



Informatica® Multidomain MDM
10.4

구성 가이드

Informatica Multidomain MDM 구성 가이드

10.4

2020년3월

© 저작권 Informatica LLC 2001, 2020

이 소프트웨어와 설명서는 사용 및 공개에 대한 제한 사항이 포함되어 있는 별도의 사용권 계약에 따라서만 제공됩니다. 본 문서의 어떤 부분도 Informatica LLC의 사전 통지 없이 어떠한 형태나 수단(전자적, 사진 복사, 녹음 등)으로 복제되거나 전송될 수 없습니다.

미국 정부 권한. 미국 정부 고객에게 제공되는 프로그램, 소프트웨어, 데이터베이스, 관련 문서 및 기술 데이터는 해당하는 연방 입수 규정 및 기관별 보안 규정에 따라 "상용 컴퓨터 소프트웨어" 또는 "상용 기술 데이터"입니다. 따라서 사용, 복제, 공개, 수정 및 조정은 해당하는 정부 계약에 규정된 제한 사항 및 라이선스 조건을 따르며, 정부 계약 조건에 의해 적용 가능한 한도 내에서, FAR 52.227-19, 상용 소프트웨어 라이선스에 규정된 추가 권한이 적용됩니다.

Informatica, Informatica 로고 및 ActiveVOS는 미국과 전 세계 여러 관할 국가에서 Informatica LLC의 상표 또는 등록 상표입니다. Informatica 상표의 현재 목록은 <https://www.informatica.com/trademarks.html>에서 확인할 수 있습니다. 다른 회사 및 제품명은 해당 소유자의 상표 또는 등록 상표일 수 있습니다.

이 소프트웨어 및/또는 설명서의 일부에는 타사의 저작권이 적용될 수 있습니다. 필요한 타사 고지 사항은 제품에 포함되어 있습니다.

이 설명서의 정보는 예고 없이 변경될 수 있습니다. 이 문서에서 문제가 발견되는 경우 infa_documentation@informatica.com으로 보고해 주십시오.

Informatica 제품은 제품이 제공될 당시의 계약 조건에 따라 보증됩니다. Informatica는 상품성과 특정 목적에의 적합성에 대한 보증 그리고 비침해에 대한 보증 또는 조건을 포함하여 어떠한 종류의 명시적이거나 묵시적인 보증 없이 이 문서의 정보를 "있는 그대로" 제공합니다.

발행 날짜: 2020-06-05

목차

서문	27
Informatica 리소스	27
Informatica 네트워크	27
Informatica 기술 자료	27
Informatica 설명서	27
Informatica Product Availability Matrix	28
Informatica Velocity	28
Informatica Marketplace	28
Informatica 글로벌 고객 지원 센터	28
파트 I: 소개	29
장 1: Informatica MDM Hub 관리	30
Informatica MDM Hub 관리 개요	30
Informatica MDM Hub 관리의 단계	30
시작 단계	31
구성 단계	31
프로덕션 단계	31
장 2: MDM Hub 콘솔 시작	32
개요	32
MDM Hub 콘솔 정보	32
Hub 콘솔 시작	32
MDM Hub 콘솔 탐색	33
프로세스 및 작업 영역 보기 간에 전환	33
작업 영역 보기에서 도구 시작	34
메타데이터 변경을 위한 잠금 획득	34
대상 데이터베이스 변경	37
다른 사용자로 로그인	37
사용자 암호 변경	37
탐색 창에서 탐색 트리 사용	37
명령 단추를 사용하여 개체 추가, 편집 및 제거	39
MDM Hub 콘솔 인터페이스 사용자 지정	40
버전 정보 표시	40
Informatica MDM Hub 작업 영역 및 도구	41
구성 작업 영역의 도구	41
모델 작업 영역의 도구	41
보안 액세스 관리자 작업 영역의 도구	42
데이터 스튜어드 작업 영역의 도구	42
유틸리티 작업 영역의 도구	43

장 3: 다국어 데이터 지원 구성	44
다국어 데이터 지원 구성 개요	44
유니코드 데이터베이스 구성(Oracle만 해당)	44
미국 외 인구집단에 대한 일치 설정 구성	45
일치 처리에 대한 인코딩 구성	45
단일 기본 개체 내에서 여러 채우기 사용	45
Windows 레지스트리에서 ANSI 코드 페이지 구성	46
유니코드에 대한 정리 설정	47
UTF-8을 사용하는 경우 UNIX에 대한 로캘 권장 사항	47
손상된 데이터 문제 해결	47
Oracle 환경의 언어 설정	48
NLS_LANG 구문	48
Windows 레지스트리에서 NLS_LANG 구성	48
NLS_LANG을 환경 변수로 구성(Windows)	48
LANG 환경 변수 및 로캘 설정(UNIX)	49
파트 II: Hub 콘솔 도구 구성	50
장 4: Hub 콘솔 도구에 대한 액세스 권한 구성	51
Hub 콘솔 도구에 대한 액세스 권한 구성 개요	51
사용자 구성	51
도구 및 프로세스에 대한 사용자 액세스 권한	52
도구 액세스 도구 시작	52
도구 및 프로세스에 대한 사용자 액세스 권한 부여	52
도구 및 프로세스에 대한 사용자 액세스 권한 취소	52
장 5: Hub 콘솔 도구에서 사용자 지정 단추 구현	53
개요	53
Hub 콘솔의 사용자 지정 단추 정보	53
사용자가 사용자 지정 단추를 클릭하면 수행되는 작업	54
Hub 콘솔에 사용자 지정 단추가 나타나는 방식	54
사용자 지정 단추 추가	54
사용자 지정 함수 작성	54
사용자 지정 단추 모양 제어	57
사용자 지정 단추 배포	57
파트 III: 데이터 모델 작성	59
장 6: Hub 저장소 정보	60
개요	60
Hub 저장소의 데이터베이스	60
Hub 저장소 데이터베이스 간의 관계	61

Hub 저장소 데이터베이스 생성.....	61
버전 요구 사항.....	61
장 7: 연산 참조 저장소 및 데이터 소스 구성.....	62
연산 참조 저장소 및 데이터 소스 구성 개요.....	62
시작하기 전에.....	62
데이터베이스 도구 정보.....	62
데이터베이스 도구 시작.....	63
연산 참조 저장소 구성.....	63
Microsoft SQL Server에 대한 연산 참조 저장소 연결 속성.....	63
Oracle에 대한 연산 참조 저장소 연결 속성.....	64
IBM DB2에 대한 연산 참조 저장소 연결 속성.....	65
연산 참조 저장소 등록.....	66
연산 참조 저장소 등록 속성 편집.....	67
연산 참조 저장소 속성 편집.....	68
연산 참조 저장소 연결.....	70
암호 변경.....	70
암호 암호화.....	71
연산 참조에 대한 프로덕션 모드.....	72
연산 참조 저장소 등록 해제.....	73
IBM DB2에서 연산 참조 저장소 삭제.....	73
데이터 소스 구성.....	73
WebLogic에서 데이터 소스 관리.....	73
데이터 소스 생성.....	74
데이터 소스 제거.....	74
장 8: 스키마 작성.....	75
개요.....	75
시작하기 전에.....	75
스키마 정보.....	75
연산 참조 저장소의 테이블 유형.....	76
스키마 개체를 정의하기 위한 요구 사항.....	78
스키마 관리자 시작.....	93
기본 개체 구성.....	94
Hub 저장소의 기본 개체와 다른 테이블 간 관계.....	94
기본 개체 정의 프로세스 개요.....	95
기본 개체 열.....	95
교차 참조 테이블.....	96
기록 테이블.....	100
기본 개체 속성.....	101
기본 개체 생성.....	104
기본 개체 속성 편집.....	104

기본 개체의 사용자 지정 인덱스.....	105
기본 개체의 영향 분석 보기.....	106
기본 개체 삭제.....	106
테이블의 열 구성.....	107
열 정보.....	107
열 편집기로 이동.....	111
열 추가.....	112
다른 테이블에서 열 정의 가져오기.....	112
열 속성 편집.....	113
열 표시 순서 변경.....	114
열 삭제.....	114
기본 개체 간의 외래 키 관계 구성.....	115
외래 키 관계 정보.....	115
외래 키 관계를 정의하기 위한 프로세스 개요.....	115
외래 키 관계 추가.....	115
외래 키 관계 편집.....	116
관계 세부 정보.....	116
외래 키 관계에 대한 조회 구성.....	117
외래 키 관계 삭제.....	117
스키마 보기.....	118
스키마 뷰어 시작.....	118
스키마 다이어그램 확대/축소.....	119
스키마 다이어그램 보기 전환.....	119
관련 디자인 개체 및 일괄 작업으로 이동.....	119
스키마 뷰어 옵션 구성.....	120
스키마 다이어그램을 JPG 이미지로 저장.....	120
스키마 다이어그램 인쇄.....	121
장 9: 쿼리 및 패키지.....	122
쿼리 및 패키지 개요.....	122
쿼리 도구.....	123
패키지 도구.....	123
쿼리 및 패키지 유지 관리.....	124
쿼리 그룹.....	125
쿼리 그룹 추가.....	125
쿼리 그룹 편집.....	125
쿼리 그룹 삭제.....	125
일반 쿼리.....	126
일반 쿼리 추가.....	126
일반 쿼리 구체화.....	127
쿼리 결과 보기.....	132
쿼리의 영향 보기.....	132

쿼리 삭제.	132
사용자 지정 쿼리.	133
사용자 지정 쿼리의 SQL 구문.	133
SQL 유효성 검사.	134
사용자 지정 쿼리 추가.	134
사용자 지정 쿼리 편집.	135
패키지.	135
표시 패키지.	135
업데이트 패키지.	135
패키지 추가.	136
패키지 편집.	137
쿼리 변경 후 패키지 새로 고침.	137
패키지 삭제.	138
조인 쿼리 지정.	138

장 10: 시간 표시 막대. 139

개요.	139
지침.	140
예제.	140
레코드 버전.	141
레코드 버전 예제.	142
시간 표시 막대 세분성.	143
기록 및 상태 관리.	144
시간 표시 막대 적용 규칙.	144
유효 기간 계산.	144
규칙 1. 유효 기간 없는 레코드 추가.	145
규칙 2. 유효 기간 없는 레코드 추가.	145
규칙 3. 유효 기간에 대한 레코드 버전 추가.	145
규칙 4. 유효 기간이 교차하고 연장된 시작 날짜가 있는 레코드 버전 추가.	146
규칙 5. 유효 기간이 교차하고 이전 종료 날짜가 있는 레코드 버전 추가.	146
규칙 6. 유효 기간 내에 포함된 레코드 버전 추가.	147
규칙 7. 유효 기간을 포함하는 레코드 버전 추가.	148
규칙 8. 비연속 유효 기간이 있는 레코드 버전 추가.	148
규칙 9. 유효 기간 내에 보류 중 상태인 레코드 버전 추가.	149
규칙 10. 레코드 버전이 잠긴 경우 레코드 버전 추가.	149
규칙 11. 버전이 보류 중 상태인 경우 레코드 버전 추가.	150
규칙 12. 연속 기본 개체의 레코드 버전 삭제 또는 업데이트.	150
규칙 13. 데이터 업데이트.	151
규칙 14. 유효 기간 업데이트.	151
규칙 15. 유효 기간 추가.	152
기본 개체 시간 표시 막대 구성.	153
1단계. Hub 콘솔에서 시간 표시 막대 활성화.	153

2단계. 속성 파일 구성.....	153
일괄 작업으로 여러 레코드 버전 로드.....	154
여러 버전의 레코드를 로드하기 위한 로드 일괄 설정.....	154
다중 레코드 버전의 일괄 로드 예제.....	154
레코드 버전의 유효 기간 편집.....	155
레코드 버전의 유효 기간 늘리기.....	155
레코드 버전의 유효 기간 줄이기.....	156
레코드 버전 추가.....	157
레코드 버전 추가 예제.....	158
레코드의 데이터 업데이트.....	158
레코드 데이터 업데이트 예제.....	158
관계 업데이트.....	159
관계 레코드의 사용자 지정 열 업데이트 예제.....	159
관계 레코드의 시스템 열 업데이트 예제.....	160
관계 종료.....	161
관계 종료 예제.....	161
관계 기간 삭제.....	162
관계 삭제 예제.....	162
모든 관계 기간 삭제.....	163
모든 관계 기간 삭제 예제.....	163
시간 표시 막대 추출 사용.....	164
시간 표시 막대 추출 속성 구성.....	165

장 11: 상태 관리 및 BPM 워크플로우 도구..... 166

상태 관리 및 BPM 워크플로우 도구 개요.....	166
예제.....	167
MDM Hub의 상태 관리.....	167
레코드 상태.....	167
보류 중인 레코드 보호.....	169
데이터 로드 규칙.....	170
레코드 상태와 기본 개체 레코드 값의 존속.....	170
BPM 워크플로우 도구.....	170
ActiveVOS에 대해 MDM Hub 구성.....	171
상태 관리 활성화.....	171
워크플로우 엔진 추가.....	172
기본 및 보조 워크플로우 어댑터 설정.....	172
사용자 역할 구성.....	173
보류 중인 레코드에서 일치 활성화.....	175
상태 전환에 대한 메시지 트리거.....	176
상태 전환에 대한 메시지 트리거 활성화.....	176
레코드 승격.....	176
데이터 스튜어드 도구에서 레코드 승격.....	177

일괄 처리 뷰어를 사용하여 승격 일괄 작업 설정.....	177
일괄 그룹 도구를 사용하여 승격 일괄 작업 설정.....	177

장 12: 데이터 암호화..... 179

데이터 암호화 개요.....	179
데이터 암호화 아키텍처.....	179
데이터 암호화 제한.....	180
데이터 암호화 유틸리티.....	180
데이터 암호화 구성.....	181
1단계. DataEncryptor 인터페이스 구현.....	181
2단계. 데이터 암호화 속성 파일 구성.....	182
3단계. Hub 서버에 대한 데이터 암호화 구성.....	182
4단계. 처리 서버에 대한 데이터 암호화 구성.....	183
서비스 통합 프레임워크 API 요청 및 응답.....	183
샘플 데이터 암호화 속성 파일.....	184

장 13: 계층..... 186

계층 개요.....	186
계층 예제.....	187
계층 구성 정보.....	187
시작하기 전에.....	188
구성 단계 개요.....	188
계층 관리자를 위한 데이터 준비.....	188
계층 관리자를 위한 데이터 준비 방법의 사용 사례.....	189
시나리오.....	189
방법론.....	189
HM 리포지토리 기본 개체 생성.....	193
기본 항목 아이콘 업로드.....	193
항목 아이콘 구성.....	194
항목 아이콘 추가.....	194
항목 아이콘 수정.....	194
항목 아이콘 삭제.....	194
항목.....	195
항목 기본 개체.....	195
항목 기본 개체 예제.....	195
항목 기본 개체 작성.....	196
기본 개체를 항목 기본 개체로 변환.....	197
항목 유형.....	198
항목 유형 예제.....	199
항목 유형 생성.....	200
항목 유형 편집.....	201
항목 유형 삭제.....	201

항목에 대한 옵션 표시.....	201
항목 기본 개체를 기본 개체로 변환.....	202
계층 유형.....	202
계층 추가.....	202
계층 삭제.....	203
관계 개체.....	203
관계 기본 개체.....	203
관계 기본 개체 생성.....	204
기본 개체를 관계 기본 개체로 변환.....	205
관계 기본 개체를 기본 개체로 변환.....	206
외래 키 관계 기본 개체.....	206
외래 키 관계 기본 개체 생성.....	207
관계 유형.....	207
관계 유형 예제.....	209
관계 유형 생성.....	210
관계 유형 편집.....	211
관계 유형 삭제.....	211
패키지.....	211
계층 관리자 데이터 구성.....	212
항목, 관계 및 FK 관계 개체 패키지 생성.....	212
항목 또는 관계 유형에 패키지 할당.....	214
프로필 정보.....	215
프로필 추가.....	215
프로필 편집.....	216
프로필 유효성 검사.....	216
프로필 복사.....	217
프로필 삭제.....	217
프로필에서 관계 유형 삭제.....	217
프로필에서 항목 유형 삭제.....	217
항목 및 관계 유형에 패키지 할당.....	218

장 14: 계층 관리자 자습서..... 219

계층 구성 개요.....	219
자습서 예제 정보.....	220
외래 키 관계.....	221
1단계. 제품 항목 기본 개체 작성.....	221
2단계. 항목 유형 작성.....	222
제품 항목 유형을 작성합니다.....	224
제품 그룹 항목 유형 작성.....	225
3단계. 제품 관계 개체 작성.....	225
4단계. 관계 유형 작성.....	226
제품 그룹이 제품 그룹의 상위 항목 관계 유형 작성.....	227

제품 그룹이 제품의 상위 항목 관계 유형 작성.	228
5단계. 계층 유형 작성.	229
6단계. 계층 프로필에 관계 유형 추가.	229
7단계. 패키지 작성.	230
제품 항목 개체 패키지 작성.	230
제품 관계 패키지 작성.	231
8단계. 패키지 할당.	231
제품 항목 유형에 PKG 제품 패키지 할당.	232
제품 그룹 항목 유형에 PKG 제품 패키지 할당.	233
제품 그룹이 제품 그룹의 상위 항목 관계 유형에 PKG 제품 관계 패키지 할당.	233
제품 그룹이 제품의 상위 항목 관계에 PKG 제품 관계 패키지 할당.	235
9단계. 계층 관리자의 데이터 표시 구성.	235
제품 항목 유형 레이블 구성.	236
제품 그룹 항목 유형에 대해 레이블 구성.	237
제품 항목 유형에 대한 도구 설명 텍스트 구성.	237
제품 그룹 항목 유형에 대한 도구 설명 텍스트 구성.	238
제품 그룹이 제품 그룹의 상위 항목 관계 유형에 대한 도구 설명 텍스트 구성.	239
제품 그룹이 제품의 상위 항목 관계 유형에 대한 도구 설명 텍스트 구성.	240
항목 목록 구성.	240
관계 목록 구성.	241
항목 검색 필드 구성.	242
관계 검색 필드 구성.	243
항목 검색 결과 구성.	244
관계 검색 결과 구성.	246
Hub 콘솔 계층 관리자의 항목 세부 정보 구성.	247
IDD 계층 관리자의 항목 세부 정보 구성.	248
관계 세부 정보 구성.	248
편집 가능한 항목 필드 구성.	249
관계 필드 편집 가능성.	250
항목 작성 필드 구성.	250
관계 작성 필드 구성.	251
계층 관리.	252

파트 IV: 데이터 흐름 구성..... 253

장 15: MDM Hub 프로세스..... 254

MDM Hub 프로세스 개요.	254
Informatica MDM Hub 프로세스 정보.	254
일괄 처리의 전체 데이터 흐름.	254
기본 개체 레코드의 통합 상태.	255
셀 데이터 존속 및 우선 순위 순서.	256
ROWID_OBJECT 존속.	256

랜드 프로세스.....	257
랜드 프로세스 정보.....	257
랜드 프로세스 관리.....	258
준비 프로세스.....	258
MDM Hub 준비.....	259
Informatica 플랫폼 준비.....	260
로드 프로세스.....	260
로드 프로세스 정보.....	260
로드 프로세스와 연결된 테이블.....	261
초기 데이터 로드 및 증분 로드.....	261
트러스트 설정 및 유효성 검사 규칙.....	262
로드 프로세스의 런타임 실행 흐름.....	263
로드 프로세스의 기타 고려 사항.....	266
토큰화 프로세스.....	267
일치 토큰 및 일치 키.....	267
일치 키 테이블.....	267
일치 키 예.....	268
토큰화 프로세스는 유사 항목 일치 기본 개체에만 적용됨.....	268
토큰화 프로세스의 주요 개념.....	268
토큰화 및 병합 프로세스 성능 최적화.....	270
일치 키 테이블 정리.....	271
일치 프로세스.....	273
일치 규칙.....	274
정확히 일치하는 기본 개체 및 유사 항목 일치 기본 개체.....	275
일치 프로세스에 사용되는 지원 테이블.....	276
인구집단 집합.....	276
중복 데이터에 대한 일치.....	276
일치 그룹 빌드 및 전이적 일치.....	277
수동 통합의 최대 일치 항목 수.....	277
외부 일치 작업.....	277
분산 처리 서버.....	277
응용 프로그램 서버 또는 데이터베이스 서버 실패 처리.....	277
통합 프로세스.....	278
통합 프로세스 정보.....	278
통합 옵션.....	279
통합 및 워크플로우 통합.....	280
게시 프로세스.....	280
Informatica MDM Hub가 지원하는 JMS 모델.....	280
조정된 데이터의 아웃바운드 분포에 대한 게시 프로세스.....	280
게시 프로세스 메시지 트리거.....	280
아웃바운드 JMS 메시지 대기열.....	281
ORS 관련 XML 메시지 스키마.....	281

게시 프로세스의 런타임 흐름.....	282
장 16: 랜드 프로세스 구성.....	283
랜드 프로세스 구성 개요.....	283
소스 시스템 구성.....	283
소스 시스템 정보.....	283
시스템 및 트러스트 도구 시작.....	284
소스 시스템 속성.....	285
소스 시스템 추가.....	285
소스 시스템 속성 편집.....	285
소스 시스템 제거.....	286
랜딩 테이블 구성.....	286
랜딩 테이블 정보.....	286
랜딩 테이블 열.....	286
랜딩 테이블 속성.....	287
랜딩 테이블 추가.....	287
랜딩 테이블 속성 편집.....	288
랜딩 테이블 제거.....	288
장 17: MDM Hub 준비.....	289
MDM Hub 준비 개요.....	289
MDM Hub 준비 테이블.....	290
MDM Hub 준비 프로세스 테이블.....	290
준비 테이블 속성.....	291
매핑 속성.....	292
MDM Hub 준비 선행 조건.....	292
준비 테이블 추가.....	292
랜딩 테이블과 준비 테이블 간의 열 매핑.....	295
데이터는 정리되거나, 변경되지 않은 채로 통과됨.....	296
매핑 도구 시작.....	297
매핑 작성.....	298
기본 키 매핑.....	300
열 매핑.....	301
매핑의 레코드 필터링.....	302
정리 함수를 사용하는 매핑 관리.....	303
행 ID별 로드.....	303
감사 내역 및 델타 검색 구성.....	304
준비 테이블에 대한 감사 내역 구성.....	304
준비 테이블에 대한 델타 검색 구성.....	305
준비 테이블 관리.....	307
준비 테이블의 속성 변경.....	307
준비 테이블의 소스 시스템으로 이동.....	308

준비 테이블 제거.....	308
매핑 관리.....	309
매핑 속성 편집.....	309
매핑 복사.....	309
스키마로 이동.....	309
매핑 테스트.....	310
매핑 제거.....	310

장 18: 영구 삭제 검색..... 311

영구 삭제 검색 개요.....	311
영구 삭제 검색 유형.....	312
기본 개체의 플래그 값 삭제.....	312
트러스트 및 유효성 검사 규칙.....	312
영구 삭제 검색 테이블.....	313
영구 삭제 검색 구성.....	314
영구 삭제 검색을 위한 기본 키 열 지정.....	318
직접 삭제.....	318
직접 삭제 플래그 지정을 위한 영구 삭제 검색 구성.....	318
종료 날짜를 사용한 직접 삭제를 위한 영구 삭제 검색 설정.....	319
종료 날짜를 사용한 직접 삭제 코드 예제.....	319
직접 삭제를 위한 영구 삭제 검색 예.....	320
합의 삭제.....	321
합의 삭제 플래그 지정을 위한 영구 삭제 검색 구성.....	321
종료 날짜를 사용한 합의 삭제를 위한 영구 삭제 검색 설정.....	322
종료 날짜를 사용한 합의 삭제 코드 예제.....	322
합의 삭제를 위한 영구 삭제 검색 예.....	323
사용자 종료 내에서 영구 삭제 검색 사용.....	326

장 19: 데이터 정리 구성..... 327

데이터 정리 구성 개요.....	327
MDM Hub에서 데이터 정리 설정.....	327
데이터 정리에 대해 처리 서버 구성.....	328
정리 작업 모드.....	328
배포 데이터 정리.....	328
정리 요청.....	329
처리 서버 도구 시작.....	329
처리 서버 속성.....	329
처리 서버 추가.....	331
처리 서버에 대한 보안 통신 활성화.....	331
처리 서버 속성 편집.....	331
처리 서버 삭제.....	332
처리 서버 구성 테스트.....	332

정리 함수 구성.....	332
정리 함수 도구 시작.....	333
정리 함수 유형.....	333
정리 함수 속성.....	333
정리 함수 구성 개요.....	334
사용자 정리 라이브러리 구성.....	334
Java 정리 라이브러리 구성.....	335
정규식 함수 추가.....	336
그래프 함수 구성.....	336
함수 테스트.....	340
정리 함수에 조건 사용.....	341
조건부 실행 구성 요소 정보.....	341
조건부 실행 구성 요소를 사용하는 경우.....	341
조건부 실행 구성 요소 추가.....	341
정리 목록 구성.....	342
정리 목록 정보.....	342
정리 목록 추가.....	342
정리 목록 속성.....	343
정리 목록 속성 편집.....	344
Informatica 플랫폼에서 데이터 정리 설정.....	346
맷에 변환 추가.....	347
매핑 구성.....	348
장 20: 로드 프로세스 구성	352
개요.....	352
시작하기 전에.....	352
데이터를 로드하기 위한 태스크 구성.....	353
준비 테이블 구성.....	353
준비 테이블 열.....	353
소스 시스템 키 유지.....	355
최상위 예약된 키 지정.....	355
최상위 예약된 키 예제.....	356
셀 업데이트 활성화.....	356
준비 테이블의 열에 대한 속성.....	356
준비 테이블의 속성 변경.....	358
외래 키 열에 대한 조회.....	359
초기 데이터 로드 구성.....	360
소스 시스템에 대한 트러스트 구성.....	360
트러스트 정보.....	360
트러스트 속성.....	362
트러스트 값 설정 시 고려 사항.....	363
열 트러스트.....	363

유효성 검사 규칙 구성.....	365
유효성 검사 규칙 정보.....	365
열의 유효성 검사 규칙 활성화.....	367
유효성 검사 규칙 노드로 이동.....	367
유효성 검사 규칙 속성.....	368
유효성 검사 규칙 추가.....	370
유효성 검사 규칙 속성 편집.....	371
유효성 검사 규칙 순서 변경.....	371
유효성 검사 규칙 제거.....	372

장 21: 일치 프로세스 구성..... 373

시작하기 전에.....	373
일치 프로세스에 대한 태스크 구성.....	373
데이터 이해.....	373
일치 프로세스와 연결된 기본 개체 속성.....	374
일치 규칙을 정의하기 위한 구성 단계.....	374
다국어 데이터가 포함된 기본 개체 구성.....	374
분산 일치 구성.....	374
데이터 로드 구성.....	374
일치/병합 설정 세부 정보 대화 상자로 이동.....	375
기본 개체에 대한 일치 속성 구성.....	376
일치 속성 설정.....	376
일치 속성.....	376
긴 ROWID_OBJECT 값 지원.....	379
관련 레코드의 일치 경로 구성.....	380
일치 경로.....	380
외래 키 관계 및 필터.....	380
관계 기본 개체.....	380
테이블 간 경로.....	380
테이블 간 경로에 대한 기본 개체 예.....	381
예제 기본 개체의 열.....	381
예제 구성 단계.....	382
테이블 내 경로.....	382
테이블 내 경로에 대한 예제 기본 개체.....	383
예제 기본 개체의 열.....	383
관계 기본 개체 생성.....	383
예제 구성 단계.....	384
경로 탭으로 이동.....	384
일치 경로에 대한 필터 구성.....	385
경로 구성 요소 구성.....	387
표시 이름.....	387
실제 이름.....	387

누락된 하위 레코드 허용.	387
제약 조건.	388
경로 구성 요소 추가.	388
경로 구성 요소 편집.	388
경로 구성 요소 삭제.	389
일치 열 구성.	389
일치 열 정보.	389
유사 항목 일치 기본 개체에 대한 일치 열 구성.	392
정확히 일치하는 기본 개체에 대한 일치 열 구성.	395
일치 규칙 집합.	397
일치 규칙 집합 속성.	398
일치 규칙 집합 탭으로 이동.	400
일치 규칙 집합 추가.	401
일치 규칙 집합 속성 편집.	401
일치 규칙 집합 이름 바꾸기.	401
일치 규칙 집합 삭제.	402
일치 규칙 집합에 대한 일치 열 규칙 구성.	402
일치 열 규칙을 구성하기 위한 선행 조건.	402
정확하게 일치하는 기본 개체와 유사 항목 일치 기본 개체 간의 일치 열 규칙 차이점.	402
일치된 레코드에 대한 통합 옵션 지정.	403
유사 항목 일치 기본 개체만을 위한 일치 규칙 속성.	403
일치 규칙에 대한 일치 열 속성.	411
일치 열 규칙의 정확히 일치 열에 대한 요구 사항.	418
열 일치 규칙을 구성하기 위한 명령 단추.	418
일치 열 규칙 추가.	419
일치 열 규칙 편집.	420
일치 열 규칙 삭제.	421
일치 열 규칙의 실행 순서 변경.	421
일치 열 규칙에 대한 통합 옵션 지정.	421
열의 일치 가중치 구성.	422
열에 대한 세그먼트 일치 구성.	422
기본 키 일치 규칙 구성.	423
기본 키 일치 규칙 정보.	423
기본 키 일치 규칙 추가.	423
기본 키 일치 규칙 편집.	424
기본 키 일치 규칙 삭제.	424
일치 키의 분포 조사.	425
일치 키 분포 정보.	425
일치 키 분포 탭으로 이동.	425
일치 키 분포 탭의 구성 요소.	425
일치 키 필터링.	426
일치 프로세스에서 레코드 제외.	427

근접 검색.....	427
근접 검색 구성.....	428
경량 일치.....	429

장 22: 일치 규칙 구성 예제 431

일치 규칙 구성 예제 개요.....	431
일치 규칙 구성 시나리오.....	432
일치 규칙 구성.....	433
1단계. 데이터 검토.....	433
2단계. 일치 작업에 대한 기본 개체 식별.....	434
3단계. 일치 속성 구성.....	434
일치 속성 구성.....	434
4단계. 일치 경로 정의.....	435
일치 경로 구성 요소 추가.....	435
5단계. 일치 열 정의.....	440
일치 열 정의.....	440
6단계. 일치 규칙 집합 정의.....	445
일치 규칙 집합 정의.....	446
7단계. 일치 규칙 추가.....	447
일치 규칙 추가.....	447
8단계. 일치 규칙에 대해 병합 옵션 설정.....	449
자동 병합 일치 규칙으로 일치 규칙 설정.....	450
9단계. 일치 속성 검토.....	450
일치 속성 검토.....	451
10단계. 일치 규칙 테스트.....	452

장 23: Elasticsearch를 사용한 검색..... 454

Elasticsearch를 사용한 검색 개요.....	454
Elasticsearch 아키텍처를 사용한 검색.....	454
검색 설치 및 구성.....	455
1단계. Elasticsearch 설치 및 설정.....	456
설치 전 태스크 완료.....	456
Elasticsearch 설치.....	457
Elasticsearch JVM(Java Virtual Machine) 구성.....	457
Elasticsearch 속성 파일 구성.....	458
Elasticsearch 보호.....	459
분석 플러그 인 설치.....	459
검색에서 무시할 단어 목록 사용자 지정.....	459
검색에 포함할 동의어 목록 사용자 지정.....	459
Elasticsearch 시작.....	460
2단계. 검색에 대한 MDM Hub 속성 구성.....	460
Hub 서버 속성 구성.....	460

처리 서버 속성 구성.	462
3단계. 프로비저닝 도구를 사용하여 검색 구성.	464
Elasticsearch 클러스터 구성.	464
사용자 지정 Elasticsearch 인덱스 설정 생성(선택 사항).	465
검색 가능한 필드 구성.	468
검색 또는 쿼리 결과 표시 구성.	470
유사한 레코드를 표시하도록 레이아웃 구성(선택 사항).	471
4단계. 연산 참조 저장소 유효성 검사.	473
5단계. 검색 데이터 인덱싱.	473
키 저장소, 트러스트 저장소 및 인증서 생성(선택 사항).	473
장 24: 통합 프로세스 구성.	475
통합 프로세스 구성 개요.	475
통합 설정.	475
변경할 수 없는 Rowid 개체.	475
고유 시스템.	476
상위 항목이 병합 해제된 경우 하위 항목 병합 해제(계단식 병합 해제).	476
통합 설정 변경.	479
장 25: 보류 중인 제어 테이블.	480
장 26: 게시 프로세스 구성.	481
게시 프로세스 개요.	481
게시 프로세스에 대한 구성 단계.	482
메시지 대기열 도구 시작.	482
글로벌 메시지 대기열 설정 구성.	482
메시지 대기열 서버 구성.	483
메시지 대기열 서버 속성.	483
메시지 대기열 서버 추가.	484
메시지 대기열 서버 속성 편집.	484
메시지 대기열 서버 삭제.	485
아웃바운드 메시지 대기열 구성.	485
메시지 대기열 정보.	485
메시지 대기열 속성.	485
메시지 대기열 서버에 메시지 대기열 추가.	485
메시지 대기열 속성 편집.	486
메시지 대기열 삭제.	486
JMS 메시지의 병렬 처리 구성.	487
JMS 보안 구성.	487
메시지 트리거 구성.	487
메시지 트리거에 대한 이벤트 유형.	488
메시지 트리거에 대한 모범 사례.	489

메시지 트리거 시스템 속성.	489
메시지 트리거 추가.	490
메시지 트리거 편집.	491
메시지 트리거 삭제.	491
메시지 대기열 폴링 비활성화.	492
JMS 메시지 XML 참조.	492
ORS 관련 XML 메시지 스키마 생성.	492
XML 메시지의 요소.	492
메시지 필터링.	494
XML 메시지 예.	494
레거시 JMS 메시지 XML 참조.	505
레거시 XML에 대한 메시지 필드.	506
레거시 XML에 대한 메시지 필터링.	506
레거시 XML에 대한 메시지 예.	506

파트 V: Informatica MDM Hub 프로세스 실행. 518

장 27: 일괄 작업 사용. 519

일괄 작업 사용 개요.	519
일괄 작업 스레드 구성.	520
다중 스레드 일괄 작업 프로세스.	520
다중 스레드 일괄 작업 예.	520
다중 스레드 일괄 처리 성능.	521
다중 스레드 일괄 작업 속성.	521
일괄 작업 실행.	521
일괄 작업에 사용되는 지원 테이블.	522
차례로 일괄 작업 실행.	522
일괄 작업을 실행하기 전에 랜딩 테이블 채우기.	522
일치 작업과 이후 통합 작업.	522
상위 테이블에서 먼저 데이터 로드.	523
외래 키 관계가 있는 개체의 데이터 로드.	523
일괄 작업에 대한 모범 사례.	523
Oracle 환경에서의 통계 수집을 위한 병렬도 제한.	523
일괄 작업 생성.	524
자동으로 생성되는 일괄 작업.	524
변경이 발생할 때 생성되는 일괄 작업.	525
정보 전용 일괄 작업(Hub 콘솔에서 실행되지 않음).	525
처리 서버 구성.	525
일괄 처리 뷰어 도구를 사용하여 일괄 작업 실행.	526
일괄 처리 뷰어 도구.	526
일괄 처리 뷰어 도구 시작.	526
테이블, 데이터 또는 프로시저 유형별로 그룹화.	526

수동으로 일괄 작업 실행.....	527
작업 실행 로그 보기.....	529
작업 실행 기록 지우기.....	533
일괄 그룹 도구를 사용하여 일괄 작업 실행.....	534
일괄 그룹 정보.....	534
일괄 그룹 도구 시작.....	535
일괄 그룹 구성.....	535
일괄 그룹 목록 새로 고침.....	539
일괄 그룹 도구를 사용하여 일괄 그룹 실행.....	539
상태별로 실행 로그 필터링.....	542
일괄 그룹 삭제.....	543
일괄 작업 참조.....	543
일괄 작업의 사전순 목록.....	543
일치되지 않은 레코드를 고유 레코드로 허용.....	544
자동 일치 및 병합 작업.....	545
자동 병합 작업.....	545
일괄 병합 해제.....	546
BVT 스냅샷 작업.....	547
외부 일치 작업.....	547
일치 토큰 생성 작업.....	550
스마트 검색 데이터 초기 인덱스 작업.....	551
키 일치 작업.....	552
로드 작업.....	552
수동 병합 작업.....	556
수동 병합 해제 작업.....	557
일치 작업.....	557
일치 분석 작업.....	559
중복 데이터에 대한 일치 작업.....	560
다중 병합 작업.....	561
승격 작업.....	561
기본 개체 다시 계산 작업.....	563
BVT 다시 계산 작업.....	563
일치 테이블 재설정 작업.....	563
유효성 다시 검사 작업.....	563
준비 작업.....	564
동기화 작업.....	564

장 28: 사용자 종료..... 565

사용자 종료 개요.....	565
사용자 종료 처리.....	566
사용자 종료 JAR 파일.....	567
사용자 종료 JAR 파일 구현.....	567

사용자 종료를 MDM Hub로 업로드.	567
MDM Hub에서 사용자 종료 제거.	567
UserExitContext 클래스.	568
준비 프로세스 사용자 종료.	569
사후 랜딩 사용자 종료.	570
사전 준비 사용자 종료.	571
사후 준비 사용자 종료.	571
로드 프로세스 사용자 종료.	572
사후 로드 사용자 종료.	573
일치 프로세스 사용자 종료.	574
사전 일치 사용자 종료.	575
사후 일치 사용자 종료.	575
병합 프로세스 사용자 종료.	576
사후 병합 사용자 종료.	576
병합 해제 프로세스 사용자 종료.	577
사전 병합 해제 사용자 종료.	578
사후 병합 해제 사용자 종료.	578
태스크 관리 사용자 종료.	579
AssignTasks 사용자 종료 인터페이스.	579
GetAssignableUsersForTask 사용자 종료 인터페이스.	580
사용자 종료 내에서 서비스 통합 프레임워크 API 사용.	580
서비스 통합 프레임워크 API를 호출하기 위한 사용자 종료 생성.	580
사용자 종료 예.	581
서비스 통합 프레임워크 API.	582
사용자 종료 구현 지침.	583

파트 VI: 응용 프로그램 액세스 구성..... 585

장 29: ORS 관련 API..... 586

ORS 관련 API 개요.	586
성능 고려 사항.	587
지원되는 리포지토리 개체.	587
ORS 관련 SIF API 속성.	587
리포지토리 개체 상태.	588
보관 테이블.	588
ORS 관련 SIF API 생성 및 배포.	589
ORS 관련 SIF API 이름 바꾸기.	589
ORS 관련 클라이언트 JAR 파일 다운로드.	589
ORS 관련 클라이언트 JAR 파일과 SIF SDK 사용.	590
ORS 관련 SIF API 제거.	590

장 30: ORS 관련 메시지 스키마.....	591
ORS 관련 메시지 스키마 개요.....	591
JMS 이벤트 스키마 관리자 도구 정보.....	591
JMS 이벤트 스키마 관리자 도구 시작.....	592
SIF 관리자 도구 시작.....	592
ORS 관련 스키마 생성 및 배포.....	593
XSD 파일 다운로드.....	593
동기화되지 않은 개체 찾기.....	593
동기화되지 않은 개체에 대한 자동 검색.....	594
 장 31: 등록된 사용자 지정 코드 보기.....	 595
개요.....	595
사용자 개체.....	595
사용자 개체 레지스트리 도구 시작.....	596
사용자 종료 보기.....	596
사용자 종료 정보.....	596
사용자 종료 보기.....	596
사용자 지정 Java 정리 함수 보기.....	596
사용자 지정 Java 정리 함수 정보.....	597
사용자 지정 Java 정리 함수를 등록하는 방법.....	597
등록된 사용자 지정 Java 정리 함수 보기.....	597
사용자 지정 단추 함수 보기.....	597
사용자 지정 단추 함수 정보.....	597
사용자 지정 단추 함수를 등록하는 방법.....	597
등록된 사용자 지정 단추 함수 보기.....	597
 장 32: Informatica MDM Hub 서비스 및 이벤트 감사.....	 599
개요.....	599
통합 감사 정보.....	599
감사 가능 이벤트.....	599
감사 관리자 도구.....	600
요청 및 응답의 XML 캡처.....	600
감사는 명시적으로 활성화되어야 함.....	600
감사는 인증 후에 수행됨.....	600
감사는 올바른 형식의 유효한 XML을 사용한 호출에 대해 수행됨.....	600
암호 변경 내용 감사.....	601
감사 관리자 시작.....	601
감사 가능한 API 요청 및 메시지 대기열.....	601
감사할 시스템.....	602
감사 속성.....	602
SIF API 요청 감사.....	603

메시지 대기열 감사.	603
오류 감사.	604
글로벌 오류 감사 구성.	604
감사 로그 사용.	604
감사 로그 정보.	605
감사 로그 테이블.	605
감사 로그 보기.	607
정기적 감사 로그 제거.	607
부록 A: MDM Hub 속성.	608
MDM Hub 속성 개요.	608
Hub 서버 속성.	608
샘플 Hub 서버 속성 파일.	625
처리 서버 속성.	627
샘플 처리 서버 속성 파일.	634
연산 참조 저장소 속성.	635
부록 B: 구성 세부 정보 보기.	637
구성 세부 정보 개요 보기.	637
엔터프라이즈 관리자 시작.	637
엔터프라이즈 관리자의 속성.	638
C_REPOS_DB_RELEASE 테이블.	638
환경 보고서.	640
MDM Hub 환경 보고서 저장.	640
엔터프라이즈 관리자에서 버전 기록 보기.	640
응용 프로그램 서버 로그 사용.	640
응용 프로그램 서버 로그 수준.	641
로그 파일 롤링.	641
응용 프로그램 서버 로그 구성.	642
클라이언트용 Hub 콘솔 로그 사용.	642
부록 C: 행 수준 잠금.	643
행 수준 잠금 개요.	643
행 수준 잠금 정보.	643
기본 동작.	643
잠금 유형.	644
행 수준 잠금을 사용하기 위한 고려 사항.	644
행 수준 잠금 구성.	644
ORS에 대해 행 수준 잠금 활성화.	644
잠금 대기 시간 구성.	645
SIF 요청과 일괄 처리 사이의 상호 작용 잠금.	645
API 일괄 처리 상호 운용성이 활성화된 경우의 상호 작용.	645

API 일괄 처리 상호 운용성이 비활성화된 경우의 상호 작용.	645
부록 D: MDM Hub 로깅.	646
MDM Hub 로깅 개요.	646
로깅 설정 구성.	647
Hub 콘솔 로그.	647
Hub 서버 로그.	647
처리 서버 로그.	648
Informatica Platform 로그.	648
Entity 360 로그.	648
프로비저닝 도구 로그.	648
부록 E: 테이블 분할.	650
테이블 분할 지원.	650
부록 F: 제품 사용 툴킷을 사용하여 MDM 환경 정보 수집.	651
제품 사용 툴킷을 사용하여 MDM 환경 정보 수집 개요.	651
시스템 구성 정보.	652
MDM Hub 환경 정보.	652
Hub 서버에서 MDM Hub 데이터 수집 활성화.	653
처리 서버에서 MDM Hub 데이터 수집 활성화.	653
Hub 서버에서 MDM Hub 데이터 수집 비활성화.	653
처리 서버에서 MDM Hub 데이터 수집 비활성화.	654
부록 G: Informatica 플랫폼 준비.	655
Informatica Platform 준비 개요.	655
Informatica 플랫폼과 MDM Hub 통합.	656
Informatica 플랫폼 준비 구성 요소.	656
모델 리포지토리 개체.	657
준비 테이블 속성.	658
데이터 소스 연결 속성.	659
Informatica 플랫폼 준비 프로세스.	665
통합 선행 조건 완료.	665
준비에 사용할 MDM Hub 준비.	666
1단계. 소스 시스템 구성.	666
2단계. 준비 테이블 추가.	666
동기화에 사용할 개발자 도구 준비.	669
1단계. 프로젝트 생성.	669
Hub 저장소와 모델 리포지토리 동기화.	671
모델 리포지토리 서비스 연결 매개 변수.	671
1단계. 모델 리포지토리 서비스 연결 구성.	672
2단계. 준비 활성화.	673

3단계. 모델 리포지토리와 동기화.	674
개발자 도구에서 준비 설정 완료.	675
1단계. 생성된 개체 검토.	675
2단계. 소스 시스템 연결 생성.	681
3단계. 연결 탐색기 보기에 연결 추가.	683
4단계. 소스 연결에 대한 실제 데이터 개체 생성.	685
5단계. 대상에 대한 연결 생성.	688
6단계. 연결 탐색기 보기에 연결 추가.	690
7단계. 실제 데이터 개체에 연결 추가.	692
8단계. 맵셋에 변환 추가.	695
매핑 구성 및 실행.	696
1단계. 매핑 구성.	697
2단계. 매핑 실행.	699
준비 테이블 관리.	700
단일 준비 테이블에 대한 준비 비활성화.	700
모든 준비 테이블에 대한 Informatica 플랫폼 준비 비활성화.	700
모든 준비 테이블에 대한 Informatica 플랫폼 준비 활성화.	701
모든 준비 테이블에 대한 변경 사항을 모델 리포지토리와 동기화.	701
추가 설명서.	702
부록 H: Informatica Platform 매핑 예.	703
Informatica Platform 매핑 예제 개요.	703
기본 키 생성 예제.	703
매핑 생성.	703
입력 데이터 샘플.	706
출력 데이터 샘플.	706
부록 I: 용어.	708
인덱스.	736

서문

Informatica® *Multidomain MDM* 구성 가이드의 지침에 따라 Informatica MDM Hub 시스템을 구성하십시오. 데이터 모델 작성, 데이터 흐름 구성, 프로세스 실행 및 응용 프로그램 액세스 권한 구성을 위해 MDM Hub 콘솔 도구를 사용하는 방법을 알아보십시오.

이 가이드에서는 사용자가 *Multidomain MDM* 개요 가이드를 검토하여 MDM Hub 아키텍처 및 핵심 개념에 대해 기본적인 내용을 파악하고 있다고 가정합니다.

Informatica 리소스

Informatica는 Informatica Network 및 기타 온라인 포털을 통해 다양한 범위의 제품 리소스를 제공합니다. 리소스를 통해 Informatica 제품 및 솔루션을 최대한 활용하고 다른 Informatica 사용자 및 주제별 전문가로부터 배울 수 있습니다.

Informatica 네트워크

Informatica Network는 Informatica 기술 자료, Informatica 글로벌 고객 지원 센터 등 여러 리소스로 연결되는 관문입니다. Informatica Network를 시작하려면 <https://network.informatica.com>을 방문하십시오.

Informatica Network 멤버인 경우 다음 옵션이 가능합니다.

- 기술 자료에서 제품 리소스를 검색할 수 있습니다.
- 제품 사용 가능 여부에 대한 정보를 봅니다.
- 지원 사례를 생성하고 검토할 수 있습니다.
- 거주 지역의 Informatica 사용자 그룹 네트워크를 검색하고 동료와 협업 관계 유지

Informatica 기술 자료

Informatica 기술 자료를 사용하여 사용 방법 문서, 모범 사례, 비디오 자습서, 자주 묻는 질문에 대한 답변 등 제품 리소스를 확인할 수 있습니다.

기술 자료를 검색하려면 <https://search.informatica.com>을 방문하십시오. 기술 자료에 대한 질문, 의견 또는 아이디어가 있는 경우 KB_Feedback@informatica.com을 통해 Informatica 기술 자료 팀에 문의해 주시기 바랍니다.

Informatica 설명서

Informatica 설명서 포털에서 확장된 설명서 라이브러리를 탐색하여 현재 및 최근 제품 릴리스를 확인할 수 있습니다. 설명서 포털을 탐색하려면 <https://docs.informatica.com>을 방문하십시오.

제품 설명서에 대한 질문, 의견 또는 아이디어가 있는 경우 infa_documentation@informatica.com에서 Informatica 설명서 팀에 문의해 주시기 바랍니다.

Informatica Product Availability Matrix

PAM(Product Availability Matrix)은 제품 릴리스에서 지원하는 운영 체제 버전, 데이터베이스 및 데이터 소스 유형과 대상을 나타냅니다.

<https://network.informatica.com/community/informatica-network/product-availability-matrices>에서 Informatica PAM을 찾을 수 있습니다.

Informatica Velocity

Informatica Velocity는 수백 가지 데이터 관리 프로젝트의 실제 경험을 토대로 Informatica 전문 서비스업에서 개발한 팁과 모범 사례 모음입니다. Informatica Velocity는 전 세계의 조직과 협력하여 성공적인 데이터 관리 솔루션을 계획, 개발, 배포 및 유지 관리하는 Informatica 컨설턴트의 포괄적인 지식을 보여줍니다.

Informatica Velocity 리소스는 <http://velocity.informatica.com>에서 확인할 수 있습니다. Informatica Velocity에 대한 질문, 주석 또는 아이디어가 있으시면 Informatica 전문 서비스업(ips@informatica.com)에 문의하십시오.

Informatica Marketplace

Informatica Marketplace는 Informatica 구현을 확대 및 개선하기 위한 솔루션을 찾을 수 있는 포럼입니다.

Marketplace에서 Informatica 개발자와 파트너가 제공하는 수백 개의 솔루션을 활용하여 생산성을 향상시키고 프로젝트의 구현에 걸리는 시간을 줄일 수 있습니다. <https://marketplace.informatica.com>에서 Informatica Marketplace를 찾을 수 있습니다.

Informatica 글로벌 고객 지원 센터

전화 또는 Informatica Network를 통해 글로벌 지원 센터에 문의할 수 있습니다.

해당 지역의 Informatica 글로벌 고객 지원 전화 번호는 Informatica 웹 사이트 (<https://www.informatica.com/services-and-training/customer-success-services/contact-us.html>)를 방문하여 찾을 수 있습니다.

Informatica Network에서 온라인 지원 리소스를 찾으려면 <https://network.informatica.com>을 방문하고 eSupport 옵션을 선택하십시오.

파트 I: 소개

이 파트에 포함된 장:

- [Informatica MDM Hub 관리, 30](#)
- [MDM Hub 콘솔 시작, 32](#)
- [다국어 데이터 지원 구성, 44](#)

제 1 장

Informatica MDM Hub 관리

이 장에 포함된 항목:

- [Informatica MDM Hub 관리 개요, 30](#)
- [Informatica MDM Hub 관리의 단계, 30](#)

Informatica MDM Hub 관리 개요

Informatica MDM Hub 관리자는 Informatica MDM Hub 시스템의 구성을 일차적으로 책임집니다.

관리자는 Informatica MDM Hub 구현을 관리하는 일련의 도구로 구성된 Hub 콘솔을 통해 Informatica MDM Hub에 액세스합니다.

Informatica MDM Hub 관리자는 Hub 콘솔을 사용하여 다음 태스크를 수행합니다.

- Hub 저장소의 데이터 모델 및 기타 개체 작성
- Informatica MDM Hub 데이터 관리 프로세스 구성 및 실행
- Informatica MDM Hub 기능 및 리소스에 대한 외부 응용 프로그램 액세스 권한 구성
- 진행 중인 작업 모니터링
- 글로벌 고객 지원 센터에서 Informatica MDM Hub의 문제를 해결하는 데 필요한 로그 유지 관리

Informatica MDM Hub 관리의 단계

조직의 방법론에 따라 Informatica MDM Hub 구현의 관리 단계는 다를 수 있습니다.

MDM Hub 관리에는 다음과 같은 단계가 포함될 수 있습니다.

1. 시작. MDM Hub를 설치 및 설정합니다.
2. 구성. MDM Hub 기능을 작성 및 테스트합니다.
3. 프로덕션. 환경을 배포, 조정 및 유지 관리합니다.

시작 단계

시작 단계에는 핵심 Informatica MDM Hub 구성 요소인 Hub 저장소, Hub 서버, 처리 서버 및 정리 어댑터를 설치 및 구성하는 작업이 관련됩니다.

Hub 저장소, Hub 서버 및 처리 서버를 설치하는 방법에 대한 자세한 내용은 응용 프로그램 서버에 대한 *Multidomain MDM 설치 가이드*를 참조하십시오. 지원되는 외부 정리 엔진을 MDM Hub와 통합하도록 정리 어댑터를 설정하는 방법에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

참고: 이 장의 지침에서는 사용자가 시작 단계를 이미 완료했으며 Informatica MDM Hub 구현 구성 작업을 시작할 준비가 되었다고 가정됩니다.

구성 단계

Informatica MDM Hub를 설치 및 설정한 후 관리자는 Hub 저장소의 데이터 모델 및 기타 개체, 데이터 관리 프로세스, 외부 응용 프로그램 액세스 등의 Informatica MDM Hub 기능을 구성하고 테스트할 수 있습니다.

이 단계에는 조직의 정해진 요구 사항을 충족하기 위해 Informatica MDM Hub 기능을 빌드하고 테스트하는 동적이고 반복적인 프로세스가 포함됩니다. 이 장에 있는 대부분의 내용에서는 구성 단계와 관련된 태스크를 다룹니다.

스키마가 충분히 작성되고 Informatica MDM Hub가 올바르게 구성된 후 개발자는 Informatica MDM Hub 기능 및 리소스에 액세스하는 외부 응용 프로그램을 빌드할 수 있습니다. 외부 응용 프로그램 개발에 대한 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

프로덕션 단계

Informatica MDM Hub 구현을 충분히 구성 및 테스트했으면 관리자는 프로덕션 환경에 Informatica MDM Hub를 배포합니다.

이 단계에는 진행 중인 Informatica MDM Hub 작업을 관리하는 작업 외에도 실제 비즈니스 데이터 처리를 최적화하도록 성능을 조정하는 작업이 포함될 수 있습니다.

참고: MDM Hub 관리자는 적시에 문제를 해결할 수 있도록 필요한 모든 로그 파일에 대한 액세스 권한을 제공해야 합니다.

제 2 장

MDM Hub 콘솔 시작

이 장에 포함된 항목:

- [개요, 32](#)
- [MDM Hub 콘솔 정보, 32](#)
- [Hub 콘솔 시작, 32](#)
- [MDM Hub 콘솔 탐색, 33](#)
- [Informatica MDM Hub 작업 영역 및 도구, 41](#)

개요

이 장에서는 Hub 콘솔에 대해 소개하며, Informatica MDM Hub 구현을 구성하는 데 필요한 도구에 대한 고급 개요가 제공됩니다.

MDM Hub 콘솔 정보

관리자와 데이터 스튜어드는 Hub 콘솔이라는 Informatica MDM Hub 사용자 인터페이스를 사용하여 Informatica MDM Hub 기능에 액세스할 수 있습니다. Hub 콘솔은 도구 집합으로 구성됩니다. 각 도구를 사용하여 특정 작업이나 일련의 관련 작업을 수행할 수 있습니다.

참고: Hub 콘솔에서 사용 가능한 도구는 Informatica 사용권 계약에 따라 다릅니다.

Hub 콘솔 시작

MDM Hub에 액세스하려면 HTTP 또는 HTTPS 연결을 사용하여 Hub 콘솔을 시작합니다.

Hub 콘솔을 시작하기 전에 다음과 같은 정보가 있는지 확인합니다.

- URL의 호스트 이름 및 포트 번호

- 사용자 이름 및 암호

1. 브라우저 창을 열고 다음 URL을 입력합니다.

`http://<MDM Hub 호스트>:<포트 번호>/cmx/`

Hub 콘솔 실행 페이지가 표시됩니다.

2. 사용자 이름 및 암호를 입력한 후 **다운로드**를 클릭합니다.

Hub 콘솔을 실행하는 데 필요한 MDM Hub 응용 프로그램 JAR 파일이 다운로드됩니다.

참고: MDM Hub 응용 프로그램 JAR 파일을 다운로드할 수 없는 경우에는 MDM 관리자에게 문의하십시오. 관리자는 다음 디렉터리에서 JAR 파일을 배포할 수 있습니다. <MDM Hub 설치 디렉터리>/hub/server/resources/hub

3. 응용 프로그램 JAR 파일을 실행합니다.

응용 프로그램에 대한 최대 메모리 할당 풀을 지정하려면 다음 명령을 실행합니다.

```
java -Xmx<n>G -jar hubConsole.jar
```

여기서 <n>은 GB 단위의 최대 메모리 할당입니다.

Informatica MDM Hub 로그인 대화 상자가 표시됩니다.

4. 사용자 이름과 암호를 입력하고 **확인**을 클릭합니다.

데이터베이스 변경 대화 상자가 표시됩니다.

5. 대상 데이터베이스를 선택합니다.

대상 데이터베이스는 MDM Hub 마스터 데이터베이스입니다.

6. 목록에서 언어를 선택하고 **연결**을 클릭합니다.

Hub 콘솔 사용자 인스턴스가 선택한 언어로 표시됩니다. Hub 콘솔 사용자 인터페이스에서 표시되는 언어를 변경해야 하는 경우 언어를 선택하여 Hub 콘솔을 다시 시작합니다.

MDM Hub 콘솔 탐색

Hub 콘솔은 Informatica MDM Hub 구현을 구성하고 관리하는 데 사용하는 도구 컬렉션입니다.

각 도구를 통해 Informatica MDM Hub 구현의 특정 영역에 맞춰 작업을 수행할 수 있습니다.

프로세스 및 작업 영역 보기 간에 전환

Informatica MDM Hub의 도구는 다음과 같은 두 가지 방식으로 그룹화됩니다.

보기	설명
작업 영역별	작업 영역별로 유사한 도구가 함께 그룹화됩니다. 작업 영역은 관련 도구의 논리적 컬렉션입니다.
프로세스별	태스크를 완료하는 데 필요한 도구 및 단계를 안내하는 논리적 워크플로우로 도구가 그룹화됩니다.

Hub 콘솔 창의 왼쪽에 있는 탭을 클릭하여 **프로세스** 보기와 **작업 영역** 보기 사이를 전환할 수 있습니다.

참고: Informatica MDM Hub에 로그인하면 Informatica MDM Hub 보안 관리자가 사용자에게 사용 권한을 부여한 도구가 포함된 작업 영역 및 프로세스만 표시됩니다.

작업 영역 보기

도구를 작업 영역별로 보려면

- Hub 콘솔 창의 왼쪽에 있는 **작업 영역** 탭을 클릭합니다.
Hub 콘솔의 **작업 영역** 탭에 사용 가능한 작업 영역 목록이 표시됩니다. **작업 영역** 보기에는 Hub 콘솔 도구가 유사한 기능별로 구성되어 있습니다.

작업 영역 이름과 도구 설명은 도구가 그룹화된 방식대로 메타데이터를 기반으로 합니다. 사용자 지정 도구 그룹이 있을 수도 있습니다.

프로세스 보기

도구를 프로세스별로 보려면

- Hub 콘솔 창의 왼쪽에 있는 **프로세스** 탭을 클릭합니다.
Hub 콘솔의 **프로세스** 탭에 사용 가능한 프로세스 목록이 표시됩니다. 도구는 일반적인 시퀀스나 프로세스로 구성되어 있습니다.

프로세스는 특정 태스크를 완료하기 위한 도구의 논리적 시퀀스를 단계별로 안내합니다. 동일한 도구가 여러 프로세스에 속할 수 있으며 한 프로세스에서 여러 번 나타날 수도 있습니다.

작업 영역 보기에서 도구 시작

작업 영역 보기에서 Hub 콘솔을 시작합니다.

1. **작업 영역** 보기에서 시작할 도구가 포함된 작업 영역을 확장합니다.
2. 필요한 경우 작업 영역 노드를 확장하여 해당 작업 영역과 연결된 도구를 표시합니다.
3. 도구를 클릭합니다.

다른 데이터베이스를 필요로 하는 도구를 선택한 경우 Hub 콘솔에 해당 데이터베이스를 선택하라는 메시지가 표시됩니다.

구성 작업 영역, 데이터베이스, 사용자, 보안 공급자, 도구 액세스, 메시지 대기열, 리포지토리 관리자, 엔터프라이즈 관리자 및 워크플로우 관리자의 모든 도구는 사용하려면 MDM Hub 마스터 데이터베이스에 연결해야 합니다. 그 이외의 모든 도구는 사용하려면 연산 참조 저장소에 연결해야 합니다.

Hub 콘솔에 선택한 도구가 표시됩니다.

메타데이터 변경을 위한 잠금 획득

Hub 저장소에서 메타데이터를 변경하려면 Hub 콘솔을 통해 리포지토리 테이블에서 잠금을 획득해야 합니다.

데이터 스튜어드 도구를 제외하고 Hub 콘솔을 통해 액세스하는 모든 도구는 읽기 전용 모드입니다. 도구를 사용하여 Hub 저장소의 메타데이터를 변경하려면 리포지토리 테이블에서 잠금을 획득해야 합니다.

동시에 Hub 저장소를 변경하려면 배타적 사용자 또는 다중 사용자에 대한 잠금을 획득하십시오. 다른 사용자가 유지하고 있는 쓰기 또는 배타적 잠금을 강제로 해제할 수 있습니다.

잠금 유형

쓰기 잠금 메뉴는 두 가지 유형의 잠금을 제공합니다.

다음 테이블에서는 Hub 콘솔에서 액세스할 수 있는 잠금 유형에 대해 설명합니다.

잠금 유형	설명
배타적 잠금	한 사용자만 기본 연산 참조 저장소를 변경할 수 있고 배타적 잠금이 적용되어 있는 동안 다른 사용자는 연산 참조 저장소를 변경할 수 없도록 합니다.
쓰기 잠금	여러 사용자가 동시에 기본 메타데이터를 변경할 수 있도록 합니다. MDM Hub 마스터 데이터베이스나 연산 참조 저장소에 대해 쓰기 잠금을 획득할 수 있습니다.

참고: 프로텍션 모드에 있는 연산 참조 저장소에 대한 잠금은 획득할 수 없습니다. 연산 참조 저장소가 프로텍션 모드에 있는 경우 쓰기 잠금을 획득하려고 하면 잠금을 획득할 수 없다는 메시지가 표시됩니다.

잠금이 필요한 도구

데이터베이스의 구성을 변경하려면 먼저 MDM Hub 마스터 데이터베이스 및 연산 참조 저장소의 도구에 대한 잠금을 획득해야 합니다.

MDM Hub 마스터 데이터베이스의 구성을 변경하려면 다음 도구에 대한 잠금을 획득해야 합니다.

- 데이터베이스
- 리포지토리 관리자
- 메시지 대기열
- 보안 공급자
- 도구 액세스
- 사용자

연산 참조 저장소의 구성을 변경하려면 다음 도구에 대한 잠금을 획득해야 합니다.

- 일괄 그룹
- 정리 함수
- 계층
- 계층 관리자
- 매핑
- 패키지
- 처리 서버
- 쿼리
- 역할
- 스키마 관리자
- 스키마 뷰어
- 보안 리소스
- SIF 관리자
- 시스템 및 트러스트

- 사용자 및 그룹

참고: 데이터 관리자, 병합 관리자 및 계층 관리자에는 쓰기 잠금이 필요하지 않습니다. 이러한 도구에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오. 감사 관리자에도 쓰기 잠금이 필요하지 않습니다.

자동 잠금 만료

Hub 콘솔은 현재 연결에 대해 잠금을 60초마다 새로 고칩니다. 사용자는 수동으로 잠금을 해제할 수 있습니다. 사용자가 잠금을 유지한 채 다른 데이터베이스로 전환하면 잠금이 자동으로 해제됩니다. Hub 콘솔이 종료되면 1분 후 잠금이 만료됩니다.

서버 캐싱 및 MDM Hub 콘솔 잠금

Hub 콘솔에 잠금이 없으면 Hub 서버가 성능상 이유로 메타데이터 및 기타 구성 설정을 캐시합니다. Hub 콘솔 사용자가 쓰기 잠금이나 배타적 잠금을 획득하면 캐싱이 비활성화되고 캐시가 비워지며 대신 Informatica MDM Hub가 데이터베이스에서 이 정보를 검색합니다. 모든 잠금이 해제되면 캐싱이 다시 활성화됩니다.

둘 이상의 Hub 콘솔이 동일한 연산 참조 저장소를 사용하는 경우 Hub 서버에 대한 쓰기 잠금으로 다른 Hub 서버에서 캐싱이 비활성화되지 않습니다.

쓰기 잠금 획득

쓰기 잠금을 사용하면 여러 사용자가 Hub 콘솔에서 데이터를 동시에 편집할 수 있습니다.

그러나 쓰기 잠금을 사용할 경우 이러한 사용자가 동일한 데이터를 동시에 편집하게 될 수도 있습니다. 이 경우 가장 최근에 저장된 변경 내용이 유지됩니다.

1. 쓰기 잠금 > 잠금 획득을 클릭합니다.

- 다른 사용자가 이미 잠금을 획득한 경우 해당 사용자의 로그인 이름 및 시스템 주소가 표시됩니다.
- 연산 참조 저장소가 프로덕션 모드에 있는 경우에는 잠금을 획득할 수 없음을 설명하는 메시지가 표시됩니다.
- 잠금을 획득하면 도구가 읽기-쓰기 모드로 전환됩니다. 여러 사용자가 각 연산 참조 저장소 또는 MDM Hub 마스터 데이터베이스에 대한 쓰기 잠금을 유지할 수 있습니다.

2. 작업을 완료했으면 쓰기 잠금 > 잠금 해제를 클릭합니다.

배타적 잠금 획득

Hub 콘솔에서 배타적 잠금을 획득할 수 있습니다.

1. 쓰기 잠금 > 잠금 지우기를 클릭하여 다른 사용자가 유지하고 있는 쓰기 잠금을 지웁니다.

2. 쓰기 잠금 > 배타적 잠금 획득을 클릭합니다.

연산 참조 저장소가 프로덕션 모드에 있는 경우에는 배타적 잠금을 획득할 수 없음을 설명하는 메시지가 표시됩니다.

3. 작업을 완료했으면 쓰기 잠금 > 잠금 해제를 클릭합니다.

잠금 해제

Hub 콘솔에서 잠금을 해제할 수 있습니다.

- ▶ 쓰기 잠금 > 잠금 해제를 클릭합니다.

잠금 해제

Hub 콘솔에서 잠금을 지울 수 있습니다. 다른 사용자에게 해당 쓰기 잠금이 해제되기 전에 변경 내용을 저장하라는 경고가 표시되지 않으므로 필요한 경우에만 잠금을 지웁니다.

▶ 쓰기 잠금 > 잠금 지우기를 클릭합니다.

Hub 콘솔에서 연산 참조 저장소에 대한 잠금을 해제합니다.

대상 데이터베이스 변경

Hub 콘솔 창의 아래쪽에 있는 상태 표시줄에는 연결되어 있는 대상 데이터베이스의 이름과 사용자가 로그인하는 데 사용한 사용자 이름이 표시됩니다.

1. 상태 표시줄에서 데이터베이스 이름을 클릭합니다.

Hub 콘솔에 대상 데이터베이스를 선택하라는 메시지가 표시됩니다.

2. 연결할 MDM Hub 마스터 데이터베이스 또는 연산 참조 저장소를 선택합니다.

3. 연결을 클릭합니다.

다른 사용자로 로그인

Hub 콘솔에서 다른 사용자로 로그인할 수 있습니다.

1. 다음 옵션 중 하나를 선택합니다.

- 상태 표시줄에서 사용자 이름을 클릭합니다.
- 옵션 > 다음으로 다시 로그인을 클릭합니다.

2. 사용할 사용자 계정의 사용자 이름과 암호를 지정합니다.

3. 확인을 클릭합니다.

사용자 암호 변경

현재 Hub 콘솔에 로그인되어 있는 사용자의 암호를 변경할 수 있습니다.

1. 옵션 > 암호 변경을 클릭합니다.

2. 사용할 암호를 지정합니다.

3. 확인을 클릭합니다.

탐색 창에서 탐색 트리 사용

Hub 콘솔의 탐색 트리를 사용하여 개체 컬렉션을 계층 형식으로 보고 관리할 수 있습니다.

탐색 트리는 Hub 콘솔의 탐색 창에 있습니다. 탐색 트리에서 명명된 각 개체는 노드로 표시됩니다. 다른 노드를 포함한 노드를 상위 노드라고 합니다. 상위 노드에 속한 노드를 하위 노드라고 합니다.

하위 노드 표시 및 숨기기

상위 노드 아래에 하위 노드를 표시하려면

- 상위 노드 옆의 더하기(+) 기호를 클릭합니다.

상위 노드 아래에 하위 노드를 숨기려면

- 상위 노드 옆의 빼기(-) 기호를 클릭합니다.

표시 이름을 기준으로 정렬

표시 이름은 탐색 트리에 표시되는 개체의 이름입니다. 트리 옵션 영역에서 **정렬 기준**을 클릭하고 적절한 정렬 옵션을 선택하여 개체가 탐색 트리에 표시되는 순서를 변경할 수 있습니다.

다음 정렬 옵션 중 하나를 선택합니다.

- **표시 이름(a-z)**는 표시 이름에 따라 트리에서 개체를 알파벳 순으로 정렬합니다.
- **표시 이름(z-a)**는 표시 이름에 따라 트리에서 개체를 알파벳 역순으로 정렬합니다.

필터 옵션

탐색 창 의 아래쪽에 있는 **필터** 영역을 클릭하고 적절한 필터 옵션을 선택하여 탐색 트리에 표시되는 항목을 필터링할 수 있습니다.

다음 필터 옵션 중 하나를 선택합니다.

- **필터 없음(모든 항목)**. 이전에 정의된 모든 필터를 제거합니다.
- **한 항목**. 탐색 트리 위에 한 항목을 선택할 수 있는 드롭다운 목록을 표시합니다.
예를 들어, 스키마 관리자에서 테이블 유형이나 테이블을 선택할 수 있습니다.
테이블 유형을 선택한 경우 아래쪽 화살표를 클릭하여 테이블 유형 목록을 표시하고 필터에 사용할 테이블 유형을 선택합니다.
- **일부 항목**. 하나 이상의 항목을 선택할 수 있습니다.

항목 필터링

한 항목 필터 옵션 또는 **일부 항목** 필터 옵션을 선택한 경우 필터링할 항목을 선택할 수 있습니다.

예를 들어, 스키마 관리자에서 테이블 유형이나 테이블 이름을 기반으로 테이블을 선택할 수 있습니다. **일부 항목**을 선택하면 Hub 콘솔의 탐색 트리 위에 **항목 필터 정의** 단추가 표시됩니다.

1. **항목 필터 정의** 단추를 클릭합니다.
2. 필터에 포함할 항목을 선택한 다음 **확인**을 클릭합니다.

항목 보기 변경

Hub 콘솔의 일부 도구는 탐색 트리 아래에 **보기** 또는 **보기 기준** 영역이 표시됩니다.

- 스키마 관리자에서는 탐색 트리 아래의 **보기** 영역을 클릭하고 적절한 명령을 선택하여 공용 Informatica MDM Hub 항목을 표시하거나 숨길 수 있습니다.
예를 들어 모든 시스템 테이블을 볼 수 있습니다.
- 매핑 도구에서는 항목을 매핑, 준비 테이블 또는 랜딩 테이블별로 볼 수 있습니다.
- 패키지 도구에서는 항목을 패키지 또는 테이블별로 볼 수 있습니다.
- 사용자 및 그룹 도구에서는 하위 그룹과 하위 사용자를 표시할 수 있습니다.
- 일괄 처리 뷰어에서는 작업을 테이블, 날짜 또는 프로시저 유형별로 그룹화할 수 있습니다.

항목 검색

필터가 없거나 **일부 항목** 필터가 선택되어 있으면 **탐색** 창에 **찾기** 상자가 포함됩니다. **찾기** 상자를 사용하여 전체 또는 부분 이름으로 항목을 찾습니다. 예를 들어, 스키마 관리자에서 테이블 이름에 "조회"가 포함된 모든 테이블을 찾을 수 있습니다. 찾기 프로세스는 대/소문자를 구분합니다.

1. **찾기** 상자를 클릭합니다.
찾기 창을 엽니다.
2. 찾을 이름 또는 이름의 처음 몇 글자를 입력합니다.
3. **F3 - 찾기** 단추를 클릭합니다.
트리에서 첫 번째로 일치하는 항목이 강조 표시됩니다.
4. 그 다음 일치 항목으로 이동하려면 **F3 - 찾기** 단추를 다시 클릭합니다.
5. 찾기 창을 닫으려면 **찾기** 상자를 클릭합니다.

탐색 트리에서 개체에 대해 명령 실행

탐색 트리의 개체에 대해 명령을 실행하려면 다음 작업 중 하나를 수행합니다.

- 개체 이름을 마우스 오른쪽 단추로 클릭하여 개체에 대해 수행할 명령의 팝업 메뉴를 표시합니다.
- 탐색 트리에서 개체를 선택한 다음 창의 위쪽에 있는 **Hub** 콘솔 메뉴에서 명령을 선택합니다.





예를 들어, 스키마 관리자의 탐색 트리에서 특정 개체 유형을 마우스 오른쪽 단추로 클릭하여 선택한 개체에 대해 사용 가능한 명령의 팝업 메뉴를 표시할 수 있습니다.

명령 단추를 사용하여 개체 추가, 편집 및 제거

명령 단추를 사용하여 **Hub** 콘솔에서 개체를 추가, 편집 및 삭제할 수 있습니다.

명령 단추

Hub 콘솔 창에서 개체를 생성, 수정 또는 제거할 수 있는 액세스 권한이 있고 쓰기 잠금을 획득한 경우 **속성** 창에 다음과 같은 명령 단추가 일부 또는 모두 표시될 수 있습니다. 이 외에 다른 명령 단추도 있습니다.

단추	이름	설명
	추가	새 개체를 추가합니다.
	편집	속성 창에서 선택한 항목의 속성을 편집합니다. 속성이 편집 가능함을 나타냅니다.
	제거	선택한 항목을 제거합니다.
	저장	변경 내용을 저장합니다.

참고: 명령 단추로 수행할 수 있는 작업에 대한 설명을 보려면 단추 위에 마우스를 놓아 도구 설명을 표시합니다.

개체 추가

Hub 콘솔에서 개체를 추가할 수 있습니다.

1. 쓰기 잠금을 획득합니다.
2. **추가** 단추를 클릭합니다.
Hub 콘솔에 **개체 추가** 창이 표시됩니다. 여기서 개체는 추가할 개체 유형의 이름입니다.
3. 개체 속성을 지정합니다.
4. **확인**을 클릭합니다.

개체 속성 편집

Hub 콘솔에서 개체의 속성을 편집할 수 있습니다.

1. 쓰기 잠금을 획득하고 편집할 개체를 선택합니다.
2. 편집할 각 속성에 대해 옆에 있는 **편집** 단추를 클릭하고 새 값을 지정합니다.
3. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

개체 제거

Hub 콘솔에서 개체를 제거할 수 있습니다.

1. 쓰기 잠금을 획득하고 제거할 개체를 선택합니다.
2. **제거** 단추를 클릭합니다.
개체가 다른 개체에 종속되거나 관련된 경우 **영향 분석기** 대화 상자가 표시됩니다. 개체가 다른 개체에 종속되지 않거나 관련되지 않은 경우 **Informatica MDM Hub 콘솔** 대화 상자가 표시됩니다.
3. 적절한 옵션을 선택합니다.

MDM Hub 콘솔 인터페이스 사용자 지정

Hub 콘솔 인터페이스를 사용자 지정할 수 있습니다.

1. **옵션 > 옵션**을 클릭합니다.
옵션 대화 상자가 표시됩니다.
2. 다음 탭에서 원하는 옵션을 지정합니다.
 - **일반** 탭: 마법사 시작 화면을 표시할지 여부 또는 창 크기 및 위치를 저장할지 여부를 지정합니다.
 - **빠른 실행** 탭: 메뉴 아래의 빠른 실행 도구 모음에 아이콘으로 표시할 도구를 지정합니다.

버전 정보 표시

설치된 버전의 Multidomain MDM에 대한 버전 세부 정보를 볼 수 있습니다.

1. Hub 콘솔에서 **도움말 > 정보**를 선택합니다.
Informatica Multidomain MDM 대화 상자가 열립니다.
2. **설치 세부 정보**를 클릭합니다.
설치 세부 정보 대화 상자가 열립니다.
3. **닫기**를 클릭합니다.

4. 닫기를 클릭합니다.





Informatica MDM Hub 작업 영역 및 도구

이 섹션에서는 Informatica MDM Hub 작업 영역 및 도구에 대한 개요를 제공합니다.




구성 작업 영역의 도구






구성 작업 영역에는 Hub 구성에 유용한 도구가 포함되어 있습니다.

다음 테이블에는 구성 작업 영역의 도구와 해당 아이콘이 나열되어 있습니다.




아이콘	도구 이름	설명
	데이터베이스	Register and manage 연산 참조 저장소를 등록 및 관리합니다.
	사용자	사용자를 정의하고 사용자가 액세스할 수 있는 데이터베이스를 지정합니다. 글로벌 암호 정책과 개별 암호 정책을 관리합니다. MDM Hub에서는 사용자에 대해 외부 인증(예: LDAP)을 지원합니다.
	보안 공급자	보안 공급자, 즉 MDM Hub에 액세스하는 사용자에게 대한 보안 서비스(인증, 권한 부여 및 사용자 프로필 서비스)를 제공하는 타사 조직을 구성합니다.
	도구 액세스	사용자가 액세스할 수 있는 Hub 콘솔 도구 및 프로세스를 정의합니다. 기본적으로 새 사용자 계정은 액세스 권한이 명시적으로 할당되기 전까지는 어떤 도구에도 액세스할 수 없습니다.
	메시지 대기열	MDM Hub에 대한 인바운드 및 아웃바운드 메시지 대기열 인터페이스를 정의합니다.
	리포지토리 관리자	연산 참조 저장소 메타데이터의 유효성을 검사하고, 리포지토리 간에 변경 내용을 승격하고, 개체를 리포지토리로 가져오고, 리포지토리를 내보냅니다. 자세한 내용은 <i>Multidomain MDM 리포지토리 관리자 가이드</i> 를 참조하십시오.
	엔터프라이즈 관리자	Hub 서버, 처리 서버, MDM Hub 마스터 데이터베이스 및 연산 참조 저장소에 대한 구성 세부 정보 및 버전 정보를 봅니다.

모델 작업 영역의 도구




아이콘	도구 이름	설명
	스키마	기본 개체, 관계, 기록 및 보안 요구 사항, 준비 및 랜딩 테이블, 유효성 검사 규칙, 일치 조건, 기타 데이터 모델 특성을 정의합니다.
	스키마 뷰어	현재 스키마를 보고 탐색합니다.
	시스템 및 트러스트	Informatica MDM Hub에서 병합할 데이터를 제공할 수 있는 소스 시스템의 이름을 지정합니다. 또한 각 기본 개체 열의 각 소스 시스템과 관련된 트러스트 설정을 정의합니다.

아이콘	도구 이름	설명
	쿼리	패키지에 사용되는 쿼리 그룹 및 쿼리를 정의합니다.
	패키지	패키지(테이블 뷰)를 정의합니다.
	정리 함수	데이터에 대해 수행할 정리 함수를 정의합니다.
	매핑	정리 함수 출력을 준비 테이블의 대상 열에 매핑합니다.
	계층	계층 관리자에서 데이터 관계를 보고 조작하는 데 필요한 구조를 설정합니다.

보안 액세스 관리자 작업 영역의 도구







아이콘	도구 이름	설명
	보안 리소스	MDM Hub의 보안 리소스를 관리합니다. 각 MDM Hub 리소스의 상태(개인, 보안)를 구성하고, 보안 리소스를 구성하는 리소스 그룹을 정의합니다.
	역할	리소스와 리소스 그룹에 대한 역할 및 권한 할당을 정의합니다. 사용자와 사용자 그룹에 역할을 할당합니다.
	사용자 및 그룹	단일 Hub 저장소 내의 사용자와 사용자 그룹을 관리합니다.

데이터 스튜어드 작업 영역의 도구

아이콘	도구 이름	설명
	데이터 관리자	통합된 데이터의 내용 관리, 교차 참조 보기, 데이터 편집, 기록 보기, 통합된 레코드 병합 해제 등을 수행합니다. 참고: 데이터 관리자 도구를 사용하여 암호화된 열의 데이터를 편집하거나 업데이트할 수 없습니다.
	병합 관리자	수동 병합을 위해 대기된 일치 레코드를 검토하고 병합합니다.
	계층 관리자	Hub 저장소의 계층 관계를 정의 및 관리합니다.

데이터 스튜어드 작업 영역의 도구에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

유틸리티 작업 영역의 도구

아이콘	도구 이름	설명
	일괄 그룹	일괄 그룹, 즉 단일 명령으로 실행할 수 있는 개별 일괄 작업(예: 준비, 로드 및 일치 작업) 컬렉션을 구성하고 실행합니다.
	일괄 처리 뷰어	일괄 작업을 실행하여 데이터 정리, 로드, 일치 또는 자동 병합을 수행하고 작업 로그를 봅니다.
	처리 서버	이름, 포트, 서버 유형 및 서버의 온라인/오프라인 상태를 포함한 처리 서버 정보를 봅니다.
	감사 관리자	응용 프로그램 요청 및 메시지 대기열 이벤트의 감사와 디버깅을 구성합니다.
	SIF 관리자	연산 참조 저장소 관련 SIF(서비스 통합 프레임워크) 요청 API를 생성합니다. SIF 관리자는 연산 참조 저장소의 패키지, 원격 패키지, 매핑 및 정리 함수에 대한 SIF 요청 API를 지원하는 코드를 생성하고 배포합니다. 생성된 연산 참조 저장소 관련 API는 Siperian API JAR(siperian-api.jar)를 통해 웹 서비스로 사용할 수 있습니다.
	사용자 개체 레지스트리	연산 참조 저장소에 대해 등록된 사용자 종료, 사용자 지정 Java 정리 함수 및 사용자 지정 GUI 함수를 봅니다.

제 3 장

다국어 데이터 지원 구성

이 장에 포함된 항목:

- [다국어 데이터 지원 구성 개요, 44](#)
- [유니코드 데이터베이스 구성\(Oracle만 해당\), 44](#)
- [미국 외 인구집단에 대한 일치 설정 구성, 45](#)
- [Windows 레지스트리에서 ANSI 코드 페이지 구성, 46](#)
- [유니코드에 대한 정리 설정, 47](#)
- [UTF-8을 사용하는 경우 UNIX에 대한 로캘 권장 사항, 47](#)
- [손상된 데이터 문제 해결, 47](#)
- [Oracle 환경의 언어 설정, 48](#)

다국어 데이터 지원 구성 개요

여러 국가의 데이터가 있는 경우 MDM Hub 구현에서 문자 집합을 구성할 수 있습니다. 사용자가 사용하는 데이터베이스는 선택한 문자 집합을 지원해야 합니다.

다양한 문자 집합을 사용하는 여러 국가의 데이터가 포함된 Oracle 환경이 있는 경우 UTF-8(Unicode Transfer Format) 인코딩을 사용해야 합니다. 또한 NLS_LANG 설정을 구성하여 클라이언트 Oracle 소프트웨어의 로캘 동작을 지정해야 합니다.

유니코드 데이터베이스 구성(Oracle만 해당)

MDM Hub 구현에서 Oracle 데이터베이스를 사용하는 경우 사용하려는 문자 집합을 구성해야 합니다. 구현에서 다른 문자 집합을 사용하는 여러 국가의 데이터와 같이 혼합 로캘 정보를 사용하는 경우 UTF-8(Unicode Transfer Format) 인코딩을 사용하도록 MDM Hub 및 데이터베이스를 구성해야 합니다. 하지만 데이터베이스에 단일 로캘의 데이터가 포함되어 있는 경우 UTF-8 데이터베이스는 필요하지 않을 수 있습니다.

1. UTF-8 데이터베이스를 생성하고 다음 설정을 선택합니다.

- 데이터베이스 문자 집합: AL32UTF8
- 국가별 문자 집합: AL16UTF16

참고: Oracle에서는 Oracle 10g의 데이터베이스 문자 집합으로 AL32UTF8을 사용하도록 권장합니다. 이전 Oracle 릴리스에 대한 내용은 Oracle 문서를 참조하십시오.

2. 서버 및 클라이언트 모두에서 데이터베이스 문자 집합과 일치하도록 **NLS_LANG**를 설정합니다.
예를 들어 미국에 있는 경우 **NLS_LANG**를 아래 값으로 설정합니다.

AMERICAN_AMERICA.AL32UTF8

하지만 일본어 구현인 경우에는 **NLS_LANG**을 아래 값으로 설정합니다.

JAPANESE_JAPAN.AL32UTF8

3. 클라이언트의 지역 글꼴 설정 구성을 확인합니다.
예를 들어 중국 데이터인 경우 중국어 글꼴을 설치합니다.
4. 다중 바이트 문자 집합을 사용하는 경우 유니코드 값을 지원하려면 **C_REPOS_DB_RELEASE** 테이블에서 아래 설정을 변경합니다.

column_length_in_bytes_ind = 0

미국 외 인구집단에 대한 일치 설정 구성

MDM Hub 구현에서 미국 외 인구집단을 사용하는 경우 인구집단 집합을 구성하고 일치 처리에 대한 인코딩을 활성화합니다. 인구집단 집합은 특정 인구집단에 대해 일반적인 이름, 주소 및 기타 식별 정보에 대한 인텔리전스를 캡슐화합니다. 인구집단 집합에 대한 자세한 내용은 [“인구집단 집합” 페이지 276](#)을 참조하십시오.

MDM Hub에는 **demo.ysp** 파일 및 **<population>.ysp** 파일이 포함되어 있습니다. **demo.ysp** 파일에는 인구집단이 포함되어 있으며, 이 파일은 데모 전용이므로 실제 일치 규칙에 사용해서는 안 됩니다. MDM Hub 구현에서 라이선스를 구입한 **<population>.ysp** 인구집단 파일을 사용합니다. 인구집단 파일이 없는 경우에는 Informatica 글로벌 고객 지원 센터에 문의하여 현재 구현에 적절한 인구집단 파일을 받으십시오.

아래 고려 사항을 기반으로 인구집단 집합을 결정합니다.

- 데이터가 모두 한 국가의 데이터이고 Informatica에서 해당 국가의 인구집단 집합을 제공하면 해당 인구집단을 사용합니다.
- 데이터가 대부분 한 국가의 데이터이고 일부만 여러 국가의 혼합된 데이터인 경우에는 대다수에 해당하는 인구집단을 사용합니다.
- 많은 양의 데이터가 여러 국가의 혼합된 데이터인 경우에는 이러한 서로 다른 데이터 집합 간에 일치시키는 것이 의미가 있을지 고려해 보십시오. 의미가 있다면 국제 인구집단을 사용합니다.

일치 채우기를 활성화하는 것에 대한 자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

일치 처리에 대한 인코딩 구성

일치 중 UTF-8 문자 처리를 활성화하려면 처리 서버 설정을 편집하십시오.

1. 텍스트 편집기를 사용하여 다음 파일을 엽니다. <MDM Hub 설치 디렉터리>\hub\cleanse\resources\cmxcleanse.properties
2. 다음 설정을 수동으로 추가합니다.

cmx.server.match.server_encoding = 1

단일 기본 개체 내에서 여러 채우기 사용

MDM Hub의 단일 기본 개체 내에 여러 개의 채우기를 사용할 수 있습니다.

기본 개체의 데이터가 다른 채우기에서 제공된 경우 유용합니다. 예를 들어, 기록의 70%는 미국에서 제공되고 30%는 중국에서 제공됩니다. 채우기는 레코드별로 다를 수 있습니다.

기본 개체 내에서 여러 채우기를 사용하려면 다음 단계를 수행하십시오.

1. Informatica 글로벌 고객 지원 센터에 문의하여 구현에 적합한 <population>.ysp 파일을 채우기 활성화 지침과 함께 구하십시오.
2. C_REPOS_SSA_POPULATION 메타데이터 테이블에서 사용할 각 채우기를 활성화합니다.
3. 적용 가능한 채우기 파일을 다음 위치에 복사합니다.
UNIX의 경우. <MDM Hub 설치 디렉터리>/hub/cleanse/
Windows의 경우. <MDM Hub 설치 디렉터리>\cleanse\resources\match
4. 응용 프로그램 서버를 다시 시작합니다.
5. 스키마 관리자에서 각 레코드에 사용할 채우기가 들어 있는 기본 개체에 SIP_POP라는 VARCHAR 열을 추가합니다.
참고: VARCHAR 열의 너비는 사용 중인 가장 긴 채우기 이름의 길이와 맞아야 합니다. 대부분의 구현에서 너비 30이면 충분합니다.
6. 일치 열을 이름이 SIP_POP인 정확히 일치 열로 구성합니다.
7. 기본이 아닌 채우기를 사용하는 기본 개체의 각 레코드에 대해 SIP_POP 열에 대신 사용할 채우기 이름을 입력합니다.
다음 방법 중 하나로 SIP_POP 열의 값을 지정할 수 있습니다.
 - 랜딩 테이블에 UTF-8 데이터를 추가합니다.
 - 준비 프로세스 중 값을 계산하는 정리 함수를 사용합니다.
 - 외부 응용 프로그램에서 SIF 요청을 호출합니다.
 - 데이터 관리자 도구를 통해 열 값을 수동으로 편집합니다.**참고:** SIP_POP 열의 데이터는 대/소문자를 구분하지 않지만 MDM Hub는 기본 채우기를 사용하여 NULL 값이나 빈 문자열과 같은 잘못된 값을 처리합니다.
8. 기본 개체에서 일치 토큰 생성 프로세스를 실행하여 일치 키 테이블을 업데이트합니다.
9. 기본 개체에서 일치 프로세스를 실행 합니다.
참고: 일치 프로세스에서는 동일한 채우기를 공유하는 레코드만 비교합니다. 예를 들어, 일치 프로세스는 중국 레코드를 중국 레코드와 비교하고 미국 레코드를 미국 레코드와 비교합니다.

Windows 레지스트리에서 ANSI 코드 페이지 구성

Windows에서 Windows 레지스트리 편집기를 통해 ANSI 코드 페이지를 구성합니다.

1. 명령 프롬프트에 regedit를 입력하고 **확인**을 클릭합니다.
2. 다음 레지스트리 항목으로 이동합니다.
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Nls\CodePage\ACP
3. ANSI 코드 페이지를 변경하려면 Windows 제어판에서 로캘 및 언어 설정을 구성합니다. 지침은 실행 중인 Windows의 버전에 따라 다릅니다. 자세한 지침은 Microsoft Windows 설명서를 참조하십시오.

유니코드에 대한 정리 설정

Informatica Address Verification 정리 라이브러리를 사용하는 경우 Informatica Address Verification에 대한 적절한 데이터베이스와 잠금 해제 코드가 있어야 합니다. 구현에 필요한 모든 국가에 해당하는 Informatica Address Verification 데이터베이스를 확보해야 합니다. 자세한 내용은 Informatica 글로벌 고객 지원 센터에 문의하십시오.

Trillium을 사용하는 경우에는 올바른 템플릿을 사용하여 프로젝트를 생성해야 합니다. 지원되는 국가를 확인하려면 Trillium 설치 설명서를 참조하십시오. Trillium에서 국가별 프로젝트를 직접 가져올 수 있습니다.

UTF-8을 사용하는 경우 UNIX에 대한 로캘 권장 사항

많은 UNIX 시스템에서 호환되지 않는 문자 인코딩을 사용하여 현지 문자를 이진 데이터로 나타냅니다. 즉, 한국어 시스템의 텍스트 문자열은 중국어 시스템에서 볼 수 없습니다. 그러나 모든 언어에 대해 UTF-8 인코딩을 사용하도록 UNIX 시스템을 구성할 수 있습니다. UTF-8 텍스트 인코딩은 여러 언어를 지원하므로 한 언어가 다른 언어에 지장을 주지 않습니다.

UTF-8을 사용하도록 시스템 로캘 설정을 구성하려면 다음 단계를 수행하십시오.

1. 다음 명령을 실행합니다.

```
locale -a
```

2. 접미사 .utf8를 사용하여 원하는 언어의 로캘을 찾을 수 있는지 확인합니다.

```
localedef -f UTF-8 -i en_US en_US.utf8
```

3. UTF-8을 허용하는 로캘이 있는 경우 UNIX 시스템이 해당 로캘을 사용하도록 지시합니다.

```
export LC_ALL="en_US.utf8"
export LANG="en_US.utf8"
export LANGUAGE="en_US.utf8"
```

손상된 데이터 문제 해결

SQL*Loader를 사용하여 국제 문자 집합으로 표시된 데이터를 로드할 때 로드된 데이터가 손상된 경우 cmxcleanse.properties 파일의 속성을 설정하여 이 문제를 해결할 수 있습니다.

1. 다음 디렉터리로 이동합니다.

```
<MDM 설치 디렉터리>/hub/cleanse/resources
```

2. 편집기에서 cmxcleanse.properties 파일을 엽니다.

3. 파일의 끝에 다음 속성을 입력하고 로드할 데이터에 일치하는 국제 문제 집합에 대한 유니코드 식별자를 지정합니다.

```
cmx.server.stage.sqlldr.charset=AL32UTF8
```

4. 파일을 저장합니다.

5. 준비 프로세스를 실행합니다.

준비 프로세스가 지정된 문자 집합이 포함된 SQL*Loader에 대한 제어 파일을 생성합니다.

6. 데이터를 다시 로드할 때 이 제어 파일을 사용합니다.

Oracle 환경의 언어 설정

클라이언트 Oracle 소프트웨어의 로캘 동작을 지정하려면 클라이언트의 언어, 지역 및 문자 집합을 지정하는 NLS_LANG 설정을 지정해야 합니다. NLS_LANG 설정을 구성하는 방법은 운영 체제에 따라 다릅니다.

Windows

Windows 레지스트리 편집기를 통해 또는 환경 변수로 NLS_LANG 설정을 구성할 수 있습니다.

UNIX

LANG 환경 변수를 설정하고 필요한 로캘을 설치해야 합니다.

NLS_LANG 구문

NLS_LANG 설정에는 다음 형식이 사용됩니다.

NLS_LANG = <LANGUAGE>_<TERRITORY>.<CHARACTERSET>

다음 표에는 매개 변수가 설명되어 있습니다.

매개 변수	설명
LANGUAGE	Oracle 메시지와 월 및 요일 이름에 사용되는 언어를 지정합니다.
TERRITORY	지역, 주 및 날짜 번호 계산 규칙, 통화 및 숫자 형식을 지정합니다.
CHARACTERSET	클라이언트 응용 프로그램이 사용하는 문자 집합을 제어합니다. 또는 이 매개 변수는 Windows 코드 페이지와 일치하거나 유니코드 응용 프로그램의 경우 UTF-8로 설정할 수 있습니다.

참고: NLS_LANG 설정으로 정의된 문자 집합에 따라 클라이언트의 문자 집합이 변경되지는 않습니다. 대신 Oracle 데이터베이스는 정의된 문자 집합으로 데이터를 변환합니다. NLS_LANG 설정은 서버에서 문자 집합을 상속받지 않습니다.

Windows 레지스트리에서 NLS_LANG 구성

Windows 시스템에서 각 Oracle 홈에 대해 NLS_LANG 레지스트리 하위 키를 설정했는지 확인합니다.

이 하위 키는 Windows 레지스트리 편집기를 통해 수정할 수 있습니다.

- 명령 프롬프트에 regedit를 입력하고 **확인**을 클릭합니다.
- 다음 레지스트리 항목으로 이동합니다.

Oracle 10g의 경우:

HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_<Oracle_Home_name>

- NLS_LANG 하위 키를 편집합니다.

NLS_LANG을 환경 변수로 구성(Windows)

Informatica에서 권장하는 방법은 아니지만 시스템 속성에서 NLS_LANG을 시스템 또는 사용자 환경 변수로 설정할 수 있습니다. 모든 Oracle 홈은 구성된 설정을 사용합니다.

시스템 또는 사용자 환경 변수를 확인 및 수정하려면 다음 단계를 수행합니다.

- 내 컴퓨터**를 마우스 오른쪽 단추로 클릭한 다음 **속성**을 선택합니다.
- 고급** 탭에서 **환경 변수**를 클릭합니다.

사용자 변수 목록에는 현재 로그인한 Windows 사용자에게 대한 설정이 포함되어 있습니다.

시스템 변수 목록에는 모든 사용자에게 대한 시스템 수준의 변수가 포함되어 있습니다.

3. 설정을 필요한 대로 변경합니다.

이러한 환경 변수는 Windows 레지스트리에 지정된 매개 변수보다 우선하므로 꼭 필요한 경우가 아니면 이 위치에 Oracle 매개 변수를 설정하지 마십시오. 특히 ORACLE_HOME 매개 변수는 UNIX에는 설정되지만 Windows에는 설정되지 않습니다.

LANG 환경 변수 및 로캘 설정(UNIX)

MDM Hub 전체에서 UTF-8 데이터의 일관된 처리를 보장하려면 응용 프로그램 서버를 호스팅하는 UNIX 서버에 대한 올바른 로캘 및 LANG 환경 변수를 설정하십시오.

데이터베이스 및 서버를 포함한 MDM Hub의 모든 시스템에서 다음 환경 변수를 설정합니다.

- `export LC_ALL=en_US.UTF-8`
- `export LANG=en_US.UTF-8`
- `export LANGUAGE=en_US.UTF-8`

Oracle 환경에서는 다음 환경 변수를 설정합니다.

- `export NLS_LANG=AMERICAN_AMERICA.AL32UTF8`

예를 들어, 미국의 기본 LANG 환경 변수는 `export LANG = en_US`입니다.

따라서 UTF-8을 사용할 때 다음 명령을 사용하여 LANG 환경 변수를 구성합니다.

```
export LANG=en_US.UTF-8
```

시스템에 여러 응용 프로그램이 설치되어 있고 올바른 로캘도 모두 설치된 경우, 응용 프로그램을 시작하는 프로필에 대해 올바른 환경 변수를 설정할 수 있습니다. 동일한 사용자 프로필이 여러 응용 프로그램을 시작하는 경우 응용 프로그램의 시작 스크립트에서 로컬로 환경 변수를 설정할 수 있습니다. 이렇게 하면 환경 변수가 응용 프로그램 프로세스 컨텍스트 내에서만 로컬로 적용됩니다.

일반적으로 모든 LANG 환경 변수는 동일한 설정이지만 다른 설정을 사용할 수도 있습니다. 예를 들어 인터페이스 언어는 영어지만 정렬해야 하는 데이터가 프랑스어인 경우에는 LC_MESSAGES를 en_US로, LC_COLLATE를 fr_FR로 설정합니다. 다른 LANG 설정을 사용하지 않아도 되면 LC_ALL 또는 LANG을 설정합니다.

응용 프로그램에서는 사용할 로캘을 결정하기 위해 다음과 같은 규칙을 사용합니다.

- LC_ALL 환경 변수가 정의되어 있고 null이 아닌 경우 응용 프로그램은 LC_ALL의 값을 사용합니다.
- 적절한 구성 요소 관련 환경 변수(예: LC_COLLATE)가 설정되어 있고 값이 null이 아닌 경우 응용 프로그램에서 이 환경 변수 값이 사용됩니다.
- LANG 환경 변수가 정의되어 있고 null이 아닌 경우 응용 프로그램은 LANG의 값을 사용합니다.
- LANG 환경 변수가 설정되어 있지 않거나 값이 null인 경우 응용 프로그램은 구현-종속 기본 로캘을 사용합니다.

참고: 시나리오마다 다른 로캘을 사용해야 하는 경우에는 LC_ALL을 설정하지 마십시오.

파트 II: Hub 콘솔 도구 구성

이 파트에 포함된 장:

- [Hub 콘솔 도구에 대한 액세스 권한 구성, 51](#)
- [Hub 콘솔 도구에서 사용자 지정 단추 구현, 53](#)

제 4 장

Hub 콘솔 도구에 대한 액세스 권한 구성

이 장에 포함된 항목:

- [Hub 콘솔 도구에 대한 액세스 권한 구성 개요, 51](#)
- [사용자 구성, 51](#)
- [도구 및 프로세스에 대한 사용자 액세스 권한, 52](#)

Hub 콘솔 도구에 대한 액세스 권한 구성 개요

MDM Hub 사용자가 Hub 콘솔 도구에 액세스하는 방법을 제어할 수 있습니다. 예를 들어 데이터 스튜어드는 일반적으로 데이터 관리자 및 병합 관리자 도구에만 액세스할 수 있습니다.

구성 작업 영역의 "도구 액세스" 도구를 사용하여 Hub 콘솔 도구에 대한 액세스 권한을 구성할 수 있습니다. "도구 액세스" 도구를 사용하려면 마스터 데이터베이스에 연결되어 있어야 합니다.

도구 액세스 도구는 관리자로 구성되지 않은 MDM Hub 사용자에게만 적용됩니다. 사용자 구성에 대한 자세한 내용은 *Multidomain MDM 보안 가이드*를 참조하십시오.

사용자 구성

MDM Hub에서 사용자를 작성, 편집 및 삭제할 수 있습니다.

구성 작업 영역의 사용자 도구를 사용하여 MDM Hub 사용자의 사용자 계정을 구성하고, 암호를 변경하고, 외부 인증을 활성화할 수 있습니다. 다른 방법을 사용하여 MDM Hub로 사용자 정보를 직접 가져올 경우 Hub 저장소에 문제가 발생할 수 있으므로 다른 방법을 사용하지 않는 것이 좋습니다.

또한 사용자 도구를 사용하여 외부 응용 프로그램 사용자에게 대한 사용자 계정을 구성할 수도 있습니다. 외부 응용 프로그램 사용자는 타사 응용 프로그램을 통해 MDM Hub 데이터에 간접적으로 액세스하는 MDM Hub 사용자입니다.

사용자 구성에 대한 자세한 내용은 *Multidomain MDM 보안 가이드*를 참조하십시오.

도구 및 프로세스에 대한 사용자 액세스 권한

구성 작업 영역의 도구 액세스 도구를 사용하여 Hub 콘솔 도구에 대한 액세스 권한을 구성할 수 있습니다.

도구 액세스 도구 시작

도구 액세스 도구를 시작하려면

1. Hub 콘솔에서 마스터 데이터베이스에 연결합니다(아직 연결하지 않은 경우).
2. 구성 작업 영역을 확장하고 **도구 액세스**를 클릭합니다.
Hub 콘솔에 도구 액세스 도구가 표시됩니다.

도구 및 프로세스에 대한 사용자 액세스 권한 부여

특정 MDM Hub 사용자에게 Hub 콘솔 도구 및 프로세스에 대한 액세스 권한을 부여하려면

1. 쓰기 잠금을 획득합니다.
2. 도구 액세스 도구에서 사용자 목록을 스크롤하여 구성하려는 사용자를 선택합니다.
3. 다음 작업 중 하나를 수행합니다.
 - **사용 가능한 프로세스** 목록에서 액세스 권한을 부여할 프로세스를 선택합니다.
 - **사용 가능한 작업 영역** 목록에서 액세스 권한을 부여할 도구가 포함된 작업 영역을 선택합니다.
4. **도구 추가** 또는 **프로세스 추가**를 클릭합니다.

선택한 도구 또는 프로세스가 **액세스 가능한 도구 및 프로세스** 목록에 추가됩니다. 프로세스에 액세스 권한을 부여할 경우 MDM Hub가 프로세스에서 사용하는 모든 도구에 액세스 권한을 부여합니다. 도구에 액세스 권한을 부여할 경우 MDM Hub가 도구를 사용하는 모든 프로세스에 액세스 권한을 부여합니다.

사용자는 액세스 권한이 있는 모든 연산 참조 저장소에서 이러한 프로세스와 도구에 액세스할 수 있습니다. 한 연산 참조 저장소와 다른 연산 참조 저장소에서 서로 다른 도구에 액세스하도록 사용자 권한을 부여할 수는 없습니다.

참고: 작업 영역의 모든 도구에 액세스 권한을 부여하지 않을 경우 **액세스 가능한 도구 및 프로세스** 목록에서 연결된 작업 영역을 확장하고 선택된 도구에 대한 액세스 권한을 취소합니다.

도구 및 프로세스에 대한 사용자 액세스 권한 취소

특정 MDM Hub 사용자에게 대해 Hub 콘솔 도구 및 프로세스에 대한 사용자 액세스 권한을 취소하려면

1. 쓰기 잠금을 획득합니다.
2. 도구 액세스 도구에서 사용자 목록을 스크롤하여 구성하려는 사용자를 선택합니다.
3. [**액세스 가능한 도구 및 프로세스**] 목록을 스크롤하여 액세스를 취소할 프로세스, 작업 영역 또는 도구를 선택합니다.
도구를 선택하려면 연결된 작업 영역을 확장합니다.
4. **도구 제거** 또는 **프로세스 제거**를 클릭합니다.
액세스 권한 제거를 확인하는 메시지가 도구 액세스 도구에 표시됩니다.
5. **예**를 클릭합니다.

도구 액세스 도구가 [**사용 가능한 도구 및 프로세스**] 목록에서 선택한 항목을 제거합니다. 프로세스에 액세스 권한을 취소할 경우 MDM Hub가 프로세스에서 사용하는 모든 도구에 액세스 권한을 취소합니다. 도구에 액세스 권한을 취소할 경우 MDM Hub가 도구를 사용하는 모든 프로세스에 액세스 권한을 취소합니다.

제 5 장

Hub 콘솔 도구에서 사용자 지정 단추 구현

이 장에 포함된 항목:

- [개요, 53](#)
- [Hub 콘솔의 사용자 지정 단추 정보, 53](#)
- [사용자 지정 단추 추가, 54](#)
- [사용자 지정 단추 모양 제어, 57](#)
- [사용자 지정 단추 배포, 57](#)

개요

이 장에서는 Informatica MDM Hub 구현에서 요청 시 외부 서비스를 호출할 수 있도록 Hub 콘솔의 도구에 사용자 지정 단추를 추가하는 방법에 대해 설명합니다.

Hub 콘솔의 사용자 지정 단추 정보

Informatica MDM Hub 구현에서는 Hub 콘솔 사용자에게 Informatica MDM Hub 구현을 확장하는 데 사용할 수 있는 사용자 지정 단추를 제공할 수 있습니다.

사용자는 사용자 지정 단추를 사용하여 특수화된 데이터 서비스에 필요할 때 실시간으로 액세스할 수 있습니다. 사용자 지정 단추는 병합 관리자와 계층 관리자에 추가할 수 있습니다.

사용자는 사용자 지정 단추를 사용하여 데이터 검색 또는 결과 계산 같은 특정 외부 서비스를 호출하고, 워크플로우 실행 같은 특수화된 작업과 기타 태스크를 수행할 수 있습니다. 사용자 지정 단추는 엔터프라이즈 응용 프로그램(예: CRM 또는 ERP 응용 프로그램), 외부 서비스 공급자(예: 외환 계산기, 금융시장 지수 게시자 또는 정부 기관) 및 Informatica MDM Hub를 비롯한 다양한 서비스 공급자가 데이터 서비스에 액세스하도록 디자인할 수 있습니다. 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

예를 들어 공급업체에서 웹 서비스로 제공되었으며 병합 관리자 화면에서 현재 선택된 고객 레코드의 데이터를 정리하는 특수화된 정리 함수를 호출하기 위해 사용자 지정 단추를 추가할 수 있습니다. 사용자가 단추를 클릭하면 기본 코드에 의해 선택된 레코드에서 관련 데이터가 캡처되고 인증 정보를 포함한 요청이 웹 서비스에 필요한 형식으로 생성되어 처리를 위해 웹 서비스에 제출됩니다. 결과가 반환되면 Hub에서는 해당 정보를 별도의

Swing 대화 상자(생성했거나 클라이언트 사용자 지정 함수로 구현한 경우)에 Informatica MDM Hub의 고객 rowid_object와 함께 표시합니다.

사용자 지정 단추는 기본적으로 설치되지 않으며 Informatica MDM Hub 구현에 따라 필요하지 않을 수도 있습니다. 각 사용자 지정 단추에 대해 Java 인터페이스를 구현하고 해당 구현을 JAR 파일에 패키징한 다음 명령줄 유틸리티를 실행하여 배포해야 합니다. Hub 콘솔에 표시되는 사용자 지정 단추의 모양을 제어하려면 텍스트나 아이콘 그래픽을 JPG, PNG 또는 GIF 같은 Swing 호환 그래픽 형식으로 제공하면 됩니다.

사용자가 사용자 지정 단추를 클릭하면 수행되는 작업

사용자가 Hub 콘솔에서 고객 레코드를 선택한 다음 사용자 지정 단추를 클릭하면 Hub 콘솔은 요청을 실행하여 해당 내용 및 컨텍스트를 Java 외부(사용자 지정) 서비스에 전달합니다. 데이터 유형의 예로는 기본 개체의 레코드 키 및 기타 데이터, 패키지 정보 등이 포함됩니다. 실행은 비동기적으로 이루어지므로 요청이 처리되는 동안 사용자는 Hub 콘솔에서 작업을 계속할 수 있습니다.

사용자 지정 코드에서는 서비스 응답을 적절하게 처리할 수 있습니다. 예를 들어 결과를 로깅하거나, 별도의 Swing 대화 상자에서 사용자에게 데이터를 표시하거나(사용자 지정 코딩되고 사용자 지정 함수가 클라이언트 측에서 실행되는 경우), 사용자가 결과를 데이터 항목 필드에 복사하여 붙여넣을 수 있도록 하거나, 데이터를 올바른 비즈니스 개체로 되돌리는 실시간 PUT 문을 실행하는 등의 처리가 수행될 수 있습니다.

Hub 콘솔에 사용자 지정 단추가 나타나는 방식

이 섹션에서는 구현된 사용자 지정 단추가 Hub 콘솔의 병합 관리자 및 계층 관리자 도구에 나타나는 방식을 보여 줍니다.

병합 관리자의 사용자 지정 단추

사용자 지정 단추는 병합 관리자에서 일반적인 병합 관리자 단추가 표시되는 동일한 위치인 위쪽 패널 오른쪽에 표시됩니다.

계층 관리자의 사용자 지정 단추

계층 관리자 화면의 상단 패널 맨 위에는 다른 계층 관리자 단추와 동일한 위치에 사용자 지정 단추가 표시됩니다.

사용자 지정 단추 추가

Informatica MDM Hub 구현에서 Hub 콘솔에 사용자 지정 단추를 추가하려면 다음 태스크를 완료합니다.

1. 요청 및 응답 메시지의 형식과 매개 변수 등 호출할 외부 서비스에 대한 세부 정보를 확인합니다.
2. 사용자 지정 단추로 실행할 비즈니스 논리를 작성하여 패키징합니다.
3. Hub 콘솔의 해당하는 도구에 표시되도록 패키지를 배포합니다.

외부 서비스 단추가 Hub 콘솔에 표시되면 이 단추를 클릭하여 해당 서비스를 호출할 수 있습니다.

사용자 지정 함수 작성

외부 서비스 호출을 만들려면 사용자가 Hub 콘솔에서 사용자 지정 단추를 클릭할 때 응용 프로그램 논리를 실행하는 사용자 지정 함수를 작성합니다. 응용 프로그램 논리는 다음 Java 인터페이스를 구현합니다.

```
com.siperian.mrm.customfunctions.api.CustomFunction
```

이 인터페이스에 대한 자세한 내용은 Informatica MDM Hub 분포 시 함께 제공되는 Javadoc를 참조하십시오.

서버 기반 및 클라이언트 기반 사용자 지정 함수

다음에 대해 응용 프로그램 논리가 실행됩니다.

환경	설명
클라이언트	UI 기반 사용자 지정 함수 - 응답 정보를 표시하는 별도의 대화 상자 등 사용자 인터페이스에 요소를 표시할 경우 권장됩니다.
서버	서버 기반 사용자 지정 함수 - 네트워크나 성능상 이유로 서버에서 외부 서비스를 호출하는 것이 더 나은 경우 권장됩니다.

사용자 지정 함수 예

이 섹션에서는 `com.siperian.mrm.customfunctions.api.CustomFunction` 인터페이스를 구현하는 두 가지 사용자 지정 함수에 대한 Java 코드 예를 제공합니다. 이 코드는 (표준 오류) 정보를 서버 로그 또는 Hub 콘솔 로그에 출력합니다.

클라이언트 기반 사용자 지정 함수 예

다음 샘플 코드의 클라이언트 함수 클래스 이름은 `com.siperian.mrm.customfunctions.test.TestFunction`입니다.

```
package com.siperian.mrm.customfunctions.test;
import java.awt.Frame;
import java.util.Properties;

import javax.swing.Icon;
import com.siperian.mrm.customfunctions.api.CustomFunction;

public class TestFunctionClient implements CustomFunction {

    public void executeClient(Properties properties, Frame frame, String username,
String password, String orsId, String baseObjectRowid, String baseObjectUid, String
packageRowid, String packageUid, String[] recordIds) {
        System.err.println("Called custom test function on the client with the following
parameters:");
        System.err.println("Username/Password: '" + username + "/" + password + "'");
        System.err.println("ORS Database ID: '" + orsId + "'");
        System.err.println("Base Object Rowid: '" + baseObjectRowid + "'");
        System.err.println("Base Object UID: '" + baseObjectUid + "'");
        System.err.println("Package Rowid: '" + packageRowid + "'");
        System.err.println("Package UID: '" + packageUid + "'");
        System.err.println("Record Ids: ");
        for(int i = 0; i < recordIds.length; i++) {
            System.err.println(" " + recordIds[i] + "");
        }
        System.err.println("Properties: " + properties.toString());
    }

    public void executeServer(Properties properties, String username, String password,
String orsId, String baseObjectRowid, String baseObjectUid, String packageRowid,
String packageUid, String[] recordIds) {
        System.err.println("This method will never be called because getExecutionType()
returns CLIENT_FUNCTION");
    }

    public String getActionText() { return "Test Client"; }
    public int getExecutionType() { return CLIENT_FUNCTION; }
    public Icon getGuiIcon() { return null; }
}
```

서버 기반 함수 예제

다음 코드의 서버 함수 클래스 이름은 `com.siperian.mrm.customfunctions.test.TestFunctionClient`입니다.

```
package com.siperian.mrm.customfunctions.test;
import java.awt.Frame;
import java.util.Properties;
import javax.swing.Icon;

import com.siperian.mrm.customfunctions.api.CustomFunction;

/**
 * This is a sample custom function that is executed on the Server.
 * To deploy this function, put it in a jar file and upload the jar file
 * to the DB using DeployCustomFunction.
 */
public class TestFunction implements CustomFunction {
    public String getActionText() {
        return "Test Server";
    }
    public Icon getGuiIcon() {
        return null;
    }
    public void executeClient(Properties properties, Frame frame, String username,
String password, String orsId, String baseObjectRowid, String baseObjectUid,
String packageRowid, String packageUid, String[] recordIds) {
        System.err.println("This method will never be called because getExecutionType()
returns SERVER_FUNCTION");
    }
    public void executeServer(Properties properties, String username, String password,
String orsId, String baseObjectRowid, String baseObjectUid, String packageRowid,
String packageUid, String[] recordIds) {
        System.err.println("Called custom test function on the server with the following
parameters:");
        System.err.println("Username/Password: '" + username + "'/'" + password + "'");
        System.err.println(" ORS Database ID: '" + orsId + "'");
        System.err.println("Base Object Rowid: '" + baseObjectRowid + "'");
        System.err.println(" Base Object UID: '" + baseObjectUid + "'");
        System.err.println("    Package Rowid: '" + packageRowid + "'");
        System.err.println("    Package UID: '" + packageUid + "'");
        System.err.println("    Record Ids: ");
        for(int i = 0; i < recordIds.length; i++) {
            System.err.println("        '" + recordIds[i] + "'");
        }
        System.err.println("    Properties: " + properties.toString());
    }
    public int getExecutionType() {
        return SERVER_FUNCTION;
    }
}
```

사용자 지정 단추 모양 제어

Hub 콘솔에서 사용자 지정 단추의 모양을 제어하려면

`com.siperian.mrm.customfunctions.api.CustomFunction` 인터페이스에서 다음 메서드 중 하나를 구현합니다.

메서드	설명
<code>getActionText</code>	단추 레이블의 텍스트를 지정합니다. 사용자 지정 단추에 기본 시각적 모양을 사용합니다.
<code>getGuilcon</code>	아이콘 그래픽을 Swing 호환 그래픽 형식(예: JPG, PNG 또는 GIF)으로 지정합니다. 이 이미지 파일은 해당 사용자 지정 함수의 JAR 파일과 함께 번들로 제공될 수 있습니다.

사용자 지정 단추는 Hub 콘솔에서 이름을 기준으로 알파벳 순서로 표시됩니다.

사용자 지정 단추 배포

Hub 콘솔에서 사용자 지정 단추를 표시하려면 먼저 명령줄에서 `DeployCustomFunction` 유틸리티를 사용하여 명시적으로 추가해야 합니다.

사용자 지정 단추를 배포하려면

1. 명령 프롬프트를 엽니다.
2. 사용자가 생성한 JAR 파일을 로드하고 등록하는 `DeployCustomFunction` 유틸리티를 실행합니다.

참고: `DeployCustomFunction`을 실행하려면 두 가지 JAR 파일, 즉 `siperian-server.jar` 및 JDBC 드라이버(이 경우 `ojdbc14.jar`)가 이 두 파일을 가리키는 디렉터리 경로와 함께 클래스 경로에 있어야 합니다.

명령 프롬프트에서 다음 명령을 지정합니다.

```
java -cp siperian-server.jar; ojdbc14.jar  
com.siperian.mrm.customfunctions.dbadapters.DeployCustomFunction
```

Informatica MDM Hub 구현 환경에 대해 구성된 설정을 기반으로 프롬프트에 응답합니다. 예를 들면 다음과 같습니다.

```
Database Type:oracle  
Host:localhost  
Port(1521):  
Service:orcl  
Username:ds_uil  
Password:!!cmx!!  
(L)ist, (A)dd, (U)pdate, (C)hange Type, (S)et Properties, (D)elete or (Q)uit:l  
No custom actions  
(L)ist, (A)dd, (U)pdate Jar, (C)hange Type, (S)et Properties, (D)elete or (Q)uit:q
```

3. 각 프롬프트에서 Informatica MDM Hub 구현 환경에 대해 구성된 설정을 기반으로 다음 정보를 지정합니다.
 - 데이터베이스 호스트
 - 포트
 - 서비스
 - 로그인 사용자 이름(스키마 이름)
 - 로그인 암호

4. 프롬프트가 표시되면 데이터베이스 연결 정보, 즉 데이터베이스 호스트, 포트, 서비스, 로그인 사용자 이름 및 암호를 지정합니다.
5. DeployCustomFunction 도구에 다음 옵션으로 구성된 메뉴가 표시됩니다.

레이블	설명
(L)ist	현재 정의된 사용자 지정 단추의 목록을 표시합니다.
(A)dd	새 사용자 지정 단추를 추가합니다. DeployCustomFunction 도구에서 다음을 지정하라는 메시지를 표시합니다. <ul style="list-style-type: none"> - 사용자 지정 단추에 대한 JAR 파일 - com.siperian.mrm.customfunctions.api.CustomFunction 인터페이스를 구현하는 사용자 지정 함수 클래스의 이름 - 사용자 지정 단추의 유형: m - 병합 관리자, d - 데이터 관리자, h - 계층 관리자(하나 이상의 문자를 지정할 수 있음)
(U)pdate	기존 사용자 지정 단추에 대한 JAR 파일을 업데이트합니다. DeployCustomFunction 도구에서 다음을 지정하라는 메시지를 표시합니다. <ul style="list-style-type: none"> - 업데이트할 사용자 지정 단추의 rowID - 사용자 지정 단추에 대한 JAR 파일 - com.siperian.mrm.customfunctions.api.CustomFunction 인터페이스를 구현하는 사용자 지정 함수 클래스의 이름 - 사용자 지정 단추의 유형: m - 병합 관리자, h - 계층 관리자(하나 이상의 문자를 지정할 수 있음)
(C)hange Type	기존 사용자 지정 단추의 유형을 변경합니다. DeployCustomFunction 도구에서 다음을 지정하라는 메시지를 표시합니다. <ul style="list-style-type: none"> - 업데이트할 사용자 지정 단추의 rowID - 사용자 지정 단추의 유형: m - 병합 관리자 및/또는 h - 계층 관리자(하나 이상의 문자를 지정할 수 있음)
(S)et Properties	실행 시 사용자 지정 함수에 필요한 이름/값 쌍(이름=값)을 정의하는 속성 파일을 지정합니다. DeployCustomFunction 도구에서 사용할 속성 파일을 지정하라는 메시지를 표시합니다.
(D)elete	기존 사용자 지정 단추를 삭제합니다. DeployCustomFunction 도구에서 삭제할 사용자 지정 단추의 rowID를 지정하라는 메시지를 표시합니다.
(Q)uit	DeployCustomFunction 도구를 종료합니다.

6. 작업 선택을 마치면 **(Q)uit**를 선택합니다.
7. 방금 추가한 사용자 지정 단추를 표시하려면 브라우저 창을 새로 고칩니다.
8. 사용자 지정 단추를 테스트하여 단추가 올바르게 작동하는지 확인합니다.

파트 III: 데이터 모델 작성

이 파트에 포함된 장:

- [Hub 저장소 정보, 60](#)
- [연산 참조 저장소 및 데이터 소스 구성, 62](#)
- [스키마 작성, 75](#)
- [쿼리 및 패키지, 122](#)
- [시간 표시 막대, 139](#)
- [상태 관리 및 BPM 워크플로우 도구, 166](#)
- [데이터 암호화, 179](#)
- [계층, 186](#)
- [계층 관리자 자습서, 219](#)

제 6 장

Hub 저장소 정보

이 장에 포함된 항목:

- [개요, 60](#)
- [Hub 저장소의 데이터베이스, 60](#)
- [Hub 저장소 데이터베이스 간의 관계, 61](#)
- [Hub 저장소 데이터베이스 생성, 61](#)
- [버전 요구 사항, 61](#)

개요

Hub 저장소는 Informatica MDM Hub에서 비즈니스 데이터가 저장 및 통합되는 곳입니다.

Hub 저장소에는 Informatica MDM Hub 구현의 일부인 모든 데이터베이스에 대한 일반적인 정보가 들어 있습니다.

Hub 저장소의 데이터베이스

Hub 저장소는 다음을 포함하는 데이터베이스 컬렉션입니다.

요소	설명
마스터 데이터 베이스	Informatica MDM Hub 환경 구성 설정(사용자 계정, 보안 구성, 연산 참조 저장소 레지스트리, 메시지 대기열 설정 등)을 포함합니다. 지정된 Informatica MDM Hub 환경에는 마스터 데이터베이스가 하나만 있을 수 있습니다. 마스터 데이터베이스의 기본 이름은 CMX_SYSTEM입니다. Hub 콘솔에서 구성 작업 영역의 도구(데이터베이스, 사용자, 보안 공급자, 도구 액세스 및 메시지 대기열)로 마스터 데이터베이스의 구성 설정을 관리합니다.
연산 참조 저장소 (연산 참조 저장소)	Informatica MDM Hub에서 BVT(최선의 진실, Best Version of the Truth)를 정의하는 데 사용되는 처리 규칙 및 보조 논리와 함께 마스터 데이터, 콘텐츠 메타데이터, 마스터 데이터 처리 규칙 및 마스터 데이터 개체 집합 관리 규칙을 포함하는 데이터베이스입니다. Informatica MDM Hub 구성에는 ORS 데이터베이스가 하나 이상 있을 수 있습니다. 연산 참조 저장소의 기본 이름은 CMX_ORS입니다.

마스터 데이터베이스 내에서 전역적으로 **Hub** 저장소 데이터베이스의 사용자를 생성한 다음 특정 연산 참조 저장소에 할당합니다. 마스터 데이터베이스에도 사용자 계정이 잠기기 전에 허용되는 잘못된 로그인 시도 횟수와 같은 사이트 수준 정보가 저장됩니다.

Hub 저장소 데이터베이스 간의 관계

Informatica MDM Hub 구현 환경에는 마스터 데이터베이스 한 개가 포함되고 ORS가 없거나 하나 이상 있을 수 있습니다.

ORS가 없을 경우 Hub 콘솔에서 구성 작업 영역 도구만 사용할 수 있습니다. Informatica MDM Hub 구현 환경에 연산 참조 저장소가 여러 개 있을 수 있습니다. 예를 들어 개발 및 프로덕션용으로 별도의 연산 참조 저장소가 있거나 지리적 위치별로 또는 조직의 여러 부분에 별도의 연산 참조 저장소가 있을 수 있습니다.

하나의 마스터 데이터베이스에서 여러 개의 연산 참조 저장소를 액세스하고 관리할 수 있습니다. 마스터 데이터베이스는 각 연산 참조 저장소에 대한 연결 설정과 속성을 저장합니다.

참고: 연산 참조 저장소는 한 마스터 데이터베이스에서만 등록할 수 있습니다. 여러 마스터 데이터베이스에서 동일한 연산 참조 저장소를 공유할 수 없습니다. 즉, 하나의 연산 참조 저장소를 여러 마스터 데이터베이스와 연결할 수 없습니다.

Hub 저장소 데이터베이스 생성

Informatica MDM Hub를 설치할 때 처음 데이터베이스를 생성하고 구성합니다.

- 마스터 데이터베이스와 ORS 하나를 생성하려면 `setup.sql` 스크립트를 실행해야 합니다.
- 개별 ORS를 생성하려면 `setup_ors.sql` 스크립트를 실행해야 합니다.

자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

버전 요구 사항

여러 버전의 Informatica MDM Hub가 동일한 환경에서 함께 작동할 수 없습니다.

Informatica MDM Hub 소프트웨어와 Hub 저장소의 데이터베이스를 포함하여 모든 설치 구성 요소는 동일한 버전이어야 합니다.

사이트에서 여러 버전의 Informatica MDM Hub를 사용하려면 각 버전을 별개의 환경에 설치해야 합니다. 다른 버전의 데이터베이스를 사용하여 작업하려고 하면 데이터베이스를 최신 버전으로 업그레이드하라는 메시지가 표시됩니다.

제 7 장

연산 참조 저장소 및 데이터 소스 구성

이 장에 포함된 항목:

- [연산 참조 저장소 및 데이터 소스 구성 개요, 62](#)
- [시작하기 전에, 62](#)
- [데이터베이스 도구 정보, 62](#)
- [데이터베이스 도구 시작, 63](#)
- [연산 참조 저장소 구성, 63](#)
- [데이터 소스 구성, 73](#)

연산 참조 저장소 및 데이터 소스 구성 개요

Hub 콘솔의 데이터베이스 도구를 사용하여 Hub 저장소에 대한 연산 참조 저장소 및 데이터 소스를 구성할 수 있습니다. 연산 참조 저장소를 생성한 후에는 데이터베이스 도구에서 등록 및 정의해야 합니다. 또한 데이터베이스 도구를 사용하여 데이터 소스를 생성하거나 제거할 수도 있습니다.

시작하기 전에

시작하기 전에 MDM Hub를 설치하고, MDM Hub 마스터 데이터베이스 및 하나 이상의 연산 참조 저장소를 생성해야 합니다. MDM Hub 마스터 데이터베이스 및 연산 참조 저장소를 생성하려면 *Multidomain MDM 설치 가이드*의 지침을 참조하십시오.

데이터베이스 도구 정보

Hub 저장소를 생성한 후 Hub 콘솔의 데이터베이스 도구를 사용하여 연산 참조 저장소를 등록 및 정의합니다.

MDM Hub가 연산 참조 저장소에 연결할 수 있도록 데이터베이스 도구를 사용하여 연산 참조 저장소를 등록합니다. 연산 참조 저장소를 등록하면 MDM Hub 마스터 데이터베이스에 데이터베이스 연결 속성이 저장됩니다.

데이터베이스 도구를 사용하여 연산 참조 저장소의 데이터 소스를 생성합니다. 연산 참조 저장소 데이터 소스에는 연산 참조 저장소의 속성 집합이 포함되며, 데이터베이스 서버 위치, 데이터베이스 이름, 서버와 통신하는 데 사용되는 네트워크 프로토콜, 데이터베이스 사용자 ID 및 암호와 같은 속성이 포함됩니다.

참고: 데이터베이스 도구는 연산 참조 저장소를 데이터베이스로 참조합니다.

데이터베이스 도구 시작

Hub 콘솔에서 데이터베이스 도구를 시작합니다.

1. Hub 콘솔에서 MDM Hub 마스터 데이터베이스에 연결합니다.
2. 구성 작업 영역을 확장한 다음 **데이터베이스**를 클릭합니다.

Hub 콘솔에 등록된 연산 참조 저장소가 표시되는 데이터베이스 도구가 표시됩니다.

데이터베이스 도구에는 다음과 같은 데이터베이스 정보가 표시됩니다.

데이터베이스 정보	설명
데이터베이스 수	Hub 저장소에 정의된 연산 참조 저장소 수입니다.
데이터베이스 목록	등록된 MDM Hub 연산 참조 저장소 데이터베이스 목록입니다.
데이터베이스 속성	선택한 연산 참조 저장소의 데이터베이스 속성입니다.

연산 참조 저장소 구성

Hub 콘솔의 데이터베이스 도구를 사용하여 Hub 저장소에 연산 참조 저장소를 구성할 수 있습니다.

연산 참조 저장소를 구성하는 데 도움이 필요한 경우 데이터베이스 관리자에게 문의하십시오. 연산 참조 저장소에 대한 자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

Microsoft SQL Server에 대한 연산 참조 저장소 연결 속성

Microsoft SQL Server에서 연산 참조 저장소를 등록할 경우 다음 연결 속성을 구성합니다.

데이터베이스 표시 이름

연산 참조 저장소의 이름으로서, Hub 콘솔에 표시되어야 합니다.

시스템 식별자

Hub 저장소 인스턴스에서 레코드를 고유하게 식별하기 위해 키에 지정되는 접두사입니다.

데이터베이스 호스트 이름

Microsoft SQL Server 데이터베이스를 호스팅하는 서버의 IP 주소 또는 이름입니다.

포트

Microsoft SQL Server 데이터베이스의 포트입니다. 기본값은 1433입니다.

데이터베이스 이름

연산 참조 저장소의 이름입니다.

사용자 이름

연산 참조 저장소에 대한 사용자 이름입니다. 기본적으로 이 이름은 연산 참조 저장소를 생성할 때 지정하는 사용자 이름입니다. 이 사용자는 **Hub** 저장소의 모든 연산 참조 저장소 데이터베이스 개체를 소유합니다.

참고: Microsoft SQL Server에 대한 사용자 이름을 제공할 필요가 없습니다.

암호

연산 참조 저장소에 대한 사용자 이름과 연결된 암호입니다.

DDM 연결 URL

선택 사항입니다. **Dynamic Data Masking** 서버의 URL입니다. **Dynamic Data Masking**을 사용하지 않는 경우 빈 상태로 둡니다.

또한 **요약** 페이지에서 다음 추가 속성을 구성할 수 있습니다.

연결 URL

연결 URL입니다. 연결 마법사가 기본적으로 연결 URL을 생성합니다.

등록 후 데이터 소스 작성

등록 후 응용 프로그램 서버에 데이터 소스를 작성하려면 선택합니다.

Oracle에 대한 연산 참조 저장소 연결 속성

Oracle에서 연산 참조 저장소를 등록할 경우 다음 연결 속성을 구성합니다.

데이터베이스 표시 이름

연산 참조 저장소의 이름으로서, **Hub** 콘솔에 표시되어야 합니다.

시스템 식별자

Hub 저장소 인스턴스에서 레코드를 고유하게 식별하기 위해 키에 지정되는 접두사입니다.

데이터베이스 호스트 이름

Microsoft SQL Server 데이터베이스를 호스팅하는 서버의 IP 주소 또는 이름입니다.

SID

서버에서 실행 중인 **Oracle** 데이터베이스의 인스턴스를 나타내는 **Oracle** 시스템 식별자입니다. 이 필드는 **SID** 연결 유형을 선택한 경우에 표시됩니다.

서비스

Oracle 데이터베이스에 연결하는 데 사용되는 **Oracle** 서비스의 이름입니다. 이 필드는 서비스 연결 유형을 선택한 경우에 표시됩니다.

포트

Oracle 데이터베이스 서버에서 실행 중인 **Oracle** 수신기의 TCP 포트입니다. 기본값은 1521입니다.

Oracle TNS 이름

응용 프로그램 서버의 **TNSNAMES.ORA** 파일에 정의된 대로 네트워크에서 데이터베이스로 알려진 이름입니다.

예를 들면 **mydatabase.mycompany.com**입니다.

Oracle 데이터베이스를 설치할 때 **Oracle TNS** 이름을 설정합니다. **Oracle TNS** 이름에 대한 자세한 내용은 **Oracle** 설명서를 참조하십시오.

스키마 이름

연산 참조 저장소의 이름입니다.

참고: **스키마 이름** 및 **사용자 이름**은 모두 연산 참조 저장소의 이름으로 연산 참조 저장소를 생성할 때 지정하는 것입니다. 이 정보가 필요한 경우 데이터베이스 관리자에게 문의하십시오.

암호

연산 참조 저장소에 대한 사용자 이름과 연결된 암호입니다.

Oracle의 경우 이 암호는 대/소문자를 구분하지 않습니다.

기본적으로 이는 연산 참조 저장소를 생성할 때 지정하는 암호입니다.

DDM 연결 URL

선택 사항입니다. Dynamic Data Masking 서버의 URL입니다. Dynamic Data Masking을 사용하지 않는 경우 빈 상태로 둡니다.

또한 **요약** 페이지에서 다음 추가 속성을 구성할 수 있습니다.

연결 URL

연결 URL입니다. 연결 마법사가 기본적으로 연결 URL을 생성합니다. 다음 예에서는 연결 URL의 형식을 보여 줍니다.

서비스 연결 유형:

`jdbc:oracle:thin:@//database_host:port/service_name`

SID 연결 유형:

`jdbc:oracle:thin:@//database_host:port/sid`

서비스 연결 유형의 사용자 지정 URL을 지정할 수 있습니다. 예:

`jdbc:oracle:thin:@//orclhost:1521/mdmorcl.mydomain.com`

등록 후 데이터 소스 작성

등록 후 응용 프로그램 서버에 데이터 소스를 작성하려면 선택합니다.

참고: 해당 옵션을 선택하지 않는 경우에는 데이터 소스를 수동으로 구성해야 합니다.

IBM DB2에 대한 연산 참조 저장소 연결 속성

IBM DB2에서 연산 참조 저장소를 등록할 경우 다음 연결 속성을 구성합니다.

데이터베이스 표시 이름

연산 참조 저장소의 이름으로서, Hub 콘솔에 표시되어야 합니다.

시스템 식별자

Hub 저장소 인스턴스에서 레코드를 고유하게 식별하기 위해 키에 지정되는 접두사입니다.

데이터베이스 서버 이름

IBM DB2 데이터베이스를 호스팅하는 서버의 이름 또는 IP 주소입니다.

데이터베이스 이름

생성한 데이터베이스의 이름입니다.

데이터베이스 호스트 이름

IBM DB2 데이터베이스를 호스팅하는 서버의 이름 또는 IP 주소입니다.

포트

IBM DB2 데이터베이스 서버용 TCP 포트입니다. 기본값은 5000입니다.

스키마 이름

연산 참조 저장소의 이름입니다.

참고: **스키마 이름**과 **사용자 이름**은 모두 연산 참조 저장소의 이름으로 연산 참조 저장소를 생성할 때 지정된 것입니다. 이 정보가 필요한 경우 데이터베이스 관리자에게 문의하십시오.

사용자 이름

연산 참조 저장소에 대한 사용자 이름입니다. 기본적으로 이 이름은 연산 참조 저장소를 생성하는 데 사용한 스크립트에서 지정한 사용자 이름입니다. 이 사용자는 Hub 저장소의 모든 연산 참조 저장소 데이터베이스 개체를 소유합니다.

암호

연산 참조 저장소에 대한 사용자 이름과 연결된 암호입니다.

IBM DB2의 경우 암호는 대/소문자를 구분합니다.

기본적으로 이는 연산 참조 저장소를 생성할 때 지정하는 암호입니다.

DDM 연결 URL

선택 사항입니다. Dynamic Data Masking 서버의 URL입니다. Dynamic Data Masking을 사용하지 않는 경우 빈 상태로 둡니다.

또한 **요약** 페이지에서 다음 추가 속성을 구성할 수 있습니다.

연결 URL

연결 URL입니다. 연결 마법사가 기본적으로 연결 URL을 생성합니다. 다음 예에서는 연결 URL의 형식을 보여 줍니다.

```
jdbc:db2://database_host:port/db_name
```

등록 후 데이터 소스 작성

등록 후 응용 프로그램 서버에 데이터 소스를 작성하려면 선택합니다.

참고: 해당 옵션을 선택하지 않는 경우에는 데이터 소스를 수동으로 구성해야 합니다.

연산 참조 저장소 등록

연산 참조 저장소를 통해 Hub 콘솔을 등록할 수 있습니다. 둘 이상의 MDM Hub 마스터 데이터베이스가 있는 연산 참조 저장소는 등록할 수 없습니다.

1. Hub 콘솔을 시작합니다.

데이터베이스 변경 대화 상자가 표시됩니다.

2. MDM Hub 마스터 데이터베이스를 선택하고 **연결**을 클릭합니다.
3. 구성 작업 영역에서 **데이터베이스** 도구를 시작합니다.
4. 쓰기 잠금을 획득합니다.
5. **데이터베이스 등록** 단추를 클릭합니다.

Informatica MDM Hub 연결 마법사가 표시되어 데이터베이스 유형을 선택하라는 메시지를 표시합니다.

6. 데이터베이스 유형을 선택하고 **다음**을 클릭합니다.
7. Oracle 데이터베이스에 대한 연결을 작성한 경우 연결 방법을 선택하고 **다음**을 클릭합니다.

- 서비스 이름을 사용하여 Oracle에 연결하려면 **서비스**를 선택합니다.
- Oracle 시스템 ID를 사용하여 Oracle에 연결하려면 **SID**를 선택합니다.

참고: 서비스 및 SID 이름에 대한 자세한 내용은 Oracle 설명서를 참조하십시오.

연결 속성 페이지가 표시됩니다.

- 데이터베이스의 연결 속성을 구성합니다.
- 요약** 페이지의 변경 사항을 검토하고 추가 연결 속성을 지정합니다.
- 마침을 클릭합니다.

데이터베이스 등록 대화 상자가 표시됩니다.

- 확인을 클릭합니다.

MDM Hub이 연산 참조 저장소를 등록합니다.

- 등록한 연산 참조 저장소를 선택하고 **데이터베이스 연결 테스트** 단추를 클릭하여 데이터베이스 설정을 테스트합니다.

WebSphere를 사용하는 경우 데이터베이스 연결을 테스트하기 전에 WebSphere를 다시 시작해야 합니다.

데이터베이스 테스트 대화 상자에 데이터베이스 연결 테스트의 결과가 표시됩니다.

- 확인을 클릭합니다.

연산 참조 저장소가 등록되고 데이터베이스에 대한 연결이 테스트됩니다.

참고: 다른 곳에서 사용되는 연산 참조 저장소를 등록하고 연산 참조 저장소에 등록된 처리 서버가 있는 경우 다른 서버를 등록하지 못할 수도 있습니다. 처리 서버 중 하나를 다시 등록해야 합니다. 그러면 c_repos_db_release의 데이터가 업데이트됩니다.

연산 참조 저장소 등록 속성 편집

특정 연산 참조 저장소 등록 속성을 편집할 수 있습니다. 편집할 수 없는 속성을 편집하려면 연산 참조 저장소를 등록 해제했다가 새 속성으로 다시 등록합니다.

- 데이터베이스 도구를 시작합니다.
- 쓰기 잠금을 획득합니다.
- 구성할 연산 참조 저장소를 선택합니다.
- 데이터베이스 연결 속성 편집** 단추를 클릭합니다.

데이터베이스 등록 대화 상자가 선택한 연산 참조 저장소에 대해 표시됩니다.

- 데이터베이스 등록 설정을 편집합니다.

- Oracle의 경우 다음 데이터베이스 등록 설정을 편집할 수 있습니다.

속성	설명
데이터베이스 표시 이름	연산 참조 저장소에 대한 이름으로 Hub 콘솔에 표시되어야 합니다.
시스템 식별자	Hub 저장소 인스턴스에서 레코드를 고유하게 식별하기 위해 키에 할당되는 접두사입니다.
Oracle TNS 이름	네트워크에서 데이터베이스로 알려진 이름입니다. TNS 이름은 TNSNAMES.ORA 파일에 정의되어 있습니다.
암호	연산 참조 저장소가 생성될 때 지정된 사용자 이름과 연결된 암호입니다.
DDM 연결 URL	선택 사항입니다. Dynamic Data Masking 서버의 URL입니다. Dynamic Data Masking을 사용하지 않는 경우 빈 상태로 둡니다.

- Microsoft SQL Server에 대해 다음 데이터베이스 등록 설정을 편집합니다.

속성	설명
데이터베이스 표시 이름	연산 참조 저장소에 대한 이름으로 Hub 콘솔에 표시되어야 합니다.
시스템 식별자	Hub 저장소 인스턴스에서 레코드를 고유하게 식별하기 위해 키에 할당되는 접두사입니다.
암호	연산 참조 저장소가 생성될 때 지정된 사용자 이름과 연결된 암호입니다.
DDM 연결 URL	선택 사항입니다. Dynamic Data Masking 서버의 URL입니다. Dynamic Data Masking을 사용하지 않는 경우 빈 상태로 둡니다.

6. 변경한 설정으로 응용 프로그램 서버의 데이터 소스를 업데이트하려면 **등록 후 데이터 소스 업데이트** 옵션을 활성화하고 **확인**을 클릭합니다.
Hub 콘솔에 데이터 소스 매개 변수 및 해당 연결 풀 매개 변수를 기본값으로 재설정하라는 메시지가 표시됩니다.
7. 기본 데이터 소스 및 연결 풀 매개 변수로 재설정하려면 **예**를 클릭합니다. 데이터 소스 및 연결 풀 매개 변수를 유지하려면 **아니오**를 클릭합니다.
데이터베이스 등록 업데이트 대화 상자가 표시됩니다.
8. **확인**을 클릭합니다.
데이터베이스 도구에서 변경 내용이 저장됩니다.
9. 업데이트된 데이터베이스 연결 설정을 테스트합니다.

연산 참조 저장소 속성 편집

등록된 연산 참조 저장소의 속성을 변경할 수 있습니다.

1. **데이터베이스** 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 구성할 연산 참조 저장소를 선택합니다.
데이터베이스 도구에 선택한 연산 참조 저장소의 데이터베이스 속성이 표시됩니다.

다음 테이블에는 데이터베이스 속성이 설명되어 있습니다.

속성	설명
데이터베이스 유형	데이터베이스 유형입니다(예: Oracle 또는 Microsoft SQL Server).
데이터베이스 ID	<p>연산 참조 저장소의 ID입니다. MDM Hub에서는 SIF 요청에 데이터베이스 ID를 사용합니다. 데이터베이스 ID 조회는 대/소문자를 구분합니다.</p> <p>Oracle 데이터베이스 ID의 형식은 다음과 같습니다.</p> <ul style="list-style-type: none"> - Oracle 연결 유형이 SID인 경우 형식은 <code>hostname-sid-databaseName</code>입니다. - Oracle 연결 유형이 Service인 경우 형식은 <code>hostname-sid-databaseName</code>입니다. <p>Microsoft SQL Server 데이터베이스 ID의 형식은 다음과 같습니다.</p> <p><code>hostname-databaseName</code></p> <p>연산 참조 저장소를 등록하면 MDM Hub에서 호스트, 서버 및 데이터베이스 이름을 정규화합니다.</p> <p>MDM Hub는 호스트 이름을 소문자로 변환하고, 데이터베이스 이름을 대문자로 변환합니다. 이는 스키마 및 테이블과 같은 데이터베이스 개체의 표준입니다.</p>
JNDI 데이터 소스 이름	<p>선택한 연산 참조 저장소의 데이터 소스 JNDI 이름입니다. MDM Hub가 응용 프로그램 서버에서 JDBC 연결에 대해 구성하는 JNDI 이름입니다.</p> <p>Oracle JNDI 데이터 소스 이름의 형식은 다음과 같습니다.</p> <ul style="list-style-type: none"> - Oracle 연결 유형이 SID인 경우 형식은 <code>jdbc/siperian-hostname-sid-databaseName-ds</code>입니다. - Oracle 연결 유형이 Service인 경우 형식은 <code>jdbc/siperian-servicename-databaseName-ds</code>입니다. <p>Microsoft SQL Server JNDI 데이터 소스의 형식은 다음과 같습니다.</p> <p><code>jdbc/siperian-hostname-databaseName-ds</code></p>
시스템 식별자	Hub 저장소 인스턴스에서 레코드를 고유하게 식별하기 위해 키에 할당되는 접두사입니다.
GETLIST 제한 (레코드)	<code>searchQuery</code> , <code>searchMatch</code> 및 <code>getLookupValues</code> 와 같은 SIF 검색 요청을 통해 반환되는 레코드 수에 대한 제한입니다. 기본값은 200이고 최대값은 5,999입니다.
프로덕션 모드	<p>연산 참조 저장소가 프로덕션 모드에 있는지 아니면 일반 모드에 있는지를 지정합니다.</p> <p>비활성화하면 권한 있는 사용자가 Hub 콘솔에서 연산 참조 저장소의 메타데이터를 편집할 수 있습니다. 기본적으로 비활성화되어 있습니다.</p> <p>활성화하면 사용자가 연산 참조 저장소의 메타데이터를 편집할 수 없습니다. 사용자가 프로덕션 모드에 있는 연산 참조 저장소에 대한 쓰기 잠금을 획득하려고 하면 Hub 콘솔에 잠금을 획득할 수 없음을 설명하는 메시지가 표시됩니다.</p> <p>참고: MDM Hub 관리자 사용자만 연산 참조 저장소에 대해 프로덕션 모드를 활성화하거나 비활성화할 수 있습니다.</p>
전환 모드	<p>연산 참조 저장소가 전환 모드에서 실행 중인지 여부를 지정합니다. 전환 모드는 연산 참조 저장소에 대해 프로덕션 모드를 활성화한 경우에 사용할 수 있습니다.</p> <p>활성화하면 사용자가 리포지토리 관리자의 승격 작업을 수행할 수 있습니다.</p> <p>비활성화하면 사용자가 리포지토리 관리자의 승격 작업을 수행할 수 없습니다.</p>

속성	설명
일괄 API 상호 운용성	MDM Hub가 연산 참조 저장소에 대해 행 수준 잠금을 사용하여 SIF API 쓰기 호출 및 일괄 작업을 동시에 실행할 수 있는지 여부를 지정합니다. 활성화하면 SIF API 쓰기 호출에 대해 행 수준 잠금을 사용할 수 있으므로 동시성 및 성능이 향상됩니다. 비활성화하면 행 수준 잠금을 사용할 수 없습니다.
ZDT 사용	연산 참조 저장소가 ZDT(Zero Downtime) 모드에서 실행 중인지 여부를 지정합니다. 활성화하면 연산 참조 저장소가 ZDT 모드에서 실행됩니다. 비활성화하면 연산 참조 저장소가 ZDT 모드에서 실행되지 않습니다.

4. 속성을 변경하려면 속성 옆의 **편집** 단추를 클릭하고 속성을 편집합니다.

5. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

연산 참조 저장소에 대해 프로덕션 모드를 활성화하면 데이터베이스 도구의 목록에서 연산 참조 저장소 옆에 잠금 아이콘이 표시됩니다.

연산 참조 저장소 연결

연산 참조 저장소에 연결할 수 있는지 확인해야 합니다.

연산 참조 저장소 연결을 테스트할 때 [데이터베이스 테스트] 명령은 다음 연결 속성에 대해 테스트합니다.

- 데이터베이스 연결 매개 변수
- 데이터 소스의 존재 여부
- 데이터 소스 연결
- 연산 참조 저장소 버전의 유효성

연산 참조 저장소 연결 테스트

연산 참조 저장소에 대한 연결을 테스트할 수 있습니다.

1. **데이터베이스** 도구를 시작합니다.
2. 테스트할 연산 참조 저장소를 선택합니다.
3. **데이터베이스 연결 테스트** 단추를 클릭합니다.

데이터베이스 테스트 대화 상자가 표시됩니다.

참고: WebSphere의 경우 Hub 콘솔을 통한 테스트 연결이 실패하면 WebSphere 콘솔에서 연결을 확인합니다. JNDI 이름은 대/소문자를 구분하며 Hub 콘솔에서 생성한 것과 일치해야 합니다.

4. **확인**을 클릭합니다.

암호 변경

MDM Hub를 설치한 후 MDM Hub 마스터 데이터베이스 또는 연산 참조 저장소의 암호를 변경할 수 있습니다.

Oracle 환경에서 MDM Hub 마스터 데이터베이스의 암호 변경

Oracle 환경에서 MDM Hub 마스터 데이터베이스의 암호를 변경하려면 데이터베이스 서버에서 암호를 변경한 다음 응용 프로그램 서버에서 암호를 업데이트합니다.

1. 데이터베이스 서버에서 **CMX_SYSTEM** 데이터베이스의 암호를 변경합니다.
2. 응용 프로그램 서버의 **Administration Console**에 로그인합니다.
3. 데이터 소스 연결 정보를 편집하여 1단계에서 생성된 **CMX_SYSTEM** 암호를 지정합니다. 변경 내용을 저장합니다.

IBM DB2 환경에서 MDM Hub 마스터 데이터베이스의 암호 변경

IBM DB2 환경에서 MDM Hub 마스터 데이터베이스의 암호를 변경하려면 운영 체제에서 암호를 변경한 다음 응용 프로그램 서버에서 암호를 업데이트합니다.

1. 운영 체제 환경에서 **CMX_SYSTEM**의 암호를 변경합니다.
2. 응용 프로그램 서버의 **Administration Console**에 로그인합니다.
3. 데이터 소스 연결 정보를 편집하여 1단계에서 생성된 **CMX_SYSTEM** 암호를 지정합니다. 변경 내용을 저장합니다.

연산 참조 저장소의 암호 변경

연산 참조 저장소의 암호를 변경할 수 있습니다.

1. IBM DB2 환경이나 운영 체제 인증이 활성화된 **Microsoft SQL Server** 환경의 경우 운영 체제 암호를 변경합니다. 이 암호는 연산 참조 저장소의 암호와 일치해야 합니다.
2. 데이터베이스 서버에서 연산 참조 저장소 스키마의 암호를 변경합니다.
3. Hub 콘솔을 시작하고 대상 데이터베이스로 **MDM Hub** 마스터 데이터베이스를 선택합니다.
4. **데이터베이스** 도구를 시작합니다.
5. 쓰기 잠금을 획득합니다.
6. 구성할 연산 참조 저장소를 선택합니다.
7. **데이터베이스 속성** 패널에서 선택한 연산 참조 저장소에 대해 표시되는 **JNDI** 데이터 소스 이름을 적어 둡니다.
8. 응용 프로그램 서버의 관리 콘솔에 로그인합니다. 연산 참조 저장소에 대한 데이터 소스 연결 정보를 편집하여, 적어둔 **JNDI** 데이터 소스 이름에 대해 새 암호를 지정한 다음 변경 내용을 저장합니다.
9. 데이터베이스 도구에서 데이터베이스에 대한 연결을 테스트합니다.

암호 암호화

스키마 암호를 성공적으로 변경하려면 응용 프로그램 서버에 정의된 데이터 소스에서 암호를 변경해야 합니다.

암호는 응용 프로그램 서버에서 보호하기 때문에 암호화되지 않습니다. 응용 프로그램 서버에서 데이터 소스를 업데이트할 뿐만 아니라 암호를 암호화하여 여러 테이블에 저장해야 합니다.

스키마에 대한 암호 암호화

데이터베이스 스키마 암호를 암호화하여 보호할 수 있습니다.

- ▶ 데이터베이스 스키마 암호를 암호화하려면 명령 프롬프트에서 다음 명령을 실행합니다.

```
java -classpath siperian-api.jar;siperian-common.jar;siperian-server.jar
com.delos.util.PublicKeyBasedEncryptionHelper <plain text password> <Hub Server installation
directory>
```

결과가 터미널 창에 출력됩니다.

```
Plaintext Password: password
Encrypted Password: encrypted password
```

스키마의 암호 업데이트

MDM Hub 마스터 데이터베이스 암호 또는 연산 참조 저장소 암호를 업데이트할 수 있습니다.

1. 마스터 데이터베이스 암호 또는 연산 참조 저장소 암호를 업데이트하려면 **cmx_system** 사용자로 연결한 후 다음 문을 실행합니다.

Oracle 및 IBM Db2의 경우.

```
UPDATE C_REPOS_DATABASE SET PASSWORD = '<new_password>' WHERE USER_NAME = <user_name>;
COMMIT;
```

Microsoft SQL Server의 경우.

```
UPDATE [dbo].[C_REPOS_DATABASE] SET PASSWORD = '<new_password>' WHERE USER_NAME = <user_name>
```

2. 응용 프로그램 서버를 다시 시작합니다.

연산 참조에 대한 프로덕션 모드

MDM Hub 관리자는 Hub 콘솔을 사용하여 연산 참조 저장소에 대해 프로덕션 모드를 활성화할 수 있습니다. 연산 참조 저장소가 프로덕션 모드에 있을 때는 연산 참조 저장소의 디자인이 잠깁니다.

연산 참조 저장소가 프로덕션 모드에 있는 경우 관리자 권한이 없는 사용자는 쓰기 잠금을 획득할 수 없습니다. 연산 참조 저장소의 스키마 정의를 변경할 수 없습니다. 프로덕션 모드에 있는 연산 참조 저장소에 대한 잠금을 획득하려고 하면 Hub 콘솔에 연산 참조 저장소가 프로덕션 모드에 있기 때문에 잠금을 획득할 수 없음을 설명하는 메시지가 표시됩니다.

연산 참조에 대한 프로덕션 모드 활성화 또는 비활성화

Hub 콘솔을 사용하여 연산 참조 저장소에 대해 프로덕션 모드를 활성화합니다.

연산 참조 저장소에 대해 프로덕션 모드를 활성화하려면 데이터베이스 도구를 실행하고 마스터 데이터베이스에 대한 잠금을 획득할 수 있는 관리자 권한이 있어야 합니다.

1. MDM Hub 구현에 대한 관리자 권한으로 Hub 콘솔에 로그인합니다.
2. **데이터베이스** 도구를 시작합니다.
3. 연산 참조 저장소에 대한 배타적 잠금을 해제합니다.
연산 참조 저장소에 배타적 잠금이 있는 경우 연산 참조 저장소에 대해 프로덕션 모드를 활성화하거나 비활성화할 수 없습니다.
4. 쓰기 잠금을 획득합니다.
5. 구성할 연산 참조 저장소를 선택합니다.
데이터베이스 도구에 선택한 연산 참조 저장소의 데이터베이스 속성이 표시됩니다.
6. **프로덕션 모드** 옵션을 활성화하거나 비활성화합니다.
7. **저장**을 클릭합니다.

연산 참조 저장소 등록 해제

데이터베이스 도구를 사용하여 ORS(연산 참조 저장소)를 등록 해제할 수 있습니다. ORS를 등록 해제할 때 MDM Hub는 MDM Hub 마스터 데이터베이스에서 ORS에 대한 연결 정보를 제거합니다. ORS를 등록 해제할 때 MDM Hub는 응용 프로그램 서버 환경에서 데이터 소스 정의도 제거합니다.

1. MDM Hub 콘솔에서 **쓰기 잠금 > 잠금 획득**을 클릭합니다.
2. 구성 작업 영역에서 **데이터베이스** 도구를 선택합니다.
데이터베이스 정보 페이지가 표시됩니다.
3. 데이터베이스 목록에서 등록을 해제할 ORS를 선택합니다.
4. **데이터베이스 등록 해제**를 클릭합니다. WebLogic을 사용하는 경우 응용 프로그램 서버에 대한 사용자 이름과 암호를 입력합니다.
데이터베이스 도구에 ORS 등록 해제를 확인하는 메시지가 표시됩니다.
5. **예**를 클릭합니다.

IBM DB2에서 연산 참조 저장소 삭제

연산 참조 저장소를 사용하지 않으려는 경우 이를 삭제하여 데이터베이스에서 제거할 수 있습니다.

1. 연산 참조 저장소의 등록을 해제하려면 MDM Hub 콘솔을 사용합니다.
2. IBM DB2 명령 창을 엽니다.
3. 다음 명령을 입력합니다.

```
CALL SYSPROC.ADMIN_DROP_SCHEMA('<Operational Reference Store name>', NULL, 'ERRORSCHEMA',  
'ERRORTABLE')
```

이 명령에 지정된 연산 참조 저장소가 삭제됩니다.

데이터 소스 구성

모든 연산 참조 저장소에는 응용 프로그램 서버 환경의 데이터 소스 정의가 필요합니다. MDM Hub에서 데이터 소스는 데이터베이스 서버 위치, 데이터베이스 이름, 데이터베이스 사용자 ID 및 암호와 같은 연산 참조 저장소의 속성을 지정합니다.

MDM Hub 데이터 소스는 응용 프로그램 서버 환경에 정의된 JDBC 리소스를 가리킵니다. JDBC 데이터 소스에 대한 자세한 내용은 응용 프로그램 서버 설명서를 참조하십시오.

WebLogic에서 데이터 소스 관리

WebLogic의 경우 데이터 소스를 추가, 삭제 또는 업데이트하려고 할 때마다 MDM Hub에 WebLogic 관리자 이름 및 암호를 지정하라는 메시지가 표시됩니다.

데이터베이스 도구에서 여러 작업을 수행하는 경우 이 대화 상자에 마지막으로 입력한 사용자 이름은 유지되지만 암호는 매번 입력해야 합니다.

데이터 소스 생성

다른 응용 프로그램 서버를 사용하여 연산 참조 저장소를 생성하거나, 연산 참조 저장소를 등록할 때 데이터 소스를 생성하지 않은 경우 명시적으로 데이터 소스를 생성해야 할 수 있습니다.

1. **데이터베이스** 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 데이터베이스 목록에서 연산 참조 저장소를 마우스 오른쪽 단추로 클릭한 다음 **데이터 소스 생성**을 클릭합니다.
4. WebLogic을 사용하는 경우 메시지가 표시되면 WebLogic 사용자 이름과 암호를 입력합니다.
데이터베이스 도구에서 데이터 소스를 생성하고 진행률 메시지를 표시합니다.
5. **확인**을 클릭합니다.

데이터 소스 제거

연산 참조 저장소를 등록하고 데이터 소스를 구성한 경우 데이터베이스 도구를 사용하여 수동으로 응용 프로그램 서버에서 데이터 소스 정의를 제거할 수 있습니다.

그러나 데이터 소스 정의를 제거한 후에도 연산 참조 저장소가 여전히 Hub 콘솔에 표시됩니다. Hub 콘솔에서 연산 참조 저장소를 완전히 제거하려면 연산 참조 저장소를 등록 해제합니다.

1. **데이터베이스** 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 데이터베이스 목록에서 연산 참조 저장소를 마우스 오른쪽 단추로 클릭한 다음 **데이터 소스 제거**를 클릭합니다.
4. WebLogic을 실행하는 경우 메시지가 표시되면 WebLogic 사용자 이름과 암호를 입력합니다.
데이터베이스 도구에서 데이터 소스를 제거하고 진행률 메시지를 표시합니다.
5. **확인**을 클릭합니다.

제 8 장

스키마 작성

이 장에 포함된 항목:

- [개요, 75](#)
- [시작하기 전에, 75](#)
- [스키마 정보, 75](#)
- [스키마 관리자 시작, 93](#)
- [기본 개체 구성, 94](#)
- [테이블의 열 구성, 107](#)
- [기본 개체 간의 외래 키 관계 구성, 115](#)
- [스키마 보기, 118](#)

개요

이 장에서는 Informatica MDM Hub에서 스키마를 디자인하고 작성하는 방법에 대해 설명합니다.

시작하기 전에

시작하기 전에 *Multidomain MDM 설치 가이드*의 지침을 사용하여 Informatica MDM Hub를 설치하고 Hub 저장소(연산 참조 저장소 포함)를 생성해야 합니다.

스키마 정보

스키마는 MDM Hub 구현에 사용되는 데이터 모델입니다.

MDM Hub에서는 특정 스키마를 강요하거나 요구하지 않습니다. 스키마는 MDM Hub 내에 존재하며 MDM Hub에 데이터를 제공하는 소스 시스템과는 독립적입니다.

참고: MDM Hub 구현의 스키마 설계 프로세스는 이 문서의 범위에 포함되지 않습니다. 이 문서에서는 조직의 요구 사항에 맞게 업계 표준 데이터 모델링 방법론을 사용하여 데이터 모델을 이미 개발했다고 가정합니다.

Informatica 스키마는 수직적 비즈니스 부문의 데이터 구조를 지원하는 리포지토리 기반의 유연한 모델입니다. Hub 저장소는 MDM Hub 기능을 지원하는 데이터베이스입니다. 설치된 모든 MDM Hub에는 MDM Hub 마스터 데이터베이스 하나와 연산 참조 저장소 데이터베이스 하나 이상이 포함된 Hub 저장소가 있습니다. 시스템 구성에 따라 여러 개의 연산 참조 저장소 데이터베이스가 설치되어 있을 수 있습니다. 예를 들어 개발 연산 참조 저장소, 테스트 연산 참조 저장소 및 프로덕션 연산 참조 저장소가 있을 수 있습니다.

스키마 구현을 시작하기 전에 기본 MDM Hub 스키마 및 스키마를 이루는 구성 요소의 기본 구조를 이해 하고 있어야 합니다.

참고: Hub 콘솔 도구를 사용하여 통합된 스키마를 정의하고 관리해야 합니다. 데이터베이스를 바로 변경할 수는 없습니다. 예를 들어 테이블과 열을 정의하려면 스키마 관리자를 사용해야 합니다.

연산 참조 저장소의 테이블 유형

연산 참조 저장소에는 사용자가 구성하는 테이블과 시스템 지원 테이블이 포함됩니다.

구성 가능한 테이블

구성 가능한 테이블을 사용하여 비즈니스 참조 데이터를 모델링할 수 있습니다. 이러한 테이블은 명시적으로 생성 및 구성해야 합니다.

다음 테이블에는 사용자가 구성할 수 있는 MDM Hub 테이블 유형이 설명되어 있습니다.

테이블 유형	설명
기본 개체로	핵심 비즈니스 항목(예: 고객, 제품 또는 직원)이나 조회 테이블(예: 국가 또는 시/도)에 대한 데이터를 저장하는 데 사용됩니다. 기본 개체에서는 여러 소스 시스템의 데이터를 통합하고 트러스트 설정을 사용하여 각 기본 개체 셀에서 가장 신뢰할 수 있는 값을 확인할 수 있습니다. 기본 개체 간에는 일대다 관계를 정의할 수 있습니다. 기본 개체를 작성 및 구성해야 합니다.
랜딩 테이블	소스 시스템에서 일괄 로드를 받는 데 사용됩니다. 랜딩 테이블을 작성 및 구성해야 합니다.
준비 테이블	데이터를 기본 개체로 로드하는 데 사용됩니다. 랜딩 테이블에서 준비 테이블로 데이터를 이동할 때 데이터를 정리 및 표준화해야 하는 방법을 지정하기 위해 랜딩 테이블과 준비 테이블 간에 매핑을 정의합니다. 준비 테이블을 작성 및 구성해야 합니다.

하부 구조 테이블

MDM Hub 하부 구조 테이블은 Hub 저장소에서 데이터의 흐름을 관리 및 지원합니다. 기본 개체가 구성될 때마다 MDM Hub에서 MDM Hub 하부 구조 테이블을 작성, 구성 및 유지 관리합니다.

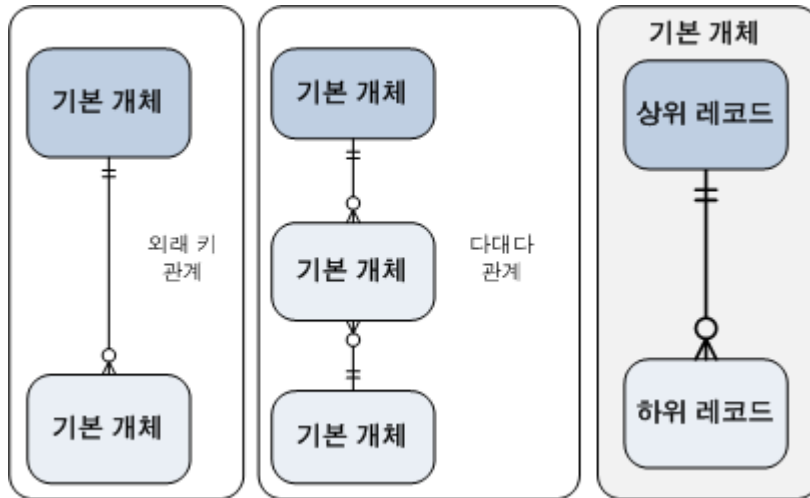
다음 테이블에는 MDM Hub 하부 구조 테이블 유형이 설명되어 있습니다.

테이블 유형	설명
교차 참조 테이블	<p>기본 개체에 있는 각 레코드의 출처를 추적하는 데 사용됩니다.</p> <p>교차 참조 테이블 이름은 다음 패턴에 따라 지정됩니다.</p> <p>C_<기본 개체 이름>_XREF</p> <p>여기서 <기본 개체 이름>은 기본 개체의 루트 이름입니다. 예를 들어, 기본 개체 이름이 PARTY인 경우 교차 참조 테이블 이름은 C_PARTY_XREF입니다. 기본 개체를 생성하면 MDM Hub에서 교차 참조 테이블을 작성하여 소스 시스템에서 가져오는 데이터에 대한 정보를 저장합니다.</p>
기록 테이블	<p>기본 개체에 대해 기록이 활성화되어 있는 경우에 사용됩니다.</p> <p>기록 테이블 이름은 다음 패턴에 따라 지정됩니다.</p> <ul style="list-style-type: none"> - C_<기본 개체 이름>_HIST. 기본 개체 기록 테이블입니다. - C_<기본 개체 이름>_HXRF. 교차 참조 기록 테이블입니다. <p>여기서 <기본 개체 이름>은 기본 개체의 루트 이름입니다. 예를 들어, PARTY라는 기본 개체에는 C_PARTY_HIST 및 C_PARTY_HXRF 기록 테이블이 있습니다.</p> <p>MDM Hub에서 다른 유형의 기록 테이블을 작성 및 유지 관리합니다. 기록 테이블은 병합 및 병합 해제 기록, 사전 정리된 데이터 기록, 기본 개체 기록 및 교차 참조 기록을 포함한 상세한 변경 추적 옵션을 제공합니다.</p>
일치 키 테이블	<p>MDM Hub가 모든 기본 개체 레코드에 대해 생성하는 일치 키를 포함합니다. 일치 키 테이블 이름은 다음 패턴에 따라 지정됩니다.</p> <p>C_<기본 개체 이름>_STRP</p> <p>여기서 <기본 개체 이름>은 기본 개체의 루트 이름입니다. 예를 들어, 기본 개체 이름이 PARTY인 경우 일치 키 테이블 이름은 C_PARTY_STRP입니다.</p>
일치 테이블	<p>기본 개체에 대해 일치 프로세스를 실행하여 얻은 기본 개체의 일치된 레코드 쌍이 포함됩니다. 일치 테이블 이름은 다음 패턴에 따라 지정됩니다.</p> <p>C_<기본 개체 이름>_MTCH</p> <p>여기서 <기본 개체 이름>은 기본 개체의 루트 이름입니다. 예를 들어, 기본 개체 이름이 PARTY인 경우 일치 테이블 이름은 C_PARTY_MTCH입니다.</p>
외부 일치 테이블	<p>외부 일치 작업에 대한 데이터를 포함합니다. 사용할 수 있는 외부 일치 테이블 유형은 다음과 같습니다.</p> <ul style="list-style-type: none"> - EMI 테이블. 외부 일치 입력 테이블입니다. 기본 개체의 레코드와 일치시킬 레코드가 포함됩니다. - EMO 테이블. 외부 일치 출력 테이블입니다. 외부 일치 작업에 대한 출력 데이터를 포함합니다. EMO 테이블의 각 행은 일치된 레코드 쌍을 나타냅니다. 레코드의 쌍 중 하나는 EMI 테이블에서 가져오고 다른 쌍은 기본 개체에서 가져옵니다. <p>외부 일치 테이블 이름은 다음 패턴에 따라 지정됩니다.</p> <ul style="list-style-type: none"> - C_<기본 개체 이름>_EMI - C_<기본 개체 이름>_EMO <p>여기서 <기본 개체 이름>은 기본 개체의 루트 이름입니다. 예를 들어, 기본 개체 이름이 PARTY인 경우 일치 테이블 이름은 C_PARTY_EMI 및 C_PARTY_EMO입니다.</p>
임시 테이블	<p>MDM Hub가 일괄 작업 중에 데이터를 처리해야 하는 임시 테이블을 작성합니다. MDM Hub가 데이터 처리를 완료한 후에는 더 이상 필요하지 않기 때문에 일괄 프로세스에서 임시 테이블을 제거합니다. 일괄 프로세스, 응용 프로그램 서버 또는 데이터베이스 서버가 실패할 경우 임시 테이블이 삭제되지 않을 수도 있습니다.</p> <p>임시 테이블에는 T\$_ 접두사가 있습니다. 일괄 프로세스가 완료된 후 리포지토리에 일괄 프로세스에서 작성한 임시 테이블이 포함된 경우 이를 수동으로 제거할 수 있습니다. 일부 임시 테이블에는 T\$_ 접두사가 없습니다. 예를 들어, 정리 임시 테이블에는 대신 _CL 접미사가 있고, 델타 임시 테이블에는 _DLT 접미사가 있습니다.</p>

지원되는 데이터 간 관계

MDM Hub은 동일한 기본 개체의 레코드 간 계층 관계 외에 테이블 간 일대다 및 다대다 관계를 지원합니다. MDM Hub에서 여러 방법으로 레코드 간 관계를 정의할 수 있습니다.

다음 이미지에서는 기본 개체 간 다른 관계를 보여 줍니다.



다음 테이블에는 데이터 간 관계 유형이 설명되어 있습니다.

관계 유형	설명
기본 개체 간의 외래 키 관계	한 기본 개체(하위 테이블)에 다른 기본 개체(상위 테이블)의 기본 키 열에 있는 값과 일치하는 값을 포함하는 외래 키 열이 포함되어 있습니다.
동일한 기본 개체 내의 레코드	기본 개체 내에서 레코드가 서로 계층적으로 관련되어 있습니다. 기본 개체 내에서 다대다 관계를 정의할 수 있습니다.

Hub 콘솔에서 관계를 구성한 후 관계를 사용하여 레코드 간 일치 경로를 정의함으로써 일치 열 규칙을 구성할 수 있습니다.

스키마 개체를 정의하기 위한 요구 사항

이 섹션에서는 스키마 개체를 구성하는 데 필요한 요구 사항에 대해 설명합니다.

Hub 콘솔에서만 스키마 변경

Hub 콘솔 도구를 사용하여 모든 모델 변경이 수행되고 데이터베이스에는 직접적인 변경이 없을 경우 MDM Hub가 스키마 일관성을 유지 관리합니다. MDM Hub는 스키마를 유지 관리하는 데 필요한 모든 도구를 제공합니다.

스키마 변경 전 고려할 사항

스키마 변경 시 데이터에 대한 위험이 수반될 수 있으므로 스키마 변경은 관리 및 제어된 방식으로 수행되어야 합니다. 스키마를 변경하기 전에 변경을 계획하고 변경으로 인한 영향을 분석해야 합니다. 변경 전에는 데이터베이스를 백업해야 합니다.

스키마 변경에 쓰기 잠금을 설정해야 함

스키마를 변경하기 위해서는 쓰기 잠금이 필요합니다.

데이터베이스 개체 이름 규칙

데이터베이스 개체 이름은 24자 이하여야 합니다.

데이터베이스 개체 이름에 대해 예약된 문자열

MDM Hub는 기본 개체에 사용할 이름에 추가되는 접두사 및 접미사를 사용하는 메타데이터 개체를 생성합니다. 혼동 및 데이터 손실을 방지하려면 데이터베이스 개체 이름에 다음과 같은 문자열을 독립형 이름이나 열 이름 일부(접두사 또는 접미사)로 사용하면 안 됩니다.

_BVTB	_OPL	_TMGB	BVTX_	TCMO_	TMF_
_BVTC	_ORAW	_TMIN	BVTC_	TCRN_	TMMA_
_BVTV	_STRP	_TML0	BVTXV_	TCRO_	TMR_
BO1	_STRPT	_TMMA	CLC_	TCSN_	TPBR_
BO2	_T	_TMNX	COCS	TCSO_	TRBX_
_C	_TBKF	_TMP0	CSC_	TCVN_	TUCA_
_CL	_TBVB	_TMST	CTL	TCVO_	TUCC_
C_REPOS_	_TBVC	_TNPMA	EXP_	TCXN_	TUCF_
C_REPAR_	_TBVV	_TPMA	GG	TCXO_	TUCR_
_D	_TC0	_TPRL	HMRG	TDCC_	TUCT_
_DLT	_TC1	_TRAW	LNK	TDEL_	TUCX_
_EMI	_TDEL	_TRLG	M	TDUMP_	TUDL_
_EMO	_TEMI	_TRLT	PRL	TFK_	TUGR_
_GOV	_TEMO	_TSD	S_	TFX_	TUHM_
_HIST	_TEMP	_TSI	T_verify_	TGA_	TUID_
_HUID	_TEST	_TSNU	TBDL_	TGB_	TUK_
_HXRF	_TGA	_TSTR	TBOX_	TGB1_	TUPT_
_JOBS	_TGA1	_TUID	TBXR_	TGC_	TUTR_
_L	_TGB	_TVXR	TCBN_	TGC1_	TVXRD_
_LINK	_TGB1	_VCT	TCBO_	TGD_	TXDL_
_LMH	_TGC	_XREF	TCCN_	TGF_	TXPR_
_LMT	_TGC1	BV0_	TCCO_	TGM_	V_
_MTBM	_TMG0	BV1_	TCGN_	TGMD_	-
_MTCH	_TMG1	BV2_	TCGO_	TGV_	-
_MTFL	_TMG2	BV3_	TCHN_	TGV1_	-

_MTFU	_TMG3	BV5_	TCHO_	TLL	-
_MVLE	_TMGA	BVLNK_	TCMN_	TMA_	-

예약된 열 이름

사용자 정의 열 이름의 어떤 부분도 예약된 단어나 예약된 열 이름이 될 수 없습니다.

예를 들어 **LAST_UPDATE_DATE**는 예약된 열 이름이며 사용자 정의 열에 사용하면 안 됩니다. 또한 사용자 정의 열 이름의 어떤 부분에도 **LAST_UPDATE_DATE**가 포함되면 안 됩니다(예: **S_LAST_UPDATE_DATE**).

예약된 열 이름을 사용할 경우 경고 메시지가 표시됩니다. 예:

"The column physical name "XREF_LUD" is a reserved name. Reserved names cannot be used."

기본 개체 열 이름에 다음 열 이름을 일부 또는 전체 사용할 수 없습니다.

AFFECTED_LEVEL_CODE	PERIOD_REFERENCE TIME
AFFECTED_ROWID_COLUMN	PK_SRC_OBJECT
AFFECTED_ROWID_OBJECT	PKEY_SRC_OBJECT
AFFECTED_ROWID_XREF	PKEY_SRC_OBJECT1
AFFECTED_SRC_VALUE	PKEY_SRC_OBJECT2
AFFECTED_TGT_VALUE	PREFERRED_KEY_IND
AUTOLINK_IND	PROMOTE_IND
AUTOMERGE_IND	PUT_UPDATE_MERGE_IND
CASE_INSENSITIVE_INDX_IND	REPOINTED_IND
CHECKIN_ID	ROOT_IND
CONSOLIDATION_IND	ROU_IND
CREATE_DATE	ROWID
CREATOR	ROWID_GROUP
CROSS_PERIOD_ID	ROWID_JOB
CTL_ROWID_OBJECT	ROWID_KEY_CONSTRAINT
DATA_COUNT	ROWID_MATCH_RULE
DATA_ROW	ROWID_OBJECT
DELETED_BY	ROWID_OBJECT1
DELETED_DATE	ROWID_OBJECT2
DELETED_IND	ROWID_OBJECT_MATCHED

DEP_PKEY_SRC_OBJECT	ROWID_OBJECT_NUM
DEP_ROWID_SYSTEM	ROWID_PERIOD
DIRTY_IND	ROWID_SYSTEM
ERROR_DESCRIPTION	ROWID_TASK
FILE_NAME	ROWID_USER
FIRSTV	ROWID_XREF
GENERATED_XREF	ROWID_XREF1
GROUP_ID	ROWID_XREF2
GVI_NO	ROWKEY
HIST_CREATE_DATE	RULE_NO
HIST_UPDATE_DATE	SDSRCFLG
HSI_ACTION	SEQ
HUB_STATE_IND	SOURCE_KEY
INTERACTION_ID	SOURCE_NAME
INVALID_IND	SRC_LUD
LAST_ROWID_SYSTEM	SRC_ROWID
LAST_UPDATE_DATE	SRC_ROWID_OBJECT
LASTV	SRC_ROWID_XREF
LOST_VALUE	SSA_DATA
MATCH_REVERSE_IND	SSA_KEY
MERGE_DATE	STRIP_DATE
MERGE_OPERATION_ID	TGT_ROWID_OBJECT
MERGE_UPDATE_NULL_ALLOW_IND	TIMELINE_ACTION
MERGE_VIA_UNMERGE_IND	TOTAL_BO_IND
MRG_SRC_ROWID_OBJECT	TREE_UNMERGE_IND
MRG_TGT_ROWID_OBJECT	UNLINK_IND
NULL_INDICATOR_BITMAP	UNMERGE_DATE
NUM_CONTR	UNMERGE_IND

OLD_AFFECTED	UNMERGE_OPERATION_ID
ONLINE_IND	UPDATED_BY
ORIG_ROWID_OBJECT	WIN_VALUE
ORIG_ROWID_OBJECT_MATCHED	XREF_LUD
ORIG_TGT_ROWID_OBJECT	-

랜딩 테이블 열 이름에 다음 열 이름을 일부 또는 전체 사용할 수 없습니다.

DEP_PKEY_SRC_OBJECT	ROWID_JOB
ERROR_DESCRIPTION	SRC_ROWID
PKEY_SRC_OBJECT	VERSION_SEQ
ROWID	ONLINE_IND

다음 단어를 열 이름으로 사용할 수 없습니다.

CLASS	ROWID
-------	-------

Oracle 환경의 예약어

Oracle 환경에서는 다음 단어를 열 이름으로 사용할 수 없습니다.

ABORT	DIGITS	MAXLOGMEMBERS	ROLES
ACCEPT	DISABLE	MAXTRANS	ROLLBACK
ACCESS	DISMOUNT	MAXVALUE	ROW
ADD	DISPOSE	MIN	ROWID
ADMIN	DISTINCT	MINEXTENTS	ROWLABEL
AFTER	DO	MINUS	ROWNUM
ALL	DOUBLE	MINVALUE	ROWS
ALLOCATE	DROP	MLSLABEL	ROWTYPE
ALTER	DUMP	MOD	RUN
ANALYZE	EACH	MODE	SAVEPOINT
AND	ELSE	MODIFY	SCHEMA
ANY	ELSIF	MOUNT	SCN

ARCHIVE	ENABLE	NATURAL	SECTION
ARCHIVELOG	END	NEXT	SEGMENT
ARRAY	ENTRY	NEXTVAL	SELECT
ARRAYLEN	ESCAPE	NOARCHIVELOG	SEPARATE
AS	EVENTS	NOAUDIT	SEQUENCE
ASC	EXCEPT	NOCACHE	SESSION
ASSERT	EXCEPTION	NOCOMPRESS	SET
ASSIGN	EXCEPTIONS	NOCYCLE	SHARE
AT	EXCEPTION _INIT	NOMAXVALUE	SHARED
AUDIT	EXCLUSIVE	NOMINVALUE	SIZE
AUTHORIZATION	EXEC	NONE	SMALLINT
AVG	EXECUTE	NOORDER	SNAPSHOT
BACKUP	EXISTS	NORESETLOGS	SOME
BASE_TABLE	EXIT	NORMAL	SORT
BECOME	EXPLAIN	NOSORT	SPACE
BEFORE	EXTENT	NOT	SQL
BEGIN	EXTERNALLY	NOTFOUND	SQLBUF
BETWEEN	FALSE	NOWAIT	SQLCODE
BINARY_INTEGER	FETCH	NULL	SQLERRM
BLOB	FILE	NUMBER	SQLERROR
BLOCK	FLUSH	NUMBER_BASE	SQLSTATE
BODY	FOR	NUMERIC	START
BOOLEAN	FORCE	OF	STATEMENT
BY	FOREIGN	OFF	STATEMENT_ID
CACHE	FORM	OFFLINE	STATISTICS
CANCEL	FORTRAN	OLD	STDDEV
CASCADE	FOUND	ON	STOP
CASE	FREELIST	ONLINE	STORAGE

CHANGE	FREELISTS	ONLY	SUBTYPE
CHAR	FROM	OPEN	SUCCESSFUL
CHARACTER	FUNCTION	OPTIMAL	SUM
CHAR_BASE	GENERIC	OPTION	SWITCH
CHECK	GO	OR	SYNONYM
CHECKPOINT	GOTO	ORDER	SYSDATE
CLOB	GRANT	OTHERS	SYSTEM
CLOSE	GROUP	OUT	TABAUTH
CLUSTER	GROUPS	OWN	TABLE
CLUSTERS	HAVING	PACKAGE	TABLES
COBOL	IDENTIFIED	PARALLEL	TABLESPACE
COLAUTH	IF	PARTITION	TASK
COLUMN	IMMEDIATE	PCTFREE	TEMPORARY
COLUMNS	IN	PCTINCREASE	TERMINATE
COMMENT	INCLUDING	PCTUSED	THEN
COMMIT	INCREMENT	PLAN	THREAD
COMPILE	INDEX	PLI	TIME
COMPRESS	INDEXES	POSITIVE	TO
CONNECT	INDICATOR	PRAGMA	TRACING
CONSTANT	INITIAL	PRECISION	TRANSACTION
CONSTRAINT	INTRANS	PRIMARY	TRIGGER
CONSTRAINTS	INSERT	PRIOR	TRIGGERS
CONTENTS	INSTANCE	PRIVATE	TRUE
CONTINUE	INT	PRIVILEGES	TRUNCATE
CONTROLFILE	INTEGER	PROCEDURE	UID
COUNT	INTERSECT	PROFILE	UNDER
CRASH	INTO	PUBLIC	UNION
CREATE	IS	QUOTA	UNIQUE

CURRENT	KEY	RAISE	UNLIMITED
CURRVAL	LANGUAGE	RANGE	UNTIL
CURSOR	LAYER	RAW	UPDATE
CYCLE	LEVEL	READ	USE
DATABASE	LIKE	REAL	USER
DATAFILE	LIMITED	RECORD	USING
DATA_BASE	LINK	RECOVER	VALIDATE
DATE	LISTS	REFERENCES	VALUES
DBA	LOCK	REFERENCING	VARCHAR
DEBUGOFF	LOGFILE	RELEASE	VARCHAR 2
DEBUGON	LONG	REMR	VARIANCE
DEC	LOOP	RENAME	VIEW
DECIMAL	MANAGE	RESETLOGS	VIEWS
DECLARE	MANUAL	RESOURCE	WHEN
DEFAULT	MAX	RESTRICTED	WHENEVER
DEFINITION	MAXDATAFILES	RETURN	WHERE
DELAY	MAXEXTENTS	REUSE	WHILE
DELETE	MAXINSTANCES	REVERSE	WITH
DELTA	MAXLOGFILES	REVOKE	WORK
DESC	MAXLOGHISTORY	ROLE	WRITE
-	-	-	XOR

IBM DB2 환경의 예약어

IBM DB2 환경에서는 다음 단어를 열 이름으로 사용할 수 없습니다.

ABSOLUTE	DESTROY	LOCALTIME	SAVE
ACTION	DESTRUCTOR	LOCALTIMESTAMP	SAVEPOINT
ADA	DETERMINISTIC	LOCATOR	SCHEMA
ADD	DIAGNOSTICS	LOWER	SCOPE

ADMIN	DICTIONARY	MAP	SCROLL
AFTER	DISCONNECT	MATCH	SEARCH
AGGREGATE	DISK	MAX	SECOND
ALIAS	DISTINCT	MDMALIAS	SECTION
ALL	DISTRIBUTED	MDMNODE	SELECT
ALLOCATE	DOMAIN	MIN	SEQUENCE
ALTER	DOUBLE	MINUTE	SESSION
AND	DROP	MODIFIES	SESSION_USER
ANY	DUMMY	MONTH	SET
ARE	DUMP	NAMES	SETS
ARRAY	DYNAMIC	NATIONAL	SETUSER
AS	EACH	NATURAL	SHUTDOWN
ASC	ELSE	NCHAR	SIZE
ASSERTION	END	NCLOB	SMALLINT
AT	END-EXEC	NEXT	SOME
AUTHORIZATION	EQUALS	NO	SPACE
AVG	ERRLVL	NOCHECK	SPECIFIC
BACKUP	ESCAPE	NONCLUSTERED	SPECIFICTYPE
BEFORE	EVERY	NONE	SQL
BEGIN	EXCEPT	NOT	SQLCA
BETWEEN	EXCEPTION	NULL	SQLCODE
BINARY	EXEC	NULLIF	SQLERROR
BIT	EXECUTE	NUMERIC	SQLEXCEPTION
BIT_LENGTH	EXISTS	OBJECT	SQLSTATE
BLOB	EXIT	OCTET_LENGTH	SQLWARNING
BOOLEAN	EXTERNAL	OF	START
BOTH	EXTRACT	OFF	STATEMENT
BREADTH	FALSE	OFFSETS	STATIC

BREAK	FETCH	OLD	STATISTICS
BROWSE	FILE	ON	STRUCTURE
BULK	FILLFACTOR	ONLY	SUBSTRING
BY	FIRST	OPEN	SUM
CALL	FOR	OPENDATASOURCE	SYSTEM_USER
CASCADE	FOREIGN	OPENQUERY	TABLE
CASCADED	FORTRAN	OPENROWSET	TEMPORARY
CASE	FOUND	OPENXML	TERMINATE
CAST	FREE	OPERATION	TEXTSIZE
CATALOG	FREETEXT	OPTION	THAN
CHAR	FREETEXTABLE	OR	THEN
CHARACTER	FROM	ORDER	TIME
CHARACTER_LENGTH	FULL	ORDINALITY	TIMESTAMP
CHAR_LENGTH	FUNCTION	OUTER	TIMEZONE_HOUR
CHECK	GENERAL	OUTPUT	TIMEZONE_MINUTE
CHECKPOINT	GET	OVER	TO
CLASS	GLOBAL	OVERLAPS	TOP
CLOB	GO	PAD	TRAILING
CLOSE	GOTO	PARAMETER	TRAN
CLUSTERED	GRANT	PARAMETERS	TRANSACTION
COALESCE	GROUP	PARTIAL	TRANSLATE
COLLATE	GROUPING	PASCAL	TRANSLATION
COLLATION	HAVING	PATH	TREAT
COLUMN	HOLDLOCK	PERCENT	TRIGGER
COMMIT	HOST	PLAN	TRIM
COMPLETION	HOURL	POSITION	TRUE
COMPUTE	IDENTITY	POSTFIX	TRUNCATE
CONNECT	IDENTITYCOL	PRECISION	TSEQUEL

CONNECTION	IDENTITY_INSERT	PREFIX	UNDER
CONSTRAINT	IF	PREORDER	UNION
CONSTRAINTS	IGNORE	PREPARE	UNIQUE
CONSTRUCTOR	IMMEDIATE	PRESERVE	UNKNOWN
CONTAINS	IN	PRIMARY	UNNEST
CONTAINSTABLE	INCLUDE	PRINT	UPDATE
CONTINUE	INDEX	PRIOR	UPDATETEXT
CONVERT	INDICATOR	PRIVILEGES	UPPER
CORRESPONDING	INITIALIZE	PROC	USAGE
COUNT	INITIALLY	PROCEDURE	USE
CREATE	INNER	PUBLIC	USER
CROSS	INOUT	RAISERROR	USING
CUBE	INPUT	READ	VALUE
CURRENT	INSENSITIVE	READS	VALUES
CURRENT_DATE	INSERT	READSTEXT	VARCHAR
CURRENT_ROLE	INT	REAL	VARIABLE
CURRENT_TIME	INTEGER	RECONFIGURE	VARYING
CURRENT_TIMESTAMP	INTERSECT	RECURSIVE	VIEW
CURRENT_USER	INTERVAL	REF	WAITFOR
CURSOR	INTO	REFERENCES	WHEN
CYCLE	IS	REFERENCING	WHENEVER
DATA	ISOLATION	RELATIVE	WHERE
DATABASE	ITERATE	REPLICATION	WHILE
DATE	JOIN	RESTORE	WITH
DAY	KEY	RESTRICT	WITHOUT
DBCC	KILL	RESULT	WORK
DEALLOCATE	LANGUAGE	RETURN	WRITE
DEC	LARGE	RETURNS	WRITETEXT

DECIMAL	LAST	REVOKE	YEAR
DECLARE	LATERAL	RIGHT	ZONE
DEFAULT	LEADING	ROLE	
DEFERRABLE	LEFT	ROLLBACK	
DEFERRED	LESS	ROLLUP	
DENY	LEVEL	ROUTINE	
DEPTH	LIKE	ROW	
DEREF	LIMIT	ROWCOUNT	
DESC	LINENO	ROWGUIDCOL	
DESCRIBE	LOAD	ROWS	
DESCRIPTOR	LOCAL	RULE	

Microsoft SQL Server 환경의 예약어

Microsoft SQL Server 환경에서는 다음 단어를 열 이름으로 사용할 수 없습니다.

ABSOLUTE	DESC	LEVEL	ROLLUP
ACTION	DESCRIBE	LIKE	ROUTINE
ADA	DESCRIPTOR	LIMIT	ROW
ADD	DESTROY	LINENO	ROWCOUNT
ADMIN	DESTRUCTOR	LOAD	ROWGUIDCOL
AFTER	DETERMINISTIC	LOCAL	ROWS
AGGREGATE	DIAGNOSTICS	LOCALTIME	RULE
ALIAS	DICTIONARY	LOCALTIMESTAMP	SAVE
ALL	DISCONNECT	LOCATOR	SAVEPOINT
ALLOCATE	DISK	LOWER	SCHEMA
ALTER	DISTINCT	MAP	SCOPE
AND	DISTRIBUTED	MATCH	SCROLL
ANY	DOMAIN	MAX	SEARCH
ARE	DOUBLE	MIN	SECOND

ARRAY	DROP	MINUTE	SECTION
AS	DUMMY	MODIFIES	SELECT
ASC	DUMP	MONTH	SEQUENCE
ASSERTION	DYNAMIC	NAMES	SESSION
AT	EACH	NATIONAL	SESSION_USER
AUTHORIZATION	ELSE	NATURAL	SET
AVG	END	NCHAR	SETS
BACKUP	END-EXEC	NCLOB	SETUSER
BEFORE	EQUALS	NEXT	SHUTDOWN
BEGIN	ERRLV	NO	SIZE
BETWEEN	ESCAPE	NOCHECK	SMALLINT
BINARY	EVERY	NONCLUSTERED	SOME
BIT	EXCEPT	NONE	SPACE
BIT_LENGTH	EXCEPTION	NOT	SPECIFIC
BLOB	EXEC	NULL	SPECIFICTYPE
BOOLEAN	EXECUTE	NULLIF	SQL
BOTH	EXISTS	NUMERIC	SQLCA
BREADTH	EXIT	OBJECT	SQLCODE
BREAK	EXTERNAL	OCTET_LENGTH	SQLERROR
BROWSE	EXTRACT	OF	SQLEXCEPTION
BULK	FALSE	OFF	SQLSTATE
BY	FETCH	OFFSETS	SQLWARNING
CALL	FILE	OLD	START
CASCADE	FILLFACTOR	ON	STATEMENT
CASCADE	FIRST	ONLY	STATIC
CASE	FOR	OPEN	STATISTICS
CAST	FOREIGN	OPENDATASOURCE	STRUCTURE
CATALOG	FORTAN	OPENQUERY	SUBSTRING

CHAR	FOUND	OPENROWSET	SUM
CHARACTER	FREE	OPENXML	SYSTEM_USER
CHARACTER_LENGTH	FREETEXT	OPERATION	TABLE
CHAR_LENGTH	FREETEXTABLE	OPTION	TEMPORARY
CHECK	FROM	OR	TERMINATE
CHECKPOINT	FULL	ORDER	TEXTSIZE
CLASS	FUNCTION	ORDINALITY	THAN
CLOB	GENERAL	OUT	THEN
CLOSE	GET	OUTER	TIME
CLUSTERED	GLOBAL	OUTPUT	TIMESTAMP
COALESCE	GO	OVER	TIMEZONE_HOUR
COLLATE	GOTO	OVERLAPS	TIMEZONE_MINUTE
COLLATION	GRANT	PAD	TO
COLUMN	GROUP	PARAMETER	TOP
COMMIT	GROUPING	PARAMETERS	TRAILING
COMPLETION	HAVING	PARTIAL	TRAN
COMPUTE	HOLDLOCK	PASCAL	TRANSACTION
CONNECT	HOST	PATH	TRANSLATE
CONNECTION	HOURL	PERCENT	TRANSLATION
CONSTRAINT	IDENTITY	PLAN	TREAT
CONSTRAINTS	IDENTITYCOL	POSITION	TRIGGER
CONSTRUCTOR	IDENTITY_INSERT	POSTFIX	TRIM
CONTAINS	IF	PRECISION	TRUE
CONTAINSTABLE	IGNORE	PREFIX	TRUNCATE
CONTINUE	IMMEDIATE	PREORDER	TSEQUEL
CONVERT	IN	PREPARE	UNDER
CORRESPONDING	INCLUDE	PRESERVE	UNION
COUNT	INDEX	PRIMARY	UNIQUE

CREATE	INDICATOR	PRINT	UNKNOWN
CROSS	INITIALIZE	PRIOR	UNNEST
CUBE	INITIALLY	PRIVILEGES	UPDATE
CURRENT	INNER	PROC	UPDATETEXT
CURRENT_DATE	INOUT	PROCEDURE	UPPER
CURRENT_ROLE	INPUT	PUBLIC	USAGE
CURRENT_TIME	INSENSITIVE	RAISERROR	USE
CURRENT_TIMESTAMP	INSERT	READ	USER
CURRENT_USER	INT	READS	USING
CURSOR	INTEGER	READTEXT	VALUE
CYCLE	INTERSECT	REAL	VALUES
DATA	INTERVAL	RECONFIGURE	VARCHAR
DATABASE	INTO	RECURSIVE	VARIABLE
DATE	IS	REF	VARYING
DAY	ISOLATION	REFERENCES	VIEW
DBCC	ITERATE	REFERENCING	WAITFOR
DEALLOCATE	JOIN	RELATIVE	WHEN
DEC	KEY	REPLICATION	WHENEVER
DECIMAL	KILL	RESTORE	WHERE
DECLARE	LANGUAGE	RESTRICT	WHILE
DEFAULT	LARGE	RESULT	WITH
DEFERRABLE	LAST	RETURN	WITHOUT
DEFERRED	LATERAL	RETURNS	WORK
DELETE	LEADING	REVOKE	WRITE
DENY	LEFT	RIGHT	WRITETEXT
DEPTH	LESS	ROLE	YEAR
DEREF	-	ROLLBACK	ZONE

예약된 특수 문자

MDM Hub는 모든 특수 문자를 허용하지만 일부 특수 문자는 Informatica Data Director에서 문제를 야기할 수 있습니다. 그러므로 다음 특수 문자는 MDM Hub 특성 표시 이름에 사용하지 마십시오.

- !(느낌표)
- '(작은따옴표)
- #(숫자 기호)
- &(앰퍼샌드)
- <(보다 작음 기호)

단, SIF CleansePut 및 Put 요청에서 &, < 특수 문자를 사용하려면 이 문자를 다음과 같은 유효한 문자열 값으로 대체하면 됩니다.

- &를 &로 대체
- <를 <로 대체

기술적 이유로 열 추가

순수하게 기술적 이유로 기본 개체에 열을 추가해야 하는 경우가 있습니다. 예를 들어, 세그먼트 일치 시 세그먼트 열을 추가해야 합니다.

혼란을 방지하려면 기술적인 이유로 기본 개체에 추가한 열과 다른 비즈니스 용도로 추가한 열을 구분하는 것이 좋습니다. 기술적인 이유로 추가한 열을 필터링하려면 특정 식별자(예: CSTM_)를 열 이름의 접두사로 사용하십시오.

스키마 관리자 시작

Hub 콘솔에서 스키마 관리자를 사용하여 스키마, 준비 테이블 및 랜딩 테이블을 정의할 수 있습니다.

또한 스키마 관리자는 일치 및 병합, 유효성 검사 및 메시지 대기열에 대한 규칙을 정의하는 데 사용됩니다.

스키마 관리자를 시작하려면

1. Hub 콘솔에서 모델 작업 영역을 확장한 후 **스키마**를 클릭합니다.
2. Hub 콘솔에 스키마 관리자가 표시됩니다.

스키마 관리자는 두 개의 창으로 구분됩니다.

창	설명
탐색 창	핵심 스키마 개체 즉 기본 개체 및 랜딩 테이블을 트리 보기로 표시합니다. 트리에서 개체를 확장하면 해당 개체에 사용 가능한 속성 그룹이 표시됩니다.
속성 창	왼쪽 창에 선택한 개체의 속성이 표시됩니다. 스키마 트리에서 아무 노드나 클릭하면 오른쪽 창에 해당 속성 페이지가 표시됩니다. 이 페이지에서 속성을 보고 편집할 수 있습니다.

연산 참조 저장소에서 테이블을 정의할 경우 스키마 관리자를 사용해야 합니다.

기본 개체 구성

MDM Hub에서는 고객, 계정 또는 제품과 같은 핵심 비즈니스 항목이 기본 개체라는 테이블에 표시됩니다. 기본 개체는 개별 항목에 대한 데이터 컬렉션을 포함하는 Hub 저장소 테이블입니다.

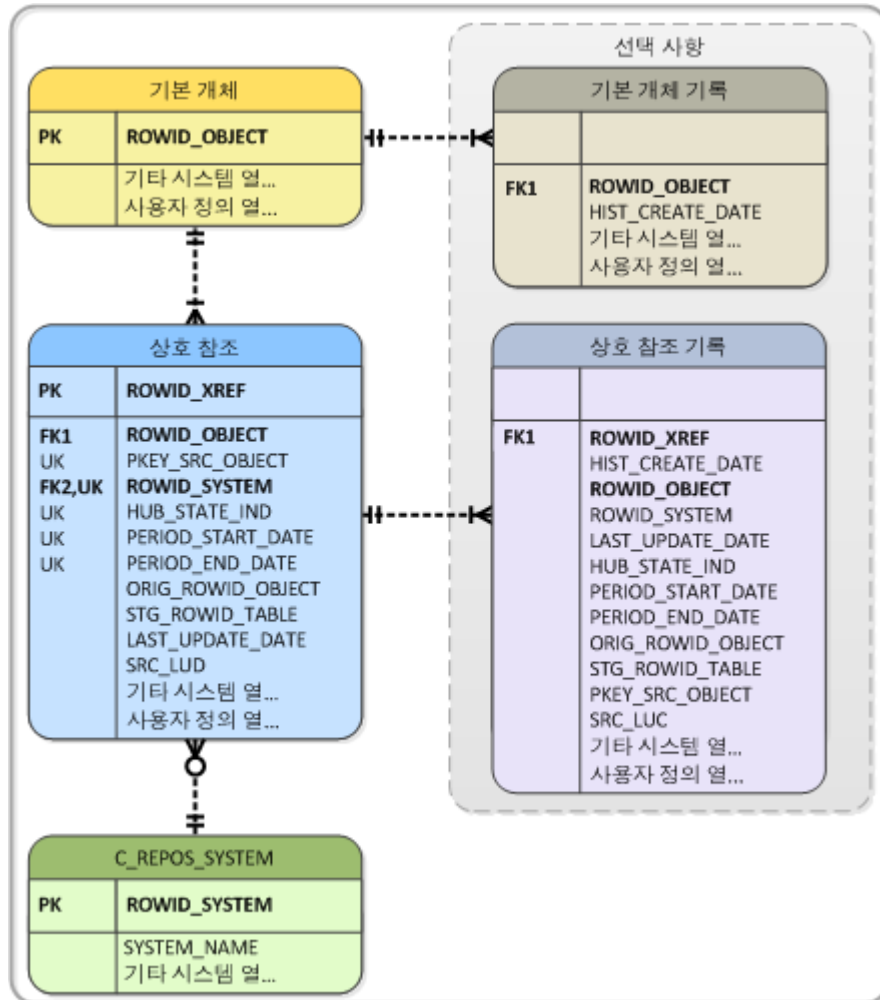
각 항목에는 단일의 마스터 레코드가 포함됩니다. 마스터 레코드는 BVT(최선의 진실)를 나타냅니다. 개별 항목 별로 마스터 레코드에 통합해야 하는 여러 버전의 진실을 포함하는 레코드가 기본 개체에 있을 수도 있습니다. 통합은 중복 레코드를 모든 소스 레코드에서 가장 신뢰할 수 있는 셀 값이 포함된 하나의 통합된 레코드에 병합하는 프로세스입니다.

기본 개체를 정의하려면 Hub 콘솔에서 스키마 관리자를 사용합니다. 기본 개체를 데이터베이스에서 바로 구성할 수는 없습니다.

스키마를 변경하는 경우 메타데이터 유효성 검사에서 경고가 생성되고 MDM Hub가 C_REPOS_MET_VALID_RESULT 테이블에 항목을 추가합니다.

Hub 저장소의 기본 개체와 다른 테이블 간 관계

다음 그림은 Hub 저장소에서 기본 개체와 다른 테이블과의 관계를 보여 줍니다.



기본 개체 정의 프로세스 개요

스키마 관리자를 사용하여 기본 개체를 정의할 수 있습니다.

1. 스키마 관리자를 사용하여 기본 개체 테이블을 생성합니다.
스키마 관리자가 자동으로 시스템 열을 추가합니다.
2. 비즈니스 데이터를 포함하는 사용자 정의 열을 추가합니다.
참고: 열 이름은 26자를 초과할 수 없습니다.
3. 열 속성을 구성하는 동안 여러 소스 시스템이 같은 셀에 대해 서로 다른 값을 제공할 경우 가장 신뢰할 수 있는 값을 결정하기 위해 트러스트를 사용하는 열을 지정합니다.
4. 기본 개체의 경우 소스 시스템마다 하나의 준비 테이블을 생성합니다. 각 준비 테이블의 경우 포함해야 하는 기본 개체 열을 선택합니다.
5. 소스 시스템에서 데이터를 저장하는 데 필요한 랜딩 테이블을 생성합니다.
6. 랜딩 테이블을 준비 테이블에 매핑합니다.
열에 데이터 정리가 필요한 경우 매핑에서 정리 함수를 지정합니다.
각 준비 테이블은 하나의 랜딩 테이블(중재 정리 함수 포함)에서 가져와야 합니다. 단 같은 랜딩 테이블에서 둘 이상의 준비 테이블에 데이터를 제공할 수 있습니다. 랜딩 테이블의 기본 키 열을 준비 테이블의 PKEY_SRC_OBJECT 열에 매핑합니다.
7. ETL 도구나 기타 몇 개의 프로세스를 사용하여 데이터로 각 랜딩 테이블을 채웁니다.

기본 개체 열

기본 개체에는 시스템 열 및 사용자 정의 열이 있습니다. 시스템 열은 스키마 관리자에서 생성하고 유지 관리하는 열입니다. 사용자 정의 열은 사용자가 추가한 열입니다.

다음 테이블에서는 기본 개체 시스템 열에 대해 설명합니다.

실제 이름	MDM Hub 데이터 유형(크기)	설명
ROWID_OBJECT	CHAR(14)	기본 키입니다. Informatica MDM Hub는 기본 개체에 레코드가 삽입되면 고유한 값을 할당합니다.
CREATOR	VARCHAR (50)	레코드를 생성한 사용자 또는 프로세스입니다.
CREATE_DATE	TIMESTAMP	레코드가 생성된 날짜입니다.
UPDATED_BY	VARCHAR (50)	레코드에 대해 가장 최근의 업데이트를 담당한 사용자 또는 프로세스입니다.
LAST_UPDATE_DATE	TIMESTAMP	레코드의 셀에 대한 가장 최근의 업데이트 날짜입니다.

실제 이름	MDM Hub 데이터 유형(크기)	설명
CONSOLIDATION_IND	INT	해당 레코드의 통합 상태를 나타내는 정수 값입니다. 유효한 값은 다음과 같습니다. <ul style="list-style-type: none"> - 1 = 고유. 레코드가 BVT(최선의 진실)를 나타냅니다. - 2 = 통합 준비 완료. - 3 = 일치 준비 완료. 해당 레코드는 현재 실행 중인 일치 프로세스의 일치 후보입니다. - 4 = 일치에 사용 가능. 해당 레코드는 새 레코드이며 일치 프로세스를 거쳐야 합니다. - 9 = 대기 중. 데이터 스튜어드가 해당 레코드를 대기 중 상태로 전환했습니다.
DELETED_IND	INT	나중에 사용하기 위해 예약된 열입니다.
DELETED_BY	VARCHAR (50)	나중에 사용하기 위해 예약된 열입니다.
DELETED_DATE	TIMESTAMP	나중에 사용하기 위해 예약된 열입니다.
LAST_ROWID_SYSTEM	CHAR(14)	기본 개체 레코드의 가장 최근에 처리된 셀에 대한 업데이트를 담당하는 시스템의 식별자입니다. LAST_ROWID_SYSTEM 은 C_REPOS_SYSTEM 테이블의 ROWID_SYSTEM 열을 참조하는 외래 키입니다.
DIRTY_IND	INT	DIRTY_IND 시스템 열은 더 이상 사용되지 않습니다. 대신 MDM Hub는 더티 테이블이라는 시스템 테이블을 사용하여 토큰화하는 데 필요한 레코드를 확인합니다.
INTERACTION_ID	INT	상태 사용 기본 개체용입니다. INTERACTION_ID는 보류 중인 교차 참조 레코드를 원래 교차 참조 레코드와 동일한 프로세스에 속하지 않는 업데이트로부터 보호하는 상호 작용 식별자입니다.
HUB_STATE_IND	INT	상태 사용 기본 개체용입니다. 해당 레코드의 상태를 나타내는 정수 값입니다. 유효한 값은 다음과 같습니다. <ul style="list-style-type: none"> - 0=보류 중 - 1=활성 - -1=삭제됨 기본값은 1입니다.
CM_DIRTY_IND	INT	ZDT(Zero Downtime) 프로세스를 사용하여 업데이트할 때 데이터가 변경되었는지 여부를 나타냅니다.

참고: sourceKey 및 ROWID_OBJECT와 같은 시스템 열의 값에는 ~ 및 와 같은 특수 문자가 포함되면 안 됩니다.

교차 참조 테이블

이 섹션에서는 Hub 저장소의 교차 참조 테이블에 대해 설명합니다.

교차 참조 테이블 정보

각 기본 개체에는 한 개의 교차 참조(XREF) 테이블이 연결되어 있습니다. XREF 테이블에서는 연계와 다른 유효 기간의 레코드 버전(시간 표시 막대가 활성화되어 있는 경우)을 추적합니다.

Informatica MDM Hub에서는 기본 개체를 생성할 때 교차 참조 테이블이 자동으로 생성됩니다. Informatica MDM Hub에서는 교차 참조 테이블을 사용하여 모든 소스 시스템 식별자를 적절한 ROWID_OBJECT 값으로 변환합니다.

교차 참조 테이블의 레코드

교차 참조 테이블의 각 행은 소스 시스템의 개별 레코드를 나타냅니다. CRM 시스템과 ERP 시스템 모두에서 전 화 번호를 가져오는 경우와 같이 여러 소스에서 단일 열의 데이터를 제공하는 경우 교차 참조 테이블에는 각 소스 시스템의 개별 레코드가 포함됩니다. 시간 표시 막대가 사용되는 기본 개체의 경우 교차 참조 테이블에는 기본 개체 레코드의 각 버전에 해당하는 개별 레코드도 포함됩니다. 각 기본 개체 레코드에는 하나 이상의 교차 참조 레코드가 연결되어 있습니다.

교차 참조 레코드에는 다음이 포함됩니다.

- 레코드를 제공한 소스 시스템의 식별자
- 소스 시스템에 있는 해당 레코드의 기본 키 값
- 해당 시스템에서 제공한 가장 최신 셀 값
- 레코드의 원래 ROWID_OBJECT 값
- 레코드의 원래 GBID 값(해당되는 경우)
- 레코드 유효 기간의 시작 날짜 및 종료 날짜(해당되는 경우)

로드 프로세스 및 교차 참조 테이블

로드 프로세스에서는 교차 참조 테이블을 채웁니다. 삽입형 로드 중에는 교차 참조 테이블에 새 레코드가 추가되고, 업데이트형 로드 중에는 영향을 받는 교차 참조 레코드에 변경 내용이 기록됩니다.

데이터 스튜어드 도구 및 교차 참조 테이블

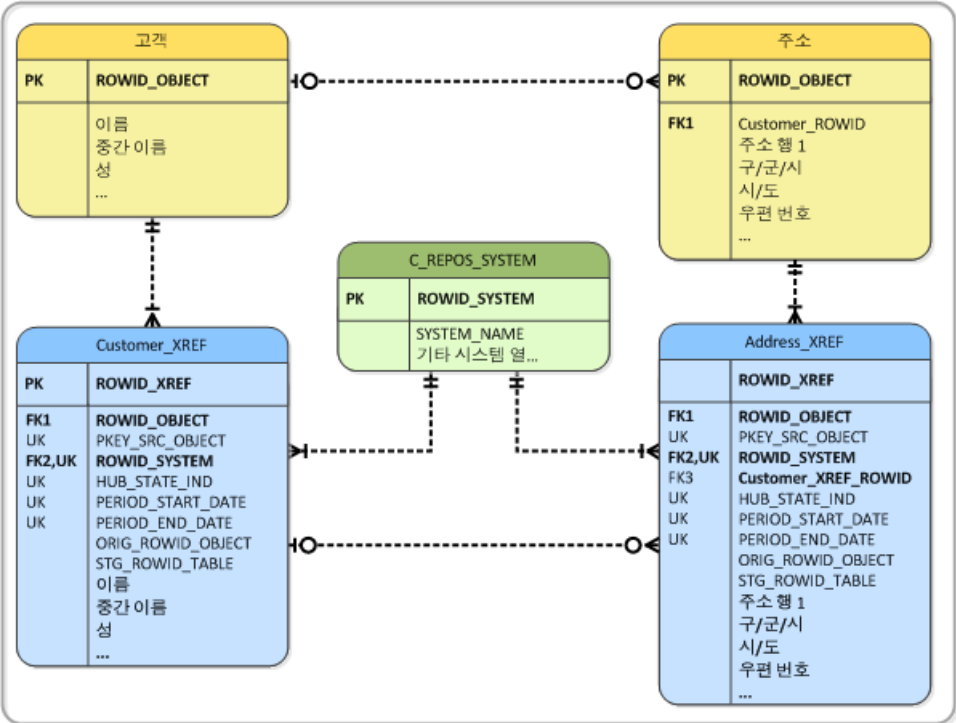
교차 참조 레코드는 병합 관리자에 표시되며 데이터 관리자를 사용하여 수정할 수 있습니다. 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

기본 개체와 교차 참조 테이블 간의 관계

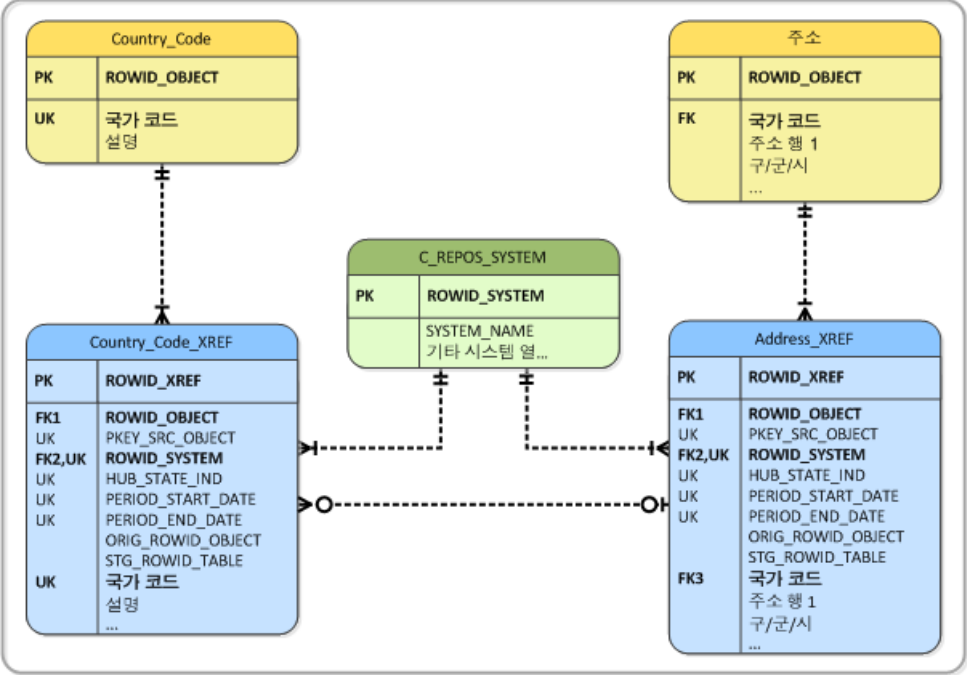
기본 개체 및 교차 참조 테이블은 다음과 같은 두 가지 방법으로 연결할 수 있습니다.

- ROWID 사용
- 고유 키 사용

다음 그림은 ROWID를 기준으로 하는 C_REPOS_SYSTEM과 기본 개체, 교차 참조 테이블 간 관계에 대한 예를 보여 줍니다.



다음 그림은 고유 키를 기준으로 하는 C_REPOS_SYSTEM과 기본 개체, 교차 참조 테이블 간 관계에 대한 예를 보여 줍니다.



교차 참조 테이블 열

교차 참조 테이블에는 다음과 같은 시스템 열이 있습니다.

참고: 교차 참조 테이블에는 PKEY_SRC_OBJECT 열과 ROWID_SYSTEM 열의 조합을 나타내는 고유 키가 있습니다.

실제 이름	MDM Hub 데이터 유형(크기)	설명
ROWID_XREF	NUMBER(38)	교차 참조 테이블에서 이 레코드를 고유하게 식별하는 기본 키입니다.
PKEY_SRC_OBJECT	VARCHAR2(255)	소스 시스템의 기본 키 값입니다. 여러 필드나 여러 열이 포함된 키를 단일 키 값으로 연결해야 합니다. MDM Hub 내부 정리 프로세스 또는 외부 정리 프로세스(예: ETL 도구)나 다른 데이터 로드 유틸리티를 사용하여 키를 연결합니다. 편집 결과인 관리 시스템 교차 참조 레코드의 경우 PKEY_SRC_OBJECT가 SYS0:<ROWID_OBJECT>입니다. 여기서 ROWID_OBJECT는 편집된 레코드의 행 ID입니다. PKEY_SRC_OBJECT가 SYS0:<ROWID_OBJECT>이고 PUT_UPDATE_MERGE_IND가 1이면 MDM Hub가 병합 해제 중에 교차 참조 레코드를 삭제합니다.
ROWID_SYSTEM	CHAR(14)	ORS를 채울 수 있는 각 소스 시스템의 MDM Hub 식별자 및 설명을 저장하는 MDM Hub 리포지토리 테이블 C_REPOS_SYSTEM에 대한 외래 키입니다.
ROWID_OBJECT	CHAR(14)	기본 개체에 대한 외래 키입니다. MDM Hub가 연결된 기본 개체 레코드에 ROWID_OBJECT를 할당합니다.
ORIG_ROWID_OBJECT	CHAR(14)	교차 참조 레코드의 원래 ROWID_OBJECT입니다.
GBID_Column_Name_GOV	VARCHAR 또는 INT	원래 글로벌 식별자 값입니다.
STG_ROWID_TABLE	CHAR(14)	MDM Hub가 교차 참조 레코드를 로드할 때 해당 조회 구성이 사용된 준비 테이블의 이름입니다.
S_Foreign_Key_Column_Name	VARCHAR(255)	외래 키 열을 조회하는 데 사용되는 값이 포함된 새도 열입니다.
SRC_LUD	TIMESTAMP	소스를 마지막으로 업데이트한 날짜입니다. MDM Hub는 소스 시스템에서 교차 참조 레코드로 업데이트를 보낼 때 SRC_LUD를 업데이트합니다.
CREATOR	VARCHAR2(50)	교차 참조 레코드의 작성을 담당하는 사용자 또는 프로세스입니다.
CREATE_DATE	TIMESTAMP	교차 참조 레코드가 작성된 날짜입니다.
UPDATED_BY	VARCHAR2(50)	교차 참조 레코드에 대한 가장 최근의 업데이트를 담당한 사용자 또는 프로세스입니다.

실제 이름	MDM Hub 데이터 유형(크기)	설명
LAST_UPDATE_DATE	TIMESTAMP	교차 참조 레코드의 셀에 대한 가장 최근의 업데이트 날짜입니다. MDM Hub는 로드 프로세스 및 통합 프로세스 중에 LAST_UPDATE_DATE를 업데이트합니다(해당하는 경우).
DELETED_IND	NUMBER(38)	나중에 사용하기 위해 예약되어 있습니다.
DELETED_BY	VARCHAR2(50)	나중에 사용하기 위해 예약되어 있습니다.
DELETED_DATE	TIMESTAMP	나중에 사용하기 위해 예약되어 있습니다.
PERIOD_START_DATE	TIMESTAMP	레코드 유효 기간의 시작 날짜입니다. PERIOD_START_DATE 값은 시간 표시 막대가 활성화된 기본 개체의 교차 참조 레코드에 필요합니다.
PERIOD_END_DATE	TIMESTAMP	레코드 유효 기간의 끝 날짜입니다. PERIOD_END_DATE 값은 시간 표시 막대가 활성화된 기본 개체의 교차 참조 레코드에 필요합니다.
PUT_UPDATE_MERGE_IND	NUMBER(38)	값이 1이면 Put API 호출에서 Put API 요청에 ROWID_OBJECT 값을 지정하여 해당 특정 레코드를 업데이트했음을 나타냅니다.
INTERACTION_ID	NUMBER(38)	상태 사용 기본 개체용입니다. 보류 중인 교차 참조 레코드를 원래 교차 참조 레코드와 동일하지 않은 프로세스에 속하는 업데이트로부터 보호하는 데 사용되는 상호 작용 식별자입니다.
HUB_STATE_IND	NUMBER(38)	상태 사용 기본 개체용입니다. 레코드의 상태를 나타내는 정수 값입니다. 유효한 값은 다음과 같습니다. - 0 = 보류 중 - 1 = 활성 - -1 = 삭제됨 기본값은 1입니다.
PROMOTE_IND	NUMBER(38)	상태 사용 기본 개체용입니다. 승격 상태를 나타내는 정수 값입니다. 승격 프로세스에서는 PROMOTE_IND를 사용하여 레코드를 ACTIVE 상태로 승격할지 여부를 결정합니다. 유효한 값은 다음과 같습니다. - 0 = 레코드를 승격하지 않습니다. - 1 = 이 레코드를 ACTIVE로 승격합니다. 승격 프로세스에서 레코드를 승격하지 않은 경우 MDM Hub는 PROMOTE_IND를 0으로 변경하지 않습니다.

기록 테이블

기록 테이블은 Hub 저장소에 있습니다.

기본 개체에 대해 기록을 활성화하면 Informatica MDM Hub가 기본 개체의 기록 테이블 및 교차 참조 테이블을 유지 관리합니다. Informatica MDM Hub는 기록 테이블에 병합 및 병합 해제 기록, 사전 정리된 데이터 기록, 기본 개체 기록 및 교차 참조 기록과 같은 자세한 변경 내용 추적 옵션을 제공합니다.

자동 병합 작업을 실행하면 Informatica MDM Hub에서 각 병합 작업에 대해 기록 테이블에 레코드를 생성합니다. 마찬가지로 Informatica MDM Hub가 하위 기본 개체의 외래 키를 업데이트할 때마다 해당 기록 테이블에 레코드가 삽입됩니다.

기본 개체 기록 테이블

기록 사용 기본 개체에는 기본 개체의 데이터 변경 사항에 대한 기록 정보를 포함하는

C_baseObjectName_HIST라는 단일 기록 테이블이 있습니다. 기본 개체에서 레코드가 추가되거나 업데이트될 때마다 해당 이벤트를 캡처하기 위해 새 레코드가 기본 개체 기록 테이블에 삽입됩니다.

교차 참조 기록 테이블

기록 사용 기본 개체에는 교차 참조 테이블의 데이터 변경 사항에 대한 기록 정보를 포함하는

C_baseObjectName_HXRF라는 단일 교차 참조 기록 테이블이 있습니다. 교차 참조 테이블에서 레코드가 변경될 때마다 해당 이벤트를 캡처하기 위해 새 레코드가 교차 참조 기록 테이블에 삽입됩니다.

기본 개체 속성

이 섹션에서는 기본 개체의 기본 속성과 고급 속성에 대해 설명합니다.

기본 개체의 기본 속성

이 섹션에서는 기본 개체의 기본 속성에 대해 설명합니다.

항목 유형

추가해야 하는 테이블 유형입니다. **기본 개체**를 선택합니다.

표시 이름

Hub 콘솔에 표시되어야 하는 기본 개체 이름입니다. 설명적인 이름을 입력합니다. 표시 이름은 67자보다 작아야 합니다.

실제 이름

데이터베이스에서 사용되는 실제 테이블 이름입니다. Informatica MDM Hub에서는 사용자가 입력한 표시 이름을 기준으로 테이블의 실제 이름을 제안합니다. 예약된 이름 접미사는 사용하지 않아야 합니다.

데이터 테이블스페이스

데이터 테이블스페이스의 이름입니다. 읽기 전용.

인덱스 테이블스페이스

인덱스 테이블스페이스의 이름입니다. 읽기 전용.

설명

기본 개체에 대한 간략한 설명입니다.

기록 활성화

기본 개체에 대한 기록을 활성화할지 여부를 지정합니다. 활성화된 경우 Informatica MDM Hub에서는 해당 기본 개체에 대해 삽입, 업데이트 또는 삭제된 레코드의 로그를 보관합니다. 기록 테이블의 정보는 감사 목적으로 사용할 수 있습니다.

시간 표시 막대

기본 개체에 대한 시간 표시 막대를 활성화할지 여부를 지정합니다. 기본값은 **시간 표시 막대 없음**입니다. 시간 표시 막대를 활성화하려면 **동적 시간 표시 막대**를 선택합니다. 시간 표시 막대가 활성화된 경우 MDM Hub에서는 항목 및 관계를 포함한 기본 개체 레코드의 버전을 관리합니다.

비연속 유효 기간 허용

레코드에 비연속 유효 기간을 사용합니다. 레코드에 비연속 유효 기간을 지정할 수 있습니다. 레코드에 비연속 유효 기간을 사용할 수 없습니다. 기본적으로 이 속성은 비활성화되어 있습니다.

기본 개체의 고급 속성

이 섹션에서는 기본 개체의 고급 속성에 대해 설명합니다.

전체 토큰화 비율

변경된 레코드의 백분율이 전체 토큰화 비율보다 높으면 전체 다시 토큰화가 수행됩니다. 토큰화할 레코드의 수가 이 임계값을 초과하면 Informatica MDM Hub에서는 일치 키 테이블에서 다시 토큰화해야 하는 레코드를 삭제하고 해당 레코드의 토큰을 계산한 다음 이를 일치 키 테이블에 다시 삽입합니다. 기본값은 60입니다.

참고: 삭제 프로세스는 느리게 진행될 수 있습니다. 그러나 처리 서버의 속도가 빠르고 처리 서버와 데이터베이스 서버 간의 네트워크 연결 속도도 빠른 경우에는 훨씬 더 낮은 토큰화 임계값(예: 10%)으로 테스트할 수 있습니다. 이렇게 하면 성능이 향상되는지 여부를 확인할 수 있습니다.

제약 조건이 비활성화되도록 허용

초기 로드/업데이트 중이나 실시간 동시 액세스가 없는 경우 성능 향상을 위해 기본 개체에 대한 참조 무결성 제약 조건을 비활성화할 수 있습니다. 기본값은 제약 조건이 비활성화됨을 나타내는 1입니다.

중복 일치 임계값

이 매개 변수는 초기 데이터 로드 중 "중복 데이터에 대한 일치" 작업에만 사용됩니다. 기본값은 0입니다.

이 기능을 활성화하려면 이 값을 2 이상으로 설정해야 합니다. 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

로드 일괄 처리 크기

로드 프로세스에서는 기본 개체의 레코드를 일괄적으로 삽입 및 업데이트합니다. 로드 일괄 처리 크기는 각 일괄 처리 주기 중에 로드할 레코드 수를 지정합니다(기본값: 1000000).

최대 일치 경과 시간(분)

이 속성은 일치 규칙을 실행할 때의 제한 시간(분)을 지정합니다. 시간 제한에 도달하면 일치 프로세스가 종료됩니다(일치 규칙이 실행될 때마다 수동으로 또는 일괄 작업을 통해). 일치 프로세스가 일괄 작업의 일부로 실행된 경우 시스템은 다음 일치 프로세스를 진행합니다. 이 일치 프로세스가 단일 일치 프로세스인 경우에는 더 이상 진행되지 않습니다. 기본값은 20입니다. 일치 규칙 및 데이터가 매우 복잡한 경우에만 이 값을 늘리십시오. 일반적으로 규칙은 기본값인 20분 내에 완료될 수 있습니다.

병렬도

이 속성은 기본 개체 테이블 및 해당 관련 테이블에 대해 설정되는 병렬도를 지정합니다. 이 속성은 모든 일괄 처리에 영향을 주지는 않지만, 이 속성을 사용할 경우 성능이 향상될 수 있습니다. 그러나 이 속성의 사용 여부는 데이터베이스 서버 시스템의 CPU 수와 사용 가능한 메모리 양에 따라 제약됩니다. 기본값은 1입니다.

참고: Microsoft SQL Server 환경에서는 병렬도 수준을 구성할 수 없습니다.

상위 항목 병합 시 대기열에 다시 넣기

기본값은 "없음"입니다. "상위 항목 병합 시 대기열에 다시 넣기" 속성에는 다음 값 중 하나를 설정할 수 있습니다.

- **없음**
하위 기본 개체에 대해 값을 "없음"으로 설정하면 상위 항목을 병합할 때 하위 레코드의 통합 표시기가 변경되지 않습니다.
- **통합되지 않은 항목만**
하위 기본 개체에 대해 값을 "통합되지 않은 항목만"으로 설정하면 상위 항목을 병합할 때 통합 표시기가 1로 설정된 레코드만 제외하고 하위 레코드의 통합 표시기가 4로 변경됩니다. 통합 표시기가 1로 설정된 하위 레코드를 대기열에 다시 넣으려면 수동으로 "상위 항목 병합 시 대기열에 다시 넣기" 설정을 2로 설정해야 합니다.

- **ALL.**

하위 기본 개체에 대해 값을 **ALL**로 설정하면 상위 항목을 병합할 때 통합 표시기가 **1**인 레코드를 포함하여 모든 하위 레코드의 통합 표시기가 **4**로 변경되므로 모든 하위 레코드가 다시 일치될 수 있습니다.

로드 시 일치 토큰 생성

로드 프로시저가 완료된 후 토큰 생성 프로시저가 실행될 수 있도록 합니다. 종속된 하위 항목이 없는 기본 개체에 대해 로드 프로세스 후 일치 토큰 생성 기능을 활성화할 수 있습니다. 상위 항목을 토큰화하려면 먼저 종속된 하위 항목을 로드해야 합니다. 로드 프로세스를 수행해야 하는 시간이 제한되어 있으면 로드 프로세스 후 일치 토큰 생성 기능을 비활성화하십시오. 기본값은 비활성화입니다.

일치 플래그 감사 테이블

일치 플래그 감사 테이블을 생성할지 여부를 지정합니다.

- 이 속성을 활성화하면 감사 테이블(*BusinessObjectName_FMHA*)이 생성되고 병합 관리자에서 자동 병합을 위해 수동 일치 레코드를 대기열에 넣은 사용자의 사용자 ID로 채워집니다.
- 이 속성을 비활성화하면 **Updated_By** 열이 자동 병합 일괄 작업을 실행한 사람의 사용자 ID로 설정됩니다.

API 잠금 대기 간격(초)

SIF 요청이 행 수준 잠금을 획득할 때까지 대기할 최대 시간(초)을 지정합니다. **ORS**에 대해 행 수준 잠금이 활성화된 경우에만 적용됩니다.

일괄 잠금 대기 간격(초)

일괄 작업이 행 수준 잠금을 획득할 때까지 대기할 최대 시간(초)을 지정합니다. **ORS**에 대해 행 수준 잠금이 활성화된 경우에만 적용됩니다.

상태 관리 활성화

Informatica MDM Hub에서 기본 개체의 레코드에 대해 시스템 상태를 관리할지 여부를 지정합니다. 기본적으로 상태 관리는 비활성화되어 있습니다. 승인 워크플로우를 지원하기 위해 해당 기본 개체에 대한 상태 관리를 활성화하려면 이 확인란을 선택하십시오. 이 속성이 활성화된 기본 개체를 이 문서에서는 *상태 사용 기본 개체*라고 합니다.

참고: 기본 개체에 사용자 지정 쿼리가 있는 경우 해당 기본 개체에 대해 상태 관리를 비활성화하면 사용자 지정 쿼리에 **hub_state_ind**가 포함되어 있지 않더라도 항상 경고 팝업 창이 표시됩니다.

교차 참조 승격 기록 활성화

상태 사용 기본 개체의 경우, **PENDING(0)**에서 **ACTIVE(1)**로 상태가 전환된 교차 참조 레코드에 대한 승격 기록을 Informatica MDM Hub에서 유지 관리할지 여부를 지정합니다. 기본적으로 이 옵션은 비활성화되어 있습니다.

기본 개체 스타일

기본 개체 스타일은 병합 스타일입니다. 병합 스타일 기본 개체를 MDM Hub의 일치 및 병합 기능과 함께 사용할 수 있습니다. 이 옵션은 기본적으로 선택되어 있습니다.

조회 표시기

MDM Hub용 Informatica Data Director에서 값이 검색되는 방식을 지정합니다.

- 이 속성이 활성화되어 있으면 Informatica Data Director에서 조회 값이 포함된 드롭다운 목록이 표시됩니다.
- 이 속성이 비활성화되어 있으면 Informatica Data Director에서 사용자에게 데이터 테이블의 값을 선택하도록 알려 주는 검색 마법사가 표시됩니다.

기본 개체 생성

스키마에서 기본 개체를 생성합니다.

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 관리자의 왼쪽 창에서 마우스 오른쪽 단추를 클릭하고 팝업 메뉴에서 **항목 추가**를 선택합니다.
스키마 관리자에 테이블 추가 대화 상자가 표시됩니다.
4. 기초적인 기본 개체 속성을 지정합니다.
5. **확인**을 클릭합니다.

ORS(연산 참조 저장소)에 새 기본 테이블과 모든 관련 지원 테이블이 생성된 다음 새 기본 개체 테이블이 스키마 트리에 추가됩니다.

기본 개체 속성 편집

기존 기본 개체의 속성을 편집할 수 있습니다. 기본 개체 속성을 편집한 후 연산 참조 저장소의 유효성을 검사합니다.

참고: 연산 참조 저장소의 유효성을 검사하지 않은 경우 응용 프로그램에서 비즈니스 항목 서비스를 사용하여 데이터를 검색하면 서비스 호출이 오류와 함께 실패할 수 있습니다.

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 수정할 기본 개체를 선택합니다.
스키마 관리자에 기본 개체 속성 페이지의 기본 탭이 표시됩니다.
4. 다음 지침에 따라 기초적인 기본 개체 속성을 편집합니다.
 - **편집** 단추를 클릭하고 표시 이름 필드에 새 값을 지정합니다.
 - **편집** 단추를 클릭하고 설명 필드에 새 값을 지정합니다.
 - Informatica MDM Hub에서 삽입, 업데이트 또는 삭제된 레코드의 로그를 유지하도록 하려면 **기록 활성화** 확인란을 선택합니다. 기록 테이블은 감사에 사용됩니다.
 - 시간 표시 막대 드롭다운 목록에서 다음 옵션 중 하나를 선택합니다.
 - **시간 표시 막대 없음** - 기본 개체에 대해 시간 표시 막대를 비활성화합니다.
 - **동적 시간 표시 막대** - 기본 개체에 대해 시간 표시 막대를 활성화합니다.
 - 항목 및 관계를 비롯한 기본 개체 레코드에 대해 비연속 유효 기간을 허용하려면 **비연속 유효 기간 허용** 확인란을 선택합니다. 이 속성은 시간 표시 막대가 사용되는 기본 개체에 대해서만 설정할 수 있습니다.
5. 다른 기본 개체 속성을 수정하려면 **고급** 탭을 클릭합니다.
6. 이 기본 개체의 고급 속성을 지정합니다.
7. 왼쪽 창의 기본 개체 이름 아래에서 일치/병합 설정을 클릭합니다.
8. 일치/병합 개체 속성을 지정합니다. 최소한 다음 속성을 구성하십시오.
 - 수동 통합의 최대 일치 항목 수
 - 일치 작업 일괄 처리 주기당 행 수속성을 편집하려면 **편집** 단추를 클릭하고 새 값을 입력합니다.
9. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

스키마를 변경하는 경우 메타데이터 유효성 검사에서 경고가 생성되고 MDM Hub가 C_REPOS_MET_VALID_RESULT 테이블에 항목을 추가합니다.

기본 개체의 사용자 지정 인덱스

사용자 지정 인덱스를 생성하여 일괄 작업 및 SIF API 성능을 향상시킬 수 있습니다.

Informatica MDM Hub는 기본 키 및 고유 열의 시스템 인덱스를 생성합니다. 선택적으로, 다른 열의 사용자 지정 인덱스를 생성하여 성능을 향상시킬 수 있습니다. 사용자 지정 인덱스 값은 고유하지 않을 수도 있습니다.

예를 들어, 외부 응용 프로그램에서 SearchQuery SIF API 요청을 호출하여 기본 개체를 성으로 검색할 수 있습니다. 성 열의 사용자 지정 인덱스를 생성하면 검색 성능이 향상됩니다.

일괄 작업에서는 시스템 인덱스 및 등록된 사용자 지정 인덱스를 삭제한 다음 다시 생성하여 성능을 향상시킵니다. RegisterCustomIndex SIF API를 사용하여 사용자 지정 인덱스를 등록합니다.

기본 개체의 글로벌 비즈니스 식별자 인덱스가 있는 경우 사용자 지정 인덱스를 생성할 수 없습니다. 글로벌 비즈니스 식별자로 사용할 기본 개체 열을 활성화하면 기본 개체에 글로벌 비즈니스 식별자 인덱스가 포함됩니다.

Microsoft SQL Server 제한 때문에 Microsoft SQL Server에서 실행되는 MDM Hub에서 열이 16개가 넘는 사용자 지정 인덱스는 생성할 수 없습니다.

사용자 지정 인덱스 설정 노드로 이동

사용자 지정 인덱스 설정 노드로 이동하려면

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 **기본 개체** 노드를 확장한 후 사용할 기본 개체의 노드를 확장합니다.
4. **사용자 지정 인덱스 설정** 노드를 클릭합니다.

스키마 관리자에 [사용자 지정 인덱스 설정] 페이지가 표시됩니다.

사용자 지정 인덱스 생성

기본 개체의 사용자 지정 인덱스를 생성하려면 기본 개체에 대해 사용자 지정 인덱스 설정을 구성합니다. 기본 개체에 대해 둘 이상의 사용자 지정 인덱스를 생성할 수 있습니다.

1. **모델** 작업 영역에서 **스키마**를 선택합니다.
2. 스키마 관리자에서 작업할 기본 개체의 사용자 지정 인덱스 설정 노드로 이동합니다.
3. **추가** 단추를 클릭합니다.
스키마 관리자에서 NI_C_<base_object_name>_inc라는 사용자 지정 인덱스를 생성합니다. 여기서 *inc*는 증분 숫자입니다.
4. **인덱스의 열** 창의 기본 개체 열 목록에서 사용자 지정 인덱스에 필요한 열을 선택합니다. **저장** 단추를 클릭합니다.

사용자 지정 인덱스 편집

사용자 지정 인덱스를 변경하려면 기존 사용자 지정 인덱스를 삭제하고 필요한 열을 사용하여 새 사용자 지정 인덱스를 추가해야 합니다.

사용자 지정 인덱스 삭제

사용자 지정 인덱스를 삭제하려면

1. 스키마 관리자에서 작업할 기본 개체의 사용자 지정 인덱스 설정 노드로 이동합니다.
2. 인덱스 목록에서 삭제할 사용자 지정 인덱스를 선택합니다.

3. **삭제** 단추를 클릭합니다.
스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.
4. **예**를 클릭합니다.

MDM Hub 외부에서 사용자 지정 인덱스 생성

MDM Hub 외부에서 사용자 지정 인덱스를 생성하려면 데이터베이스용 데이터베이스 유틸리티를 사용합니다.

MDM Hub 외부에서 인덱스를 생성하여 특수 작업을 지원할 수 있습니다. 예를 들어, 인덱스 식으로 함수 기반 인덱스(예: `Upper(Last_Name)`)를 생성할 수 있습니다. MDM Hub 외부에서 사용자 지정 인덱스를 생성하면 해당 인덱스를 등록할 수 없습니다. MDM Hub는 등록되지 않은 인덱스를 자동으로 유지 관리할 수 없습니다. 인덱스를 유지 관리하지 않으면 일괄 작업 성능이 저하될 수 있습니다.

기본 개체의 영향 분석 보기

스키마 관리자를 사용하여 기본 개체와 연결된 테이블, 패키지 및 쿼리를 모두 볼 수 있습니다.

일반적으로 이 작업은 연결된 다른 개체를 실수로 삭제하지 않도록 기본 개체를 삭제하기 전에 수행합니다.

참고: 기본 쿼리에서 열이 삭제되면 종속 쿼리 및 패키지가 완전히 제거됩니다.

기본 개체의 영향 분석을 보려면

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 보려는 기본 개체를 선택합니다.
4. 마우스 오른쪽 단추를 클릭하고 **영향 분석**을 선택합니다.
스키마 관리자에 테이블 영향 분석 대화 상자가 표시됩니다.
5. **닫기**를 클릭합니다.

기본 개체 삭제

기본 개체를 삭제하려면

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 삭제할 기본 개체를 선택합니다.
4. 마우스 오른쪽 단추를 클릭하고 **제거**를 선택합니다.
스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.
5. **예**를 선택합니다.
스키마 관리자에서 기본 개체를 삭제하기 전에 영향 분석을 확인할 것인지 묻는 메시지가 나타납니다.
6. 영향 분석을 확인하지 않고 기본 개체를 삭제하려면 **아니오**를 선택합니다.
스키마 관리자가 삭제된 기본 개체를 스키마 트리에서 제거합니다.

테이블의 열 구성

테이블(기본 개체 또는 랜딩 테이블)을 생성한 후에는 스키마 관리자를 사용하여 해당 테이블의 열을 정의합니다. 테이블의 열을 정의하려면 반드시 스키마 관리자를 사용해야 하며 데이터베이스에서 직접 열을 구성할 수는 없습니다.

참고: 스키마 관리자에서는 교차 참조 테이블과 기록 테이블의 열을 볼 수도 있지만 이러한 열을 편집할 수는 없습니다.

열 정보

이 섹션에서는 테이블 열에 대한 일반적인 정보를 제공합니다.

ORS 테이블의 열 유형

Hub 저장소의 테이블에는 다음 두 가지 유형의 열이 포함됩니다.

열	설명
시스템 열	Informatica MDM Hub에서 자동으로 생성되어 유지 관리되는 열. 시스템 열에는 메타데이터가 포함됩니다.
사용자 정의 열	시스템 열이 아닌 테이블의 모든 열. 사용자 정의 열은 스키마 관리자에서 추가되며 일반적으로 비즈니스 데이터를 포함합니다.

참고: 시스템 열에는 Informatica MDM Hub 메타데이터가 포함됩니다. Informatica MDM Hub 메타데이터는 어떤 식으로든 변경하지 마십시오. 변경할 경우 Informatica MDM Hub가 예측할 수 없는 방식으로 동작하여 데이터를 잃게 될 수 있습니다.

열의 데이터 유형

MDM Hub에는 열의 데이터 유형 집합이 있습니다. MDM Hub 데이터 유형은 Oracle, IBM DB2, Microsoft SQL Server 데이터 유형으로 직접 매핑합니다. 데이터베이스 데이터 유형에 대한 자세한 내용은 데이터베이스용 제품 설명서를 참조하십시오.

참고: 열의 데이터 유형을 변경하려면 Hub 콘솔을 사용하십시오. 데이터베이스의 데이터 유형을 변경하지 마십시오. 데이터베이스의 데이터 유형을 변경하는 경우 메타데이터 유효성 검사 프로세스에 문제가 발생할 수 있습니다.

다음 테이블에서는 MDM Hub 데이터 유형이 데이터베이스 데이터 유형으로 매핑되는 방법을 보여 줍니다.

MDM Hub	Oracle	IBM DB2	Microsoft SQL Server
CHAR	CHAR	CHARACTER	NCHAR ¹
VARCHAR	VARCHAR2	VARCHAR	NVARCHAR
NVARCHAR2	NVARCHAR2	VARGRAPHIC	NVARCHAR
NCHAR	NCHAR	GRAPHIC	NCHAR ¹
DATE	DATE	TIMESTAMP	DATETIME2

MDM Hub	Oracle	IBM DB2	Microsoft SQL Server
TIMESTAMP ²	TIMESTAMP	TIMESTAMP	DATETIME2
NUMBER	NUMBER	DECIMAL	NUMERIC
INT	INTEGER	DECIMAL(31,0)	BIGINT

1. GBID(글로벌 식별자) 열의 경우 CHAR 또는 NCHAR 대신 VARCHAR을 사용하여 Microsoft SQL Server의 행 너비 제한으로 발생할 수 있는 문제를 방지합니다.

2. 날짜가 있는 시스템 열의 경우 데이터 유형은 TIMESTAMP입니다. 날짜가 있는 사용자 정의 열의 경우 TIMESTAMP 데이터 유형을 사용할 수 없습니다. 대신 DATE를 사용합니다.

열 속성

Informatica MDM Hub 열의 속성을 구성할 수 있습니다.

참고: MDM Hub에서 빈 문자열은 빈 문자열을 적용하는 데이터베이스 유형에 상관없이 null 값에 해당합니다.

Informatica MDM Hub 열에는 다음과 같은 속성이 있습니다.

표시 이름

Hub 콘솔에 표시되는 열의 이름입니다.

실제 이름

기본 개체에서 사용되는 열의 이름입니다. Hub 콘솔은 표시 이름을 기반으로 열의 실제 이름을 제안합니다.

실제 열 이름은 예약된 열 이름일 수 없으며, 달러 기호 '\$' 문자를 포함할 수 없습니다.

Null 가능

활성화하면 열이 null 값을 포함할 수 있습니다. null 가능 속성을 활성화하지 않은 경우 기본값을 지정해야 합니다.

null 가능 속성이 열에서 비활성화된 경우, Data Director에서 또는 SIF PUT 작업 중에 레코드가 업데이트 되면 업데이트된 필드에 대한 값으로만 교차 참조 레코드가 생성됩니다. Null이 아니어야 하는 필드를 포함 하여 다른 모든 교차 참조 레코드 필드는 Null 값을 갖습니다. 레코드에 대한 BVT(최선의 진실, Best Version of the Truth) 계산 도중 null이 아니어야 하는 필드에 null 값이 우선 적용될 수 있으며 오류가 발생합니다.

BVT 계산 시 null이 아니어야 하는 필드를 고려하려면 cmxserver.properties 파일에서 cmx.server.put.autopopulate.missing.user.columns.bo.list 속성을 설정합니다. null 가능 속성이 비활성화된 열이 있는 기본 개체의 이름을 쉼표로 구분된 목록으로 하여 속성 값을 설정합니다. 속성이 설정되면 MDM Hub는 교차 참조 레코드의 null 값을 연결된 기본 개체의 값으로 업데이트합니다. 그러면 BVT 계산 도중 null이 아니어야 하는 필드에 null 값이 우선 적용되지 않습니다.

데이터 유형

열의 데이터 유형입니다. 문자 데이터 유형의 경우 길이 속성을 지정할 수 있습니다. 특정 숫자 데이터 유형의 경우 전체 자릿수 속성 및 소수 자릿수 속성을 지정할 수 있습니다.

길이

문자 데이터 유형의 경우 허용되는 문자 수를 지정합니다.

정밀도

숫자 데이터 유형의 경우 모든 소수 자릿수를 포함하여 숫자에 허용되는 10진수를 지정합니다.

소수자릿수

숫자 데이터 유형의 경우 소수점 뒤에 허용되는 10진수를 지정합니다.

기본값 있음

활성화하면 기본값을 지정할 수 있습니다.

기본값

열의 기본값입니다. 열이 null 가능이 아니고 해당 열에 값이 제공되지 않은 경우 Informatica MDM Hub에서는 기본값을 사용합니다.

고유 속성을 활성화하거나 null 가능 속성을 비활성화하는 경우 열의 기본값을 지정해야 합니다.

트러스트

활성화하면 Informatica MDM Hub가 트러스트 점수가 가장 높은 소스 시스템 값을 사용합니다.

활성화하지 않으면 Informatica MDM Hub가 가장 최근에 업데이트된 값을 사용합니다.

고유

활성화하면 Informatica MDM Hub가 열에 고유 제약 조건을 적용합니다. 고유 제약 조건을 활성화하면 고유 열에 중복 값이 없게 됩니다. 대부분의 조직에서는 소스 시스템의 기본 키를 조회 값으로 사용합니다. [고유] 속성을 활성화하면 Informatica MDM Hub가 열에 중복 값이 있는 모든 레코드를 거부합니다. [고유] 속성을 활성화한 경우 열의 기본값을 구성해야 합니다.

통합된 기본 개체에 대해 [고유] 속성을 활성화하면 Informatica MDM Hub가 다른 시스템에서 동일한 키를 로드할 수 있기 때문에 기본 개체에 대한 삽입이 실패할 수 있습니다. 모든 소스 시스템에서 이 열의 고유 키가 있어야 합니다.

유효성 검사

활성화하면 Informatica MDM Hub가 로드 프로세스 동안 유효성 검사 규칙을 적용하여 올바르지 않은 셀 값의 트러스트 점수를 다운그레이드합니다. [유효성 검사] 속성을 활성화한 경우 유효성 검사 규칙을 구성해야 합니다.

Null 값 적용

null 값이 이 열에 대한 가장 신뢰할 수 있는 값인 경우 사용하려는 기본값으로 이 속성을 설정합니다. 프로세스가 준비 테이블의 열에 대해 **Null 업데이트 허용** 속성 설정을 확인할 수 없는 경우 프로세스에서는 이 속성을 사용합니다.

- **True.** 활성화하면 이 열에 대한 가장 신뢰할 수 있는 값이 null 값인 경우 프로세스는 null 값을 기본 개체 레코드에 쓸 수 있습니다.
- **False.** 기본값입니다. 비활성화하면 다른 소스 시스템이 이 열에 대해 null이 아닌 값을 적용하는 한 프로세스는 기본 개체 레코드에 null 값을 쓸 수 없습니다.

Null 값 허용 속성은 **Null 업데이트 허용** 속성과 같은 방식으로 작동됩니다. **Null 업데이트 허용** 속성에 대한 자세한 내용은 [“준비 테이블의 열에 대한 속성” 페이지 356](#)을 참조하십시오.

GBID

활성화하면 Informatica MDM Hub에서 이 열을 기본 개체의 GBID(글로벌 비즈니스 식별자)로 사용합니다. API 액세스 및 일괄 로드에는 GBID 열을 개수 제한 없이 구성할 수 있습니다.

Oracle에서는 GBID를 활성화하면 열을 INT 데이터 유형 또는 길이가 255자인 CHAR, NCHAR, VARCHAR 및 NVARCHAR2 데이터 유형으로 구성해야 합니다.

IBM DB2에서는 GBID를 활성화하면 열을 INT 데이터 유형 또는 길이가 255자인 VARCHAR 및 NVARCHAR 데이터 유형으로 구성해야 합니다.

Microsoft SQL Server에서는 GBID를 활성화하면 열을 INT 데이터 유형 또는 길이가 255자인 VARCHAR 및 NVARCHAR2 데이터 유형으로 구성해야 합니다.

임의의 기본 개체 열에 대해 GBID를 활성화하면 Informatica MDM Hub가 기본 개체의 인덱스를 생성합니다. 기본 개체에 인덱스가 있기 때문에 Informatica MDM Hub에서는 기본 개체의 사용자 지정 인덱스를 생성할 수 없도록 합니다.

넣기 가능

시스템 열에 대해 활성화하면 일괄 작업 및 SIF API 요청에서 시스템 열에 데이터를 삽입하거나 시스템 열의 데이터를 업데이트할 수 있습니다. 다음 시스템 열에 대해서는 [넣기 가능] 속성을 활성화할 수 없습니다.

- ROWID_OBJECT
- CONSOLIDATION_IND
- HUB_STATE_IND
- LAST_ROWID_SYSTEM
- CM_DIRTY_IND

넣기 또는 정리 넣기 API 요청에서 시스템 열을 업데이트하려고 하지만 사용자가 [넣기 가능] 속성을 활성화하지 않으면 API 요청이 실패합니다.

사용자 정의 열과 INTERACTION_ID 및 LAST_UPDATE_DATE 시스템 열은 항상 넣기 가능합니다.

GBID(글로벌 식별자) 열

GBID(글로벌 비즈니스 식별자) 열에는 비즈니스 요구에 따라 레코드를 전역적으로 고유하게 식별할 수 있는 일반적인 식별자(키 값)가 포함됩니다. 예를 들어 다음과 같은 항목이 여기에 포함됩니다.

- ERP(SAP 또는 Siebel 고객 번호) 또는 CRM 시스템과 같이 MDM Hub 외부의 응용 프로그램에서 정의되는 식별자입니다.
- 업종별 코드(AMA 번호, DEA 번호 등) 또는 정부 발행 식별자(주민등록번호, 납세자 번호, 운전면허번호 등)와 같이 외부 기관에 의해 정의된 식별자

참고: GBID 열로 구성하려면 열이 INTEGER, CHAR, VARCHAR, NCHAR 또는 NVARCHAR 열 유형이어야 합니다. 정수가 아닌 열은 정확히 255자여야 합니다.

스키마 관리자에서 기본 개체에 GBID 열을 여러 개 정의할 수 있습니다. 예를 들어 직원 테이블에 주민등록번호 및 운전면허번호에 대한 열이 포함되거나 공급업체 테이블에 납세자 번호가 포함될 수도 있습니다.

MID(마스터 식별자)는 다른 구성 요소(예: CIF, 레거시 Hub, MDM Hub, 상대방 Hub 등)에서 사용되는 참조 시스템 또는 레코드 시스템에 의해 생성된 일반적인 식별자입니다. MDM Hub에서 MID는 ROWID_OBJECT로, 이는 다양한 소스 시스템의 개별 레코드를 고유하게 식별합니다.

GBID는 ROWID_OBJECT를 대체하지 않습니다. GBID는 MDM Hub 구현 환경을 외부 시스템과 통합할 수 있는 추가적인 방법을 제공하므로 *Multidomain MDM 서비스 통합 프레임워크 가이드*에 설명되어 있는 것과 같이 SIF 요청을 사용하여 자신이 선택한 고유한 식별자를 통해 데이터를 쿼리하고 액세스할 수 있습니다. 또한 이미 정의된 식별자를 사용하여 GBID 열을 구성하면 식별자를 사용자 정의할 필요가 없어집니다.

GBID는 데이터의 추적 가능성을 향상시킵니다. 추적 가능성이란 데이터를 추적하는 능력을 말하며, 이를 통해 연결을 확인하여 통합된 레코드에 데이터를 제공한 시스템과 이러한 시스템의 레코드를 알아낼 수 있습니다. 기본 개체에서 GBID 열을 정의하면 스키마 관리자가 현재 및 원래 GBID 값을 추적하기 위해 <GBID_column_name> 및 <GBID_column_name>_GOV의 두 열을 교차 참조 테이블에 생성합니다.

예를 들어 납세자 번호가 다른 고객 두 명을 한 회사에 병합했는데 납세자 번호 하나는 유지하지만 다른 하나는 더 이상 사용하지 않는다고 가정해 봅니다. 납세자 번호 열을 GBID로 정의하면 MDM Hub이 현재 및 과거의 납세자 번호를 모두 추적할 수 있으므로 과거의 값을 사용하여 SIF 요청을 통해 데이터에 액세스할 수 있습니다.

참고: MDM Hub에서는 GBID 열에 대해 데이터 확인 또는 오류 검색을 수행하지 않습니다. 따라서 소스 시스템에 중복 GBID 값이 있을 경우 이러한 중복 값은 MDM Hub에 전달됩니다.

준비 테이블의 열

준비 테이블의 열은 열 편집기를 사용하여 정의할 수 없습니다. 준비 테이블 열은 특수한 경우로, 준비 테이블의 대상 개체에 있는 일부 또는 모든 열을 기반으로 합니다. 준비 테이블에 의해 채워질 수 있는 대상 테이블의 열을 선택하려면 준비 테이블 추가/편집 창을 사용합니다. 그러면 Informatica MDM Hub에서는 각 준비 테이블 열을 대상 테이블의 해당 열과 동일한 데이터 유형으로 생성합니다.

기본 개체의 최대 열 수

자동 통합에 대해 구성된 일치 규칙이 있는 경우에는 기본 개체에 200개 이상의 사용자 정의 열이 포함될 수 없습니다.

열 편집기로 이동

기본 개체 및 랜딩 테이블에 대해 열을 구성하려면



1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 열을 추가할 개체의 스키마 트리를 확장합니다.
4. 열을 선택합니다.
스키마 관리자의 속성 창에 열 정의가 표시됩니다.

시스템 열 표시 확인란을 선택한 경우 열 편집기의 시스템 열 옆에 "잠금" 아이콘이 표시됩니다.

열 편집기의 명령 단추

열 편집기의 속성 창에는 다음과 같은 명령 단추가 포함됩니다.

단추	이름	설명
	열 추가	새 열을 추가합니다.
	열 제거	기존 열을 제거합니다.
	열 설명 편집	선택한 열의 설명을 편집할 수 있습니다.
	위로 이동	선택한 열을 표시 순서에서 위로 이동합니다.
	아래로 이동	선택한 열을 표시 순서에서 아래로 이동합니다.
	스키마 가져오기	다른 테이블에서 열 정의를 가져와 새 열을 추가할 수 있습니다.
	테이블 열 보기 확장	테이블 열 보기를 확장합니다.

단추	이름	설명
	테이블 열 보기 복원	테이블 열 보기를 복원합니다.
	저장	열 정의에 대한 변경 내용을 저장합니다.

시스템 열 표시 또는 숨기기

[시스템 열 표시] 확인란을 전환하여 시스템 열을 표시하거나 숨길 수 있습니다.

테이블 열 보기 확장

속성 창을 확장하여 모든 열 속성을 하나의 창에 표시할 수 있습니다. 기본적으로 스키마 관리자에서 열 정의는 축소된 보기로 표시됩니다.

확장된 테이블 열 보기를 표시하려면

- **테이블 열 보기 확장** 단추를 클릭합니다.
스키마 관리자에 확장된 테이블 열 보기가 표시됩니다.

기본 테이블 열 보기를 표시하려면

- **테이블 열 보기 복원** 단추를 클릭합니다.
스키마 관리자에 기본 테이블 열 보기가 표시됩니다.

열 추가

열을 추가하려면

1. 구성할 테이블에 대한 열 편집기로 이동합니다.
2. 쓰기 잠금을 획득합니다.
3. **추가** 단추를 클릭합니다.
스키마 관리자에 빈 행이 표시됩니다.
4. 각 열의 속성을 지정합니다.
5. **저장** 단추를 클릭하여 추가한 열을 저장합니다.

다른 테이블에서 열 정의 가져오기

다른 테이블에서 일부 열 정의를 가져올 수 있습니다.

1. 구성할 테이블에 대한 열 편집기로 이동합니다.
2. 쓰기 잠금을 획득합니다.
3. **스키마 가져오기** 단추를 클릭합니다.
스키마 가져오기 대화 상자가 열립니다.
4. 가져올 스키마의 연결 속성을 지정합니다.
여기서 지정하는 연결 정보에 대한 자세한 내용이 필요할 경우 데이터베이스 관리자에게 문의하십시오.

IBM DB2 환경의 경우 사용자 이름 및 암호 필드에 대한 설정은 MDM Hub 구현에 프록시 사용자가 구성되어 있는지 여부에 따라 다릅니다.

- 프록시 사용자가 구성되지 않은 경우 사용자 이름은 스키마 이름과 동일합니다.
- 프록시 사용자가 구성된 경우 사용자 지정 사용자 이름 및 암호를 지정합니다.

프록시 사용자 지원에 대한 자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

5. 다음을 클릭합니다.

참고: 데이터베이스를 현재 작업 중인 연산 참조 저장소와 동일하게 입력하지 않아도 되며 데이터베이스가 연산 참조 저장소일 필요도 없습니다.

유일한 제한은 현재 작업 중인 데이터베이스와 다른 유형의 관계형 데이터베이스에서는 가져올 수 없다는 것입니다. 예를 들어 데이터베이스가 Oracle 데이터베이스일 경우 다른 Oracle 데이터베이스의 열만 가져올 수 있습니다.

스키마 관리자에 가져올 수 있는 테이블의 목록이 표시됩니다.

6. 가져올 테이블을 선택합니다.

7. 다음을 클릭합니다.

스키마 관리자에 선택한 테이블의 열 목록이 표시됩니다.

8. 가져오려는 열을 선택합니다.

9. 마침을 클릭합니다.

10. 저장을 클릭합니다.

열 속성 편집

열을 추가하고 저장하고 나서는 특정 열 속성을 변경할 수 있습니다.

참고: 테이블을 정의하고 변경 내용을 저장한 후에는 CHAR, VARCHAR, NCHAR 또는 NVARCHAR2 필드의 길이를 줄이거나 NUMBER 필드의 소수 자릿수 또는 전체 자릿수를 변경할 수 없습니다.

테이블에 데이터가 채워진 후에 스키마를 변경하는 경우 계획되고 제어되는 방식으로 열에 대한 변경 내용을 관리하고 적절한 데이터베이스 백업을 수행한 후에 변경을 수행합니다.

1. 구성할 테이블에 대한 열 편집기로 이동합니다.
2. 쓰기 잠금을 획득합니다.
3. 각 열의 다음 속성을 변경할 수 있습니다.

속성	설명
표시 이름	Hub 콘솔에 표시할 열의 이름입니다.
길이	CHAR, VARCHAR, NCHAR 또는 NVARCHAR2 필드의 길이는 늘리기만 가능합니다.
기본값	열에 값이 제공되지 않았지만 해당 열이 NULL일 수 없는 경우에 사용됩니다. 참고: 기본값을 활성화해도 기본값이 활성화되기 전에 로드된 레코드에는 영향을 주지 않습니다. 기존 NULL 값은 NULL 상태를 유지합니다. NULL인 열 값이 없게 하려면 가능한 경우 준비 테이블에서 데이터를 다시 로드합니다.

속성	설명
트러스트	<p>열이 트러스트된 열인지 여부를 지정합니다.</p> <p>트러스트를 활성화한 경우 메타데이터를 동기화해야 합니다. 이미 데이터가 있는 테이블의 열에 대해 트러스트를 활성화할 경우 트러스트 설정이 변경되었으며 테이블에 다른 데이터를 로드하기 전에 일괄 처리 뷰어 도구에서 트러스트 동기화 일괄 작업을 실행해야 한다는 경고가 표시됩니다.</p> <p>Informatica MDM Hub가 자동으로 일괄 처리 뷰어 도구에서 동기화 작업을 사용할 수 있도록 합니다.</p> <p>다른 추가 로드 작업을 실행하기 전에 동기화 프로세스를 실행해야 합니다. 그렇지 않으면 열을 채우기 위해 사용되는 트러스트된 값이 잘못됩니다.</p> <p>트러스트를 비활성화한 경우 일부 기본 메타데이터 테이블에서 열이 제거되고 데이터가 손실됩니다.</p> <p>실수로 트러스트를 비활성화하고 변경 내용을 저장한 경우에는 트러스트를 다시 활성화하고 동기화 작업을 즉시 실행하여 메타데이터를 다시 생성하는 방식으로 오류를 수정해야 합니다.</p>
고유	<p>열이 고유한 값을 포함하는지 여부를 지정합니다. 열이 고유한 값을 포함해야 하는 경우 고유 속성을 활성화합니다. 열에 대해 고유 속성을 활성화하면 열에 중복 값이 없게 됩니다. 중복 값이 포함된 열에 대해서는 고유 속성을 활성화할 수 없습니다. 특히, 병합될 수 있는 기본 개체에 대해서는 고유 속성을 사용하지 마십시오.</p>
유효성 검사	<p>열의 유효성을 검사해야 하는지 여부를 지정합니다. 유효성 검사를 비활성화하면 연결된 열의 메타데이터가 손실됩니다.</p>
Put 가능	<p>SIF 요청을 사용하고 Hub 콘솔을 통해 실행되는 일괄 작업을 사용하여 데이터를 넣을(삽입 또는 업데이트) 시스템 열에 대해 [Put 가능] 속성을 활성화하십시오. ROWID_OBJECT, CONSOLIDATION_IND, HUB_STATE_IND, LAST_ROWID_SYSTEM 및 CM_DIRTY_IND를 제외한 모든 시스템 열에 적용됩니다.</p>

4. 저장 단추를 클릭하여 변경 내용을 저장합니다.

열 표시 순서 변경

열을 표시 순서의 위 또는 아래로 이동할 수 있습니다. 표시 순서를 변경해도 데이터베이스의 실제 테이블에는 영향이 없습니다.

열 표시 순서를 변경하려면

1. 구성할 테이블에 대한 열 편집기로 이동합니다.
2. 쓰기 잠금을 획득합니다.
3. 이동할 열을 선택합니다.
4. 다음 작업 중 하나를 수행합니다.
 - 선택한 열을 표시 순서의 위로 이동하려면 **위로 이동** 단추를 클릭합니다.
 - 선택한 열을 표시 순서의 아래로 이동하려면 **아래로 이동** 단추를 클릭합니다.
5. 저장 단추를 클릭하여 변경 내용을 저장합니다.

열 삭제

열을 삭제할 때는 특별한 주의가 필요합니다. 열을 제거하면 열에 로드된 모든 데이터가 손실됩니다. 열 삭제 프로세스는 영향을 받을 수 있는 기본 테이블의 개수로 인해 오래 걸릴 수 있습니다.

기본 개체 및 랜딩 테이블에서 열을 삭제하려면

1. Hub 콘솔에서 스키마 도구를 시작합니다.
2. 구성할 테이블에 대한 열 편집기로 이동합니다.
3. 쓰기 잠금을 획득합니다.

4. 속성 창의 열 정의 목록에서 삭제할 열을 선택합니다.
5. **열 제거** 단추를 클릭합니다.
열이 분석되고 영향 분석기가 표시됩니다.
6. 영향 분석기에서 **확인**을 클릭하여 열을 삭제합니다.
열이 제거됩니다.
7. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

기본 개체 간의 외래 키 관계 구성

이 섹션에서는 Informatica MDM Hub 구현의 기본 개체 간 외래 키 관계를 구성하는 방법에 대해 설명합니다.

외래 키 관계에 대한 일반적인 개요는 [“외래 키 관계를 정의하기 위한 프로세스 개요” 페이지 115](#)를 참조하십시오.

외래 키 관계 정보

Informatica MDM Hub에서 외래 키 관계는 일치 열을 통해 두 기본 개체 간에 연관을 설정합니다. 외래 키 관계에서 한 기본 개체(하위 개체)에 다른 기본 개체(상위 개체)의 기본 키 열에 있는 값과 일치하는 값을 포함하는 외래 키 열이 포함되어 있습니다. 하위 기본 개체의 외래 키가 상위 기본 개체의 ROWID_OBJECT를 가리키는 경우 연관된 하위 교차 참조 테이블의 외래 키가 연관된 상위 교차 참조 테이블의 ROWID_XREF를 가리킵니다.

연산 참조 저장소 테이블의 외래 키 관계 유형

Hub 저장소 테이블의 외래 키 관계에는 두 가지 유형이 있습니다.

유형	설명
시스템 외래 키 관계	Informatica MDM Hub에서 자동으로 정의 및 적용되어 스키마의 참조 무결성을 보호합니다.
사용자 정의 외래 키 관계	이 섹션 뒷부분의 지침에 따라 수동으로 정의된 사용자 지정 외래 키 관계입니다.

외래 키 관계를 정의하기 위한 프로세스 개요

외래 키 관계를 생성하려면

1. 상위 테이블을 생성합니다.
2. 하위 테이블을 생성합니다.
3. 이 테이블 간 외래 키 관계를 정의합니다.

상위 테이블에서 생성된 키가 하위 테이블에 포함된 경우 로드 프로세스에서는 적절한 기본 키 값을 상위 테이블에서 하위 테이블로 복사합니다.

외래 키 관계 추가

두 기본 개체 간에 외래 키 관계를 추가할 수 있습니다.

1. 스키마 관리자를 시작합니다.

2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 기본 개체(관계의 하위 개체가 될 기본 개체)를 확장합니다.
4. **관계**를 마우스 오른쪽 단추로 클릭합니다.
스키마 관리자의 관계 페이지에 속성 탭이 표시됩니다.
5. **추가** 단추를 클릭합니다.
스키마 관리자에 관계 추가 대화 상자가 표시됩니다.
6. 다음을 선택하여 새 관계를 정의합니다.
 - 관계 시작 트리의 열
 - 관계 끝 트리의 열
7. 이 외래 키 관계에 대한 인덱스를 생성하려면 **연결된 인덱스 생성** 확인란을 선택합니다. 메타데이터는 인덱스가 있는 연산 참조 저장소에 정의됩니다.
8. **확인**을 클릭합니다.
9. **다이어그램** 탭을 클릭하여 외래 키 관계 다이어그램을 봅니다.
10. **저장** 단추를 클릭하여 변경 내용을 저장합니다.
참고: 관계를 생성한 후 되돌아가 또 다른 관계를 생성하려고 하면 이 열은 사용 중이므로 표시되지 않습니다. 관계를 삭제하면 이 열이 표시됩니다.

외래 키 관계 편집

외래 키 관계에서 조회 표시 이름만 변경할 수 있습니다. 다른 속성을 변경하려면 관계를 삭제한 다음 다시 추가하고 원하는 속성을 지정하십시오.
두 기본 개체 사이 외래 키 관계의 조회 표시 이름을 편집하려면

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 기본 개체를 확장하고 **관계**를 마우스 오른쪽 단추로 클릭합니다.
스키마 관리자의 관계 페이지에 속성 탭이 표시됩니다.
4. **속성** 탭에서 속성을 보려는 외래 키 관계를 클릭합니다.
스키마 관리자에 관계 세부 정보가 표시됩니다.
5. 조회 표시 이름 옆의 **편집** 단추를 클릭하고 새 값을 지정합니다.
6. 원하는 경우 **연결된 인덱스 있음** 확인란을 선택하여 이 외래 키 관계에 인덱스를 추가하거나 이 확인란 선택을 취소하여 기존 인덱스를 제거합니다.
7. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

관계 세부 정보

관계 세부 정보 화면에서 테이블과 외래 키 간의 관계 세부 정보를 볼 수 있습니다.

설명

외래 키와 개체 간의 관계 설명입니다.

제약 조건 테이블

외래 키 제약 조건이 정의된 테이블의 이름입니다.

제약 조건 열

외래 키 제약 조건이 정의된 열의 이름입니다.

상위 테이블

외래 키에 대한 상위 테이블 이름입니다.

상위 열

외래 키에 대한 상위 열 이름으로, 상위 테이블에 있습니다.

관계 적용

- 제한. 연결된 하위 레코드를 사용할 수 있는 경우 **MDM Hub**에서 상위 테이블의 레코드 삭제를 제한합니다. 상위 테이블의 레코드를 삭제하려면 먼저 하위 테이블에서 연결된 레코드를 삭제합니다.
- 계단식 삭제. **C_REPOS_TABLE**에서 상위 레코드가 삭제되는 경우 **MDM Hub**에서 적절한 메타데이터를 모두 삭제합니다. 상위 테이블의 레코드를 삭제하면 **MDM Hub**에서 상위 테이블의 레코드와 하위 테이블의 연결된 레코드를 삭제합니다.

유형

- 적용됨. **MDM Hub**에서 적용된 관계를 생성합니다. 적용된 관계는 데이터베이스의 데이터베이스 제약 조건입니다. 예를 들어 기본 개체와 해당 상호 참조 테이블 간의 관계는 적용을 받습니다. 적용된 관계에 발생하는 문제는 데이터베이스 계층을 통해 보고됩니다.
- 가상. 사용자는 가상 관계를 생성합니다. **MDM Hub**에서는 가상 관계에 대한 제약 조건을 생성하지 않습니다. 가상 관계의 경우 **MDM Hub**에서는 외래 키 관계에 대한 메타데이터를 내부적으로 저장합니다. 가상 관계에 발생하는 문제는 응용 프로그램 서버를 실행한 경우에만 보고됩니다.

연결된 인덱스 있음

연결된 인덱스 있음 확인란을 선택하여 외래 키 관계에 인덱스를 추가합니다. **연결된 인덱스 있음** 확인란을 지워 외래 키 관계에서 기존 인덱스를 제거합니다.

조회 표시 이름

조회 테이블에 대해 **Hub** 콘솔에 표시되는 이름입니다. **편집** 단추를 클릭하여 표시 이름을 변경할 수 있습니다.

외래 키 관계에 대한 조회 구성

외래 키 관계를 생성한 후에는 열에 대한 조회를 구성할 수 있습니다. 조회를 구성하면 **Informatica MDM Hub**가 로드 프로세스 중 상위 테이블에서 데이터 값을 검색하게 됩니다. 예를 들어 **Address** 준비 테이블에 **CONSUMER_CODE_FK** 열이 포함되어 있는 경우 **Informatica MDM Hub**가 **Consumer** 기본 개체의 **ROWID_OBJECT** 열에 대한 조회를 수행하여 **Consumer** 테이블에 있는 연결된 상위 레코드의 **ROWID_OBJECT** 값을 검색하도록 할 수 있습니다.

외래 키 관계 삭제

이전에 추가한 사용자 정의 외래 키 관계를 삭제할 수 있습니다. **Informatica MDM Hub**에서 자동으로 정의한 후 스키마의 참조 무결성을 보호하기 위해 적용한 시스템 외래 키 관계는 삭제할 수 없습니다.

두 기본 개체 간의 외래 키 관계를 삭제하려면

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 기본 개체를 확장하고 **관계**를 마우스 오른쪽 단추로 클릭합니다.
4. 속성 탭에서 삭제할 외래 키 관계를 클릭합니다.
5. **삭제** 단추를 클릭합니다.

스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.

6. **예**를 클릭합니다.

스키마 관리자가 외래 키 관계를 삭제합니다.

7. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

스키마 보기

Hub 콘솔에서 스키마 뷰어 도구를 사용하여 연산 참조 저장소의 스키마를 시각화할 수 있습니다. 스키마 뷰어는 특히 복잡한 스키마를 시각화할 때 유용합니다.

스키마 뷰어 시작

참고: 또한 *Multidomain MDM 리포지토리 관리자 가이드*에 설명된 대로 리포지토리 관리자 내에서 스키마 뷰어를 실행할 수도 있습니다. 그러나 스키마 뷰어가 시작되고 나면 스키마 뷰어 사용 지침은 실행 위치에 관계없이 동일합니다.

스키마 뷰어 도구를 시작하려면

- Hub 콘솔에서 모델 작업 영역을 확장한 후 **스키마 뷰어**를 클릭합니다.
Hub 콘솔에서 스키마 뷰어가 시작되고 데이터 모델이 로드되며 진행률 대화 상자가 표시됩니다.
데이터 모델이 로드되고 나면 Hub 콘솔에 스키마 뷰어 도구가 표시됩니다.





스키마 뷰어의 창




스키마 뷰어는 두 개의 창으로 구분됩니다.

창	설명
다이어그램 창	스키마의 상세한 다이어그램을 표시합니다.
개요 창	스키마의 개요를 표시합니다. 회색 상자는 전체 스키마 다이어그램 중 다이어그램 창에 현재 표시된 부분을 강조 표시합니다. 회색 상자를 스키마의 특정 부분까지 끌어 표시 영역을 이동할 수 있습니다.

스키마 뷰어의 명령 단추

스키마 뷰어의 다이어그램 창에는 다음과 같은 명령 단추가 포함됩니다.

단추	이름	설명
	확대	스키마 다이어그램의 작은 영역을 확대합니다.
	축소	스키마 다이어그램의 큰 부분을 축소하여 표시합니다.
	모두 확대/축소	전체 스키마 다이어그램을 축소하여 표시합니다.
	레이아웃	계층 보기 및 직각 보기 간 전환합니다.

단추	이름	설명
	옵션	열 이름을 표시하거나 숨기고 계층 보기의 방향을 제어합니다.
	저장	스키마 다이어그램을 저장합니다.
	인쇄	스키마 다이어그램을 인쇄합니다.

스키마 다이어그램 확대/축소

스키마 다이어그램을 확대하거나 축소할 수 있습니다.

확대

스키마 다이어그램의 일부 영역을 확대하려면

- **확대** 단추를 클릭합니다.
스키마 뷰어에서 해당 화면 부분이 확대됩니다.

참고: 개요 창의 회색 강조 표시 상자는 다이어그램 창에 표시된 스키마 부분을 나타내기 위해 더 작아집니다.

축소

스키마 다이어그램을 축소하려면

- **축소** 단추를 클릭합니다.
스키마 뷰어에서 스키마 다이어그램이 축소됩니다.

참고: 개요 창의 회색 상자는 보기 영역이 더 커졌음을 나타내기 위해 커집니다.

모두 확대/축소

스키마 다이어그램을 모두 확대/축소하여 다이어그램 창에 전체 스키마 다이어그램을 표시하려면

- **모두 확대/축소** 단추를 클릭합니다.
스키마 뷰어가 축소되어 전체 스키마 다이어그램이 표시됩니다.

스키마 다이어그램 보기 전환

스키마 뷰어에는 스키마 다이어그램이 두 가지 보기로 표시됩니다.

- 계층 보기(기본값)
- 직각 보기

보기 전환

계층 보기와 직각 보기 사이에서 전환하려면

- **레이아웃** 단추를 클릭합니다.
스키마 뷰어에 다른 보기가 표시됩니다.

관련 디자인 개체 및 일괄 작업으로 이동

스키마 뷰어의 개체를 마우스 오른쪽 단추로 클릭하여 컨텍스트 메뉴를 표시합니다.

컨텍스트 메뉴에는 다음 명령이 표시됩니다.

명령	설명
BaseObject로 이동	스키마 관리자를 실행하고 확장된 기본 개체 노드에 이 기본 개체를 표시합니다.
준비 테이블로 이동	스키마 관리자를 실행하고 연결된 기본 개체에 선택한 준비 테이블을 표시합니다.
매핑으로 이동	매핑 도구를 실행하고 선택한 매핑의 속성을 표시합니다.
작업으로 이동	일괄 처리 뷰어를 실행하고 선택한 일괄 작업의 속성을 표시합니다.
일괄 그룹으로 이동	일괄 그룹 도구를 실행합니다.

스키마 뷰어 옵션 구성

스키마 뷰어 옵션을 구성하려면

1. **옵션** 단추를 클릭합니다.
옵션 대화 상자가 표시됩니다.
2. 원하는 옵션을 지정합니다.

항	설명
열 이름 표시	항목 상자에 열 이름을 표시할지 여부를 제어합니다. 항목 상자에 열 이름을 표시하려면 이 옵션을 선택합니다. 항목 상자에서 열 이름을 숨기고 항목 이름만 표시하려면 이 옵션의 선택을 취소합니다.
방향	스키마 계층의 방향을 제어합니다. 다음 값 중 하나입니다. <ul style="list-style-type: none"> - 위에서 아래로(기본값) - 계층이 위에서 아래의 순서로 표시됩니다(최상위 노드가 맨 위에 표시됨). - 아래에서 위로 - 계층이 아래에서 위의 순서로 표시됩니다(최상위 노드가 맨 아래에 표시됨). - 왼쪽에서 오른쪽으로 - 계층이 왼쪽에서 오른쪽의 순서로 표시됩니다(최상위 노드가 왼쪽에 표시됨). - 오른쪽에서 왼쪽으로 - 계층이 오른쪽에서 왼쪽의 순서로 표시됩니다(최상위 노드가 오른쪽에 표시됨).

3. **확인**을 클릭합니다.

스키마 다이어그램을 JPG 이미지로 저장

스키마 다이어그램을 JPG 이미지로 저장하려면

1. **저장** 단추를 클릭합니다.
스키마 뷰어에 [저장] 대화 상자가 표시됩니다.
2. JPG 파일을 저장할, 파일 시스템의 위치로 이동합니다.
3. JPG 파일의 설명 이름을 지정합니다.
4. **저장**을 클릭합니다.
스키마 뷰어가 파일을 저장합니다.

스키마 다이어그램 인쇄

스키마 다이어그램을 인쇄하려면

1. **인쇄** 단추를 클릭합니다.
스키마 뷰어에 [인쇄] 대화 상자가 표시됩니다.
2. 원하는 인쇄 옵션을 선택합니다.

창	설명
인쇄 영역	인쇄할 범위: 모두 인쇄 - 전체 스키마 다이어그램을 인쇄합니다. 표시된 범위 인쇄 - 다이어그램 창에 현재 표시된 스키마 다이어그램 부분만 인쇄합니다.
페이지 설정	용지, 방향 및 여백 등과 같은 페이지 출력 옵션입니다.
프린터 설정	사용자 환경에서 사용 가능한 프린터를 기준으로 하는 프린터 옵션입니다.

3. **인쇄**를 클릭합니다.
스키마 뷰어가 스키마 다이어그램을 프린터로 전송합니다.

제 9 장

쿼리 및 패키지

이 장에 포함된 항목:

- [쿼리 및 패키지 개요, 122](#)
- [쿼리 그룹, 125](#)
- [일반 쿼리, 126](#)
- [사용자 지정 쿼리, 133](#)
- [패키지, 135](#)

쿼리 및 패키지 개요

MDM Hub에서 쿼리는 Hub 저장소에서 날짜를 검색하는 요청입니다. 요청 형식은 SQL SELECT 문입니다. 쿼리를 요청할 때 MDM Hub는 Hub 저장소를 포함하는 데이터베이스에 SQL 쿼리 문을 전송하고 데이터베이스는 쿼리 결과를 반환합니다. 패키지는 쿼리 결과의 공용 뷰입니다.

일반 쿼리

일반 쿼리는 쿼리 마법사 및 빌딩 블록을 사용하여 정의하는 쿼리 유형입니다. SQL 지식이 필요하지 않습니다. 쿼리 도구는 선택한 빌딩 블록에서 SQL SELECT 문을 생성합니다. 생성된 SELECT 문은 지원되는 모든 데이터베이스에서 작동합니다.

사용자 지정 쿼리

사용자 지정 쿼리는 고유의 SQL SELECT 문을 정의하는 쿼리 유형입니다. 데이터베이스 관련 SQL 구문과 문법을 사용할 경우 사용자 지정 쿼리를 생성합니다.

쿼리 그룹

쿼리 그룹은 쿼리를 위한 사용자 정의 컨테이너입니다. 쿼리 그룹을 사용하여 쿼리를 구성하면 검색 및 실행이 더 쉬워집니다.

팁: 쿼리를 생성하기 전에 쿼리 그룹을 생성하면 쿼리를 생성할 때 그룹을 선택할 수 있습니다.

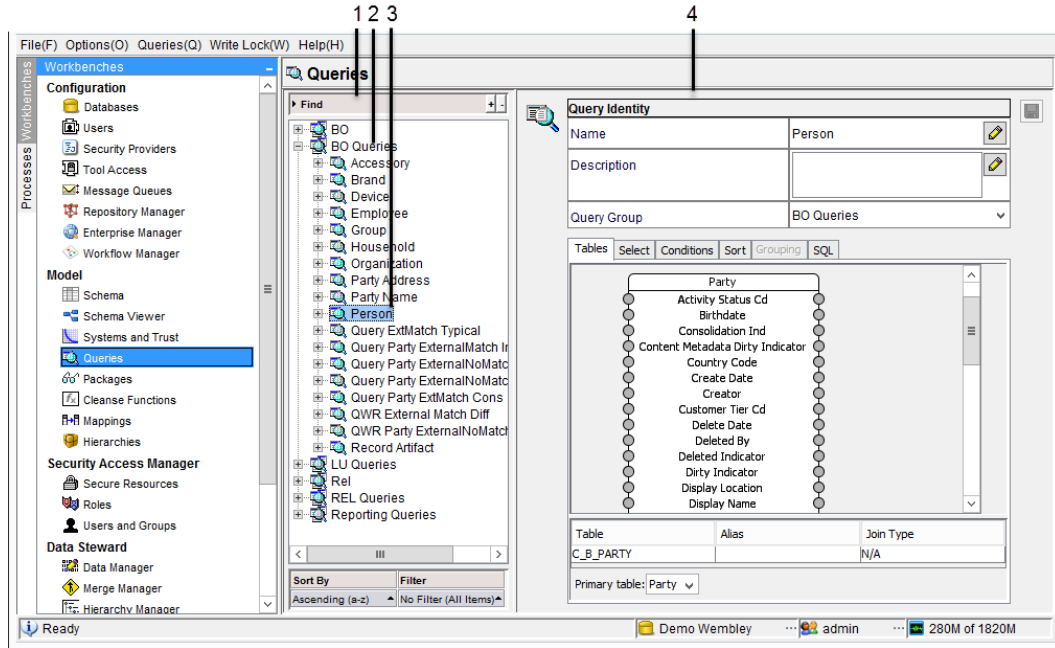
패키지

패키지는 쿼리 결과의 공용 뷰입니다. 데이터 스튜디오는 데이터 관리자 및 병합 관리자 도구에서 패키지를 사용합니다. 외부 응용 프로그램에서 패키지를 사용할 수도 있습니다.

쿼리 도구

쿼리 도구를 사용하여 일반 쿼리, 사용자 지정 쿼리 및 쿼리 그룹을 추가, 편집 및 삭제합니다. 쿼리 결과를 보거나 쿼리에 종속된 패키지를 볼 수도 있습니다.

다음 이미지는 쿼리 도구에서 쿼리가 선택된 상태를 보여 줍니다.

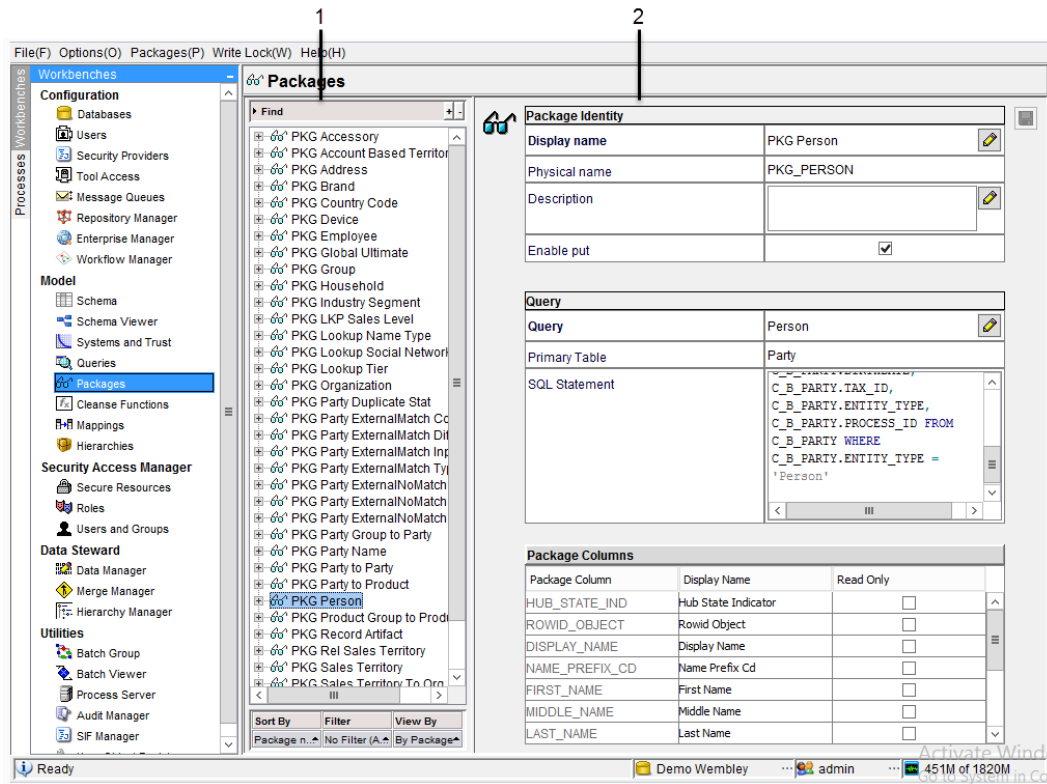


1. 탐색 창. 사용자 정의 쿼리 그룹 및 쿼리가 포함되어 있습니다.
2. 쿼리 그룹입니다.
3. 쿼리입니다.
4. 속성 창. 선택한 쿼리 그룹 또는 쿼리에 대한 속성이 포함되어 있습니다.

패키지 도구

패키지 도구를 사용하여 패키지를 추가, 편집 및 삭제합니다.

다음 이미지는 패키지 도구에서 패키지가 선택된 상태를 보여 줍니다.



1. 탐색 창. 사용자 정의 패키지가 포함되어 있습니다.
2. 속성 창. 선택한 패키지에 대한 속성이 포함되어 있습니다.

쿼리 및 패키지 유지 관리

쿼리 및 패키지를 생성한 후 데이터베이스 스키마를 변경할 경우 MDM Hub는 쿼리 및 패키지를 업데이트합니다. 업데이트는 스키마 변경 유형에 따라 다릅니다. 일부 쿼리 및 패키지는 수동으로 편집해야 할 수 있습니다.

다음 테이블에는 스키마 변경 유형이 나열되어 있으며 MDM Hub에서 수행하는 작업과 사용자가 수행해야 할 작업을 알 수 있습니다.

스키마 변경	일반 쿼리 및 패키지	사용자 지정 쿼리 및 패키지
수정된 열 이름	수정된 열 이름을 사용하도록 일반 쿼리 및 패키지를 업데이트합니다.	수정된 열 이름을 사용하도록 사용자 지정 쿼리 및 패키지를 업데이트합니다.
삭제된 열	일반 쿼리 및 패키지에서 삭제된 열을 제거합니다.	사용자 지정 쿼리는 업데이트하지 않습니다. 삭제된 열을 사용하는 사용자 지정 쿼리 및 패키지는 더 이상 유효하지 않습니다. 참고: 쿼리 및 패키지를 편집하여 삭제된 열을 제거해야 합니다.
삭제된 기본 개체	삭제된 기본 개체를 사용하는 일반 쿼리 및 패키지를 삭제합니다.	사용자 지정 쿼리는 업데이트하지 않습니다. 삭제된 기본 개체를 사용하는 사용자 지정 쿼리 및 개체는 더 이상 유효하지 않습니다. 참고: 쿼리 및 종속 패키지를 삭제해야 합니다.

쿼리 그룹

쿼리 그룹은 쿼리를 위한 사용자 정의 컨테이너입니다. 쿼리 그룹을 사용하여 쿼리를 구성하면 검색 및 사용이 더 쉬워집니다.

예를 들어 다음 쿼리 유형에 대해 별도의 그룹을 생성할 수 있습니다.

- 업데이트 패키지에 사용하기 적합한 일반 쿼리
- 하나 이상의 기본 개체 테이블에서 선택하는 쿼리
- 조회 테이블에서 선택하는 쿼리

쿼리 그룹 추가

쿼리 그룹을 사용하여 쿼리를 구성합니다.

1. **모델** 작업 영역에서 **쿼리**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 마우스 오른쪽 단추를 클릭하고 **새 쿼리 그룹**을 클릭합니다.
4. 쿼리 그룹 추가 창에서 쿼리 그룹 이름을 입력하고 필요한 경우 설명을 입력합니다.
5. **확인**을 클릭합니다.

쿼리 그룹이 탐색 창에 알파벳 순으로 표시됩니다.

쿼리 그룹 편집

쿼리 그룹 이름 및 설명을 편집할 수 있습니다.

1. **모델** 작업 영역에서 **쿼리**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 쿼리 그룹을 선택합니다.
속성 창에 속성이 표시됩니다.
4. 속성을 편집하려면 해당 **편집** 아이콘을 클릭하여 텍스트를 편집하고 **편집 허용** 아이콘을 클릭합니다.
5. **저장** 아이콘을 클릭합니다.

쿼리 그룹 삭제

쿼리 그룹에 쿼리가 포함되어 있는 경우 그룹을 삭제하려면 먼저 쿼리를 이동하거나 삭제해야 합니다.

1. **모델** 작업 영역에서 **쿼리**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 대상 쿼리 그룹을 확장합니다.
4. 쿼리 그룹에 쿼리가 포함되어 있는 경우 그룹의 쿼리를 이동 또는 삭제합니다.
팁: 쿼리를 이동하려면 쿼리를 편집하고 다른 쿼리 그룹을 선택합니다.
5. 빈 쿼리 그룹을 마우스 오른쪽 단추로 클릭하고 **쿼리 그룹 삭제**를 클릭합니다.

일반 쿼리

쿼리 도구에서 빌딩 블록을 사용하여 일반 쿼리를 빌드합니다. 일반 쿼리를 생성하는 SQL을 알 필요가 없습니다.

선택하는 빌딩 블록은 테이블 이름, 열 이름 및 조건 집합을 비롯하여 데이터를 검색하는 데 사용할 조건을 지정합니다. 쿼리에는 결과를 정렬하고 결과를 그룹화하는 방법에 관한 지침도 포함될 수 있습니다. MDM Hub는 빌딩 블록에서 SQL 문을 생성합니다. SQL 문은 모든 데이터베이스에서 작동합니다.

일반 쿼리 추가

빌딩 블록을 사용하여 지원되는 모든 데이터베이스에서 이해할 수 있는 쿼리를 생성하려면 일반 쿼리를 추가합니다.

팁: SQL 문을 코딩하려면 사용자 지정 쿼리를 추가합니다.

1. **모델** 작업 영역에서 **쿼리**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. 필요한 경우 탐색 창에서 쿼리를 추가하려는 쿼리 그룹을 선택합니다.
4. 탐색 창에서 마우스 오른쪽 단추를 클릭하고 **새 쿼리**를 클릭합니다.
새 쿼리 마법사가 열립니다.
5. **시작** 화면이 표시되면 **다음**을 클릭합니다.
6. 일반 쿼리 속성을 지정하고 **다음**을 클릭합니다.

속성	설명
쿼리 이름	쿼리에 대한 설명이 포함된 이름을 입력합니다.
설명	필요한 경우 쿼리에 대한 설명을 입력합니다.
쿼리 그룹	필요한 경우 다른 쿼리 그룹을 선택합니다.
기본 테이블 선택	데이터를 검색하려는 테이블을 선택합니다.

7. 열의 하위 집합을 검색하려면 열을 선택합니다.
 - a. **쿼리 열 선택** 화면에서 포함할 열을 선택하고 모든 기타 열의 확인란을 선택 취소합니다.
 - b. PUT 사용 패키지의 쿼리를 사용하려면 **Rowid 개체** 열을 선택합니다.
참고: Rowid 개체는 PUT 사용 패키지의 필수 열입니다.
 8. **마침**을 클릭합니다.
해당 쿼리가 선택한 쿼리 그룹 내에 표시됩니다.
 9. 쿼리의 결과를 보려면 탐색 창에서 쿼리를 확장하고 **보기**를 클릭합니다.
쿼리 도구에 쿼리 결과가 표시됩니다.
- 쿼리를 더 구체화하려면 쿼리를 편집하고 쿼리 빌딩 블록을 사용합니다.

일반 쿼리 구체화

일반 쿼리를 생성한 후 빌딩 블록을 사용하여 쿼리를 구체화할 수 있습니다. 쿼리를 편집하여 빌딩 블록을 엮습니다. 각 빌딩 블록은 속성 창의 탭에 포함되어 있습니다.

다음 테이블에는 탭이 나열되어 있으며 빌딩 블록을 설명하고 해당하는 SQL 구문을 식별합니다.

탭 이름	빌딩 블록 설명	SQL 구문
테이블	이 쿼리와 연결된 테이블입니다.	FROM 절
선택	이 쿼리와 연결된 열입니다. 함수와 상수를 열에 추가할 수 있습니다.	SELECT <열 목록>
조건	이 쿼리와 연결된 조건입니다. 개별 레코드의 선택 조건을 결정합니다.	WHERE 절
정렬	이 쿼리 결과의 정렬 순서입니다.	ORDER BY 절
그룹화	이 쿼리 결과의 그룹화입니다.	GROUP BY 절
SQL	선택한 빌딩 블록에서 생성된 SQL 문을 표시합니다.	모든 절이 포함된 SELECT 절

일반 쿼리 편집

쿼리 조건을 구체화하려면 쿼리를 편집합니다.

1. **모델** 작업 영역에서 **쿼리**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 쿼리를 선택합니다.
속성 창에 쿼리 속성이 나타납니다.
4. 이름 또는 설명을 변경하려면 해당 **편집** 아이콘을 클릭하여 텍스트를 편집하고 **편집 허용** 아이콘을 클릭합니다.
5. 쿼리 그룹을 변경하려면 쿼리 그룹 목록에서 그룹을 선택합니다.
6. 쿼리 조건을 구체화하려면 탭을 선택하고 빌딩 블록을 정의합니다.

관련 항목:

- [“추가 테이블 선택” 페이지 128](#)
- [“열 선택” 페이지 128](#)
- [“함수 정의” 페이지 129](#)
- [“상수 정의” 페이지 129](#)
- [“비교 조건 정의” 페이지 130](#)
- [“결과의 정렬 순서 정의” 페이지 131](#)
- [“결과의 그룹화 정의” 페이지 131](#)

추가 테이블 선택

일반 쿼리를 생성할 때 쿼리할 기본 테이블을 선택했습니다. 쿼리를 편집할 때 테이블을 더 추가할 수 있습니다. 테이블 간의 외래 키 관계를 정의할 수도 있습니다.

참고: 업데이트 패키지에서 쿼리를 사용할 경우 추가 테이블을 선택하지 않습니다. 업데이트 패키지의 목적은 기본 테이블에서 데이터를 업데이트하는 것입니다.

1. 속성 창에서 **테이블** 탭을 클릭합니다.
2. **추가** 아이콘을 클릭합니다.

팁: 추가 아이콘을 사용할 수 없는 경우 쿼리가 업데이트 패키지와 연결되어 있습니다. 업데이트 패키지는 하나의 테이블만 참조할 수 있습니다.

3. **추가할 테이블 선택** 대화 상자에서 테이블을 선택하고 **확인**을 클릭합니다.
보기 영역과 목록에 테이블이 나타납니다. 목록에서 조인 유형은 교차입니다.
4. 필요한 경우 테이블 간의 외래 키 관계를 생성합니다.
 - a. 보기 영역에서 관련지를 열을 찾습니다. 조인과 호환되는 열이어야 합니다.
 - b. 하나의 열 옆의 원에서 다른 열 옆의 원으로 연결선을 끌어옵니다.
목록에서 조인 유형이 내부로 바뀝니다.
 - c. 필요한 경우 목록에서 유형을 선택하여 조인 유형을 변경합니다.

주의: 쿼리에 둘 이상의 관계가 포함된 경우 하나의 관계만 외부 조인일 수 있습니다.

5. 필요한 경우 다른 테이블을 추가합니다.
오류가 있는 테이블이나 관계를 추가할 경우 제거할 수 있습니다.

옵션	설명
관계 제거	보기 영역에서 연결선을 마우스 오른쪽 단추로 클릭하고 삭제 를 클릭합니다.
테이블 제거	보기 영역에서 테이블을 선택하고 삭제 아이콘을 클릭합니다. 기본 테이블은 삭제할 수 없습니다.

6. **저장** 아이콘을 클릭합니다.

열 선택

각 테이블에서 열의 하위 집합으로 쿼리를 제한할 수 있습니다. 제거할 열을 추가한 경우 쿼리에서 해당 열을 삭제할 수 있습니다.

1. 속성 창에서 **선택** 탭을 클릭합니다.
2. **추가** 아이콘을 클릭합니다.
테이블 열 추가 대화 상자가 열립니다.
3. 테이블 열 목록을 확장합니다.
4. 쿼리에 포함할 열을 선택합니다.
5. **확인**을 클릭합니다.
선택한 열이 테이블에 표시됩니다.

6. 필요한 경우 열의 순서를 다시 지정하거나 제거할 수 있습니다.

옵션	설명
순서 다시 지정	열을 선택하고 위로 이동 또는 아래로 이동 아이콘을 클릭합니다.
제거	열을 선택하고 삭제 아이콘을 클릭합니다.

7. **저장** 아이콘을 클릭합니다.

함수 정의

쿼리의 열에 집계 함수를 추가할 수 있습니다. 예를 들어 열에서 검색되는 데이터에서 COUNT, MIN 또는 MAX를 사용할 수 있습니다. 쿼리에서 함수를 편집하고 제거할 수도 있습니다.

아래 코드 샘플은 SQL 문의 함수를 보여 줍니다.

```
select col1, COUNT(col2) as c1 from table_name
```

- 속성 창에서 **선택** 탭을 클릭합니다.
- 함수 추가** 아이콘을 클릭합니다.
함수 추가 대화 상자가 열립니다.
- 열을 선택합니다.
- 함수를 선택합니다.
- 확인**을 클릭합니다.
함수가 테이블에 표시됩니다.
- 필요한 경우 함수를 편집하거나 순서를 다시 지정하거나 제거할 수 있습니다.

옵션	설명
편집	함수를 선택하고 편집 아이콘을 클릭합니다.
순서 다시 지정	함수를 선택하고 위로 이동 또는 아래로 이동 아이콘을 클릭합니다.
제거	함수를 선택하고 삭제 아이콘을 클릭합니다.

7. **저장** 아이콘을 클릭합니다.

상수 정의

쿼리에서 검색한 열 데이터에 적용할 상수를 추가할 수 있습니다. 쿼리에서 상수를 편집하고 제거할 수도 있습니다.

- 속성 창에서 **선택** 탭을 클릭합니다.
- 상수 추가** 아이콘을 클릭합니다.
상수 추가 대화 상자가 열립니다.
- 데이터 유형을 선택합니다.
- 값 필드가 표시되면 데이터 유형을 준수하는 값을 입력합니다.
- 확인**을 클릭합니다.
상수가 테이블에 표시됩니다.

6. 필요한 경우 상수를 편집하거나 순서를 다시 지정하거나 제거할 수 있습니다.

옵션	설명
편집	상수를 선택하고 편집 아이콘을 클릭합니다.
순서 다시 지정	상수를 선택하고 위로 이동 또는 아래로 이동 아이콘을 클릭합니다.
제거	상수를 선택하고 삭제 아이콘을 클릭합니다.

7. **저장** 아이콘을 클릭합니다.

비교 조건 정의

쿼리의 비교 조건을 정의할 수 있습니다. 비교 조건에는 열, SQL 비교 연산자 및 다른 열 또는 상수가 있습니다. 쿼리를 실행하면 쿼리가 열의 값을 조건과 비교하여 조건을 충족하는 레코드를 반환합니다.

참고: 열의 데이터가 암호화되면 문자열 또는 와일드카드 문자를 사용하는 조건을 생성할 수 없습니다.

- 속성 창에서 **조건** 탭을 클릭합니다.
- 추가** 아이콘을 클릭합니다.
조건 **추가** 대화 상자가 열립니다.
- 열 필드에서 비교를 정의하려는 열을 선택합니다.
- 연산자 필드에서 SQL 비교 연산자를 선택합니다.
- 다른 열 또는 상수가 될 수 있는 비교의 대상을 선택합니다.
 - 열을 선택하는 경우 열 편집 목록에서 열을 선택합니다.
 - 상수를 선택하는 경우 기본값은 NULL입니다. 값을 변경하려면 **편집** 아이콘을 클릭하고 데이터 유형을 선택한 후 값 필드가 표시되면 값을 입력합니다.

예를 들어 아래 이미지는 LAST_NAME 열의 값과 %JO% 문자열을 비교하는 비교 조건을 보여 줍니다.

쿼리를 실행하면 쿼리가 “Johnson”, “Vallejo”, “Major”와 같은 값을 가진 레코드를 반환합니다.

6. **확인**을 클릭합니다.
조건이 테이블에 표시됩니다.

7. 필요한 경우 조건을 편집하거나 순서를 다시 지정하거나 제거할 수 있습니다.

옵션	설명
편집	조건을 선택하고 편집 아이콘을 클릭합니다.
순서 다시 지정	조건을 선택하고 위로 이동 또는 아래로 이동 아이콘을 클릭합니다.
제거	조건을 선택하고 삭제 아이콘을 클릭합니다.

8. **저장** 아이콘을 클릭합니다.

결과의 정렬 순서 정의

데이터베이스가 쿼리 결과를 정렬하는 방법을 지정할 수 있습니다. 정렬을 수행할 열을 선택합니다.

- 속성 창에서 **정렬** 탭을 클릭합니다.
- 추가** 아이콘을 클릭합니다.
테이블 열 추가 대화 상자가 열립니다.
- 테이블을 확인하고 열 목록을 확장합니다.
- 정렬에 포함할 열을 선택합니다.
- 확인**을 클릭합니다.
선택한 열이 정렬 테이블에 표시됩니다. 기본적으로 열은 오름차순으로 정렬됩니다.
- 내림차순으로 열을 정렬하려면 **오름차순** 열의 확인란을 선택 취소합니다.
- 필요한 경우 열의 순서를 다시 지정하거나 제거할 수 있습니다.

옵션	설명
순서 다시 지정	열을 선택하고 위로 이동 또는 아래로 이동 아이콘을 클릭합니다.
제거	열을 선택하고 삭제 아이콘을 클릭합니다.

8. **저장** 아이콘을 클릭합니다.

결과의 그룹화 정의

데이터베이스가 쿼리 결과를 그룹화하는 방법을 지정할 수 있습니다. 그룹화에 포함할 열을 선택합니다.

- 속성 창에서 **그룹화** 탭을 클릭합니다.
- 추가** 아이콘을 클릭합니다.
테이블 열 추가 대화 상자가 열립니다.
- 테이블 열 목록을 확장합니다.
- 그룹에 포함할 열을 선택합니다.
- 확인**을 클릭합니다.
선택한 열이 테이블에 표시됩니다.

- 필요한 경우 열의 순서를 다시 지정하거나 제거할 수 있습니다.

옵션	설명
순서 다시 지정	열을 선택하고 위로 이동 또는 아래로 이동 아이콘을 클릭합니다.
제거	열을 선택하고 삭제 아이콘을 클릭합니다.

- 저장** 아이콘을 클릭합니다.

쿼리 SQL 보기

일반 쿼리의 경우 **쿼리** 도구는 지정한 빌딩 블록에서 SQL 문을 생성합니다. 쿼리를 업데이트할 때마다 SQL 문이 생성됩니다.

- 생성된 SQL 문을 보려면 속성 창에서 **SQL** 탭을 클릭합니다.

쿼리 결과 보기

쿼리 도구 내에서 쿼리 결과를 미리볼 수 있습니다.

- 모델** 작업 영역에서 **쿼리**를 클릭합니다.
- 탐색 창에서 쿼리를 포함하고 있는 쿼리 그룹을 확장합니다.
- 쿼리를 확장합니다.
- 보기**를 클릭합니다.

쿼리 도구에 쿼리 결과가 표시됩니다.

쿼리의 영향 보기

각 쿼리에서 해당 종속성을 볼 수 있습니다. 예를 들어 쿼리를 사용하는 패키지를 볼 수 있습니다.

- 모델** 작업 영역에서 **쿼리**를 클릭합니다.
- 탐색 창에서 쿼리를 포함하고 있는 쿼리 그룹을 확장합니다.
- 해당 쿼리를 마우스 오른쪽 단추로 클릭하고 **영향 분석**을 클릭합니다.
영향 분석 대화 상자가 열립니다.
- 쿼리를 사용할 시스템 개체를 검토합니다.
- 닫기**를 클릭합니다.

쿼리 삭제

더 이상 쿼리를 사용하지 않는 경우 삭제합니다.

- 모델** 작업 영역에서 **쿼리**를 클릭합니다.
- 쓰기 잠금을 획득합니다.
- 탐색 창에서 쿼리를 포함하고 있는 쿼리 그룹을 확장합니다.
- 해당 쿼리를 마우스 오른쪽 단추로 클릭하고 **영향 분석**을 클릭합니다.
영향 분석기 대화 상자가 열립니다.

5. 패키지 섹션에 패키지가 포함되어 있는 경우 패키지 이름을 메모하고 **닫기**를 클릭합니다.
영향 분석기 대화 상자가 닫힙니다.
6. 해당 쿼리가 패키지에 연결되었으면 패키지를 삭제합니다.
 - a. **모델** 작업 영역에서 **패키지**를 클릭합니다.
 - b. 패키지 이름을 마우스 오른쪽 단추로 클릭하고 **패키지 삭제**를 클릭합니다. 모든 종속 패키지를 제거하려면 반복하십시오.
 - c. **모델** 작업 영역에서 **쿼리**를 클릭합니다.
7. 탐색 창에서 쿼리를 마우스 오른쪽 단추로 클릭하고 **쿼리 삭제**를 클릭합니다.

사용자 지정 쿼리

*사용자 지정 쿼리*는 SQL 문을 작성하지 않고 SQL 문을 직접 제공하는 쿼리입니다. 사용자 지정 쿼리는 표시 패키지에서 사용할 수 있습니다. 사용자 지정 쿼리는 모든 개체에 대해 생성할 수 있습니다.

IBM DB2 환경에서 프록시 사용자를 사용하는 경우 프록시 사용자가 쿼리의 개체에 액세스할 수 있는지 확인하십시오. 리포지토리 관리자에 쿼리의 모든 개체에 대한 메타데이터가 없는 경우 리포지토리 관리자 마이그레이션에서 쿼리 권한에 대한 경고가 표시될 수 있습니다. 경고를 해결하려면 프록시 사용자에게 개체에 대한 액세스 권한을 수동으로 부여합니다.

사용자 지정 쿼리의 SQL 구문

Hub 콘솔은 사용자 지정 쿼리에 사용할 수 있는 SQL 구문에 몇 가지 제한을 적용합니다. 이러한 제한 이외에는 데이터베이스에서 지원하는 SQL 구문과 문법을 사용할 수 있습니다.

다음 테이블에는 SQL 구문에 대한 제한이 설명되어 있습니다.

SQL	제한
문	사용자 지정 쿼리는 SELECT 문이어야 합니다. 다른 SQL 문은 지원되지 않습니다.
열 이름	열 이름에는 영숫자와 밑줄을 포함할 수 있습니다. 공백 및 기타 특수 문자는 지원되지 않습니다.
별칭	별칭 이름에는 영숫자와 특수 문자를 포함할 수 있습니다. 공백은 지원되지 않습니다.
상수 열	작은 따옴표로 묶은 상수 열을 추가하려면 별칭을 사용해야 합니다. 예를 들어 다음 쿼리에서는 const_alias라는 별칭을 사용합니다. SELECT ID, 'CONST_COL' AS const_alias FROM c_party
집계 함수	특수 문자가 포함된 집계 함수를 추가하려면 별칭을 사용해야 합니다. 예를 들어 다음 쿼리에서는 new_rowid라는 별칭을 사용합니다. SELECT rowid_object*0 AS new_rowid FROM c_party

SQL 유효성 검사

사용자 지정 쿼리를 저장하면 MDM Hub에서 SQL 문에 대해 클라이언트 측 유효성 검사를 수행한 다음 추가적인 유효성 검사를 위해 SQL 문을 데이터베이스에 보냅니다.

클라이언트 측 SQL 유효성 검사

MDM Hub는 SQL 문이 MDM Hub에 필요한 SQL 구문을 충족하는지 유효성 검사를 수행합니다. 예를 들어 클라이언트 측 유효성 검사 프로세스에서는 문이 **SELECT** 키워드로 시작하고 열 이름에 공백이나 특수 문자가 포함되지 않았는지 확인합니다. 유효성 검사 프로세스에서 오류가 발견되면 오류 메시지가 표시됩니다.

데이터베이스 측 SQL 유효성 검사

데이터베이스에서는 SQL 문을 수신하면 SQL 문의 구문과 문법이 데이터베이스의 요구 사항을 충족하는지 확인합니다. SQL 문에서 오류를 생성하면 데이터베이스에서 해당 오류를 MDM Hub에 반환하고, Hub 콘솔에서는 데이터베이스 오류를 표시합니다.

사용자 지정 쿼리 추가

사용자 지정 쿼리에서 데이터베이스별 SQL 구문을 사용할 수 있습니다. SQL 문이 데이터베이스와 MDM Hub 콘솔의 구문 요구 사항을 준수하는지 확인합니다.

1. **모델** 작업 영역에서 **쿼리**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. 쿼리 그룹을 정의한 경우 쿼리를 추가할 그룹을 선택합니다.
4. 그룹을 마우스 오른쪽 단추로 클릭하고 **새 사용자 지정 쿼리**를 클릭합니다.
새 사용자 지정 쿼리 마법사가 표시됩니다.
5. 시작 화면이 표시되면 **다음**을 클릭합니다.
6. 다음과 같은 사용자 지정 쿼리 속성을 설정합니다.

속성	설명
쿼리 이름	쿼리에 대한 설명이 포함된 이름을 입력합니다.
설명	필요한 경우 쿼리에 대한 설명을 입력합니다.
쿼리 그룹	필요한 경우 선택한 쿼리 그룹을 변경합니다.

7. **마침**을 클릭합니다.
선택한 쿼리 그룹 아래 탐색 창에 쿼리가 표시됩니다.
8. SQL 필드 옆의 **편집** 아이콘을 클릭합니다.
9. 사용 중인 데이터베이스 환경의 구문 규칙에 따라 SQL 쿼리를 입력합니다.
10. **저장** 아이콘을 클릭합니다.
Hub 콘솔은 쿼리를 저장하고 데이터베이스에 쿼리를 보냅니다. 데이터베이스가 SQL 문에서 오류를 발견하면 쿼리 도구는 데이터베이스 오류 메시지를 표시합니다. 오류를 수정하고 변경 내용을 저장합니다.
11. 쿼리의 결과를 보려면 탐색 창에서 쿼리를 확장하고 **보기**를 클릭합니다.
쿼리 도구에 쿼리 결과가 표시됩니다.

사용자 지정 쿼리 편집

사용자 지정 쿼리를 편집할 수 있습니다.

1. **모델** 작업 영역에서 **쿼리**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 사용자 지정 쿼리를 선택합니다.
속성 창에 속성 및 **SQL** 문이 나타납니다.
4. 속성을 편집하려면 해당 **편집** 아이콘을 클릭하여 텍스트를 편집하고 **편집 허용** 아이콘을 클릭합니다.
5. **저장** 아이콘을 클릭합니다.
쿼리 도구에서 쿼리 설정의 유효성을 검사하여 오류가 있는 경우 메시지를 표시합니다.

관련 항목:

- [“쿼리 삭제” 페이지 132](#)
- [“쿼리의 영향 보기” 페이지 132](#)
- [“쿼리 결과 보기” 페이지 132](#)

패키지

패키지는 하나 이상의 쿼리에 대한 공용 뷰입니다. **Hub** 콘솔의 데이터 스튜어드 도구 및 서비스를 사용하여 마스터 데이터에 액세스하는 외부 응용 프로그램에서 패키지를 사용합니다.

두 가지 패키지(표시 패키지 및 업데이트 패키지)를 생성할 수 있습니다. 표시 패키지는 마스터 데이터의 읽기 전용 뷰를 정의합니다. 업데이트 패키지는 권한 있는 사용자가 마스터 데이터를 변경할 수 있는 뷰를 정의합니다.

권한 있는 사용자로 패키지를 제한하려면 패키지를 보안 리소스로 만듭니다. 역할 도구에서 모든 보안 리소스를 관리합니다. 자세한 내용은 *Multidomain MDM 보안 가이드*를 참조하십시오.

표시 패키지

사용자가 쿼리 결과를 데이터 스튜어드 도구 또는 외부 응용 프로그램에서 볼 수 있도록 하려면 표시 패키지를 생성합니다. 사용자는 쿼리 결과를 볼 수 있지만 데이터를 변경할 수는 없습니다.

각 권한 있는 역할의 경우 해당 역할이 사용할 수 있는 표시 패키지에 대해 읽기 권한을 설정합니다. 다른 권한은 활성화하지 마십시오. 보안 리소스 및 권한에 대한 자세한 내용은 *Multidomain MDM 보안 가이드*를 참조하십시오.

업데이트 패키지

업데이트 패키지는 **PUT** 사용 패키지, 넣기 가능 패키지 또는 병합 패키지라고도 합니다. 업데이트 패키지를 생성할 때 패키지 내에서 **PUT API**를 활성화합니다. 데이터 스튜어드는 업데이트 패키지를 사용하여 마스터 데이터를 추가, 변경 또는 병합할 수 있습니다.

권한 있는 사용자가 다음 액션을 수행할 수 있도록 하려면 업데이트 패키지를 생성합니다.

- 데이터 관리자 또는 외부 응용 프로그램에서 레코드의 데이터를 업데이트합니다.
- 데이터 관리자 또는 외부 응용 프로그램에서 레코드를 삽입합니다.
- 데이터 관리자 또는 외부 응용 프로그램에서 레코드를 병합합니다.

각 권한 있는 역할의 경우 해당 역할이 사용할 수 있는 업데이트 패키지에 대해 권한을 설정합니다. 생성, 업데이트 또는 병합 권한 중 하나 이상의 권한과 읽기 권한을 부여합니다. 보안 리소스 및 권한에 대한 자세한 내용은 *Multidomain MDM 보안 가이드*를 참조하십시오.

업데이트 패키지 요구 사항

업데이트 패키지에는 패키지 수준 및 쿼리 수준에서 몇 가지 요구 사항이 있습니다.

패키지 수준 요구 사항

업데이트 패키지는 다음 요구 사항을 준수해야 합니다.

- 업데이트 패키지에 대한 일반 쿼리를 선택합니다. 사용자 지정 쿼리 및 기타 패키지는 업데이트 패키지에서 지원되지 않습니다.
- Put 활성화** 옵션을 선택합니다.
- 일반 쿼리 내에 정의된 관계 및 테이블만 액세스합니다. 업데이트 패키지는 다른 테이블에 대한 조인을 포함할 수 없습니다.

쿼리 수준 요구 사항

일반 쿼리는 다음 요구 사항을 준수해야 합니다.

- 기본 테이블만 선택합니다. 일반 쿼리는 추가 테이블을 포함할 수 없습니다.
- 개별 열을 선택할 경우 ROWID_OBJECT 열을 추가해야 합니다.
- 일반 쿼리는 시스템 테이블, 상수 열, 집계 함수 또는 그룹화를 포함할 수 없습니다.

패키지 추가

표시 패키지 또는 업데이트 패키지를 생성할 수 있습니다.

팁: 업데이트 패키지를 생성하려면 먼저 업데이트 패키지에 대한 요구 사항을 충족하는 일반 쿼리를 생성합니다.

- 모델** 작업 영역에서 **패키지**를 클릭합니다.
- 쓰기 잠금을 획득합니다.
- 탐색 창에서 마우스 오른쪽 단추로 클릭하고 **새 패키지**를 클릭합니다.
새 패키지 마법사가 열립니다.
- 시작** 화면이 표시되면 **다음**을 클릭합니다.
- 다음 속성을 설정합니다.

속성	설명
표시 이름	패키지에 대한 설명이 포함된 이름을 입력합니다. 이 이름이 탐색 창에 표시됩니다.
실제 이름	필요한 경우 제안된 실제 이름을 변경합니다. 마법사는 지정한 표시 이름을 바탕으로 실제 이름을 제안합니다.
설명	필요한 경우 쿼리에 대한 설명을 입력합니다.
쿼리 그룹	필요한 경우 다른 쿼리 그룹을 선택합니다.
PUT 활성화	업데이트 패키지를 생성하려면 이 옵션을 선택합니다. 표시 패키지를 생성하려면 이 옵션을 선택 취소합니다.

속성	설명
보안 리소스	패키지를 사용할 수 있는 사용자를 제한하려면 이 옵션을 선택합니다. 역할 도구를 사용하여 보호된 패키지에 사용자 역할을 할당합니다.

6. 다음을 클릭합니다.

새 패키지 마법사에 **쿼리 선택** 대화 상자가 표시됩니다.

7. 패키지에서 사용하려는 쿼리를 선택합니다.

기억: 업데이트 패키지의 경우 일반 쿼리를 선택해야 합니다. 표시 패키지의 경우 일반 쿼리 또는 사용자 지정 쿼리를 선택할 수 있습니다.

- 기존 쿼리를 사용하려면 목록에서 쿼리를 선택합니다.
- 쿼리를 생성하려면 **새 쿼리**를 클릭하여 쿼리를 생성합니다.
- 쿼리 그룹을 생성하려면 **새 쿼리 그룹**을 클릭하여 쿼리 그룹을 생성합니다.

8. **마침**을 클릭합니다.

9. 패키지 결과를 미리 보려면 탐색 창에서 패키지를 확장하고 **보기**를 클릭합니다.

패키지 도구에 패키지 미리보기가 표시됩니다.

팁: 패키지가 생성되지 않고 사용자 지정 쿼리를 선택한 경우 사용자 지정 쿼리가 SQL 구문 제약 조건을 준수하는지 확인합니다.

패키지 편집

패키지 속성 및 기본 쿼리를 변경할 수 있습니다.

1. **모델** 작업 영역에서 **패키지**를 클릭합니다.

2. 쓰기 잠금을 획득합니다.

3. 탐색 창에서 패키지를 선택합니다.

속성 창에 속성이 표시됩니다.

4. 텍스트 속성을 편집하려면 해당 **편집** 아이콘을 클릭하고 **편집 허용** 아이콘을 클릭합니다.

5. **저장** 아이콘을 클릭합니다.

6. 필요한 경우 이 패키지에 대한 쿼리를 편집할 수 있습니다.

- 탐색 창에서 패키지를 확장합니다.
- 쿼리**를 클릭합니다.
- 쿼리를 편집합니다.

7. 패키지 결과를 미리 보려면 탐색 창에서 패키지를 확장하고 **보기**를 클릭합니다.

패키지 도구에 패키지 미리보기가 표시됩니다.

쿼리 변경 후 패키지 새로 고침

쿼리를 변경할 경우 쿼리를 사용하는 모든 패키지를 새로 고칩니다.

참고: 새로 고친 후 패키지가 쿼리와 동기화되지 않을 경우 쿼리와 일치하도록 열을 선택하거나 지웁니다.

1. **모델** 작업 영역에서 **패키지**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.

3. 탐색 창에서 패키지를 확장합니다.
4. 새로 고침을 클릭합니다.

패키지 삭제

더 이상 패키지가 필요하지 않은 경우 패키지를 삭제하여 기본 쿼리에서 종속성을 제거합니다.

1. 모델 작업 영역에서 **패키지**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 패키지를 마우스 오른쪽 단추로 클릭하고 **패키지 삭제**를 클릭합니다.

조인 쿼리 지정

데이터 관리자나 병합 관리자에서 데이터 스튜어드가 다른 테이블의 정보와 함께 기본 개체 정보를 볼 수 있는 패키지를 생성할 수 있습니다.

1. 기본 개체를 쿼리하는 업데이트 패키지를 생성합니다.
2. 다른 테이블과 업데이트 패키지를 조인하는 쿼리를 생성합니다.
3. 방금 생성한 쿼리를 바탕으로 표시 패키지를 생성합니다.

제 10 장

시간 표시 막대

이 장에 포함된 항목:

- [개요, 139](#)
- [지침, 140](#)
- [예제, 140](#)
- [레코드 버전, 141](#)
- [시간 표시 막대 세분성, 143](#)
- [기록 및 상태 관리, 144](#)
- [시간 표시 막대 적용 규칙, 144](#)
- [기본 개체 시간 표시 막대 구성, 153](#)
- [일괄 작업으로 여러 레코드 버전 로드, 154](#)
- [레코드 버전의 유효 기간 편집, 155](#)
- [레코드 버전 추가, 157](#)
- [레코드의 데이터 업데이트, 158](#)
- [관계 업데이트, 159](#)
- [관계 종료, 161](#)
- [관계 기간 삭제, 162](#)
- [모든 관계 기간 삭제, 163](#)
- [시간 표시 막대 추출 사용, 164](#)

개요

시간 표시 막대 관리를 통해 비즈니스 항목의 데이터 변경 이벤트 및 해당 관계를 관리할 수 있습니다. 데이터 변경 이벤트는 시간 동안 유효한 데이터에 대한 변경입니다. 데이터 변경 이벤트가 발생하면 **MDM Hub**에서 기존 데이터를 덮어쓰는 대신 여러 버전의 항목을 생성합니다.

해당 유효 기간의 측면에서 비즈니스 항목 및 해당 관계의 데이터 변경 이벤트 또는 버전을 정의할 수 있습니다. 데이터 변경은 시간의 흐름에 따라 발생하고 다른 데이터에 대한 해당 관계와는 무관합니다. 날짜를 변경하면 유효 시간이 새로 설정되거나 기존 또는 미래 유효 기간이 업데이트됩니다. 데이터의 유효 기간에 대한 변경 내용을 추적하려면 시간 표시 막대 관리를 사용합니다.

시간 표시 막대를 활성화하면 고객 주소, 전화 번호, 해당 관계 등 비즈니스 항목의 데이터 이벤트를 관리할 수 있습니다. 기본 개체 레코드의 유효 기간을 유지하기 위해 **MDM Hub**에서는 사용자가 시간 표시 막대를 활성화한 기본 개체와 연결된 교차 참조 테이블을 사용합니다.

Hub 콘솔에서 기본 개체 속성을 구성한 경우 기본 개체에 대한 시간 표시 막대를 활성화하십시오. 시간 표시 막대를 활성화할 경우 MDM Hub에서 고객 주소와 같은 데이터의 과거, 현재, 미래 변경 내용을 추적합니다.

시간 표시 막대는 기본적으로 계층 관리 관계 기본 개체에 대해 활성화됩니다.

시간 표시 막대가 활성화된 기본 개체에는 사용자가 지정하는 유효 기간과 관련된 데이터 버전이 포함됩니다. 기본 개체가 현재 값을 반영하지 못할 수 있습니다. 특정 시간에 기본 개체에 포함되는 날짜 버전은 교차 참조 테이블에 의해 결정되며, 교차 참조 테이블은 시간에 따라 달라질 수 있습니다. SIF API 호출을 사용하여 현재 유효 날짜에 대한 레코드를 가져올 수 있습니다. 또한 유효 날짜를 전달하여 지정한 유효 날짜의 날짜를 검색할 수도 있습니다.

변경 내용이 수신되어 기본 개체의 현재 유효 기간에 영향을 줄 경우 MDM Hub에서는 현재 트러스트 설정을 사용하여 BVT를 계산합니다. 그런 다음 MDM Hub에서 기본 개체 레코드를 업데이트합니다.

참고: Hub 서버에서는 항상 현재 트러스트 설정을 사용하여 과거, 현재 또는 미래 날짜에 대한 BVT를 계산합니다.

기본 개체에 대해 시간 표시 막대를 활성화한 후에는 비활성화할 수 없습니다. 채워진 기본 개체에 대해 시간 표시 막대를 활성화할 경우 Hub 서버에서 유효 기간 시작 날짜 및 종료 날짜를 null로 설정합니다.

지침

기본 개체에 대한 시간 표시 막대를 구성할 경우 MDM Hub에서 기본 개체의 데이터 변경 이벤트 및 해당 관계를 관리합니다.

관계 기본 개체의 시간 표시 막대는 기본적으로 활성화되지만 항목 기본 개체의 시간 표시 막대는 활성화되지 않습니다. 관계 기본 개체의 시간 표시 막대는 비활성화할 수 없습니다. 또한 시간 표시 막대가 활성화된 후에는 기본 개체의 시간 표시 막대를 비활성화할 수 없습니다.

기본 개체에 대해 시간 표시 막대를 활성화하면 기록 및 상태 관리가 기본적으로 활성화됩니다. 시간 표시 막대가 활성화된 기본 개체의 기록 또는 상태 관리를 비활성화할 수 없습니다.

MDM Hub에서는 시간 표시 막대가 활성화된 기본 개체에 대해 BVT(최선의 진실, Best Version of the Truth)를 자동으로 업데이트하지 않습니다. 사용자는 유효 기간을 지정하여 기간 내 레코드의 BVT를 볼 수 있습니다.

예제

조직에서 시간 표시 막대가 활성화된 CUSTOMER 기본 개체가 있다고 가정합니다. CUSTOMER 기본 개체에는 로스앤젤레스에서 2011년 1월 31일부터 2013년 10월 20일까지 거주했고, 2013년 10월 21일부터 현재까지 샌프란시스코에서 거주하고 있으며, 2015년 11월 25일부터는 라스베이거스에서 거주할 예정인 John Smith라는 사람의 레코드가 포함되어 있습니다. MDM Hub는 John Smith의 주소처럼 과거, 현재 그리고 미래의 날짜 변화를 추적합니다.

과거 데이터가 포함된 레코드

CUSTOMER 기본 개체에는 John Smith가 과거에 거주했던 로스앤젤레스 주소로 된 레코드가 포함되어 있습니다. 교차 참조 테이블에는 John Smith가 로스앤젤레스에 거주한 기간 사이의 레코드가 포함되어 있습니다.

John Smith의 교차 참조 레코드는 John Smith가 과거에 로스앤젤레스에 거주한 사실을 보여 줍니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	로스앤젤레스	2011년 1월 31일	2013년 10월 20일

John Smith의 기본 개체 레코드는 예를 들어 2012년 2월 12일과 같이 여러분이 지정하는 유효 날짜에 대한 BVT(최선의 진실, Best Version of the Truth)를 표시합니다.

Rowid 개체	고객 ID	이름	성	구/군/시
2	25	John	Smith	로스앤젤레스

과거 및 현재 데이터가 포함된 레코드

CUSTOMER 기본 개체에는 여러분이 지정하는 날짜의 BVT에 해당하는 주소로 된 John Smith 레코드가 포함되어 있습니다. 교차 참조 테이블에는 John Smith의 레코드가 하나 포함되어 있습니다. 교차 참조 테이블은 John Smith가 로스앤젤레스에 거주한 기간과 샌프란시스코에 거주한 기간의 레코드를 보여 줍니다.

John Smith의 교차 참조 레코드는 John Smith가 과거에 로스앤젤레스에 거주한 사실을 보여 줍니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	로스앤젤레스	2011년 1월 31일	2013년 10월 20일
2	2	25	John Smith	샌프란시스코	2013년 10월 21일	2015년 11월 24일

John Smith의 기본 개체는 2012년 2월 12일과 같이 여러분이 지정하는 기간의 BVT를 표시합니다.

Rowid 개체	고객 ID	이름	성	구/군/시
2	25	John	Smith	로스앤젤레스

과거, 현재 및 미래의 데이터가 포함된 레코드

CUSTOMER 기본 개체에는 여러분이 지정하는 날짜의 BVT에 해당하는 주소로 된 John Smith 레코드가 포함되어 있습니다. 교차 참조 테이블에는 John Smith가 로스앤젤레스에 거주한 기간의 레코드가 포함되어 있습니다.

과거에 로스앤젤레스에 거주했고, 현재 샌프란시스코에 거주하고, 2015년 11월 25일부터는 라스베이거스에서 거주할 예정인 John Smith의 교차 참조 레코드입니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	로스앤젤레스	2011년 1월 31일	2013년 10월 20일
2	2	25	John Smith	샌프란시스코	2013년 10월 21일	2015년 11월 24일
3	2	25	John Smith	라스베이거스	2015년 11월 25일	null

John Smith의 기본 개체는 2012년 2월 12일과 같이 여러분이 지정하는 기간의 BVT를 표시합니다.

Rowid 개체	고객 ID	이름	성	구/군/시
2	25	John	Smith	로스앤젤레스

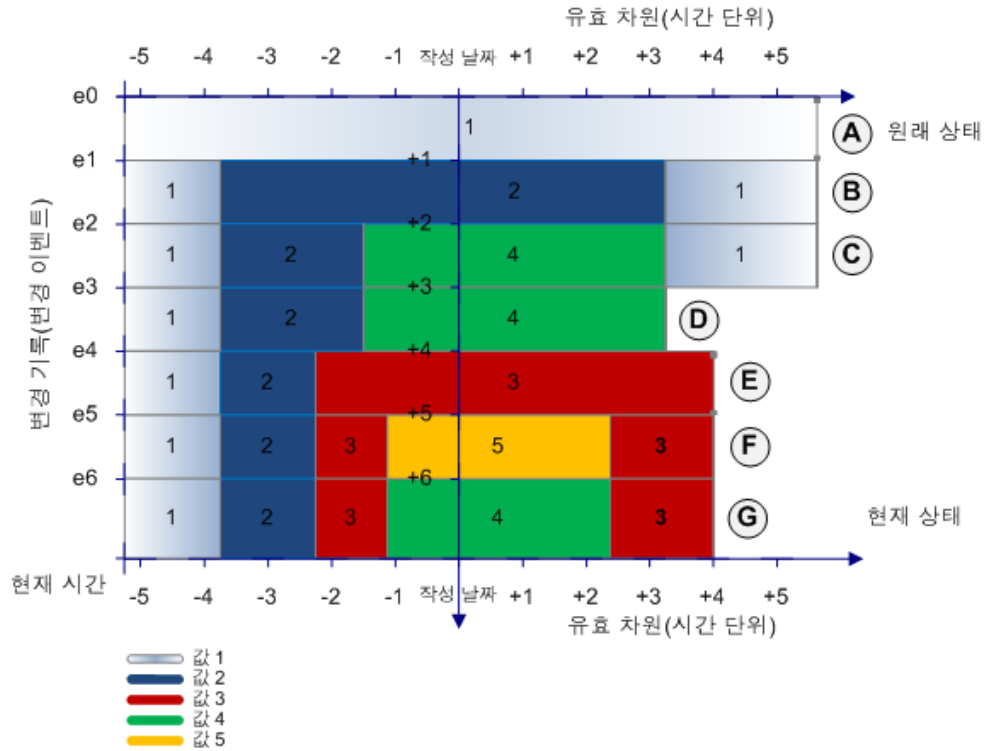
레코드 버전

기본 개체에 대한 시간 표시 막대가 활성화된 경우 데이터에 대한 2차원 정보를 볼 수 있습니다. 데이터에 대한 2차원 보기는 레코드의 유효 기간 및 기록을 기반으로 합니다. 유효 기간은 레코드가 유효한 기간입니다. 레코드의 시작 날짜와 종료 날짜를 지정하여 레코드의 유효 기간을 정의할 수 있습니다. 기록은 레코드의 수명 기간 중 발생한 과거의 데이터 이벤트입니다. 레코드의 기록을 보려면 레코드가 유효했던 과거의 날짜를 지정하십시오.

레코드 버전 예제

직에서 시간 표시 막대가 활성화된 기본 개체에 표시되는 레코드가 있다고 가정합니다. 이 레코드의 시간 표시 막대에는 여러 개의 데이터 변경 이벤트(e0, e1, e2, e3, e4, e5 및 e6)가 포함되어 있습니다. 데이터 변경 이벤트, 다시 말해서 새 레코드 값 또는 업데이트된 레코드 값은 지정된 유효 기간 유효한 새 교차 참조 레코드 버전이 됩니다.

다음 그림에서는 레코드의 전체 유효 기간과 기록 수명 및 변경 기록을 함께 나타냅니다.



레코드는 수명 기간 동안 다음 변경 사항을 거칩니다.

- A. 값이 1인 원래 레코드는 유효 기간이 지정되어 있지 않으므로 항상 유효합니다.
- B. MDM Hub에서 유효 기간 값이 2인 새 레코드 버전을 추가합니다.
- C. MDM Hub에서 유효 기간 값이 4인 새 레코드 버전을 추가합니다.
- D. MDM Hub에서 유효 기간 값이 1인 레코드 버전을 제거합니다.
- E. MDM Hub에서 유효 기간 값이 3인 새 레코드 버전을 추가합니다. MDM Hub에서 유효 기간 값이 4인 레코드 버전을 제거합니다. 새로 추가된 버전의 유효 기간에 포함되기 때문입니다. MDM Hub에서 값이 2인 레코드 버전의 유효 기간을 업데이트합니다.
- F. MDM Hub에서 유효 기간 값이 5인 새 레코드 버전을 추가합니다. MDM Hub에서 새로운 두 유효 기간에 대해 레코드 값이 3인 레코드 버전 두 개를 생성합니다.
- G. 레코드 버전의 값이 5에서 4로 변경되지만 유효 기간은 변경되지 않습니다.

시간 표시 막대 세분성

시간 표시 막대 세분성은 레코드 버전에 대한 유효 기간을 정의하기 위해 사용할 시간 단위입니다. 예를 들어, 유효 기간을 연, 월, 초 단위로 선택할 수 있습니다.

연도, 월, 일, 시간, 분 또는 초의 시간 표시 막대 세분성을 구성하여 MDM Hub 구현의 데이터 유효 기간을 지정할 수 있습니다. 연산 참조 저장소를 생성하거나 업데이트할 때 필요한 시간 표시 막대 세분성을 구성할 수 있습니다.

중요: 한 번 구성한 시간 표시 막대 세분성은 변경할 수 없습니다.

어떠한 시간 표시 막대 세분성의 유효 기간을 지정하든 시스템에서는 데이터베이스 시간 로컬이 유효 기간으로 사용됩니다. 시간 표시 막대 측정 단위에 대해 유효한 버전을 생성하려면 시작 날짜와 종료 날짜가 같아야 합니다.

다음 테이블에는 구성할 수 있는 시간 표시 막대 세분성 옵션이 설명되어 있습니다.

시간 표시 막대 세분성	설명
연도	시간 표시 막대 세분성이 연도인 경우 유효 기간을 연도 형식 yyyy(예: 2010)로 지정할 수 있습니다. 레코드의 유효 시작 날짜는 해당 연도가 시작될 때 시작되고 유효 종료 날짜는 해당 연도가 끝날 때 종료됩니다. 예를 들어 유효 시작 날짜가 2013이고 유효 종료 날짜가 2014인 경우 레코드가 01/01/2013부터 31/12/2014까지 유효합니다.
월	시간 표시 막대 세분성이 월인 경우 유효 기간을 월 형식 mm/yyyy(예: 01/2013)로 지정할 수 있습니다. 레코드의 유효 시작 날짜는 해당 월의 첫 번째 날에 시작됩니다. 레코드의 유효 종료 날짜는 해당 월의 마지막 날에 종료됩니다. 예를 들어 유효 시작 날짜가 02/2013이고 유효 종료 날짜가 04/2013인 경우 레코드가 01/02/2013부터 30/04/2013까지 유효합니다.
일	시간 표시 막대 세분성이 일인 경우 유효 기간을 날짜 형식 dd/mm/yyyy(예: 13/01/2013)로 지정할 수 있습니다. 레코드의 유효 시작 날짜는 해당 일의 시작 시간, 즉 12:00에 시작됩니다. 레코드의 유효 종료 날짜는 해당 일의 종료 시간, 즉 23:59에 종료됩니다. 예를 들어 유효 시작 날짜가 13/01/2013이고 유효 종료 날짜가 15/04/2013인 경우 레코드가 13/01/2013 12:00부터 15/04/2013 23:59까지 유효합니다.
시간	시간 표시 막대 세분성이 시간인 경우 유효 기간에 연도, 월, 일 및 시간이 포함됩니다. 시간 표시 막대 형식은 dd/mm/yyyy hh(예: 13/01/2013 15)입니다. 레코드의 유효 시작 날짜는 하루 중 시작 시간에 시작됩니다. 레코드의 유효 종료 날짜는 사용자가 지정한 시간의 마지막에 종료됩니다. 예를 들어 유효 시작 날짜가 13/01/2013 15이고 유효 종료 날짜가 15/04/2013 10인 경우 레코드가 13/01/2013 15:00부터 15/04/2013 10:59까지 유효합니다.
분	시간 표시 막대 세분성이 분인 경우 유효 기간에 연도, 월, 일, 시간 및 분이 포함됩니다. 시간 표시 막대 형식은 dd/mm/yyyy hh:mm(예: 13/01/2013 15:30)입니다. 레코드의 유효 시작 날짜는 분이 시작될 때 시작됩니다. 레코드의 유효 종료 날짜는 사용자가 지정한 분의 마지막에 종료됩니다. 예를 들어 유효 시작 날짜가 13/01/2013 15:30이고 유효 종료 날짜가 15/04/2013 10:45인 경우 레코드가 13/01/2013 15:30:00부터 15/04/2013 10:45:59까지 유효합니다.
초	시간 표시 막대 세분성이 초인 경우 유효 기간에 연도, 월, 일, 시간, 분 및 초가 포함됩니다. 시간 표시 막대 형식은 dd/mm/yyyy hh:mm:ss(예: 13/01/2013 15:30:45)입니다. 레코드의 유효 시작 날짜는 초가 시작될 때 시작됩니다. 레코드의 유효 종료 날짜는 사용자가 지정한 초의 마지막에 종료됩니다. 예를 들어 유효 시작 날짜가 13/01/2013 15:30:55이고 유효 종료 날짜가 15/04/2013 10:45:15인 경우 레코드가 13/01/2013 15:30:55:00부터 15/04/2013 10:45:15:00까지 유효합니다.

기록 및 상태 관리

시간 표시 막대가 활성화된 기본 개체 및 연결된 관계 기본 개체의 경우 기록 및 상태 관리가 기본적으로 활성화되어 있습니다.

기본 개체에 대한 시간 표시 막대가 활성화된 경우 기본 개체에 대한 기록 또는 상태를 비활성화할 수 없습니다. **MDM Hub**은 교차 참조 테이블과 연결된 기록 테이블에서 시간 표시 막대와 관련된 변경 기록을 유지 관리합니다. **MDM Hub**은 연결된 교차 참조 테이블에서 레코드의 상태(예: 활성, 보류 중 또는 삭제됨)를 관리합니다.

업데이트하는 레코드에 승인이 필요하면 데이터 변경 이벤트가 발생하며 **MDM Hub**은 이 레코드를 보류 중 상태로 저장합니다. 업데이트에 기인한 데이터 변경 이벤트는 상호 참조 버전에 영향을 줍니다. **MDM Hub**은 상태 관리 상호 작용 ID를 사용하여 상호 참조 버전을 수정하지 못하도록 합니다. 연결된 상호 참조 버전에 영향이 미치는 것은 동일한 소스 시스템에서 기간이 겹치는 데서 기인하는 것일 수도 있습니다. 보류 중 상태인 상호 참조를 승격하면 상호 참조 레코드 상태가 활성으로 변경됩니다. 레코드를 일시 삭제하면 상태가 삭제됨으로 변경됩니다.

시간 표시 막대 적용 규칙

시간 표시 막대 정보를 정의하고 유지할 때 비즈니스 항목의 시간 표시 막대 및 관계를 관리하기 위해 **MDM Hub**에서는 미리 정의된 시간 표시 막대 규칙을 적용합니다. **MDM Hub**은 기간 시작 및 종료 날짜에 다음과 같은 규칙 집합을 적용하여 로드 및 Put 작업 동안 유효 기간을 관리합니다.

어느 시점에서든 **MDM Hub**는 유효 시작 날짜와 유효 종료 날짜를 기준으로 한 버전의 레코드만 유효한 것으로 간주합니다. 일괄 처리, 서비스 통합 프레임워크 또는 **Informatica Data Director**를 사용하여 데이터를 변경하면 **MDM Hub**는 현재 유효한 데이터를 유지합니다. 또한 여러 시스템에서 기본 개체 레코드에 데이터를 제공하는 경우 **MDM Hub**는 데이터를 제공하는 유효한 레코드를 기준으로 레코드 버전을 업데이트하는 규칙을 적용합니다.

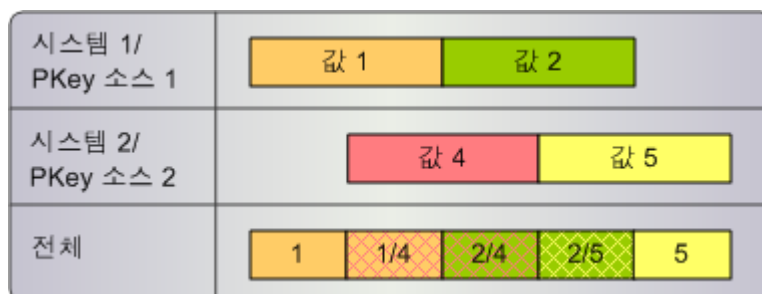
또한 사용자 종료일을 사용하여 시간 표시 막대 및 유효 날짜를 관리하기 위한 사용자 지정 규칙을 정의하고 적용할 수 있습니다.

유효 기간 계산

MDM Hub에서 기본 개체 레코드의 전체 유효 기간을 계산합니다.

데이터 변경 이벤트를 추적하는 기본 개체에 대해 로드 작업이나 Put 작업을 사용할 경우 소스 레코드마다 기간 시작 날짜 및 종료 날짜를 지정하십시오. 기본 개체 레코드는 여러 소스 시스템 또는 기본 키 소스에, 또는 양쪽에 제공자가 있을 수 있습니다. **MDM Hub**는 모든 소스 시스템의 레코드 유효 기간을 집계하여 기본 개체 레코드의 전체 유효 기간을 계산합니다.

다음 그림은 두 개의 소스 시스템 즉, PKey 1을 포함한 System 1과 PKey 2를 포함한 System 2의 제공자가 포함된 기본 개체를 보여 줍니다.



위의 그림에서 접치는 서로 다른 유효 시간에 대해 System 1에는 values 1 및 2가 있고 System 2에는 values 4 및 5가 있습니다. MDM Hub는 두 소스 시스템의 레코드 유효 기간을 집계하여 레코드의 전체 유효 기간을 계산합니다. 레코드에 대한 유효 기간 집계 결과는 각 결과 유효 기간의 BVT(최선의 진실, Best Version of Truth)입니다. 그림에 나타난 전체 유효 값 1, 1/4, 2/4, 2/5 및 5는 특정 유효 기간의 BVT입니다.

참고: 유효 기간 관리 규칙은 변경할 수 없습니다.

규칙 1. 유효 기간 없는 레코드 추가

교차 참조 레코드에 기간 시작 날짜 및 기간 종료 날짜가 없고 기존 교차 참조 레코드가 없을 경우 레코드가 항상 유효합니다.

다음 그림은 기간 시작 날짜 및 기간 종료 날짜가 없고 기존 교차 참조 레코드가 없는 상태로 삽입된 교차 참조 레코드를 보여 줍니다.

이전	레코드 없음
변경	초기값
이후	초기값

규칙 2. 유효 기간 없는 레코드 추가

교차 참조 레코드에 기간 시작 날짜 및 기간 종료 날짜가 있고 기존 교차 참조 레코드가 없을 경우 MDM Hub에서 레코드를 삽입하고, 유효 기간은 사용자가 지정합니다.

다음 그림은 기간 시작 날짜 및 기간 종료 날짜가 있고 기존 교차 참조 레코드는 없는 상태로 삽입된 교차 참조 레코드를 보여 줍니다.

이전	레코드 없음
변경	초기값
이후	초기값

규칙 3. 유효 기간에 대한 레코드 버전 추가

기존 유효 기간에 대한 레코드 버전을 추가할 수 있습니다.

다음 조건이 있을 경우 규칙 3이 적용됩니다.

- 기간 시작 및 기간 종료 날짜가 지정되어 있고 유효 시간이 동일한 교차 참조 레코드가 존재합니다.
- 삽입된 교차 참조 레코드 및 기존 교차 참조 레코드가 동일한 소스 시스템에 속합니다.

규칙 3에 따라 MDM Hub에서 다음 동작을 수행합니다.

- 교차 참조 레코드가 업데이트됩니다.

- 유효 기간이 변경되지 않습니다.

다음 이미지는 유효 기간이 같은 교차 참조 레코드가 있는 상태에서 기간 시작 날짜 및 기간 종료 날짜로 삽입된 교차 참조 레코드를 보여 줍니다.

이전	초기값
변경	기타 값
이후	기타 값

규칙 4. 유효 기간이 교차하고 연장된 시작 날짜가 있는 레코드 버전 추가

유효 기간이 교차하고 연장된 시작 날짜가 있는 레코드 버전을 추가할 수 있습니다.

다음 조건이 있을 경우 규칙 4가 적용됩니다.

- 지정된 유효 기간이 기존 교차 참조 레코드의 유효 기간과 교차합니다.
- 변경된 기간 시작 날짜가 기존 교차 참조 레코드의 기간 시작 날짜보다 느립니다.

규칙 4에 따라 MDM Hub에서 다음 동작을 수행합니다.

- 기존 교차 참조 버전의 기간 종료 날짜가 '변경된 기간 시작 날짜 - 선택된 시간 표시 막대 단위의 1'로 업데이트되고 초기 기간 시작 날짜는 변경되지 않습니다.
- 지정된 유효 기간과 함께 새 버전의 교차 참조 레코드가 삽입됩니다.

다음 이미지는 유효 기간이 기존 교차 참조 레코드의 유효 기간과 교차하고 기간 시작 날짜가 나중인 교차 참조 레코드를 보여 줍니다.

이전	초기값
변경	기타 값
이후	초기값 기타 값

규칙 5. 유효 기간이 교차하고 이전 종료 날짜가 있는 레코드 버전 추가

유효 기간이 교차하고 이전 종료 날짜가 있는 레코드 버전을 추가할 수 있습니다.

다음 조건이 있을 경우 규칙 5가 적용됩니다.

- 지정된 유효 기간이 기존 교차 참조 레코드의 유효 기간과 교차합니다.
- 변경된 종료 날짜가 교차 참조의 기존 종료 날짜보다 빠릅니다.

규칙 5에 따라 MDM Hub에서 다음 동작을 수행합니다.

- 기존 교차 참조 버전의 기간 시작 날짜는 '변경된 종료 날짜 + 선택된 시간 표시 막대 측정 단위의 1'로 업데이트됩니다.

- 기존 교차 참조 버전의 원래 기간 종료 날짜는 변경되지 않습니다.
- 새 버전의 교차 참조 레코드가 지정된 유효 기간에 추가됩니다.

다음 그림은 유효 기간이 기존 교차 참조 레코드의 유효 기간과 교차하고 기간 종료 날짜가 빠른 교차 참조 레코드를 보여 줍니다.



규칙 6. 유효 기간 내에 포함된 레코드 버전 추가

유효 기간 내에 포함된 레코드 버전을 추가할 수 있습니다.

다음 조건이 있을 경우 규칙 6이 적용됩니다.

- 새 교차 참조 버전의 유효 기간이 기존 교차 참조 레코드의 유효 기간에 포함됩니다.
- 변경된 기간 시작 날짜가 기존 교차 참조 레코드의 기간 시작 날짜보다 느립니다.
- 변경된 기간 종료 날짜가 기존 교차 참조 레코드의 기간 종료 날짜보다 빠릅니다.

규칙 6에 따라 MDM Hub에서 다음 동작을 수행합니다.

- 기존 교차 참조 버전의 기간 종료 날짜는 '변경된 날짜 - 선택된 시간 표시 막대 측정 단위의 1'로 업데이트됩니다.
- 기존 교차 참조 버전의 시간 시작 날짜는 변경되지 않습니다.
- 지정된 유효 기간과 함께 새 버전의 교차 참조 레코드가 삽입됩니다.
- 두 번째 교차 참조 레코드 버전은 '변경된 종료 날짜 + 선택된 시간 표시 막대 측정 단위의 1'로 설정된 기간 시작 날짜와 함께 삽입됩니다.
- 두 번째 교차 참조 레코드 버전의 종료 날짜가 기존 교차 참조 레코드의 기간 종료 날짜로 설정됩니다.

다음 그림은 유효 기간이 기존 교차 참조 레코드의 유효 기간에 포함되는 교차 참조 레코드를 보여 줍니다.



규칙 7. 유효 기간을 포함하는 레코드 버전 추가

레코드 버전의 기존 유효 기간을 포함할 수 있는 레코드 버전을 추가할 수 있습니다.

다음 조건이 있을 경우 규칙 7이 적용됩니다.

- 기존 교차 참조 버전의 유효 시간이 같은 날짜에 또는 더 느린 날짜에 시작됩니다.
- 기존 교차 참조 버전의 유효 기간이 새 교차 참조 버전과 같은 날짜에 또는 더 빠른 날짜에 종료됩니다.

규칙 7에 따라 MDM Hub에서 다음 동작을 수행합니다.

- 유효 기간이 새 교차 참조 버전의 시작 날짜와 같거나 그 이후에 시작되고 새 교차 참조 버전의 종료 날짜와 같거나 그 이전에 종료되는 기존 교차 참조 버전이 삭제됩니다.
- 지정된 유효 기간과 함께 새 교차 참조 버전이 삽입됩니다.
- 다른 모든 기존 교차 참조 버전의 경우 규칙 4와 5가 적용됩니다.

다음 그림은 유효 기간 시작 날짜는 새 교차 참조 레코드보다 느리고 유효 기간 종료 날짜는 새 교차 참조 레코드보다 빠른 기존 교차 참조를 보여 줍니다.



규칙 8. 비연속 유효 기간이 있는 레코드 버전 추가

비연속 유효 기간이 있는 레코드 버전을 추가할 수 있습니다.

다음 조건이 있을 경우 규칙 8이 적용됩니다.

- 지정된 유효 기간이 기존 교차 참조 레코드의 유효 기간과 연속하지 않습니다.
- 변경된 유효 기간 종료 날짜가 기간 시작 날짜보다 빠르거나 변경된 기간 시작 날짜가 기존 교차 참조 날짜의 기간 종료 날짜보다 느립니다.

규칙 8에 따라 MDM Hub에서 다음 동작을 수행합니다.

- 기존 교차 참조 레코드 버전은 변경되지 않습니다.
- 기본 개체에 비연속 유효 기간이 있는 경우 새 버전의 교차 참조 레코드가 지정된 유효 기간과 함께 삽입됩니다.
- 기본 개체에 연속 유효 기간이 있어야 하는 경우 새 버전의 교차 참조 레코드가 삽입되지 않고 오류가 생성됩니다.

다음 이미지에서는 비연속 유효 기간을 가진 교차 참조 레코드를 보여 줍니다.

이전	초기값
변경 내용	기타 값
이후	초기값 기타 값

규칙 9. 유효 기간 내에 보류 중 상태인 레코드 버전 추가

기존 유효 기간 내에 포함되어야 하는 보류 중 상태의 레코드 버전을 추가할 수 있습니다.

기존 교차 참조 레코드의 유효 기간이 새 **PENDING** 교차 참조 버전의 유효 기간 범위 내에 있으면 규칙 9가 적용됩니다.

규칙 9에 따라 **MDM Hub**에서 다음 동작을 수행합니다.

- 기존 교차 참조 레코드 버전이 변경되지는 않지만 상호 작용 ID로 잠깁니다. 이 ID는 승격 동안 사용됩니다.
- 지정된 유효 기간과 함께 교차 참조 버전이 삽입되고 **PENDING** 상태로 설정됩니다.
- 보류 중인 교차 참조 버전이 승격되면 규칙 1 - 8이 적용됩니다.

다음 그림에서는 기존 교차 참조 레코드의 유효 기간이 새 **PENDING** 교차 참조 버전의 유효 기간 범위 내에 있습니다.

이전	초기값
변경	기타 값
이후	초기값 기타 값

규칙 10. 레코드 버전이 잠긴 경우 레코드 버전 추가

상호 작용 ID로 기존 레코드 버전이 잠긴 경우 레코드 버전을 추가할 수 있습니다.

규칙 1 - 9에 따라 변경된 사항이 상호 작용 ID로 잠긴 기존 교차 참조 레코드에 영향을 미칠 경우 **MDM Hub**에서 변경을 제한합니다.

다음 그림은 상호 작용 ID로 잠긴 교차 참조 레코드가 있을 때 새 교차 참조 레코드가 삽입되면 아무 것도 변경되지 않는다는 것을 보여 줍니다.

이전	<div>초기값 1</div> <div>초기값 2</div>
변경	<div>기타 값</div>
이후	<div>초기값 1</div> <div>초기값 2</div>

규칙 11. 버전이 보류 중 상태인 경우 레코드 버전 추가

기존 버전이 보류 중 상태인 경우 활성 또는 보류 중 상태의 레코드 버전을 추가할 수 있습니다.

다음 조건이 있을 경우 규칙 11이 적용됩니다.

- 기존 교차 참조 레코드가 PENDING 상태입니다.
- 교차 참조 레코드가 ACTIVE 또는 PENDING 상태에서 삽입되었고 교차 참조 레코드의 유효 기간이 잠긴 레코드와 전혀 교차하지 않습니다.

규칙 11에 따라 MDM Hub에서 다음 동작을 수행합니다.

- 기존 교차 참조 버전은 변경되지 않습니다.
- 교차 참조 레코드의 새 버전이 상태 변경 없이 지정된 유효 기간과 함께 삽입됩니다.

다음 이미지는 기존 교차 참조 레코드가 PENDING 상태이고 새 교차 참조 레코드가 PENDING 상태로 삽입된 것을 보여 줍니다.

Before	<div>Initial Value 1</div> <div>Initial Value 2</div>
The Change	<div>Other Value</div>
After	<div>Initial Value 1</div> <div>Initial Value 2</div> <div>Other Value</div>

규칙 12. 연속 기본 개체의 레코드 버전 삭제 또는 업데이트

연속 기본 개체의 레코드 버전 연속성을 손상시키는 레코드 버전을 삭제하거나 업데이트할 수 없습니다.

기본 개체 속성 설정이 비연속 유효 기간을 허용하지 않는 경우 규칙 12에 따라 MDM Hub에서 다음 동작을 수행합니다.

- 연속성을 손상하는 유효 기간은 삭제될 수 없습니다.
- 연속성을 손상하는 교차 참조 레코드의 유효 기간에 대한 변경 내용은 저장되지 않습니다.

다음 이미지는 교차 참조 레코드 버전이 삭제된 전과 후의 연속 유효 기간을 가진 교차 참조 레코드를 보여줍니다.

Before	Initial Value 1	Initial Value 2	Initial Value 3
The Change	Initial Value 1		Initial Value 3
After	Initial Value 1	Initial Value 2	Initial Value 3

규칙 13. 데이터 업데이트

기존 레코드 버전의 데이터를 업데이트할 수 있습니다.

다음 조건이 있을 경우 규칙 13이 적용됩니다.

- 기존 교차 참조 레코드의 유효 기간이 변경되지 않습니다.
- 기존 교차 참조 데이터가 변경됩니다.
- 시간 표시 막대 작업이 1로 설정됩니다.

규칙 13에 따라 MDM Hub에서 다음 동작을 수행합니다.

- 기존 교차 참조 레코드 버전은 변경되지 않습니다.
- 기존 버전의 교차 참조 레코드에 있는 데이터가 업데이트됩니다.

다음 이미지에서는 기존 레코드 버전의 데이터 업데이트를 보여 줍니다.

Before	Initial Value
The Change	Updated Value
After	Updated Value

규칙 14. 유효 기간 업데이트

레코드 버전의 유효 기간을 업데이트할 수 있습니다.

다음 조건이 있을 경우 규칙 14가 적용됩니다.

- 기존 교차 참조 레코드의 유효 기간이 업데이트되어 유효 기간을 늘리거나 줄입니다.
- 기존 교차 참조 레코드의 데이터는 변경되지 않습니다.
- 시간 표시 막대 작업이 2로 설정됩니다.

규칙 14에 따라 MDM Hub에서 다음 동작을 수행합니다.

- 기존 교차 참조 레코드 버전이 업데이트됩니다.
- 기존 교차 참조 레코드 버전에 사용자가 지정한 유효 기간이 있고 데이터는 변경되지 않습니다.

- 기본 개체에 비연속 유효 기간이 있을 수 있고 레코드 버전의 유효 기간을 늘린 경우, 앞에 인접한 레코드 버전과 뒤에 인접한 레코드 버전의 유효 기간이 줄어듭니다. 레코드 버전의 겹침을 방지하기 위해 MDM Hub은 인접한 레코드 버전의 유효 기간을 줄입니다.
- 기본 개체에 비연속 유효 기간이 있을 수 있고 레코드 버전의 유효 기간을 줄인 경우 MDM Hub에서 레코드 버전 간에 공백을 생성합니다.
- 기본 개체에 연속 유효 기간이 있어야 하는 경우 MDM Hub에서 인접한 레코드의 유효 기간을 연장하거나 줄여서 연속성을 유지합니다.

다음 이미지에서는 기본 개체에 대한 연속성이 활성화된 경우 유효 기간에 대한 업데이트를 보여 줍니다.

이전	초기 값 1	초기 값 2
변경	업데이트된 값 1	
이후	업데이트된 값 1	업데이트된 값 2

규칙 15. 유효 기간 추가

새 유효 기간이 있는 레코드 버전을 추가할 수 있습니다.

다음 조건이 있을 경우 규칙 15가 적용됩니다.

- 새 유효 기간이 있는 레코드 버전이 교차 참조 테이블에 추가됩니다.
- 지정된 유효 기간으로 인해 유효 기간 간에 공백이 생성됩니다.
- 기본 개체 속성 설정에서 비연속 유효 기간을 허용하지 않습니다.
- 시간 표시 막대 작업이 4로 설정됩니다.

규칙 15에 따라 MDM Hub에서 다음 동작을 수행합니다.

- 새 유효 기간이 있는 교차 참조 레코드가 로드됩니다.
- 연속성이 깨진 경우 간격을 채우려면 기존 레코드 버전의 유효 기간이 연장되어 레코드 버전 간 간격을 채웁니다.
- 연속성이 깨진 경우 간격을 채우지 않으려면 새 레코드 버전이 삽입되지 않고 오류가 생성됩니다. 일괄 로드 중에 레코드가 거부 테이블로 이동합니다.

다음 이미지에서는 연결된 기본 개체에 대한 연속성이 활성화된 경우 교차 참조 테이블에 레코드를 추가하는 작업을 보여 줍니다.

이전	초기 값	
변경		기타 값
이후	초기 값	기타 값

기본 개체 시간 표시 막대 구성

기본 개체에서 레코드에 대한 시간 표시 막대를 구성하려면 MDM Hub 및 Hub 서버 속성 파일에서 시간 표시 막대를 활성화하십시오.

1. Hub 콘솔에서 기본 개체에 대한 시간 표시 막대를 활성화합니다.
2. MDM Hub 속성 파일을 구성합니다.

1단계. Hub 콘솔에서 시간 표시 막대 활성화

시간 표시 막대가 필요한 각 기본 개체에 대해 시간 표시 막대를 활성화합니다.

1. 모델 작업 영역을 열고 **스키마**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 기본 개체를 선택합니다.
스키마 관리자에 기본 개체 속성 페이지의 기본 탭이 표시됩니다.
4. 시간 표시 막대 목록에서 **동적 시간 표시 막대**를 선택합니다.
5. 유효 기간을 구성합니다.
 - 레코드에 대해 연속 유효 기간을 허용해야 할 경우 **연속 유효 기간 허용** 확인란을 비활성화합니다.
 - 레코드에 대해 비연속 유효 기간을 허용해야 할 경우 **비연속 유효 기간 허용** 확인란을 활성화합니다.
6. **저장**을 클릭합니다.

MDM Hub에서 기본 개체에 대해 시간 표시 막대를 활성화합니다.

2단계. 속성 파일 구성

기본 개체에 대한 시간 표시 막대를 활성화한 경우 BVT(최선의 진실, Best Version of the Truth) 계산 중에 사용할 최대 레코드 수를 구성합니다. BVT 계산을 위한 유효 날짜를 지정한 경우 SearchQuery, SearchHMQQuery, GetOneHop 및 GetEntityGraph SIF API에서 계산 중 사용자가 구성하는 최대 레코드 수를 사용합니다.

1. 다음 디렉터리에서 `cmxserver.properties` 파일을 엽니다.
UNIX의 경우. `<infamdm 설치 디렉터리>/hub/server/resources`
Windows의 경우. `<infamdm 설치 디렉터리>\hub\server\resources`
2. 다음 속성을 추가합니다.
`searchQuery.buildBvtTemp.MaxRowCount`
`sif.search.result.querytemptableTimeToLive.seconds`
`searchQuery.buildBvtTemp.MaxRowCount`에 대한 기본값은 10000입니다.
`sif.search.result.querytemptableTimeToLive.seconds`에 대한 기본값은 30입니다.
3. 파일을 저장하고 닫습니다.

일괄 작업으로 여러 레코드 버전 로드

하나의 일괄 작업으로 준비 테이블에서 기본 개체와 연결된 교차 참조 테이블로 여러 버전의 레코드를 로드할 수 있습니다. 레코드의 유효 기간 연속성을 위한 기본 개체 속성을 활성화했는지 확인하십시오.

MDM Hub이 여러 레코드 버전을 로드할 때 레코드의 유효 기간 연속성을 깨뜨리는 레코드 버전을 거부합니다. 레코드의 유효 기간 연속성에 영향을 미치지 않고 여러 레코드 버전을 로드하려면 연속성을 유지하는 순서대로 레코드 버전을 로드하도록 MDM Hub을 구성하십시오.

여러 버전의 레코드를 로드하기 위한 로드 일괄 설정

단일 로드 일괄 작업으로 여러 버전의 레코드를 로드하도록 MDM Hub을 구성합니다.

1. 기본 개체 레코드에 대한 연속 유효 기간을 구성합니다.
 - a. 모델 작업 영역을 열고 **스키마**를 클릭합니다.
 - b. 쓰기 잠금을 획득합니다.
 - c. 스키마 트리에서 기본 개체를 선택합니다.
스키마 관리자에 기본 개체 속성 페이지의 기본 탭이 표시됩니다.
 - d. **비연속 유효 기간 허용** 확인란을 비활성화합니다.
2. 기본 개체에 로드할 레코드 버전의 순서를 구성합니다.
 - a. 다음 디렉터리에서 `cmxserver.properties` 파일을 엽니다.
UNIX의 경우. <infamdm 설치 디렉터리>/hub/server/resources
Windows의 경우. <infamdm 설치 디렉터리>\hub\server\resources
 - b. 다음 속성을 추가합니다.
`cmx.server.batch.load.smart_resequencing = true`
 - c. 파일을 저장하고 닫습니다.

다중 레코드 버전의 일괄 로드 예제

조직에 데이터 변경 이벤트를 추적하는 CUSTOMER 기본 개체가 있습니다. CUSTOMER 기본 개체에는 2014년 3월 21일 현재 뉴욕에 거주하고 있고 2015년 1월 15일까지 뉴욕에 거주할 John Smith에 대한 레코드가 포함되어 있습니다. 과거 및 미래의 다른 유효 기간에 대한 레코드 버전을 로드하고자 합니다.

다음 유효 기간을 가진 레코드를 준비 테이블에서 CUSTOMER 기본 개체와 연결된 교차 참조 테이블로 로드하고자 합니다.

- 2011년 1월 31일 - 2013년 10월 20일
- 2013년 10월 21일 - 2014년 3월 20일
- 2015년 1월 16일 - 2015년 11월 24일
- 2015년 11월 25일 - null

여러 버전의 레코드를 단일 로드 일괄 작업에 로드하려면 기본 개체에 레코드의 유효 기간 연속성을 구성하십시오. 또한 MDM Hub가 기존 레코드 버전과 연속성을 유지하는 순서로 레코드 버전을 로드하도록 활성화하십시오.

로드 일괄 작업 전 교차 참조에는 John Smith의 뉴욕 거주를 나타내는 레코드가 1개 있습니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	뉴욕	2014년 3월 21일	2015년 1월 15일

로드 일괄 작업 후 교차 참조 테이블에는 John Smith에 대한 연속 레코드 버전이 여러 개 있습니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
3	2	25	John Smith	로스앤젤레스	2011년 1월 31일	2013년 10월 20일
2	2	25	John Smith	샌프란시스코	2013년 10월 21일	2014년 3월 20일
1	2	25	John Smith	뉴욕	2014년 3월 21일	2015년 1월 15일
4	2	25	John Smith	오스틴	2015년 1월 16일	2015년 11월 24일
5	2	25	John Smith	라스베이거스	2015년 11월 25일	null

여러 버전의 레코드가 로드된 후에는 레코드에서 John Smith가 과거에는 로스앤젤레스와 샌프란시스코에서 거주했고, 지금은 뉴욕에 거주하며, 앞으로 오스틴과 라스베이거스에서 거주할 것을 표시합니다.

레코드 버전의 유효 기간 편집

레코드 버전의 유효 기간을 편집할 수 있습니다. 레코드 버전에 대한 잘못된 값을 저장한 경우 유효 기간 값을 편집합니다. 레코드 버전의 시작 날짜 및 종료 날짜를 변경하여 레코드의 유효 기간을 변경할 수 있습니다.

레코드 버전의 유효 기간이 사용자가 지정한 날짜보다 이후에 시작되거나 이전에 종료되도록 편집할 경우 유효 기간이 감소됩니다. 레코드 버전의 유효 기간이 사용자가 지정한 날짜보다 이전에 시작되거나 이후에 종료되도록 편집할 경우 유효 기간이 증가됩니다.

연결된 기본 개체에서 레코드의 연속성이 활성화된 경우 레코드 버전의 유효 기간을 편집할 수 있습니다. 연속성을 유지하기 위해 편집한 레코드 버전과 인접한 레코드 버전의 유효 기간을 MDM Hub에서 연장하거나 줄입니다. 기본 개체에 연속하지 않는 유효 기간이 있는 경우 유효 기간을 연장하면 인접한 레코드 버전이 영향을 받습니다.

로드 일괄 작업 중에 레코드 버전의 유효 기간을 편집하려면 다음 설정을 수행하십시오.

- **TIMELINE_ACTION.** TIMELINE_ACTION 준비 테이블 열의 값을 2로 설정합니다. 속성을 2로 설정한 경우 MDM Hub에서 레코드 버전의 유효 기간을 편집할 수 있습니다. TIMELINE_ACTION 열이 해당하는 랜딩 테이블 열에서 매핑됩니다.
- **TIMELINE_FILL_ON_GAP.** 레코드 버전의 유효 기간을 편집할 경우 레코드 버전의 유효 날짜 간 연속성이 유지되는지 확인합니다. 연산 참조 저장소의 C_REPOS_TABLE 또는 Hub 콘솔의 준비 테이블 속성에 속성을 설정합니다. true로 설정한 경우 새 레코드 버전을 기본 개체에 추가할 수 있을 때 MDM Hub에서 레코드 버전의 유효 날짜 간 연속성을 유지 관리합니다. false로 설정한 경우 MDM Hub에서 레코드 버전의 유효 기간 사이의 연속성을 깨뜨리는 레코드 버전의 추가를 거부합니다. 기본값은 false입니다.

참고: 상태 관리 재정의의 시스템의 데이터 또는 레코드가 생성된 소스 시스템을 통해 유효 기간을 편집할 수 있습니다.

레코드 버전의 유효 기간 늘리기

종료 날짜 또는 시작 날짜를 편집하여 레코드 버전의 유효 기간을 늘릴 수 있습니다. 기본 개체 레코드 버전이 연속된 경우 MDM Hub에서 사용자가 편집한 레코드 버전과 인접한 레코드 버전의 유효 기간을 줄입니다.

종료 날짜를 연장할 경우 유효 기간을 늘릴 수 있습니다. 종료 날짜를 연장한 후 편집한 레코드 버전이 인접한 레코드 버전과 겹칠 수 있습니다. 편집한 레코드 버전 뒤에 있는 레코드 버전과 겹칩니다. MDM Hub에서 인접한 레코드 버전의 시작 날짜를 연장합니다. 레코드 버전이 겹치지 않으면서 연속성을 보장하기 위해 인접한 레코드 버전의 유효 기간이 줄어듭니다.

시작 날짜를 이전 날짜로 이동할 경우 유효 기간을 늘릴 수 있습니다. 시작 날짜를 이동하면 레코드 버전이 인접한 레코드 버전과 겹칠 수 있습니다. 편집한 레코드 버전 앞에 있는 레코드 버전과 겹칩니다. MDM Hub에서 인접한 레코드 버전의 종료 날짜를 이전 날짜로 이동합니다. 레코드 버전이 겹치지 않으면서 연속성을 보장하기 위해 인접한 레코드 버전의 유효 기간이 줄어듭니다.

기본 개체에서 레코드의 연속성을 활성화하지 않은 경우 레코드 버전이 겹치지 않으면 인접한 레코드 버전이 변경되지 않은 채 남아 있습니다.

유효 기간 늘리기 예제

조직에 데이터 변경 이벤트를 추적하는 CUSTOMER 기본 개체가 있습니다. CUSTOMER 기본 개체와 연결된 교차 참조 테이블에는 2014년 3월 21일부터 2014년 11월 30일까지 뉴욕에 거주하는 John Smith에 대한 레코드가 포함되어 있습니다. 또한 교차 참조 테이블에는 2014년 12월 1일부터 유효한 다른 레코드 버전이 포함되어 있습니다. 2014년 3월 21일부터 2014년 11월 30일까지 유효한 레코드 버전을 최대 2015년 2월 28일까지 유효하도록 편집하려고 합니다.

첫 번째 레코드 버전을 편집하기 전에 John Smith에 대한 두 번째 레코드 버전은 2014년 12월 1일부터 유효합니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	뉴욕	2014년 3월 21일	2014년 11월 30일
2	2	25	John Smith	로스앤젤레스	2014년 12월 1일	null

레코드 버전의 유효 기간이 2014년 11월 30일 대신 2015년 2월 28일에 끝나도록 편집할 경우 유효 기간이 연장됩니다. 레코드 버전의 겹침을 방지하기 위해 인접한 레코드 버전의 시작 날짜가 2014년 12월 1일에서 2015년 3월 1일로 변경됩니다.

첫 번째 레코드 버전의 시작 날짜를 연장하면 MDM Hub에서 첫 번째 레코드 버전이 끝난 후 두 번째 레코드 버전이 시작하도록 업데이트합니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	뉴욕	2014년 3월 21일	2015년 2월 28일
2	2	25	John Smith	로스앤젤레스	2015년 3월 1일	null

레코드 버전의 유효 기간 줄이기

시작 날짜 또는 종료 날짜를 편집하여 레코드 버전의 유효 기간을 줄일 수 있습니다. 기본 개체 레코드 버전이 연속된 경우 MDM Hub에서 사용자가 편집한 레코드 버전과 인접한 레코드 버전의 유효 기간을 연장합니다.

시작 날짜를 연장할 경우 유효 기간을 줄일 수 있습니다. 편집한 레코드 버전 앞에서 인접하는 레코드 버전이 있는 경우 레코드 버전 간에 공백이 생성됩니다. MDM Hub에서 레코드 버전의 종료 날짜를 연장하고 연속성을 유지하기 위해 간격이 생성됩니다.

종료 날짜를 이전 날짜로 이동하여 유효 기간을 줄일 수 있습니다. 편집한 레코드 버전 뒤에서 인접하는 레코드 버전이 있는 경우 레코드 버전 간에 공백이 생성됩니다. 연속성을 보장하기 위해 MDM Hub에서 인접한 레코드 버전의 시작 날짜를 이전 날짜로 이동합니다.

기본 개체에서 레코드의 연속성을 활성화하지 않은 경우 인접 레코드 버전이 변경되지 않은 채 남아 있습니다.

유효 기간 줄이기 예제

조직에 데이터 변경 이벤트를 추적하는 CUSTOMER 기본 개체가 있습니다. CUSTOMER 기본 개체에는 2014년 3월 21일 현재 뉴욕에 거주하는 John Smith에 대한 레코드가 포함되어 있습니다. 유효 기간의 연속성을 관리하고자 합니다. 2014년 12월 1일부터 유효한 레코드 버전을 로드합니다. 마지막으로 2014년 12월 1일부터 유효한 레코드 버전을 2015년 3월 1일부터 유효하도록 편집하고자 합니다.

John Smith의 교차 참조 레코드에서 John Smith가 2014년 3월 21일부터 뉴욕에 거주하고 있음을 보여 줍니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	뉴욕	2014년 3월 21일	null

2014년 12월 1일부터 유효한 레코드 버전을 로드할 경우 기존 레코드 버전이 2014년 11월 30일에 종료됩니다. 겹침을 방지하고 연속성을 유지하기 위해 기존 레코드 버전의 유효 기간이 감소합니다.

2014년 12월 1일부터 유효한 레코드 버전을 로드하면 교차 참조 테이블에서 John Smith의 기존 레코드 버전이 2014년 11월 30일에 종료됨을 표시합니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	뉴욕	2014년 3월 21일	2014년 11월 30일
2	2	25	John Smith	로스앤젤레스	2014년 12월 1일	null

2014년 12월 1일부터 유효한 레코드 버전을 2015년 3월 1일부터 유효하도록 편집할 경우 유효 기간이 감소합니다. 또한 2개 레코드 버전의 유효 기간 간에 공백이 발생합니다. MDM Hub에서 사용자가 편집한 버전과 인접한 레코드 버전의 종료 날짜를 연장합니다. 종료 날짜가 2014년 11월 30일에서 2015년 2월 28일로 변경되어 간격을 채웁니다.

레코드 버전의 유효 날짜를 2014년 12월 1일에서 2015년 3월 1일로 편집한 후에는 다음과 같습니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	뉴욕	2014년 3월 21일	2015년 2월 28일
2	2	25	John Smith	로스앤젤레스	2015년 3월 1일	null

레코드 버전 추가

새로운 기간 동안 유효한 레코드 버전을 추가할 수 있습니다.

기본 개체에서 레코드의 연속성을 활성화한 경우 MDM Hub에서 레코드 버전에 연속된 유효 날짜가 있는지 확인합니다. 연속성을 유지하기 위해 추가된 레코드 버전과 인접한 레코드 버전의 유효 기간을 MDM Hub에서 연장합니다.

새 레코드 버전 및 C_REPOS_TABLE 열을 추가하여 레코드 버전의 연속성을 보장하도록 준비 테이블 열을 구성하십시오.

로드 일괄 작업 중에 레코드 버전을 추가하려면 다음 설정을 구성하십시오.

- **TIMELINE_ACTION.** TIMELINE_ACTION 준비 테이블 열의 값을 4로 설정합니다. 속성을 4로 설정한 경우 MDM Hub에서 새 레코드 버전을 추가할 수 있습니다. TIMELINE_ACTION 열이 해당하는 랜딩 테이블 열에서 매핑됩니다.
- **TIMELINE_FILL_ON_GAP.** 새 레코드 버전을 추가할 경우 레코드 버전의 유효 날짜 간 연속성이 유지되는지 확인합니다. 연산 참조 저장소의 C_REPOS_TABLE 또는 Hub 콘솔의 준비 테이블 속성에 속성을 설정합니다. true로 설정한 경우 새 레코드 버전을 기본 개체에 추가할 수 있을 때 MDM Hub에서 레코드 버전의 유효 날짜 간 연속성을 유지 관리합니다. false로 설정한 경우 MDM Hub에서 레코드 버전의 유효 기간 사이의 연속성을 깨뜨리는 레코드 버전의 추가를 거부합니다. 기본값은 false입니다.

레코드 버전 추가 예제

조직에 데이터 변경 이벤트를 추적하는 CUSTOMER 기본 개체가 있습니다. CUSTOMER 기본 개체에는 2014년 3월 21일 현재 뉴욕에 거주하는 John Smith에 대한 레코드가 포함되어 있습니다. 유효 기간의 연속성을 관리하고자 합니다. 2014년 12월 1일부터 유효한 레코드 버전을 로드합니다.

새 레코드 버전을 로드하기 전에는 John Smith의 레코드에서 2014년 3월 21일부터 뉴욕에 거주하고 있음을 표시합니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	뉴욕	2014년 3월 21일	null

2014년 12월 1일부터 유효한 레코드 버전을 로드할 경우 인접한 레코드 버전의 유효 기간이 줄어들습니다. MDM Hub에서 기존 레코드의 유효 종료 날짜를 조정하여 레코드 버전의 겹침을 방지하고 연속성을 유지합니다.

2014년 12월 1일부터 유효한 레코드 버전을 로드하면 교차 참조 테이블에서 John Smith의 기존 레코드 버전이 2014년 11월 30일에 종료됨을 표시합니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	뉴욕	2014년 3월 21일	2014년 11월 30일
2	2	25	John Smith	로스앤젤레스	2014년 12월 1일	null

레코드의 데이터 업데이트

레코드의 데이터를 업데이트하면 MDM Hub가 레코드의 데이터를 변경합니다. 이 경우에는 새 레코드 버전이 작성되지 않습니다.

레코드 데이터 업데이트 예제

조직에 데이터 변경 이벤트를 추적하는 CUSTOMER 기본 개체가 있습니다. CUSTOMER 기본 개체에는 2014년 3월 21일부터 2015년 1월 15일까지 뉴욕에 거주하는 John Smith에 대한 레코드가 포함되어 있습니다. 거주 도시를 뉴어크로 업데이트하고자 합니다.

데이터 업데이트 전에 John Smith의 교차 참조 레코드에는 John Smith가 뉴욕에 거주하는 것으로 표시됩니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	뉴욕	2014년 3월 21일	2015년 1월 15일

데이터 업데이트 후 John Smith의 교차 참조 레코드에는 John Smith가 뉴어크에 거주하는 것으로 표시됩니다.

Rowid XREF	Rowid 개체	고객 ID	이름 성	구/군/시	기간 시작 날짜	기간 종료 날짜
1	2	25	John Smith	뉴어크	2014년 3월 21일	2015년 1월 15일

MDM Hub에서 기존 레코드 버전을 올바른 도시 이름(뉴어크)으로 업데이트합니다. John Smith 레코드의 유효 기간은 변경되지 않습니다.

관계 업데이트

관계 기본 개체 레코드의 관계 세부 정보를 업데이트하면 MDM Hub가 레코드 버전을 삽입하거나 업데이트합니다.

관계 기본 개체 레코드의 사용자 지정 열에 있는 관계 세부 정보를 업데이트하면 MDM Hub는 관계 레코드에 연결된 교차 참조 레코드를 업데이트합니다.

관계 기본 개체 레코드의 시스템 열에 있는 관계 세부 정보를 업데이트하면 MDM Hub는 연결된 교차 참조 테이블에 새 레코드 버전을 삽입합니다. 시스템 열은 관계 유형, 계층 유형, 기간 시작 날짜 및 기간 종료 날짜 같은 열입니다. 또한 소스 시스템에서 해당하는 교차 참조 레코드가 없는 관계를 업데이트하면 MDM Hub는 새 레코드 버전을 교차 참조 테이블에 추가합니다.

관계 레코드의 사용자 지정 열 업데이트 예제

조직에 Mary Adam에 대한 레코드 및 ABC 조직에 대한 레코드가 포함된 PARTY 기본 개체가 있습니다. Mary Adam과 ABC 조직 사이의 관계는 연결된 PARTY REL 관계 기본 개체에 정의되어 있습니다. 사용자 지정 열인 관계 설명 열의 값을 개인에서 조직으로 업데이트합니다.

참고: IDD 소스 시스템을 사용하도록 Informatica Data Director를 구성했고 업데이트가 계층 보기에서 수행됩니다.

SFDC 소스 시스템에서 업데이트가 발생한 경우

SFDC 소스 시스템 레코드에서 관계 설명 열에 대한 업데이트가 발생한 경우, MDM Hub는 다음과 같은 방법으로 RL PARTY CROSS-REFERENCE 테이블을 업데이트합니다.

- MDM Hub는 기존의 교차 참조 레코드는 변경하지 않습니다.
- MDM Hub는 시작 날짜와 종료 날짜를 변경하지 않고 새 교차 참조 레코드를 삽입합니다.
- MDM Hub는 레코드를 제공하는 소스 시스템에 있는 값으로 사용자 지정 열 및 새 교차 참조 레코드의 Rowid 시스템 열에 있는 값을 업데이트합니다.

RL PARTY CROSS-REFERENCE 테이블에는 해당하는 관계 설명 사용자 지정 열의 값을 변경할 레코드가 표시됩니다.

Rowid 시스템	Rowid 개체	당사자 ID1	당사자 ID2	관계 설명	기간 시작 날짜	기간 종료 날짜
SFDC	414722	2402007	2402147	개인	null	null

관계 설명 열의 값을 개인에서 조직으로 변경하면 RL PARTY CROSS-REFERENCE 테이블에 기존 레코드와 새 레코드가 표시됩니다.

Rowid 시스템	Rowid 개체	당사자 ID1	당사자 ID2	관계 설명	기간 시작 날짜	기간 종료 날짜
SFDC	414722	2402007	2402147	개인	null	null
IDD	414722	2402007	2402147	조직	null	null

IDD 소스 시스템에서 업데이트가 발생한 경우

IDD 소스 시스템의 레코드에서 관계 설명 열에 대한 업데이트가 발생한 경우, MDM Hub는 다음과 같은 방법으로 RL PARTY CROSS-REFERENCE 테이블을 업데이트합니다.

- MDM Hub가 기존 교차 참조 레코드의 사용자 지정 열에 있는 값을 업데이트합니다.
- MDM Hub는 교차 참조 레코드의 시작 날짜와 종료 날짜를 변경하지 않습니다.

RL PARTY CROSS-REFERENCE 테이블에는 해당하는 관계 설명 사용자 지정 열의 값을 변경할 레코드가 표시됩니다.

Rowid 시스템	Rowid 개체	당사자 ID1	당사자 ID2	관계 설명	기간 시작 날짜	기간 종료 날짜
IDD	414722	2402007	2402147	개인	null	null

관계 설명 열의 값을 개인에서 조직으로 변경하면 업데이트된 레코드가 RL PARTY CROSS-REFERENCE 테이블에 표시됩니다.

Rowid 시스템	Rowid 개체	당사자 ID1	당사자 ID2	관계 설명	기간 시작 날짜	기간 종료 날짜
IDD	414722	2402007	2402147	조직	null	null

관계 레코드의 시스템 열 업데이트 예제

조직에 Mary Adam에 대한 레코드 및 ABC 조직에 대한 레코드가 포함된 PARTY 기본 개체가 있습니다. Mary Adam과 ABC 조직 사이의 관계는 연결된 PARTY REL 관계 기본 개체에 정의되어 있습니다. 기간 시작 날짜와 기간 종료 날짜를 업데이트합니다.

참고: IDD 소스 시스템을 사용하도록 Informatica Data Director를 구성했고 업데이트가 계층 보기에서 수행됩니다.

기간 시작 날짜를 null에서 2016년 1월 1일로 업데이트하고 기간 종료 날짜를 null에서 2017년 1월 1일로 업데이트합니다.

SFDC 소스 시스템에서 업데이트가 발생한 경우

기간 시작 날짜 및 기간 종료 날짜에 대한 업데이트가 SFDC 소스 시스템 레코드에서 발생한 경우, MDM Hub는 다음과 같은 방법으로 RL PARTY CROSS-REFERENCE 테이블을 업데이트합니다.

- MDM Hub는 기존의 교차 참조 레코드는 변경하지 않습니다.
- MDM Hub는 시작 날짜가 2016년 1월 1일로 설정되고 종료 날짜가 2017년 1월 1일로 설정된 새 교차 참조 레코드를 삽입합니다.

RL PARTY CROSS-REFERENCE 테이블에는 해당하는 기간 시작 날짜와 기간 종료 날짜를 변경할 활성 레코드가 표시됩니다.

Rowid 시스템	Rowid 개체	당사자 ID1	당사자 ID2	Hub 상태 표시기	관계 설명	기간 시작 날짜	기간 종료 날짜
SFDC	414722	2402007	2402147	활성	개인	null	null

기간 시작 날짜와 기간 종료 날짜를 변경하면 RL PARTY CROSS-REFERENCE 테이블에 기존 레코드와 새 레코드가 표시됩니다.

Rowid 시스템	Rowid 개체	당사자 ID1	당사자 ID2	Hub 상태 표시기	관계 설명	기간 시작 날짜	기간 종료 날짜
SFDC	414722	2402007	2402147	활성	개인	null	null
IDD	414722	2402007	2402147	활성	개인	2016년 1월 1일	2017년 1월 1일

IDD 소스 시스템에서 업데이트가 발생한 경우

기간 시작 날짜 및 기간 종료 날짜에 대한 업데이트가 IDD 소스 시스템 레코드에서 발생한 경우, MDM Hub는 다음과 같은 방법으로 RL PARTY CROSS-REFERENCE 테이블을 업데이트합니다.

- MDM Hub는 기존의 교차 참조 레코드는 변경하지 않습니다.
- MDM Hub는 종료 날짜가 2015년 12월 31일로 설정된 새 교차 참조 레코드를 삽입합니다.

- MDM Hub는 시작 날짜가 2016년 1월 1일로 설정되고 종료 날짜가 2017년 1월 1일로 설정된 새 교차 참조 레코드를 삽입합니다.
- MDM Hub는 시작 날짜가 2017년 1월 2일로 설정되고 종료 날짜가 null로 설정된 새 교차 참조 레코드를 삽입합니다.

RL PARTY CROSS-REFERENCE 테이블에는 해당하는 기간 시작 날짜와 기간 종료 날짜를 변경할 활성 레코드가 표시됩니다.

Rowid	시스템	Rowid	개체	당사자 ID1	당사자 ID2	Hub	상태 표시기	관계 설명	기간 시작 날짜	기간 종료 날짜
IDD		414722		2402007	2402147	활성		개인	null	null

기간 시작 날짜와 기간 종료 날짜를 변경하면 RL PARTY CROSS-REFERENCE 테이블에 기존 레코드와 새 레코드 3개가 표시됩니다.

Rowid	시스템	Rowid	개체	당사자 ID1	당사자 ID2	Hub	상태 표시기	관계 설명	기간 시작 날짜	기간 종료 날짜
IDD		414722		2402007	2402147	활성		개인	null	null
IDD		414722		2402007	2402147	활성		개인	null	2015년 12월 31일
IDD		414722		2402007	2402147	활성		개인	2016년 1월 1일	2017년 1월 1일
IDD		414722		2402007	2402147	활성		개인	2017년 1월 2일	null

관계 종료

두 레코드 간의 관계를 종료할 수 있습니다. 관계를 종료하면 MDM Hub가 관계 기본 개체에 연결된 교차 참조 테이블에 새 레코드 버전을 삽입합니다. 또한 MDM Hub가 레코드의 삭제됨 표시기를 -999999999로 설정하고 Hub 상태 표시기를 삭제됨으로 업데이트합니다. 새 레코드 버전의 기간 시작 날짜가 관계를 종료한 날짜로 설정됩니다.

레코드가 상태 관리 재정의 시스템에 속하는 경우 새 레코드 버전의 기간 시작 날짜가 관계를 종료한 날짜에 시간 표시 막대 단위가 1 증가한 날짜로 설정됩니다. 예를 들어 기간 종료 날짜가 2014년 1월 31일이고 시간 표시 막대 세분성이 1일인 경우 상태 관리 재정의 시스템 레코드 버전의 기간 시작 날짜는 2014년 2월 1일입니다.

관계 종료 예제

조직에 Mary Adam에 대한 레코드가 포함된 PARTY라는 기본 개체가 있습니다. PARTY GROUP 기본 개체에는 Mary가 속하는 Adam 가정의 세부 정보가 포함되어 있습니다. PARTY와 PARTY GROUP 기본 개체 간의 관계는 연결된 PARTY GROUP REL 관계 기본 개체에 정의됩니다. 2015년 5월 17일에 Mary가 더 이상 Adam 가정의 일원이 아님을 알립니다. 따라서 Adam 가정과의 관계가 종료되도록 Mary의 레코드를 업데이트해야 합니다.

Mary와 Adam 가정의 관계 종료 날짜를 설정합니다. Mary의 관계 레코드에 대한 종료 날짜를 설정하면 MDM Hub가 관계 기본 개체에 연결된 교차 참조 테이블(RL PARTY GROUP CROSS-REFERENCE)에 삭제됨 상태의 레코드를 추가합니다.

PARTY GROUP REL 관계 기본 개체에 연결된 교차 참조 테이블인 RL PARTY GROUP CROSS-REFERENCE에는 다음 변경 사항이 발생합니다.

- Mary와 Adam 가정 간의 관계에 레코드가 삽입됩니다.
- 이 레코드는 2015년 5월 18일부터 유효합니다.
- 삭제됨 표시기는 -999999999로 설정됩니다.
- Hub 상태 표시기는 삭제됨으로 설정됩니다.

- 원래 레코드의 종료 날짜가 2015년 5월 17일로 업데이트됩니다.

Mary와 Adam 가정의 관계를 종료하기 전에는 유효 기간이 2006년 10월 6일부터 2999년 12월 31일까지인 레코드가 교차 참조 테이블에 표시됩니다.

Rowid	XREF	Rowid	개체	당사자 ID	삭제됨 표시기	Hub 상태 표시기	당사자 그룹 ID	기간 시작 날짜	기간 종료 날짜
63		22		147		활성	28	2006년 10월 6일	2999년 12월 31일

Mary와 Adam 가정 사이의 관계를 종료하면 2015년 5월 18일부터 Mary가 Adam 가정의 멤버가 아니라고 교차 참조 테이블에 표시됩니다.

Rowid	XREF	Rowid	개체	당사자 ID	삭제됨 표시기	Hub 상태 표시기	당사자 그룹 ID	기간 시작 날짜	기간 종료 날짜
63		22		147		활성	28	2006년 10월 6일	2015년 5월 17일
621		22		147	-999999999	삭제됨	28	2015년 5월 18일	

관계 기간 삭제

특정 기간 동안 유효한 관계 레코드 버전을 삭제할 수 있습니다. 관계 레코드 버전을 삭제하면 MDM Hub가 관계 기본 개체에 연결된 교차 참조 테이블에 새 레코드 버전을 삽입합니다. 레코드 버전의 유효 기간은 삭제한 관계 레코드 버전의 유효 기간과 동일합니다. 또한 MDM Hub가 레코드 버전의 삭제됨 표시기를 999999999로 설정하고 Hub 상태 표시기를 삭제됨으로 설정합니다.

관계 삭제 예제

조직에 Mary Adam에 대한 레코드가 포함된 PARTY라는 기본 개체가 있습니다. PARTY GROUP 기본 개체에는 Mary가 속하는 Adam 가정의 세부 정보가 포함되어 있습니다. PARTY와 PARTY GROUP 기본 개체 간의 관계는 연결된 PARTY GROUP REL 관계 기본 개체에 정의됩니다. 2006년 10월 6일부터 유효한 Mary와 Adam 가정 간의 관계를 삭제해야 합니다.

올바르지 않은 Mary의 관계 레코드 버전을 삭제하려고 합니다. Mary의 관계 레코드를 삭제하면 MDM Hub가 PARTY GROUP REL 관계 기본 개체에 연결된 교차 참조 테이블에 삭제됨 상태의 레코드를 추가합니다.

PARTY GROUP REL 관계 기본 개체에 연결된 교차 참조 테이블인 RL PARTY GROUP CROSS-REFERENCE에는 다음 변경 사항이 발생합니다.

- Mary와 Adam 가정 간의 관계에 레코드가 삽입되고 유효 기간은 변경되지 않습니다.
- 새 레코드의 삭제됨 표시기에는 -999999999가 표시됩니다.
- 새 레코드의 Hub 상태 표시기에는 삭제됨이 표시됩니다.
- 원래 레코드는 변경되지 않습니다.

Mary와 Adam 가정 사이의 관계를 삭제하기 전에는 교차 참조 테이블에 해당 관계가 활성 상태로 표시됩니다.

Rowid	XREF	Rowid	개체	당사자 ID	삭제됨 표시기	Hub 상태 표시기	당사자 그룹 ID	기간 시작 날짜	기간 종료 날짜
63		22		147		활성	28	2006년 10월 6일	2999년 12월 31일

Mary와 Adam 가정 사이의 관계를 삭제하면 교차 참조 테이블에 해당 관계가 삭제된 새 레코드가 표시됩니다.

Rowid	XREF	Rowid	개체	당사자 ID	삭제됨 표시기	Hub 상태 표시기	당사자 그룹 ID	기간 시작 날짜	기간 종료 날짜
63		22		147		활성	28	2006년 10월 6일	2999년 12월 31일
621		22		147	-999999999	삭제됨	28	2006년 10월 6일	2999년 12월 31일

관계 기본 개체에 Mary와 Adam 가정의 관계가 삭제됨으로 표시됩니다.

Rowid 개체	당사자 ID	삭제됨 표시기	Hub 상태 표시기	당사자 그룹 ID
22	147	-999999999	-1	28

모든 관계 기간 삭제

관계 레코드의 관계 기간이 모두 올바르지 않은 경우 모든 관계 기간을 삭제할 수 있습니다. 모든 관계 기간을 삭제하면 MDM Hub가 PARTY GROUP REL 관계 기본 개체에 연결된 교차 참조 테이블에서 레코드 버전의 Hub 상태를 삭제됨 상태로 변경합니다. MDM Hub는 Hub 상태 표시기 및 PARTY GROUP REL 관계 기본 개체 레코드의 삭제됨 표시기를 삭제됨으로 설정합니다.

모든 관계 기간 삭제 예제

조직에 Mary Adam에 대한 레코드가 포함된 PARTY라는 기본 개체가 있습니다. PARTY GROUP 기본 개체에는 Mary가 속하는 Adam 가정의 세부 정보가 포함되어 있습니다. PARTY와 PARTY GROUP 기본 개체 간의 관계는 연결된 PARTY GROUP REL 관계 기본 개체에 정의됩니다. 따라서 Mary와 Adam 가정 간의 관계를 삭제해야 합니다.

Mary와 Adam 가정의 관계가 올바르지 않으므로 Mary의 관계 레코드를 삭제하려고 합니다. 그러려면 유효 기간이 서로 다른 모든 관계 레코드를 삭제해야 합니다. 모든 유효 기간에 대한 Mary의 관계 레코드를 삭제하면 MDM Hub가 PARTY GROUP REL 관계 기본 개체에 연결된 교차 참조 테이블에서 레코드 버전의 Hub 상태를 삭제됨 상태로 변경합니다.

모든 관계 기간을 삭제하면 모든 원래 레코드 버전의 Hub 상태 표시기가 PARTY GROUP REL 관계 기본 개체에 연결된 교차 참조 테이블(RL PARTY GROUP CROSS-REFERENCE)에 삭제됨으로 표시됩니다.

모든 관계 기간을 삭제하기 전에는 교차 참조 테이블에 Mary와 Adam 가정의 유효한 모든 관계가 활성 상태로 표시됩니다.

Rowid XREF	Rowid 개체	당사자 ID	Hub 상태 표시기	당사자 그룹 ID	기간 시작 날짜	기간 종료 날짜
63	22	147	활성	28	2006년 10월 6일	2015년 12월 31일
66	22	147	활성	28	2016년 1월 1일	2999년 12월 31일

모든 관계 기간을 삭제한 후에는 교차 참조 테이블에 Mary와 Adam 가정의 모든 관계가 삭제됨 상태로 표시됩니다.

Rowid XREF	Rowid 개체	당사자 ID	Hub 상태 표시기	당사자 그룹 ID	기간 시작 날짜	기간 종료 날짜
63	22	147	삭제됨	28	2006년 10월 6일	2015년 12월 31일
66	22	147	삭제됨	28	2016년 1월 1일	2999년 12월 31일

관계 기본 개체에 Mary와 Adam 가정의 관계가 삭제됨으로 표시됩니다.

Rowid 개체	당사자 ID	Hub 상태 표시기	당사자 그룹 ID
22	147	-1	28

시간 표시 막대 추출 사용

시간 표시 막대 추출은 다음 방법으로 실행할 수 있습니다.

- MDM Hub 콘솔의 일괄 처리 뷰어 도구에서 BVT 버전 추출 일괄 작업을 실행합니다.
- executeBatchExtractBVTVersions SIF API 실행

BVT 버전 추출 일괄 작업 실행

영향을 미친 기본 개체에 대해 MDM Hub 콘솔의 일괄 처리 뷰어 도구에서 BVT 버전 추출 일괄 작업을 실행합니다.

참고: 일괄 처리 뷰어 도구에서 BVT 버전 추출 일괄 작업을 실행할 때는 제한 날짜를 지정할 수 없습니다. BVT 버전 추출 일괄 작업은 제한 날짜에 대해 현재 응용 프로그램 서버 시간을 사용합니다. 제한 날짜를 지정하려면 executeBatchExtractBVTVersions SIF API를 실행합니다.

일괄 처리 뷰어 도구에 대한 자세한 내용은 *Multidomain MDM 구성 가이드*를 참조하십시오.

executeBatchExtractBVTVersions SIF API 실행

요청

executeBatchExtractBVTVersions 요청은 다음 매개 변수를 포함합니다.

orsId

연산 참조 저장소 이름

tableName

기본 개체 이름

limitDate

BVT 버전 추출 일괄 작업은 마지막 업데이트 날짜가 제한 날짜 전인 교차 참조 레코드에 대해 작동됩니다.

응답

executeBatchExtractBVTVersions 응답은 다음 매개 변수를 반환합니다.

메시지

요청의 상태에 관한 메시지를 포함합니다.

RetCode

반환 코드를 포함합니다.

EJB 요청 예제

다음 EJB 요청은 BVT 버전 추출 일괄 작업을 실행합니다.

```
SiperianClient sipClient = SiperianClient.newSiperianClient(new File( context.getTestPTTStartDir() +
"siperian-client.properties" ));
ExecuteBatchExtractBVTVersionsRequest req = new ExecuteBatchExtractBVTVersionsRequest();
req.setTableName(jobContext.getTableName()); // Pass BO name as a string
req.setLimitDate(jobContext.getLimitDate()); // Pass limit date using java Date type
ExecuteBatchExtractBVTVersionsResponse executed = (ExecuteBatchExtractBVTVersionsResponse)
sipClient.process( req );
String errMessage = executed.getMessage();
int rc = executed.getRetCode();
```

SOAP 요청 예제

다음 SOAP 요청은 CMX_ORIS10A 연산 참조 저장소에 있는 C_TIMELINE 기본 개체에 대해 BVT 버전 추출 일괄 작업을 실행합니다.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:siperian.api">
  <soapenv:Header/>
```



```

<soapenv:Body>
  <urn:executeBatchExtractBTVersions>
    <urn:username>admin</urn:username>
    <urn:password>
      <urn:password>admin</urn:password>
      <urn:encrypted>>false</urn:encrypted>
    </urn:password>
    <urn:orsId>localhost-orcl-CMX_ORS10A</urn:orsId>
    <urn:tableName>C_TIMELINE</urn:tableName>
    <urn:limitDate>2015-10-29T12:59:14+02:00</urn:limitDate>
  </urn:executeBatchExtractBTVersions>
</soapenv:Body>
</soapenv:Envelope>

```

시간 표시 막대 추출 속성 구성

시간 표시 막대 추출 기능을 사용하면 `cmxserver.properties` 파일에서 다음 속성을 선택적으로 구성할 수 있습니다.

- ▶ 다음 선택적 속성을 `cmxserver.properties` 파일에 추가합니다.

속성	설명
<code>cmx.server.batch.extractbvtversions.removePreviousExtractRecords</code>	디스크 공간의 과도한 사용을 방지하기 위해 최신 스냅샷을 유지할지 여부를 지정합니다. 과도한 디스크 공간 사용을 방지하려면 <code>true</code> 로 설정합니다. 기본값은 <code>false</code> 입니다.
<code>cmx.server.batch.extractbvtversions.dropOutOfSyncBOExtractTable</code>	기본 개체 열이 변경될 경우 기존 추출 테이블이 삭제되지 않도록 할지 여부를 지정합니다. 추출 테이블이 삭제되지 않도록 하려면 <code>false</code> 로 설정합니다. 기본값은 <code>true</code> 입니다.
<code>cmx.server.batch.extractbvtversions.maxJobRejectExtractToKeep</code>	거부된 레코드의 기록을 유지할 작업의 수를 지정합니다. 기본값은 10입니다.

제 11 장

상태 관리 및 BPM 워크플로우 도구

이 장에 포함된 항목:

- [상태 관리 및 BPM 워크플로우 도구 개요, 166](#)
- [MDM Hub의 상태 관리, 167](#)
- [BPM 워크플로우 도구, 170](#)
- [ActiveVOS에 대해 MDM Hub 구성, 171](#)
- [보류 중인 레코드에서 일치 활성화, 175](#)
- [상태 전환에 대한 메시지 트리거, 176](#)
- [레코드 승격, 176](#)

상태 관리 및 BPM 워크플로우 도구 개요

업데이트된 레코드가 BVT(최선의 진실, Best Version of the Truth) 레코드에 제공되기 전에 업데이트된 항목 데이터가 변경 승인 워크플로우를 거치도록 보장할 수 있습니다. 예를 들어 비즈니스 프로세스에서 고객 데이터에 대한 업데이트가 마스터 데이터가 되기 전에 선임 관리자의 검토 및 승인을 거쳐야 할 수 있습니다.

변경 승인 워크플로우를 지원하기 위해 MDM Hub 및 IDD(Data Director)가 ActiveVOS® 서버와 통합됩니다. 미리 정의된 MDM 워크플로우, 태스크 유형 및 역할을 통해 구성 요소가 서로 동기화되도록 할 수 있습니다.

구성 요소는 다음과 같은 방식으로 변경 승인 워크플로우를 지원합니다.

- MDM Hub이 상태 관리가 활성화된 기본 개체 테이블에 있는 레코드의 상태를 관리합니다. 승인되지 않은 레코드는 보류 상태이고, 승인된 레코드는 활성 상태입니다.
- IDD 응용 프로그램에서 승인된 비즈니스 사용자는 항목 데이터를 변경하고 승인을 위해 업데이트를 보낼 수 있습니다.
- ActiveVOS Server가 MDM 워크플로우 내에서 활동을 실행하고 비즈니스 관리자 및 데이터 스튜어드를 위해 태스크를 생성합니다.

예제

비즈니스 프로세스에서는 선임 관리자가 고객 데이터에 대한 업데이트를 마스터 데이터가 되기 전에 검토하고 승인해야 합니다. IDD 응용 프로그램 및 MDM 환경은 ActiveVOS Server를 기본 워크플로우 엔진으로 사용하도록 구성됩니다.

미리 정의된 1단계 승인 워크플로우가 업데이트를 검토하고 승인할 선임 관리자 1명이 필요한 비즈니스 프로세스를 정의합니다. 다음 단계는 데이터 스튜어드가 1단계 승인 워크플로우를 시작하면 어떤 일이 발생하는지 요약합니다.

1. IDD 응용 프로그램에서 데이터 스튜어드는 고객 항목을 업데이트하고 해당 업데이트를 승인을 위해 보냅니다.
2. MDM 연산 참조 저장소에서 레코드가 보류 상태에 들어갑니다.
3. ActiveVOS Server이(가) 1단계 승인 워크플로우 프로세스에서 활동을 실행하기 시작합니다.
4. ActiveVOS Server이(가) 최종 검토 인물 활동에 도달하면 서버가 선임 관리자용 태스크를 생성합니다.
5. IDD 응용 프로그램에서 모든 선임 관리자는 태스크 받은 편지함에 태스크를 받습니다. 태스크는 업데이트된 고객 항목으로 연결됩니다.
6. 선임 관리자가 업데이트를 승인하면 다음 이벤트가 발생합니다.
 - IDD에서 태스크가 완료됩니다.
 - ActiveVOS Server이(가) 최종 검토 인물 활동을 완료 표시하고 1단계 승인 워크플로우 프로세스에서 다음 활동을 실행합니다.
 - MDM 연산 참조 저장소에서 승인된 레코드가 활성 상태에 들어갑니다.
 - MDM Hub에서 승인된 레코드가 BVT 레코드에 제공됩니다.

MDM Hub의 상태 관리

상태 관리는 MDM Hub에 있는 레코드와 관련된 상태를 할당하고 변경하는 프로세스입니다. MDM Hub에서는 기본 개체 테이블에 대한 상태 관리를 활성화할 수 있습니다. 상태가 활성화된 기본 개체 테이블 및 해당 상호 참조 테이블의 모든 레코드는 기본 상태로 할당됩니다.

다음 활동은 레코드의 상태를 변경할 수 있습니다.

- 비즈니스 사용자가 BPM 워크플로우 도구에서 태스크를 완료합니다. 완료된 승인 태스크는 관련된 레코드의 상태 변경을 트리거할 수 있습니다.
- 데이터 스튜어드가 Hub 콘솔에서 도구를 사용하여 레코드 집합을 승격, 삭제 또는 복원합니다. 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.
- 응용 프로그램에서 SiperianClient API를 사용하여 레코드 집합을 승격, 삭제 또는 복원합니다. 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

레코드 상태

사전 정의된 MDM Hub 레코드 상태는 활성, 보류 중 및 삭제됨입니다.

참고: 기본적으로 활성 상태의 승인된 레코드만 BVT(최선의 진실) 레코드에 제공됩니다. 작업을 수정하여 활성 레코드와 보류 중 레코드로부터 일치를 허용할 수 있습니다.

다음 테이블에서는 레코드 상태를 설명합니다.

레코드 상태	설명
활성	<p>기본값입니다. 활성 상태의 레코드는 검토 및 승인된 것입니다. 레코드가 BPM 승인 워크플로우를 거쳐야 한다면 활성 레코드가 해당 프로세스를 통해 승인된 것입니다.</p> <p>다음 특성이 활성 레코드에 적용됩니다.</p> <ul style="list-style-type: none"> - 기본적으로 모든 MDM Hub 프로세스에 활성 레코드가 포함됩니다. - 상호 참조 레코드 중 하나 이상이 활성 상태인 경우 기본 개체 레코드가 활성이 됩니다. - 활성 상호 참조 레코드만 통합된 기본 개체에 제공됩니다. - 테이블에서 HUB_STATE_IND 시스템 열이 활성 상태의 레코드에 대해 1을 표시합니다.
보류 중	<p>보류 중 상태의 레코드는 검토 대기 중인 것입니다. 레코드가 BPM 승인 워크플로우를 거쳐야 하는 경우 보류 중인 레코드와 관련된 검토 태스크는 여전히 열려 있습니다. 워크플로우에 참가하고 있고 승인 보류 중인 레코드는 편집할 수 없습니다.</p> <p>다음 특성이 활성 레코드에 적용됩니다.</p> <ul style="list-style-type: none"> - 보류 중 상태의 레코드가 MDM Hub에서 일반적으로 사용하도록 승인되지 않습니다. - 기본적으로 MDM Hub 프로세스에서 보류 중인 레코드가 제외됩니다. 보류 중인 레코드에 대해 대부분의 작업을 수행할 수 있지만 보류 중인 레코드를 요청해야 합니다. - 보류 중인 상호 참조 레코드만 있는 경우 MDM Hub에서 보류 중 레코드의 트러스트를 통해 기본 개체의 BVT를 정합합니다. - 보류 중 상태의 레코드를 삭제할 경우 레코드가 테이블에서 제거되며 삭제된 상태로 전환되는 것이 아니므로 복원할 수 없습니다. - 테이블에서 HUB_STATE_IND 시스템 열은 보류 중 상태의 레코드에 대해 0을 표시합니다.
삭제됨	<p>삭제된 상태의 레코드는 더 이상 MDM Hub 데이터의 일부가 아닙니다.</p> <p>다음 특성이 활성 레코드에 적용됩니다.</p> <ul style="list-style-type: none"> - 기본적으로 MDM Hub 프로세스에서 삭제된 레코드가 제외됩니다. 삭제된 레코드를 포함하려면 이러한 레코드를 요청해야 합니다. - 삭제된 상태의 레코드는 복원할 수 있습니다. - 테이블에서 HUB_STATE_IND 시스템 열이 삭제된 상태의 레코드에 대해 -1을 표시합니다.

Hub 상태 표시기

상태가 활성화된 기본 개체 테이블 및 상호 참조 테이블에는 HUB_STATE_IND라는 시스템 열이 있는데 이 열에서는 값을 사용하여 테이블에 있는 레코드에 대한 레코드 상태를 나타냅니다.

다음 테이블에서는 HUB_STATE_IND에 대해 가능한 값을 나열하고 이러한 값을 레코드 상태에 매핑합니다.

HUB_STATE_IND 값	레코드 상태
1	활성
0	보류 중
-1	삭제됨
-9	영구 삭제 후 상호 참조 기록이 활성화된 경우 사용됩니다. 상호 참조가 삭제되면 HUB_STATE_IND가 -9인 레코드가 상호 참조 기록 테이블(HXRF)에 작성됩니다. 마찬가지로 기본 개체 레코드가 영구 삭제된 경우 삭제된 기본 개체에 대해 HUB_STATE_IND가 -9인 레코드가 기록 테이블(HIST)에 추가됩니다.

상태 전환

상태 전환 규칙은 레코드가 한 상태에서 다른 상태로 변경될 수 있는지, 또 언제 변경될 수 있는지 결정합니다.

다음 테이블에서는 공통 전환을 설명하고 기본 개체 레코드와 상호 참조 레코드 간의 차이점을 보여 줍니다.

전환 전 상태	전환 후 상태	참고
활성	삭제됨	이 전환을 일시 삭제라고 합니다.
보류 중	활성	이 전환을 승격이라고 합니다.
삭제됨	활성	이 전환을 복원이라고 합니다. 상호 참조 레코드는 복원될 수 있습니다. 기본 개체 레코드는 상호 참조 레코드가 복원된 경우에만 복원될 수 있습니다.
삭제됨	보류 중	삭제된 레코드를 보류 중인 레코드로 변경할 수 없습니다. 이 전환은 간접적으로만 발생합니다. 활성 기본 개체 레코드를 삭제한 다음 삭제된 레코드에 대해 보류 중인 상호 참조 레코드를 추가한 경우 기본 개체 레코드 상태는 활성 상태에서 삭제된 상태를 거친 후 보류 중 상태로 변환됩니다.

참고: 다음 상태 전환은 올바르지 않습니다.

- 활성 상태에서 보류 중 상태로 전환. 활성 레코드는 보류 중인 레코드로 변경될 수 없습니다.
- 보류 중 상태에서 삭제된 상태로 전환. 보류 중인 레코드를 삭제할 수는 있지만 이로 인해 삭제된 상태로 전환되는 것은 아닙니다. 대신 MDM Hub에서 데이터베이스에서 레코드를 제거합니다. 이 작업을 영구 삭제라고 합니다. 영구 삭제된 레코드는 복원할 수 없습니다.

보류 중인 레코드 보호

동일한 프로세스에 속하지 않는 업데이트로부터 보류 중인 레코드를 보호할 수 있습니다. 상호 작용 ID에서 이 설정을 제어합니다.

기본 개체 테이블 또는 상호 참조 테이블에 상호 작용 ID가 포함된 경우에는 Hub 콘솔의 도구를 사용하여 기본 개체 레코드 또는 상호 참조 레코드의 상태를 보류 중 상태에서 활성 상태로 변경할 수 없습니다. 상호 참조 레코드에 상호 작용 ID가 포함된 경우 상호 작용 ID는 원래 상호 참조 레코드와 동일한 프로세스에 속하지 않는 업데이트로부터 보류 중인 상호 참조 레코드를 보호합니다. 대신 상태 관리 SIF API 요청 중 하나를 사용합니다.

참고: 상호 작용 ID는 모든 API를 통해 지정할 수 있습니다. 단 일괄 처리를 수행할 때는 지정할 수 없습니다. 예를 들어 상호 작용 ID로 보호되는 레코드는 로드 일괄 처리에 의해 업데이트될 수 없습니다.

다음 테이블에는 기존 레코드 및 수신 레코드에 대한 일치 프로세스 동안 상호 작용 ID 필드의 영향을 보여 주는 예가 포함되어 있습니다. 예제에서 interaction_ID 필드에는 버전 A, 버전 B 및 Null 값이 있습니다.

수신 레코드의 상호 작용 ID	기존 레코드의 상호 작용 ID		
	버전 A	버전 B	Null
버전 A	확인	오류	확인
버전 B	오류	확인	확인
Null	오류	오류	확인

데이터 로드 규칙

로드 일괄 처리 프로세스는 모든 상태의 레코드를 로드합니다. 상태는 준비 테이블에서 입력 열로 지정됩니다. 입력 상태는 랜딩 테이블 열로 매핑에서 지정되거나 파생될 수 있습니다. 매핑에서 입력 상태가 지정되지 않은 경우 상태는 삽입형 로드에는 대해 **ACTIVE**로 가정됩니다. 로드 일괄 처리 작업을 통해 레코드가 업데이트되고 수신 상태가 **null**인 경우 업데이트할 기존 레코드 상태가 변경되지 않습니다.

다음 테이블에는 수신 교차 참조 레코드의 상태가 첫 번째 열에 나열되고, 기존 교차 참조 레코드의 상태가 최상위 행에 나열되고, 결과 작업이 셀에 나열되어 있습니다.

XREF 상태	기존: ACTIVE	기존: PENDING	기존: DELETED	기존: XREF 없음 (rowid별로 로드)	기존: 기본 개체 레코드 없음
수신: ACTIVE	업데이트	업데이트 + 승격	업데이트 + 복원	삽입	삽입
수신: PENDING	업데이트 보류 중	업데이트 보류 중	업데이트 보류 중 + 복원	삽입 보류 중	삽입 보류 중
수신: DELETED	일시 삭제	영구 삭제	일시 삭제	권장되지 않음	권장되지 않음
수신: Undefined	Active로 간주	PENDING으로 간주	DELETED로 간주	Active로 간주	Active로 간주

참고: 영구 삭제 후 기록이 활성화된 경우 교차 참조가 삭제될 때 HUB_STATE_IND가 -9인 레코드가 교차 참조 기록 테이블(HXRF)에 작성됩니다. 마찬가지로 기본 개체 레코드가 실제로 삭제된 경우 삭제된 기본 개체에 대해 HUB_STATE_IND가 -9인 레코드가 기록 테이블(HIST)에 추가됩니다.

레코드 상태와 기본 개체 레코드 값의 존속

병합, 동기 또는 로드 프로세스 후에는 기본 개체 레코드에 다양한 레코드 상태의 여러 상호 참조 레코드가 있을 수도 있습니다. 이 경우 기본 개체 레코드의 값은 활성 상호 참조 레코드의 값만 반영됩니다.

일반적으로 다양한 상태의 여러 레코드를 확인하는 경우 MDM Hub 프로세스에서 다음 규칙을 적용합니다.

- 활성 레코드의 값은 보류 중이거나 삭제된 레코드의 값보다 우선합니다.
- 보류 중인 레코드의 값은 삭제된 레코드의 값보다 우선합니다.

참고: 일치 및 병합 작업의 경우 병합할 레코드의 소스 및 대상은 레코드 상태가 아닌, 존속 Rowid를 결정합니다.

BPM 워크플로우 도구

BPM(비즈니스 프로세스 관리) 도구는 비즈니스 프로세스를 자동화하도록 지원합니다.

기본값으로 미리 정의된 워크플로우 엔진은 ActiveVOS® 서버의 라이선스 버전으로, Multidomain MDM과 함께 제공됩니다. 설치 프로세스에서 이 ActiveVOS Server 버전을 MDM Hub 및 Data Director와 통합하고 미리 정의된 MDM 워크플로우, 태스크 유형 및 역할을 배포합니다.

Informatica ActiveVOS 워크플로우 엔진은 다음 어댑터를 지원합니다.

- 비즈니스 서비스를 통해 비즈니스 항목에 수행되는 태스크를 위한 어댑터입니다. 어댑터 이름은 **BE ActiveVOS**입니다.
- SIF API를 통해 제목 영역에서 운영되는 태스크를 위한 어댑터. 어댑터 이름은 **Informatica ActiveVOS**입니다.

또한 다음과 같이 BPM 도구의 독립 실행형 인스턴스를 통합하도록 선택할 수도 있습니다.

Informatica ActiveVOS

환경에서 Informatica ActiveVOS의 독립 실행형 인스턴스를 실행하는 경우 인스턴스를 수동으로 MDM Hub 및 Data Director와 통합할 수 있습니다. 미리 정의된 MDM 워크플로우를 배포하거나 사용자 지정 워크플로우를 생성할 수 있습니다. 자세한 내용은 *Multidomain MDM Data Director - ActiveVOS 통합 가이드*를 참조하십시오.

타사 BPM 도구

환경에서 타사 인스턴스를 실행하는 경우 인스턴스를 수동으로 MDM Hub 및 Data Director와 통합할 수 있습니다. 미리 정의된 MDM 워크플로우를 배포하거나 사용자 지정 워크플로우를 생성할 수 있습니다. 자세한 내용은 *Multidomain MDM 비즈니스 프로세스 관리자 어댑터 SDK 구현 가이드*를 참조하십시오.

중요: Informatica에서는 비즈니스 항목에 기반하는 ActiveVOS 워크플로우 어댑터로 마이그레이션할 것을 권장합니다. Siperian 워크플로우 어댑터는 더 이상 사용되지 않습니다. Informatica는 더 이상 사용되지 않는 어댑터를 계속해서 지원하지만 해당 어댑터는 더 이상 사용되지 않게 되며 Informatica는 이후 릴리스에서 지원을 중단할 것입니다. MDM Hub에서는 기본 워크플로우 엔진 및 보조 워크플로우 엔진을 지원합니다. Siperian 워크플로우 어댑터에서 비즈니스 항목에 기반하는 ActiveVOS 워크플로우 어댑터로 마이그레이션할 수 있습니다.

ActiveVOS에 대해 MDM Hub 구성

포함된 ActiveVOS Server를 사용하도록 MDM Hub을 구성하려면 다음 단계를 수행해야 합니다.

1. 연산 참조 저장소의 기본 개체 테이블에 대한 상태 관리를 활성화합니다.
2. 연산 참조 저장소를 ActiveVOS 워크플로우 엔진에 매핑합니다.
3. 사전 정의된 워크플로우 사용자 역할을 비즈니스 관리자에 할당합니다.

상태 관리 활성화

Data Director 응용 프로그램을 통해 업데이트할 수 있는 레코드를 포함한 각 기본 개체 테이블의 경우, 상태 관리를 활성화해야 합니다.

1. 모델 작업 영역에서 **스키마**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 도구에서 응용 프로그램을 통해 업데이트할 수 있는 레코드가 포함된 기본 개체 테이블을 선택합니다.
4. **고급** 탭을 클릭합니다.
5. **상태 관리 활성화** 확인란을 선택합니다.
6. 기본 개체에 속한 교차 참조 레코드가 PENDING에서 ACTIVE로 변경되었는지 추적하려면 **교차 참조 승격 이력** 확인란을 선택합니다.

워크플로우 엔진 추가

워크플로우 엔진을 MDM Hub에 추가하면 해당 워크플로우 엔진이 워크플로우 어댑터와 연결됩니다.

워크플로우 엔진을 추가하면 기본 워크플로우 엔진이 되고, 기존 기본 워크플로우 엔진은 보조 워크플로우 엔진이 됩니다. 기존 보조 워크플로우 엔진이 있는 경우 해당 워크플로우 엔진이 연산 참조 저장소에서 삭제되고 태스크 받은 편지함에서 해당 태스크가 제거됩니다.

1. 구성 작업 영역에서 **워크플로우 관리자**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. **워크플로우 엔진** 탭을 선택한 후 **추가** 단추를 클릭합니다.
4. **워크플로우 추가** 대화 상자에 워크플로우 엔진 속성을 입력합니다.

다음 테이블에는 워크플로우 엔진 속성이 설명되어 있습니다.

필드	설명
워크플로우 엔진	워크플로우 엔진의 표시 이름입니다.
어댑터 이름	비즈니스 항목을 기반으로 하는 ActiveVOS 워크플로우 어댑터의 경우 BE ActiveVOS를 선택하십시오.
호스트	Informatica ActiveVOS 인스턴스의 호스트 이름입니다.
포트	Informatica ActiveVOS 인스턴스의 게시 이름입니다.
사용자 이름	트러스트된 사용자의 사용자 이름입니다.
암호	트러스트된 사용자의 암호입니다.
프로토콜	MDM Hub와 ActiveVOS 간에 통신하는 프로토콜입니다. http 또는 https를 프로토콜로 사용할 수 있습니다.

5. **확인**을 클릭합니다.

기본 및 보조 워크플로우 어댑터 설정

BE ActiveVOS 워크플로우 어댑터로 마이그레이션하려면 BE ActiveVOS 워크플로우 엔진을 기본 워크플로우 엔진으로 선택합니다. 비즈니스 항목에 대한 워크플로우 어댑터 이름은 BE ActiveVOS입니다. 보조 워크플로우 엔진이 있는 기존 태스크를 처리할 수 있지만 태스크를 작성할 수는 없습니다. 기본 및 보조 워크플로우 엔진으로 동일한 워크플로우 엔진을 선택하지 마십시오.

ActiveVOS가 포함된 Multidomain MDM을 이전에 사용하지 않은 경우에는 BE ActiveVOS 어댑터를 기본 워크플로우 어댑터로 선택합니다. 기존 태스크를 처리할 보조 워크플로우 어댑터는 선택하지 않아도 됩니다.

참고: Data Director 응용 프로그램에서 제목 영역을 사용하는 경우 계속해서 Informatica ActiveVOS 어댑터를 기본 워크플로우 엔진으로 사용합니다.

워크플로우 엔진을 추가하면 기본 워크플로우 엔진이 되고, 기존 기본 워크플로우 엔진은 보조 워크플로우 엔진이 됩니다. 기존 보조 워크플로우 엔진이 있는 경우 해당 워크플로우 엔진이 연산 참조 저장소에서 삭제되고 태스크 받은 편지함에서 해당 태스크가 제거됩니다.

1. 구성 작업 영역에서 **워크플로우 관리자**를 클릭합니다.
2. 쓰기 잠금을 획득합니다.

3. 워크플로우 엔진 탭을 선택하고 다음 BE ActiveVOS 워크플로우 어댑터 정보가 올바른지 확인합니다.
 - ActiveVOS 서버 호스트
 - ActiveVOS 서버 포트
 - 트러스트된 사용자의 사용자 이름
 - 트러스트된 사용자의 암호
 - MDM Hub와 ActiveVOS 서버 간 통신 프로토콜
4. 연산 참조 저장소 워크플로우 매핑 탭을 선택합니다.
 탭의 테이블에 모든 연산 참조 저장소 데이터베이스(Hub 저장소의 데이터베이스)가 포함됩니다.
5. 기본 워크플로우 엔진 열에서 BE ActiveVOS 워크플로우 어댑터에 대한 워크플로우 엔진을 선택합니다.
6. 보조 워크플로우 엔진 열에서 워크플로우 엔진을 선택합니다.

사용자 역할 구성

MDM이 ActiveVOS Server를 사용하여 사용자를 인증할 수 있도록 하려면 동일한 사용자 역할이 MDM Hub, Data Director 및 ActiveVOS Server에 있어야 합니다. MDM Hub에서 사용자에게 이러한 사용자 역할을 할당하면 보안 액세스 관리자가 모든 구성 요소에 대한 사용자 액세스 권한을 관리합니다.

여러 역할을 사용자에게 할당할 수 있습니다. 역할 권한은 누적되므로 여러 역할을 가진 사용자는 각 역할과 연결된 결합된 권한을 수신합니다. 예를 들어 사용자에게 데이터 스튜어드 역할, abAdmin 역할 및 abadmin 역할을 할당할 수 있습니다. 사용자는 결합된 워크플로우 및 각 역할과 연결된 태스크 관리 권한을 수신합니다.

다음 테이블은 필수 사용자 역할을 나열하고, 워크플로우 작업을 식별하며, 각 역할과 연결된 태스크 관리 작업을 나열합니다.

사용자 역할	워크플로우 작업	태스크 관리 작업
데이터 스튜어드	업데이트, 병합, 병합 해제, 알림	<p>사용자는 해당 사용자 역할에서 사용할 수 있는 태스크에서 다음 작업을 수행할 수 있습니다.</p> <ul style="list-style-type: none"> - 요청 - 릴리스 - 편집 - 태스크 작업(예: 수락, 거부 또는 부인) <p>참고: 사용 가능한 태스크 작업은 역할 및 ActiveVOS 워크플로우에 따라 다릅니다.</p>
Manager	승인 없이 검토	<p>사용자는 해당 사용자 역할에서 사용할 수 있는 태스크에서 다음 작업을 수행할 수 있습니다.</p> <ul style="list-style-type: none"> - 요청 - 릴리스 - 편집 - 태스크 작업(예: 수락, 거부 또는 부인) <p>참고: 사용 가능한 태스크 작업은 역할 및 ActiveVOS 워크플로우에 따라 다릅니다.</p>
SrManager	최종 검토	<p>사용자는 해당 사용자 역할에서 사용할 수 있는 태스크에서 다음 작업을 수행할 수 있습니다.</p> <ul style="list-style-type: none"> - 요청 - 릴리스 - 편집 - 태스크 작업(예: 수락, 거부 또는 부인) <p>참고: 사용 가능한 태스크 작업은 역할 및 ActiveVOS 워크플로우에 따라 다릅니다.</p>

사용자 역할	워크플로우 작업	태스크 관리 작업
abAdmin	-	<p>사용자는 모든 태스크에서 다음 작업을 수행할 수 있습니다.</p> <ul style="list-style-type: none"> - 할당 - 릴리스 - 편집 <p>참고: 사용자는 기본 승인 및 병합 해제 ActiveVOS 워크플로우를 위한 태스크를 관리할 수 있습니다.</p>
abadmin	-	<p>사용자는 모든 태스크에서 다음 작업을 수행할 수 있습니다.</p> <ul style="list-style-type: none"> - 할당 - 릴리스 - 편집 <p>참고: 사용자는 기본 병합 ActiveVOS 워크플로우를 위한 태스크를 관리할 수 있습니다.</p>

참고: 비즈니스 프로세스 관리를 위해 ActiveVOS를 사용하는 경우 사용자 역할에 공백을 포함하지 말아야 합니다.

abAdmin 및 abadmin 역할에 대한 태스크 관리 권한을 활성화하려면 프로비저닝 도구에서 태스크 관리자 역할을 구성해야 합니다. 자세한 내용은 *Multidomain MDM 프로비저닝 도구 가이드*를 참조하십시오.

워크플로우 사용자 역할 할당

사전 정의된 워크플로우 사용자 역할 각각을 비즈니스 관리자 또는 데이터 스튜어드에게 할당합니다.

시작하기 전에 워크플로우에 참여해야 하는 모든 비즈니스 사용자의 MDM Hub 사용자 계정이 있는지 확인하십시오. 필요한 경우 비즈니스 사용자에게 대한 새 사용자 계정을 추가합니다. 새 사용자를 추가하는 경우 동일한 사용자를 응용 프로그램 서버의 컨테이너에 추가합니다.

1. Hub 콘솔에서 Informatica Data Director 응용 프로그램을 사용하여 연산 참조 저장소에 연결합니다.
2. 쓰기 잠금을 획득합니다.
3. 보안 액세스 관리자 작업 영역을 열고 **사용자 및 그룹**을 클릭합니다.
사용자 및 그룹 도구가 열립니다.
4. **역할에 사용자/그룹 할당** 탭을 클릭합니다.
5. 워크플로우의 사용자 역할을 선택하고 **편집**을 클릭합니다.
역할에 사용자 할당 대화 상자가 열립니다.
6. 이 역할이 필요한 사용자 또는 그룹을 선택하고 **확인**을 클릭합니다.
7. 5단계 및 6 단계를 반복해서 다른 워크플로우 역할을 사용자에게 할당합니다.

ActiveVOS 웹 응용 프로그램에 대한 액세스 활성화

Informatica ActiveVOS에는 두 개의 웹 응용 프로그램인 ActiveVOS Console 및 ActiveVOS Central이 포함됩니다. 응용 프로그램 서버에서 사용자 및 워크플로우 역할을 정의하여 이러한 웹 응용 프로그램에 대한 사용자 액세스를 활성화할 수 있습니다.

MDM Hub에서 정의된 동일한 사용자 역할 및 사용자 자격 증명을 정의합니다. 역할 및 사용자를 생성하는 방법에 대한 자세한 내용은 응용 프로그램 서버 설명서를 참조하십시오.

태스크 관리자 역할 구성

프로비저닝 도구를 사용하여 태스크 관리자 역할을 구성하고 태스크 관리자 역할에 매핑하려는 MDM Hub 역할을 지정할 수 있습니다.

시작하기 전에, 기본 ActiveVOS 워크플로우를 사용 중이라면 MDM Hub에서 abAdmin 및 abadmin MDM Hub 역할을 생성해야 합니다.

중요: 사용자가 모든 워크플로우의 모든 태스크에 대해 태스크 관리 작업을 수행할 수 있으려면 해당 사용자에게 abAdmin 및 abadmin MDM Hub 역할을 모두 할당해야 합니다.

1. 프로비저닝 도구에 로그인합니다.
2. **데이터베이스** 목록에서 구성을 연결할 데이터베이스를 선택합니다.
3. **비즈니스 항목 > 태스크**를 클릭합니다.
4. 태스크 패널에서 **태스크 관리자 역할**을 선택한 다음 **생성**을 클릭합니다.
5. 속성 패널의 **이름** 필드에 TaskAdministrator를 입력합니다.
6. **태스크 관리자 역할 활성화** 확인란을 선택합니다.
7. **MDM Hub 역할** 목록에서 태스크 관리자 역할에 매핑하려는 MDM Hub 역할을 선택합니다.

기본 ActiveVOS 워크플로우를 사용 중이라면 **abAdmin**을 선택합니다. 사용자 지정 ActiveVOS 워크플로우를 사용 중이라면 ActiveVOS에서 비즈니스 관리를 위해 사용하는 MDM Hub 역할을 선택합니다.

8. **적용**을 클릭합니다.

변경 내용은 저장되지만 MDM Hub에 게시되지 않습니다.

9. 변경 내용을 MDM Hub에 게시합니다.

- a. **게시**를 클릭합니다.

변경 내용을 게시 또는 검토하라는 메시지를 표시하는 확인 대화 상자가 나타납니다.

- b. 변경 내용을 검토하거나 검토 없이 게시합니다.

- 검토 없이 게시하려면 **게시**를 클릭합니다.
- 검토 후 게시하려면 **변경 내용 검토**를 클릭하고 화면에 나타나는 지시 사항을 따릅니다.

태스크 관리자 역할을 구성했습니다. 매핑된 MDM Hub 역할이 할당된 사용자는 Data Director의 모든 태스크에 대해 태스크 관리 작업을 수행할 수 있습니다.

보류 중인 레코드에서 일치 활성화

기본적으로 일치 프로세스에는 활성 레코드가 포함되고 보류 중인 레코드는 제외됩니다. 보류 중인 레코드를 포함할 수 있습니다.

1. 모델 작업 영역을 열고 **스키마**를 클릭합니다.
2. 스키마 도구에서 상태가 활성화된 기본 개체를 선택합니다.
3. 기본 개체에 대한 트리를 확장하고 **일치/병합 설정**을 선택합니다.
4. 일치/병합 설정 세부 정보 패널의 속성 탭에서 **보류 중인 레코드에 대한 일치 활성화** 확인란을 선택합니다.

상태 전환에 대한 메시지 트리거

MDM Hub에서는 메시지 대기열의 메시지를 사용하여 외부 응용 프로그램과 통신합니다. 기본 개체 레코드 및 상호 참조 레코드의 상태가 변경되는 경우 보고하는 메시지 트리거를 활성화할 수 있습니다.

규칙이 정의된 작업이 수행되면 메시지가 메시지 대기열에 배치됩니다. 메시지 트리거는 메시지가 배치되는 대기열을 지정합니다.

레코드의 상태 변경에 대해 다음 메시지 트리거 이벤트를 사용할 수 있습니다.

이벤트	작업
보류 중인 새 데이터 추가	보류 중인 새 레코드가 생성됩니다.
보류 중인 기존 데이터 업데이트	보류 중인 기본 개체 레코드가 업데이트됩니다.
변경된 상호 참조 레코드만 업데이트 보류	보류 중인 상호 참조 레코드가 업데이트됩니다. 이 이벤트에는 레코드 승격이 포함됩니다.
기본 개체 데이터 삭제	기본 개체 레코드가 일시 삭제됩니다.
상호 참조 데이터 삭제	상호 참조 레코드가 일시 삭제됩니다.
보류 중인 기본 개체 데이터 삭제	기본 개체 레코드가 영구 삭제됩니다.
보류 중인 상호 참조 데이터 삭제	상호 참조 레코드가 영구 삭제됩니다.

상태 전환에 대한 메시지 트리거 활성화

레코드 상태가 변경되면 외부 응용 프로그램에 알리는 메시지 트리거를 활성화할 수 있습니다.

MDM Hub에서는 메시지 대기열을 사용하여 외부 응용 프로그램과 통신합니다. 이렇게 하지 않은 경우 응용 프로그램 서버에서 메시지 대기열을 구성해야 합니다. 메시지 대기열은 게시 프로세스의 일부입니다.

1. **모델** 작업 영역을 열고 **스키마**를 클릭합니다.
2. 메시지 대기열 도구의 메시지 대기열에 대해 **보류 중 업데이트에 대한 트리거** 확인란을 선택합니다.

레코드 승격

레코드 승격은 레코드 집합에 대한 시스템 상태를 보류 중 상태에서 활성 상태로 변경하는 프로세스입니다. 데이터 스튜어드 도구 또는 승격 일괄 처리를 사용하여 레코드 승격을 수동으로 설정할 수 있습니다.

또한 승격 일괄 프로세스에서 승격된 기본 개체와 연결된 하위를 승격합니다. 하위 레코드 및 하위(두 수준 아래) 레코드를 승격할 경우 승격 일괄 작업을 두 번 실행해야 합니다. 먼저 상위 기본 개체에서 승격 일괄 작업을 실행합니다. 그런 다음 하위(두 수준 아래) 기본 개체에서 승격 일괄 작업을 다시 실행합니다.

참고: MDM Hub에서 레코드를 삭제할 경우 유사한 접근 방식을 사용하십시오. 하위 레코드를 삭제하려면 기본 개체를 삭제하십시오. 그런 다음 두 번째 삭제 작업을 실행하여 기본 개체의 하위(두 수준 아래) 레코드를 삭제하십시오.

데이터 스튜어드 도구에서 레코드 승격

데이터 스튜어드 작업 영역의 도구를 통해 **PENDING** 기본 개체 또는 교차 참조 레코드를 즉시 **ACTIVE** 상태로 승격할 수 있습니다. 또한 데이터 관리자 또는 병합 관리자를 사용하여 이 레코드를 나중에 승격하도록 플래그 지정할 수도 있습니다. **Hub** 콘솔을 사용하여 이러한 작업을 수행하는 데 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

나중에 승격할 수 있도록 기본 개체 또는 교차 참조 레코드에 플래그 지정

데이터 관리자를 사용하여 레코드가 승격되도록 플래그를 지정할 수 있습니다. 승격 일괄 작업을 실행하여 플래그가 지정된 레코드를 승격합니다.

1. 데이터 스튜어드 작업 영역을 열고 **데이터 관리자**를 클릭합니다.
2. 데이터 관리자 도구에서 기본 개체 레코드 또는 상호 참조 레코드를 클릭합니다.
3. 관련 패널에서 **승격 플래그 지정**을 클릭합니다.

참고: HUB_STATE_IND 필드가 패키지에 대해 읽기 전용으로 설정되면 데이터 관리자 및 병합 관리자 Hub 콘솔 도구에서 관련 레코드에 대한 레코드 상태 설정 단추가 비활성화됩니다. 그러나 승격 플래그 지정 단추는 레코드에 대해 HUB_STATE_IND 열을 직접적으로 변경하지 않기 때문에 활성화로 유지됩니다.

병합 관리자를 사용하여 일치된 레코드가 승격되도록 플래그 지정

병합 관리자를 사용하여 상호 참조 레코드가 승격되도록 플래그를 지정할 수 있습니다. 승격 일괄 작업을 실행하여 플래그가 지정된 레코드를 승격합니다.

1. 데이터 스튜어드 작업 영역을 열고 **병합 관리자**를 클릭합니다.
2. 일치된 레코드를 클릭합니다.
3. 일치된 레코드 패널에서 **승격 플래그 지정**을 클릭합니다.

일괄 처리 뷰어를 사용하여 승격 일괄 작업 설정

일괄 작업을 설정하여 승격되도록 플래그가 지정된 레코드를 승격할 수 있습니다.

1. 보류 중인 레코드가 승격되도록 플래그를 지정합니다.
2. 유틸리티 작업 영역을 열고 **일괄 처리 뷰어**를 클릭합니다.
3. 일괄 처리 뷰어의 기본 개체 노드에서 **승격** 일괄 작업을 클릭합니다.
4. **플래그가 지정된 레코드 abc 승격**을 선택합니다.

여기서 **abc**는 이전에 승격되도록 플래그가 지정된 관련 레코드를 나타냅니다.

5. 승격되도록 플래그가 지정된 레코드를 승격하려면 **일괄 실행**을 클릭합니다.

일괄 그룹 도구를 사용하여 승격 일괄 작업 설정

플래그가 지정된 레코드를 승격하기 위해 일괄 그룹 도구를 사용하여 승격 일괄 작업을 추가할 수 있습니다.

1. 보류 중인 레코드가 승격되도록 플래그를 지정합니다.
2. 유틸리티 작업 영역을 열고 **일괄 그룹**을 클릭합니다.
3. 쓰기 잠금을 획득합니다.
4. 일괄 그룹 트리에서 일괄 그룹 노드를 마우스 오른쪽 단추로 클릭하고 **일괄 그룹 추가**를 선택합니다.
5. 일괄 그룹 트리에서 임의의 수준을 마우스 오른쪽 단추로 클릭하고 옵션을 선택하여 수준을 일괄 그룹에 추가합니다.

일괄 그룹에 추가할 작업 선택 대화 상자가 열립니다.

6. 추가할 작업에 대한 기본 개체를 확장합니다.
7. **[XREF 테이블]**에서 플래그가 지정된 레코드 승격 작업을 선택합니다.
8. **확인**을 클릭합니다.
일괄 그룹 도구에서 선택한 작업을 일괄 그룹에 추가합니다.
9. **저장**을 클릭합니다.
설정이 완료됩니다. 일괄 그룹 작업을 실행할 수 있습니다.

제 12 장

데이터 암호화

이 장에 포함된 항목:

- [데이터 암호화 개요, 179](#)
- [데이터 암호화 아키텍처, 179](#)
- [데이터 암호화 제한, 180](#)
- [데이터 암호화 유틸리티, 180](#)
- [데이터 암호화 구성, 181](#)
- [서비스 통합 프레임워크 API 요청 및 응답, 183](#)
- [샘플 데이터 암호화 속성 파일, 184](#)

데이터 암호화 개요

MDM Hub 구현으로 중요 데이터를 저장하거나 네트워크를 통해 중요 데이터를 전송하려면 데이터 암호화에 대한 MDM Hub를 구성합니다.

데이터 암호화에 대한 MDM Hub를 구성할 때 암호화된 형식으로 데이터베이스에 중요 데이터를 저장할 수 있습니다. MDM Hub가 데이터베이스에서 Hub 서버, 처리 서버 및 제목 영역이 있는 Data Director로 데이터를 전송할 때 중요 데이터는 암호화된 형식으로 전송됩니다. 데이터 암호화는 사용자가 네트워크를 통해 전송하는 모든 중요 데이터가 보호되도록 합니다.

참고: VARCHAR 데이터 유형의 데이터만 암호화할 수 있습니다.

데이터 암호화 아키텍처

데이터베이스의 중요 데이터를 암호화된 형식으로 저장하고 이를 Hub 서버, 처리 서버 및 Data Director에 암호화된 형식으로 전송할 수 있습니다. 데이터 암호화는 MDM Hub 구현의 중요 데이터가 보호되도록 합니다.

암호화된 데이터는 사용자에게 표시되기 전에 Data Director 또는 다른 SIF(서비스 통합 프레임워크) 클라이언트에서 암호 해독됩니다. 또한 데이터베이스의 암호화된 데이터는 정리 프로세스 전에 암호 해독됩니다.

MDM Hub는 SiperianClient 개체를 통해 각 SIF API 호출을 수행합니다. 각 호출은 요청과 응답으로 구성됩니다. 나가는 요청은 암호화해야 하고 들어오는 응답은 암호 해독해야 합니다.

MDM Hub는 네트워크를 통해 다음 데이터 교환을 암호화할 수 있습니다.

- Hub 서버와 데이터베이스 간

- 정리 작업 동안 Hub 서버와 처리 서버 간
- Data Director와 Hub 서버 간

데이터 암호화 제한

기본 개체 열에 대한 데이터 암호화를 구성할 때 데이터는 암호화된 형식으로 저장됩니다.

데이터 암호화를 구성하기 전에 다음 제한 사항을 고려합니다.

- 데이터 관리자 또는 병합 관리자와 같은 Hub 콘솔 도구를 사용하여 데이터를 편집하거나 데이터를 암호화된 열에 추가하면 데이터가 암호화되지 않습니다.
- 데이터 암호화는 비즈니스 항목이 있는 Data Director에 대해 지원되지 않습니다. 데이터 암호화는 제목 영역이 있는 Data Director에 대해서만 지원됩니다.
- 데이터 암호화는 REST API에 대해 지원되지 않습니다. 데이터 암호화는 SIF(서비스 통합 프레임워크) API에 대해서만 지원됩니다.
- SIF API에서 EjbSiperianClient를 사용하지 않는 경우 요청의 데이터가 암호화되지 않고 응답이 해독되지 않습니다. 요청을 전송하기 전에 모든 중요 데이터를 암호화하고 응답을 수신한 후 암호화된 데이터를 해독해야 합니다.
- SOAP 호출은 데이터를 암호화하지 않습니다. SOAP 호출을 사용하여 데이터를 암호화된 열에 삽입하는 경우 데이터는 사전에 암호화되어 있어야 합니다.
- SearchMatch 및 SearchQuery와 같은 SOAP 호출은 암호화된 열에 있는 데이터를 검색할 수 없습니다.
- Get과 같은 SOAP 호출이 암호화된 열에서 데이터를 검색하는 경우 해당 데이터는 암호화된 형식으로 반환됩니다.
- 일치 필드가 여러 열이 연결된 형태인 경우 암호화된 데이터는 공백을 포함할 수 없습니다. 열에 데이터가 없는 경우에도 이러한 각 열은 암호화되어야 합니다.
- 와일드카드 문자 및 리터럴이 포함된 검색 쿼리는 검색 결과를 가져오지 않습니다.

데이터 암호화 유틸리티

MDM Hub에 대한 데이터 암호화를 구성하기 위해 리소스 키트를 통해 제공되는 데이터 암호화 유틸리티를 사용할 수 있습니다.

리소스 키트에 다음 데이터 암호화 샘플 및 유틸리티가 포함되어 있습니다.

데이터 암호화 라이브러리

데이터 암호화 라이브러리는 데이터 암호화 JAR 파일에서 번들로 제공될 MDM Hub에 필요합니다. 데이터 암호화 라이브러리에는 API 및 공통 클래스가 포함되어 있습니다. 특히 데이터 암호화를 구성할 때 구현해야 하는 DataEncryptor 인터페이스가 포함되어 있습니다. DataEncryptor 인터페이스는 MDM Hub에서 데이터 암호화를 위해 구현해야 하는 encrypt 및 decrypt 메서드를 정의합니다.

데이터 암호화 속성 파일의 샘플

샘플 데이터 암호화 속성 파일에는 데이터 암호화 구현에 필요한 매개 변수가 포함되어 있습니다. 이 속성 파일의 이름은 dataencryption.properties입니다. 샘플 속성 파일을 사용자 지정하여 데이터 암호화 구현 옵션을 지정할 수 있습니다.

샘플 DataEncryption 인터페이스 구현

샘플 DataEncryption 인터페이스 구현은 InformaticaDataEncryptor 클래스를 사용하여 DataEncryptor 인터페이스를 구현합니다. 사용자 지정 암호화 알고리즘을 생성하려면 샘플 구현을 참조하십시오. 데이터 암호화 속성 파일의 mainClass 속성은 사용자가 인터페이스 구현에서 사용하는 클래스 이름을 참조해야 합니다.

Ant 빌드 스크립트

Ant 빌드 스크립트인 build.xml은 데이터 암호화 JAR 파일을 생성하는 데 사용됩니다.

데이터 암호화 구성

데이터 암호화를 사용하려면 데이터 암호화를 위한 MDM Hub를 구성해야 합니다.

1. DataEncryptor 인터페이스를 구현합니다.
2. 데이터 암호화 속성 파일을 구성합니다.
3. Hub 서버에 대한 데이터 암호화를 구성합니다.
4. 처리 서버에 대한 데이터 암호화를 구성합니다.

1단계. DataEncryptor 인터페이스 구현

DataEncryptor 인터페이스를 구현해야 합니다. DataEncryptor 인터페이스는 encrypt 및 decrypt 메서드를 정의합니다. encrypt 및 decrypt 메서드의 구현은 스레드로부터 안전해야 합니다.

1. Java 통합 개발 환경에서 Java 프로젝트를 생성합니다.
2. Java 프로젝트에 다음 MDM Hub JAR 파일을 추가합니다.
 - siperian-api.jar
 - siperian-common.jarjar 파일은 다음 디렉터리에 있습니다.
UNIX의 경우. <infamdm_install_dir>/resourcekit/samples/DataEncryption/lib
Windows의 경우. <infamdm_install_dir>\resourcekit\samples\DataEncryption\lib
3. encrypt 및 decrypt 메서드를 포함하는 데이터 암호화를 위한 Java 클래스를 생성합니다.
4. 데이터 암호화 Java 클래스를 컴파일합니다.
5. 사용자 지정 데이터 암호화 JAR 파일의 클래스 파일을 패키지하려면 다음 명령을 실행합니다.

```
ant build
```
6. 생성된 파일을 정리하려면 빌드가 완료된 후 다음 명령을 실행합니다.

```
ant clean
```
7. 생성된 사용자 지정 데이터 암호화 JAR 파일에서 DataEncryptor 인터페이스를 구현합니다.

2단계. 데이터 암호화 속성 파일 구성

Informatica는 데이터 암호화 구현에 필요한 매개 변수가 포함되어 있는 샘플 데이터 암호화 속성 파일을 제공합니다. 샘플 속성 파일을 사용자 지정하여 데이터 암호화 구현 옵션을 지정할 수 있습니다.

1. 다음 디렉터리에서 `dataencryption.properties` 파일을 찾습니다.
UNIX의 경우. `<ResourceKit_install_dir>/DataEncryption/resources`
Windows의 경우. `<ResourceKit_install_dir>\DataEncryption\resources`
2. `dataencryption.properties` 파일의 백업 복사본을 생성합니다.
3. 텍스트 편집기를 사용하여 파일을 열고 데이터 암호화 매개 변수를 편집합니다.
4. `DataEncryptor` 인터페이스 구현에 대한 참조를 제공합니다.
5. 중요 데이터가 포함된 기본 개체 열, 관련된 테이블(예: 기록 및 교차 참조 테이블) 및 패키지를 암호화 및 암호 해독하도록 `EJBSiperianClient`를 구성합니다.

다음 구문을 사용하여 기본 개체 열 이름, 관련된 테이블(예: 기록 및 교차 참조 테이블) 및 패키지를 지정합니다.

```
API.<ORS_ID>.BASE_OBJECT.<BASE_OBJECT_NAME>.<COLUMN_NAME>  
API.<ORS_ID>.XREF.<CROSS_REFERENCE_TABLE_NAME>.<COLUMN_NAME>  
API.<ORS_ID>.HISTORY.<HISTORY_TABLE_NAME>.<COLUMN_NAME>  
API.<ORS_ID>.PACKAGE.<PACKAGE_NAME>.<COLUMN_NAME>  
API.<ORS_ID>.HM_ENTITY_TYPE.<HIERARCHY_MANAGER_ENTITY_NAME>.<COLUMN_NAME>
```

6. 유사 항목 일치 및 유사 항목 검색 작업을 사용하려면 사용할 일치 필드를 지정합니다.

다음 구문을 사용하여 일치 필드 이름을 지정합니다.

```
MATCHFIELD.<ORS_ID>.BASE_OBJECT.<BASE_OBJECT_NAME>.<MATCH_FIELD_NAME>
```

참고: 지정하는 기본 개체가 일치 작업 중인 기본 개체(일치 경로 구성 요소의 루트)인지 확인합니다. 일치 필드를 일치 경로 구성 요소의 루트가 아닌 기본 개체에서 가져온 경우라도 일치 경로 구성 요소의 루트인 기본 개체를 지정합니다.

일치 필드가 여러 열이 연결된 형태인 경우 열에 데이터가 없는 경우에도 이러한 각각의 열을 암호화해야 합니다.

7. 데이터를 암호화 및 암호 해독해야 하는 정리 함수의 입력 및 출력 포트를 구성합니다.

다음 구문을 사용하여 정리 함수의 입력 및 출력 포트를 지정합니다.

```
CLEANSE.<PORT_NAME>=true
```

8. 속성 파일을 동일한 이름인 `dataencryption.properties`로 저장합니다.

3단계. Hub 서버에 대한 데이터 암호화 구성

데이터 암호화를 사용하도록 Hub 서버를 구성하려면 사용자 지정 데이터 암호화 JAR 파일을 사용하도록 Hub 서버를 구성합니다.

1. 다음 디렉터리에서 `cmxserver.properties` 파일을 찾습니다.
UNIX의 경우. `<infamdm_install_directory>/hub/server/resources`
Windows의 경우. `<infamdm_install_directory>\hub\server\resources`
2. 텍스트 편집기를 사용하여 파일을 열고 `encryption.plugin.jar` 속성에 대한 데이터 암호화 JAR의 경로를 지정합니다.
`encryption.plugin.jar=<Path to the data encryption JAR>`
3. `cmxserver.properties` 속성 파일을 저장합니다.

4단계. 처리 서버에 대한 데이터 암호화 구성

데이터 암호화를 사용하도록 처리 서버를 구성하려면 사용자 지정 데이터 암호화 JAR 파일을 사용하도록 처리 서버를 구성합니다.

1. 다음 디렉터리에서 cmxcleanse.properties 파일을 찾습니다.

UNIX의 경우. <infamdm_install_directory>/hub/cleanse/resources

Windows의 경우. <infamdm_install_directory>\hub\cleanse\resources

2. 텍스트 편집기를 사용하여 파일을 열고 encryption.plugin.jar 속성에 대한 데이터 암호화 JAR의 경로를 지정합니다.

encryption.plugin.jar=<Path to the data encryption JAR>

3. cmxcleanse.properties 속성 파일을 저장합니다.

서비스 통합 프레임워크 API 요청 및 응답

MDM Hub는 데이터 암호화 JAR 파일에 포함된 데이터 암호화 Java 클래스를 사용하여 SIF(서비스 통합 프레임워크) API를 호출합니다.

다음 테이블에서는 데이터 암호화와 함께 작동하는 SIF API 요청 및 응답에 대해 설명합니다.

요청/응답	설명
PutRequest	키로 식별된 단일 레코드를 기본 개체에 삽입하거나 업데이트합니다. 입력 레코드를 암호화합니다.
MultiMergeRequest	동일한 개체를 나타내는 여러 개의 기본 개체 레코드를 병합합니다. 병합된 레코드에 대한 필드 수준 재정의의 지정할 수 있습니다. 재정의의 필드를 암호화합니다.
GetResponse	알려진 키를 사용하여 패키지에서 단일 레코드를 검색합니다. 반환된 레코드를 암호화합니다.
SearchHmQueryRequest	계층 관리자 항목 및 관계를 검색합니다. 입력 필터 매개 변수를 암호화합니다. 매개 변수 대신 SiperianObjectUidField를 사용하여 암호화를 활성화합니다.
SearchQueryRequest	패키지에서 지정한 조건을 충족하는 레코드 집합을 검색합니다. 입력 필터 매개 변수를 암호화합니다. 매개 변수 대신 SiperianObjectUidField를 사용하여 암호화를 활성화합니다.
SearchMatchRequest	일치 열 및 규칙 정의를 기반으로 패키지의 레코드를 검색합니다. 일치를 수행할 대상인 입력 필터 매개 변수 및 레코드를 암호화합니다. 매개 변수 대신 SiperianObjectUidField를 사용하여 암호화를 활성화합니다.
SearchHmQueryResponse	하나 이상의 항목과 연결된 항목 및 관계를 반환합니다. 반환된 레코드를 암호화합니다.
SearchMatchResponse	레코드 및 일치 토큰 목록을 반환합니다. 반환된 레코드를 암호화합니다.

요청/응답	설명
SearchQueryResponse	패키지에서 레코드 집합을 반환합니다. 반환된 레코드를 암호화합니다.
AddRelationshipRequest	두 항목 간의 관계를 추가합니다. 레코드로 제공된 사용자 지정 필드를 암호화합니다.
UpdateRelationshipRequest	관계의 특성을 변경하는 계층 관리자 요청입니다. 레코드로 제공된 사용자 지정 필드를 암호화합니다.
GetOneHopResponse	지정된 계층 구성의 지정된 항목 그룹과 직접적으로 관련된 항목에 대한 정보를 반환합니다. 반환된 레코드를 암호화합니다.
GetEntityGraphResponse	지정된 항목 집합과 관련된 항목 및 관계의 그래프를 반환합니다. 반환된 레코드를 암호화합니다.
GetXrefForEffectiveDateResponse	지정된 유효 날짜에 대한 여러 교차 참조 레코드를 반환합니다. 반환된 레코드를 암호화합니다.

샘플 데이터 암호화 속성 파일

리소스 키트에는 샘플 데이터 암호화 속성 파일인 `dataencryption.properties`가 포함됩니다.

샘플 데이터 암호화 속성 파일은 다음 디렉터리에 있습니다.

UNIX의 경우. `<infadm_install_dir>/hub/resourcekit/samples/DataEncryption/resources`

Windows의 경우. `<infadm_install_dir>\hub\resourcekit\samples\DataEncryption\resources`

다음 예에서는 샘플 데이터 암호화 속성 파일의 내용을 보여 줍니다.

```
# This is an example of dataencryption.properties file.
# File contains encryptor configuration and different parts of the product that should be aware of data
# encryption.

#
# Encryptor implementation
#

# main class that implements DataEncryptor interface. If this option is empty, encryption is turned off.
mainClass=com.informatica.dataencryption.sample.InformaticaDataEncryptor

#
# Part 1. SiperianClient configuration.
#
# List of API.<ORS_ID>.<BASE_OBJECT_NAME>.<COLUMN_NAME> that should be encrypted before they are sent out
# and decrypted when returned back by SiperianClient.
#
API.localhost-main-MDM_SAMPLE.BASE_OBJECT.C.PARTY.FIRST_NAME=true
API.localhost-main-MDM_SAMPLE.PACKAGE.PKG_PERSON_IDD_SEARCH.FIRST_NAME=true
API.localhost-main-MDM_SAMPLE.HISTORY.C.PARTY.FIRST_NAME=true
API.localhost-main-MDM_SAMPLE.XREF.C.PARTY.FIRST_NAME=true
API.localhost-main-MDM_SAMPLE.HM_ENTITY_TYPE.Person.FIRST_NAME=true

#
```

```
# Part 2. Cleanse functions.  
#  
# List of input and output ports of cleanse functions that must receive and produce encrypted data.  
#  
CLEANSE.firstName=true
```

제 13 장

계층

이 장에 포함된 항목:

- [계층 개요, 186](#)
- [계층 예제, 187](#)
- [계층 구성 정보, 187](#)
- [시작하기 전에, 188](#)
- [구성 단계 개요, 188](#)
- [계층 관리자를 위한 데이터 준비, 188](#)
- [계층 관리자를 위한 데이터 준비 방법의 사용 사례, 189](#)
- [HM 리포지토리 기본 개체 생성, 193](#)
- [기본 항목 아이콘 업로드, 193](#)
- [항목 아이콘 구성, 194](#)
- [항목, 195](#)
- [항목 기본 개체, 195](#)
- [항목 유형, 198](#)
- [항목에 대한 옵션 표시, 201](#)
- [항목 기본 개체를 기본 개체로 변환, 202](#)
- [계층 유형, 202](#)
- [관계 개체, 203](#)
- [관계 기본 개체, 203](#)
- [외래 키 관계 기본 개체, 206](#)
- [관계 유형, 207](#)
- [패키지, 211](#)
- [프로필 정보, 215](#)

계층 개요

MDM Hub 콘솔에서 계층을 구성할 수 있습니다. 계층은 개별 레코드 간의 관계를 보여 주는 반면, 데이터 모델은 기본 개체 간의 관계를 보여 줍니다.

예를 들어 조직 내 각 부서에 속한 직원을 추적하는 계층을 생성할 수 있습니다. 또한 다양한 제품 그룹 내 제품 계층을 생성할 수도 있습니다.

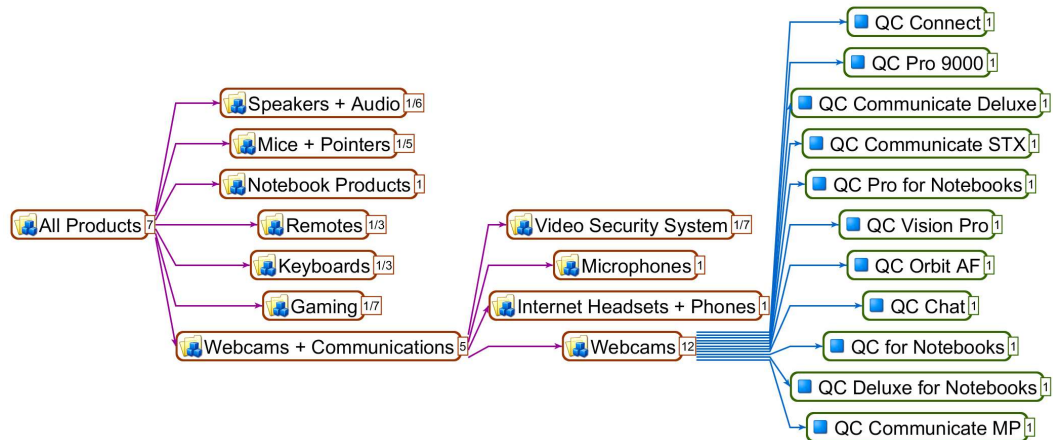
계층을 구성하려면 먼저 항목 기본 개체 등을 생성해야 합니다.

계층을 구성하려면 계층 관리자 라이선스가 있어야 합니다.

계층 예제

리소스 키트의 샘플 연산 참조 저장소에 사전 구성된 계층이 있는 스키마가 포함되어 있습니다. 이 장에서는 제품 계층 일부를 사용하여 계층 구성 단계를 설명합니다.

다음 이미지는 예제에서 사용되는 계층 일부를 보여 주는 것으로, Hub 콘솔의 계층 관리자 도구에 표시됩니다.



계층 구성 정보

MDM Hub 관리자는 계층 도구를 사용하여 계층 관리자에서 데이터 관계를 보고 조작하는 데 필요한 구조를 설정합니다.

계층 도구를 사용하여 항목 유형, 계층, 관계 유형, 패키지, 프로필 등의 계층 관리자 구성 요소를 MDM Hub 구현에 대해 정의할 수 있습니다.

계층 관리자 구성 요소를 정의한 후에는 패키지 또는 쿼리 관리자 도구를 사용하여 쿼리 조건을 업데이트할 수 있습니다.

참고: 계층 관리자에서 사용하도록 패키지를 구성하고 프로필의 유효성을 검사해야 합니다.

계층 및 해당 구성 요소의 개념을 이해하려면 다음 장에 나오는 개념을 잘 알고 있어야 합니다.

- [장 8, “스키마 작성” 페이지 75](#)
- [장 9, “쿼리 및 패키지” 페이지 122](#)
- [장 24, “통합 프로세스 구성” 페이지 475](#)

시작하기 전에

HM(계층 관리자)을 구성하려면 먼저 몇 가지 태스크를 완료해야 합니다.

다음 태스크를 완료하십시오.

- 기초로 사용할 빈 연산 참조 저장소나 유효한 ORS를 준비하고 해당 데이터베이스를 CMX_SYSTEM에 등록합니다.
- 계층 관리자 라이선스가 있는지 확인합니다. 자세한 내용은 Informatica MDM Hub 관리자에게 문의하십시오.
- 데이터 분석을 수행합니다.

구성 단계 개요

계층 관리자를 구성하려면 다음 단계를 수행합니다.

1. Hub 콘솔을 시작합니다.
2. 계층 도구를 실행합니다.
RBO(리포지토리 기본 개체) 테이블을 아직 작성하지 않은 경우 Hub 콘솔에서 해당 프로세스를 안내합니다.
3. 항목 개체 및 유형을 생성합니다.
4. 계층을 생성합니다.
5. 관계 개체 및 유형을 생성합니다.
6. 패키지를 생성합니다.
7. 프로필을 구성합니다.
8. 프로필의 유효성을 검사합니다.

참고: 계층 메뉴에는 계층 관리자의 마우스 오른쪽 단추 클릭 메뉴에 표시되는 옵션과 같은 옵션이 제공됩니다.

계층 관리자를 위한 데이터 준비

HM을 최대한 활용하려면 정보를 분석하고 다음을 수행해야 합니다.

- 데이터 소스가 올바르게 신뢰할 수 있는지 확인합니다.
- Informatica MDM Hub 및 HM에서 사용할 수 있도록 올바른 스키마를 생성합니다.
- 항목 간 계층적 외래 키 및 단일 홉 및 다중 홉 관계를 생성합니다(직접 관계 및 간접 관계). 모든 하위 항목에는 해당 항목과 관련된 올바른 상위 항목이 있어야 합니다.
데이터를 HM에 입력할 때 데이터에 '고아' 하위 항목이 포함되면 안 됩니다.
모든 계층의 유효성을 검사해야 합니다.
단일 홉 및 다중 홉 관계에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.
- HM 유형을 파생합니다.
- 여러 소스 시스템의 중복된 항목을 통합합니다.

예를 들어 항목 그룹(소스 A)이 다른 항목 그룹(소스 B)과 같지만 두 항목 그룹의 이름이 다를 수 있습니다. 항목이 같은 것으로 식별되면 두 그룹을 통합할 수 있습니다.

- 항목을 논리 범주로 그룹화합니다(예: 의사 이름을 “Physician” 범주로 그룹화).
- 데이터가 참조 무결성, 잘못된 데이터 및 데이터 변동성 관련 규칙을 준수하는지 확인합니다. 이러한 데이터베이스 개념에 대한 자세한 내용은 데이터베이스 참조 텍스트를 확인하십시오.

계층 관리자를 위한 데이터 준비 방법의 사용 사례

이 섹션에는 데이터를 Informatica MDM Hub에 입력하고 계층 관리자에서 보기 전에 데이터를 조작하는 방법을 보여 주는 예가 포함되어 있습니다.

일반적으로 회사 데이터는 여기에 나오는 예보다 훨씬 더 큼니다.

시나리오

John이 계층 관리자에서 가장 효율적인 방식으로 데이터를 보고 사용할 수 있도록 회사 데이터를 조작했습니다.

예를 간소화하기 위해 컴퓨터 구성 부품을 판매하는 회사의 제품 및 제품 유형을 포함하는 데이터 하위 집합에 대해 설명합니다.

이 회사는 세 가지 유형의 제품 즉 마우스, 트랙볼 및 키보드를 판매합니다. 이러한 각 제품 유형에는 게임용 키보드 및 TrackMan 트랙볼 등과 같은 다양한 수준의 제품과 여러 공급업체가 포함됩니다.

방법론

이 섹션에서는 데이터를 간소화하는 방법에 대해 설명합니다.

1단계: 데이터를 계층으로 구성

이 단계에서는 데이터를 계층으로 구성합니다. 그런 다음 이 데이터는 HM 구성으로 변환됩니다.

John은 제품 및 제품 그룹 계층을 분석하여 작업을 시작하고 제품 그룹별로 제품을 구성하고 해당 상위 제품 그룹별로 제품 그룹을 구성합니다. 데이터에 포함된 데이터 및 관계의 순수 양은 시각화하기 어려우므로 John은 범주를 나열하고 범주 간 관계가 있는지 여부를 확인합니다.

마케팅 부서의 데이터를 포함하는 다음 테이블에는 John이 데이터를 구성한 방법에 대한 예가 나열되어 있습니다.

ProdGroup		ProdGroup		ProdGroup		제품	
ProdNumber	설명	ProdNumber	설명	ProdNumber	설명	ProdNumber	설명
ALL	모든 제품	100	마우스 + 포인터	120	마우스	120-0001	레이저 마우스
						120-0002	나노 무선 레이저 마우스

ProdGroup		ProdGroup		ProdGroup		제품	
						120-0003	무선 광학 마우스
						120-0004	나노 무선 레이저 마 우스 II
						120-0005	노트북용 레이저 마 우스
						120-0006	레볼루션
						120-0007	충전식 무 선 마우스
						120-0008	무선 광학 마우스 II
		200	키보드	210	키보드	-	-
				220	키보드 및 마 우스 콤보	-	-

참고: 대부분의 데이터 집합에는 훨씬 더 많은 항목이 포함됩니다.

이 테이블에는 제품 BO에 저장되는 데이터가 표시됩니다. 이 BO는 HM에서 변환(또는 생성)할 BO입니다. 또한 마우스 또는 레이저 마우스 등과 같은 항목이 나열되어 있으며, 관계가 그룹화로 표시됩니다. 즉 마우스와 레이저 마우스 간 관계가 있습니다. 제목 값은 항목 유형입니다. 마우스는 제품 그룹이고 레이저 마우스는 제품입니다. 이 유형은 제품 테이블의 필드에 저장됩니다.

이 방법으로 데이터를 구성함으로써 John은 데이터에 속하는 항목 및 항목 유형 수를 명확하게 확인할 수 있으며, 해당 항목에 포함되는 관계도 확인할 수 있습니다.

주요 범주는 제품 그룹으로, 여기에는 제품 그룹(예: 마우스 및 포인터), 범주 제품 및 제품 자체(예: Trackman 휠)가 모두 포함될 수 있습니다. 이러한 항목 간 관계는 관계 개체로 캡슐화할 수 있으며, John은 이를 제품 관계라고 했습니다. John은 제품 관계에 대한 정보에서 제품 그룹이 제품 및 제품 그룹 둘 다의 상위 항목임을 나타내는 관계를 설명했습니다.

2단계: 관계 기본 개체 테이블 생성

데이터 분석을 통해 John은 다음과 같은 결론을 도출했습니다.

- 제품(BO)은 항목 개체로 변환되어야 합니다.
- 제품 그룹 및 제품은 항목 유형입니다.
- 제품 관계는 생성할 관계 개체입니다.
- 다음과 같은 관계 유형(테이블에 표시된 모든 항목은 아님)을 생성해야 합니다.
 - 제품이 제품 상위임(표시되지 않음)
 - 제품 그룹이 제품 상위임(예: 마우스-레이저 마우스)
 - 제품 그룹이 제품 그룹 상위임(예: 마우스+마우스 상위인 포인터)

John은 계층 도구에 액세스하여 작업을 시작합니다. 도구에 액세스하면 시스템이 **RBO** 테이블(관계 기본 개체 테이블)을 생성합니다. 기본적으로 **RBO** 테이블은 특정 열을 포함한 필수 기본 개체인 시스템 기본 개체입니다. 이 테이블에는 **HM** 구성 데이터(예: 1단계의 테이블에 표시된 데이터)가 저장됩니다.

기본 개체를 생성하는 방법에 대한 자세한 내용은 [“기본 개체 생성” 페이지 104](#)을 참조하십시오. 이 섹션에서는 스키마 도구에서 예제 기본 개체를 생성할 때 제공되는 옵션에 대해 설명합니다.

이전 단계에서 식별한 각 항목 개체 및 관계 개체에 대해 기본 개체를 생성 및 구성해야 합니다. 예제에서 제품에 대해 기본 개체를 생성하고 이 개체를 **HM** 항목 개체로 변환합니다. 제품 관계 **BO**가 변환되는 대신 **HM**에서 직접 생성되어야 합니다(더 간편한 프로세스임). 새로운 각 기본 개체가 기본 개체 범주의 스키마 패널에 표시됩니다. 이 프로세스를 반복하여 모든 기본 개체를 생성합니다.

다음 섹션에서는 **HM** 사용에 최적화되도록 기본 개체를 구성합니다.

3단계: 기본 개체 구성

이전 섹션에서 두 개의 기본 개체(제품 및 제품 관계)를 생성했습니다. 이 섹션에서는 이러한 기본 개체를 구성하는 방법에 대해 설명합니다.











기본 개체를 구성하는 작업에는 열 수, 열 유형, 준비 테이블의 콘텐츠, 교차 참조 테이블 이름(있는 경우) 등과 같은 개체 속성의 기준을 입력하는 작업이 포함됩니다. 기록 함수를 활성화하고 유효성 검사 규칙 및 메시지 트리거를 설정하며 사용자 지정 인덱스를 생성하고 외부 일치 테이블(있는 경우)을 구성할 수도 있습니다.

이러한 옵션을 선택할지 여부 및 옵션을 구성하는 방법은 데이터 모델 및 기본 개체 선택에 따라 다릅니다.

이 예에서는 John이 다음 섹션에 설명한 대로 기본 개체를 구성합니다.

참고: 기본 개체 생성과 관련된 일부 구성 요소는 여기에서 설명하지 않습니다. **HM**에서 사용되는 데이터와 관련된 구성 요소만 설명됩니다. 여기에서 설명되지 않는 구성 요소에 대한 자세한 내용은 [장 8, “스키마 작성” 페이지 75](#)을 참조하십시오.

다음 그림은 기본 개체 열을 보여 줍니다.

	Display Name	Physical Name	Nullable	Data Type	Length
	Rowid Object	ROWID_OBJECT	<input type="checkbox"/>	CHAR	14
	Product Number	PRODUCT_NUMBER	<input checked="" type="checkbox"/>	VARCHAR	50
	Product Name	PRODUCT_NAME	<input checked="" type="checkbox"/>	VARCHAR	100
	Product Desc	PRODUCT_DESC	<input checked="" type="checkbox"/>	VARCHAR	255
	Inception Date	INCEPTION_DATE	<input checked="" type="checkbox"/>	DATE	
	Eff Start Date	EFF_START_DATE	<input checked="" type="checkbox"/>	DATE	
	Eff End Date	EFF_END_DATE	<input checked="" type="checkbox"/>	DATE	
	Status Cd	STATUS_CD	<input checked="" type="checkbox"/>	CHAR	2
	Product Type Cd	PRODUCT_TYPE_CD	<input checked="" type="checkbox"/>	VARCHAR	50
	Product Type	PRODUCT_TYPE	<input type="checkbox"/>	VARCHAR	255

이 테이블은 **HM** 항목 개체로 변환한 후 제품 **BO**를 보여 줍니다. 이 목록에서는 제품 유형 필드만 **HM** 필드입니다.

기본 개체마다 시스템 열과 사용자 정의 열이 포함됩니다. 시스템 열은 자동으로 생성되며 필요한 열 **Rowid** 개체를 포함합니다. 이 열은 각 기본 개체 테이블의 기본 키이며 **Hub**에서 생성된 고유 값을 포함합니다. 이 값은 클래스 코드에 대한 **HM** 조회이므로 **null**일 수 없습니다. **HM**은 **ROWID_OBJECT** 값이 필수이고 **null**일 수 없도록 데이터베이스에서 외래 키 제약 조건을 만듭니다.

John은 사용자 정의 열에 대해 제품 이름, 제품 유형 및 제품 설명 등과 같은 제품에 대한 정보를 실질적으로 포함하는 논리 이름을 선택합니다. 다음 그림에 표시된 대로 준비 테이블에 이러한 같은 열과 열 값이 표시되어야 합니다.

Columns						
The selected columns will be included in this staging table. To include a lookup, use the edit button.						
	Column	Lookup System	Lookup Table	Lookup Column	Allow Null Update	Allow Null Foreign Key
	<input checked="" type="checkbox"/> Product Number				<input type="checkbox"/>	<input type="checkbox"/>
	<input checked="" type="checkbox"/> Product Name				<input type="checkbox"/>	<input type="checkbox"/>
	<input checked="" type="checkbox"/> Product Desc				<input type="checkbox"/>	<input type="checkbox"/>
	<input checked="" type="checkbox"/> Inception Date				<input type="checkbox"/>	<input type="checkbox"/>
	<input checked="" type="checkbox"/> Eff Start Date				<input type="checkbox"/>	<input type="checkbox"/>
	<input checked="" type="checkbox"/> Eff End Date				<input type="checkbox"/>	<input type="checkbox"/>
	<input checked="" type="checkbox"/> Status Cd		LU Product Status	Status Cd	<input type="checkbox"/>	<input type="checkbox"/>
	<input checked="" type="checkbox"/> Product Type Cd		LU Product Type	Product Type	<input type="checkbox"/>	<input type="checkbox"/>
	<input checked="" type="checkbox"/> Product Type		Rbo Bo Class	Bo Class Code	<input type="checkbox"/>	<input type="checkbox"/>

John이 위의 그래픽에 표시된 대로 준비 테이블의 모든 사용자 정의 열을 기본 개체의 열로 추가합니다. 조회 열에 HM에서 추가한 조회 값이 표시됩니다.

준비 테이블(상태 코드, 제품 유형 및 제품 유형 코드)의 여러 열에 조회 테이블에 대한 참조가 포함됩니다. 준비 테이블을 생성할 때 이러한 참조를 설정할 수 있습니다. 준비 테이블의 값을 하드코딩하지 않고 서버가 상위 테이블의 값을 조회하도록 하려면 조회를 사용합니다.

대부분의 조회는 HM과 관련되지 않으며 데이터 모델의 일부입니다. Rbo Bo 클래스 조회는 HM에서 추가된 항목이므로 예외입니다. HM은 제품 유형 열에 대한 조회를 추가합니다.

참고: 항목을 항목 기본 개체(HM에서 사용하도록 구성된 항목)로 변환하려면 상태 코드, 제품 유형 및 제품 유형 코드에 대한 값을 확인할 조회 테이블이 있어야 합니다.

참고: HM 항목 개체에는 시작 날짜와 종료 날짜가 필요하지 않습니다. 모든 시작 날짜와 종료 날짜는 사용자 정의됩니다. 단 관계 개체에서는 이러한 날짜를 사용합니다. 시작 날짜와 종료 날짜에 대해 서로 다른 이름을 가진 새 관계 개체를 생성하지 마십시오. 이 항목은 이미 제공되었습니다.

4단계: 항목 유형 생성

계층 도구에서 항목 유형을 생성합니다. John이 두 개의 항목 유형 즉 제품 그룹 및 제품 유형을 생성합니다.

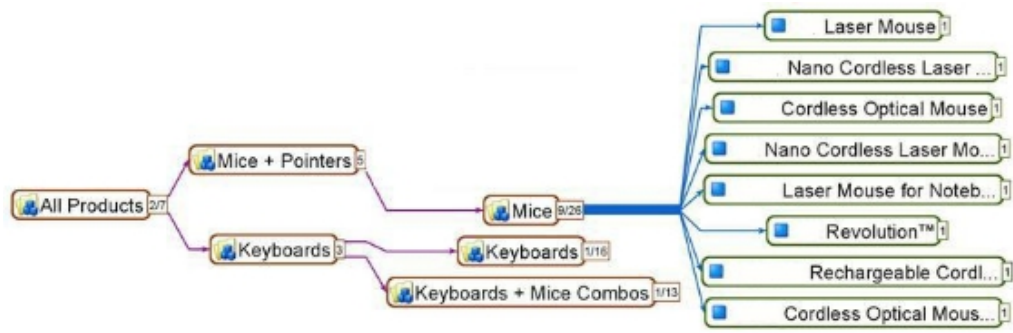
각 항목 유형에는 데이터 분석 및 디자인에서 파생된 코드가 포함됩니다. John은 하나의 유형으로 '제품'을 사용하고 또 다른 유형으로 '제품 그룹'을 사용하도록 선택합니다.

이 코드는 해당 RBO 기본 개체 테이블에서 참조되어야 합니다. 이 예에서 코드 '제품'은 C_RBO_BO_CLASS 테이블에서 참조됩니다. BO_CLASS_CODE 값은 '제품'입니다.

다음 테이블에는 RBO 테이블에 대한 HM 관계 개체와 HM 항목 개체 간 관계를 보여 줍니다.

개체	테이블	관계
제품 항목 개체	BO_CLASS_CODE	다대일(개체-테이블)
제품 관계 개체	C_RBO_BO_CLASS	다대일(개체-테이블)
	C_RBO_HIERARCHY	일대일(테이블-개체)

John이 이 섹션의 모든 단계를 완료하면 패키지 등과 같은 다른 HM 구성 요소를 생성할 준비가 완료된 것이며 HM에서 자신의 데이터를 볼 수 있습니다. 예를 들어 다음 그래픽은 John이 계층 도구에서 설정하고 계층 관리자에서 표시한 관계를 보여 줍니다. 이 예는 전적으로 마우스 장치와 관련된 계층을 보여 줍니다. 계층 관리자를 사용하는 방법에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.



HM 리포지토리 기본 개체 생성

ORS에서 계층 도구를 사용하려면 시스템에서 먼저 ORS에 대한 리포지토리 기본 개체(RBO 테이블)를 생성해야 합니다. RBO 테이블은 근본적으로 시스템 기본 개체입니다.

이러한 필수 기본 개체에는 특정 열이 포함됩니다.

이러한 RBO 테이블에 대해 쿼리 및 패키지(및 관련 쿼리)도 생성됩니다.

참고: 이러한 RBO 테이블, 쿼리 또는 패키지를 수정하지 마십시오.

RBO를 생성하려면

1. 쓰기 잠금을 획득합니다.
2. 계층 도구를 시작합니다. 모델 작업 영역을 확장하고 **계층**을 클릭합니다.

참고: 탐색 패널을 마우스 오른쪽 단추로 클릭하여 선택할 수 있는 옵션은 모두 계층 도구 메뉴에서도 선택할 수 있습니다.

계층 도구를 시작했지만 ORS에 필요한 RBO 테이블이 없는 경우 계층 도구의 안내에 따라 RBO 테이블을 생성할 수 있습니다.

다음 단계에서는 계층 도구에 표시되는 대화 상자에서 선택할 수 있는 항목에 대해 설명합니다.

1. Hub 콘솔 대화 상자에서 **예**를 선택하여 ORS에서 HM에 대한 메타데이터(RBO 테이블)를 생성합니다.
2. RBO 테이블 생성 대화 상자에서 테이블스페이스 이름을 선택한 다음 **확인**을 클릭합니다.

기본 항목 아이콘 업로드

계층 도구에서 기본 항목 아이콘을 업로드할지 묻는 메시지가 표시됩니다.

이러한 아이콘은 항목을 생성할 때 유용합니다.

1. **예**를 클릭합니다.
2. Hub 콘솔에 기본 메타데이터와 함께 계층 도구가 표시됩니다.

항목 아이콘 구성

계층 도구를 사용하면 항목 유형을 구성할 때 이후에 사용할 수 있는 사용자 고유의 항목 아이콘을 추가하거나 구성할 수 있습니다.

이러한 항목 아이콘은 계층 관리자 내에서 항목을 그래픽으로 표시하는 데 사용됩니다. 항목 아이콘은 JAR 또는 ZIP 파일에 저장해야 합니다.

항목 아이콘 추가

사용자 고유의 아이콘을 가져오려면 아이콘이 포함된 ZIP 또는 JAR 파일을 생성해야 합니다. 각 아이콘에 대해 16 x 16 크기의 작은 아이콘과 48 x 48 크기의 큰 아이콘을 생성합니다.

새 항목 아이콘을 추가하려면

1. 쓰기 잠금을 획득합니다.
2. 계층 도구를 시작합니다.
3. 탐색 창에서 마우스 오른쪽 단추를 클릭하고 **항목 아이콘 추가**를 선택합니다.
참고: 마우스 오른쪽 단추를 클릭하면 나타나는 메뉴를 표시하려면 잠금을 획득해야 합니다.
파일 찾아보기 창이 열립니다.
4. 아이콘이 포함된 JAR 또는 ZIP 파일을 찾습니다.
5. **열기**를 클릭하여 아이콘을 추가합니다.

항목 아이콘 수정

콘솔에서 직접 아이콘을 수정할 수는 없습니다.

ZIP 또는 JAR 파일을 다운로드하고 해당 내용을 수정한 후 파일을 다시 콘솔에 업로드하면 됩니다.

아이콘 그룹을 삭제하거나 이 그룹을 비활성 상태로 설정할 수 있습니다. 아이콘이 이미 항목과 연결된 경우 또는 향후 아이콘 그룹을 사용할 계획이 있는 경우 아이콘 그룹을 삭제하지 말고 비활성 상태로 설정하는 것이 좋습니다.

아이콘 그룹은 아이콘 패키지를 **비활성**으로 표시하여 비활성화할 수 있습니다. 비활성 아이콘은 UI에 표시되지 않으므로 항목 유형에 할당할 수 없습니다. 아이콘 패키지를 다시 활성화하려면 **활성**으로 표시합니다.

참고: Informatica MDM Hub는 아이콘을 삭제하기 전에 아이콘 할당의 유효성을 검사하지 않습니다. 현재 항목 유형에 할당된 아이콘을 삭제할 경우 편집 내용을 저장할 때 오류가 표시됩니다.

항목 아이콘 삭제

콘솔에서 ZIP 또는 JAR 파일의 아이콘을 개별적으로 삭제할 수는 없으며 그룹 또는 패키지로만 아이콘을 삭제할 수 있습니다.

항목 아이콘 그룹을 삭제하려면

1. 쓰기 잠금을 획득합니다.
2. 계층 도구를 시작합니다.
3. 탐색 창에서 아이콘 컬렉션을 마우스 오른쪽 단추로 클릭하고 **항목 삭제** 아이콘을 선택합니다.

항목

계층 관리자에서 항목은 모든 개체, 사람, 장소, 조직 또는 비즈니스 측면에서 의미가 있으며 데이터베이스에서 처리될 수 있는 작업 대상을 나타냅니다.

예를 들어 특정 개인의 이름, 특정 당좌 예금 계좌 번호, 특정 회사, 특정 주소 등이 여기에 해당됩니다.

항목 기본 개체

항목 기본 개체는 계층 관계를 유지 관리하는 데 사용하는 열이 포함된 기본 개체입니다. 기본 개체를 항목 기본 개체로 생성하거나 기본 개체를 항목 기본 개체로 변환하여 항목 기본 개체를 생성할 수 있습니다.

계층 도구를 사용하여 항목 기본 개체를 생성할 때 계층 도구는 계층 관리자에 필요한 열을 생성합니다. 또한 계층 도구의 옵션을 사용하여 기존 기본 개체를 항목 기본 개체로 변환할 수도 있습니다.

참고: 기본 개체에 데이터가 있는 경우 기본 개체를 항목 기본 개체로 변환할 수 없습니다.

항목 기본 개체를 추가한 후 항목 기본 개체를 보고 편집하거나 삭제하려면 스키마 관리자를 사용합니다.

항목 기본 개체를 생성하는 경우 다음 외래 키 열 중 하나를 생성하도록 선택합니다.

ROWID_BO_CLASS

행 ID 개체를 선택하는 경우 계층 도구에서 **ROWID_BO_CLASS** 열을 추가합니다.

ROWID_BO_CLASS 필드는 행 ID 값으로 채워집니다.

BO_CLASS_CODE 또는 기존 열

기본 개체 클래스 코드를 선택하면 기본 개체를 항목 기본 개체로 생성한 경우 계층 관리 도구에서 **BO_CLASS_CODE** 열을 추가합니다. 기존 기본 개체를 항목 기본 개체로 변환하는 경우 **BO_CLASS_CODE** 열을 생성하도록 선택하거나 기존 열을 선택할 수 있습니다.

기본 개체 클래스 노트에서 레코드의 항목 유형을 정의합니다. 필드가 정의한 값으로 채워집니다. 매핑을 구성하여 이 열을 채웁니다.

행 ID는 시스템에서 생성되어 할당되지만 **BO** 클래스 코드는 기억하기 더 쉽도록 사용자가 생성합니다.

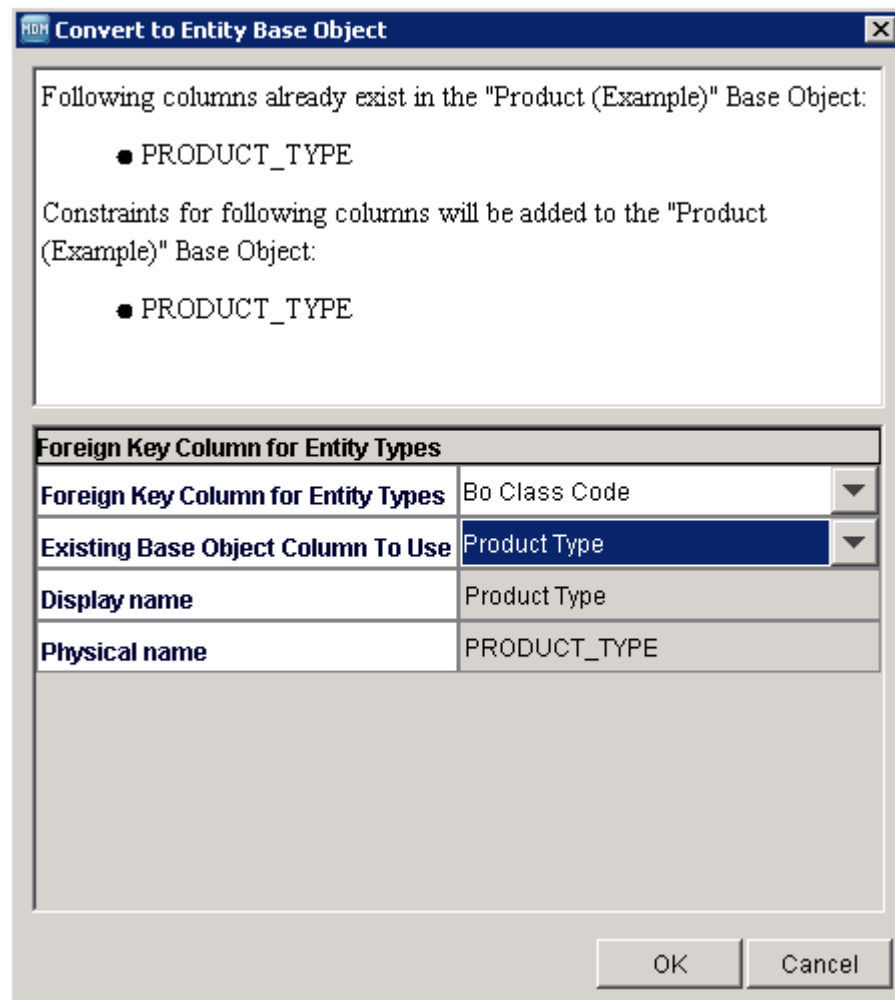
항목 기본 개체 예제

예제의 제품 계층에서 하나의 항목 기본 개체를 생성합니다.

항목 기본 개체를 생성하는 경우 항목 유형을 지정할 방법을 결정해야 합니다. 예제에서는 시스템에서 생성한 행 ID 번호를 사용하지 않고 대신 의미 있는 코드 제품 및 제품 그룹을 사용하여 항목 유형을 지정합니다.

선택한 외래 키 항목 유형을 사용할 것이므로, 먼저 **Product_Type** 열이 있는 기본 개체를 생성합니다. 새 항목 기본 개체를 생성하는 경우에는 사용자 지정 열 이름을 지정할 수 없습니다. 이 기본 개체를 항목 기본 개체로 변환하는 경우 **Product_Type** 열을 항목 유형 외래 키로 선택합니다. 이 계층의 경우 두 개의 항목 유형을 생성합니다. **Product_Type** 열의 값은 레코드가 속한 항목 유형을 정합니다.

다음 이미지는 샘플 항목 기본 개체를 변환하는 선택 사항을 보여 줍니다.



그러면 매핑에서 외래 키 열을 채울 수 있습니다.

항목 기본 개체 작성

항목 기본 개체인 기본 개체를 작성하려면 Hub 콘솔에서 **계층** 도구를 사용합니다.

1. 쓰기 잠금을 획득합니다.
2. **모델** 작업 영역에서 **계층** 도구를 선택합니다.
3. **계층 > 새 항목/관계 개체 작성**을 선택합니다.
4. **새 항목/관계 기본 개체 작성** 대화 상자에서 **새 항목 기본 개체 작성**을 선택한 다음 **확인**을 클릭합니다.

5. 새 항목 기본 개체 작성 대화 상자에서 항목 기본 개체에 대한 다음 속성을 지정합니다.

필드	설명
항목 유형	기본 개체로, 읽기 전용입니다.
표시 이름	Hub 콘솔에서 표시하는 이 기본 개체의 이름입니다.
실제 이름	데이터베이스에서 사용되는 실제 테이블 이름입니다. Informatica MDM Hub에서 입력한 표시 이름을 기준으로 하여 실제 테이블 이름을 제안합니다. 행 ID는 시스템에서 작성되어 할당되지만 BO 클래스 코드는 기억하기 더 쉽도록 사용자가 작성합니다.
데이터 테이블 스페이스	데이터 테이블스페이스의 이름입니다. 기본값은 CMX_DATA입니다.
인덱스 테이블 스페이스	인덱스 테이블스페이스의 이름입니다. 기본값은 CMX_INDX입니다.
로드 테이블스페이스	로드 테이블스페이스의 이름입니다. 기본값은 CMX_DATA입니다.
사용자 임시 테이블스페이스	사용자 임시 테이블스페이스의 이름입니다. 기본값은 CMX_USER_TEMP입니다.
설명	항목 기본 개체의 설명입니다. 선택 사항입니다.
보안 리소스	보안 리소스 도구의 기본 개체 상태를 결정합니다. 활성화된 경우 기본 개체의 상태가 보안입니다. 비활성화된 경우 기본 개체의 상태가 개인입니다. 기본값은 활성화됨입니다.
항목 유형에 대한 외래 키 열	항목 유형에 대한 외래 키 열입니다. 행 ID 개체의 경우 외래 키 관계는 MDM Hub에서 작성한 행 ID를 기반으로 합니다. BO 클래스 개체의 경우 외래 키 관계는 정의한 코드를 기반으로 합니다. 기본값은 행 ID 개체입니다.
표시 이름	Hub 콘솔에 표시되는 열의 이름입니다.
실제 이름	항목 기본 개체의 외래 키 열 이름입니다. Hub 콘솔에서는 입력한 표시 이름을 기준으로 하여 실제 열 이름을 제안합니다.

6. **확인**을 클릭합니다.

MDM Hub은 계층 관리자에서 필요로 하는 열이 있는 항목 기본 개체를 작성합니다.

스키마 도구를 사용하여 항목 기본 개체에 열을 추가합니다. 계층 관리자에서 사용하는 외래 키 열은 수정하지 마십시오. 외래 키 열을 수정하는 경우 계층 관리자가 작업을 예상대로 수행하지 못할 수도 있으며 데이터가 유실될 수 있습니다.

기본 개체를 항목 기본 개체로 변환

계층 관리자에서 사용하려면 먼저 기본 개체를 항목 기본 개체로 변환해야 합니다. 기본 개체를 항목 기본 개체로 변환하려면 먼저 기본 개체에서 상태 관리가 활성화되어야 합니다.

기본 개체에는 계층 관리자에서 필요로 하는 메타데이터가 없습니다. 계층 관리자에서 기본 개체를 사용하려면 변환 프로세스를 사용하여 이 메타데이터를 추가해야 합니다. MDM Hub 및 계층 관리자 모두에서 항목 기본 개체를 사용할 수 있습니다.

1. 계층 도구에서 쓰기 잠금을 획득합니다.

2. 탐색 창에서 아무 곳이나 마우스 오른쪽 단추를 클릭하고 **BO를 항목/관계 개체로 변환**을 선택합니다.
참고: 마우스 오른쪽 단추 메뉴를 클릭하면 나타나는 동일한 옵션을 계층 메뉴에서도 사용할 수 있습니다.
3. 기존 기본 개체 수정 대화 상자에서 **항목 기본 개체로 변환**을 선택하고 **확인**을 클릭합니다.
참고: 기본 개체 수정 필드에 옵션이 표시되지 않으면 사용할 수 있는 비계층 기본 개체가 없는 것입니다. 스키마 도구에서 생성해야 합니다.
4. **확인**을 클릭합니다.
기본 개체에 이미 계층 관리자 메타데이터가 있으면 계층 도구에서 기존 계층 관리자 메타데이터를 나타내는 메시지가 표시됩니다.
5. 항목 유형에 대한 외래 키 열 필드에서 RowId 개체 또는 BO 클래스 코드 중 추가할 열을 선택합니다.
BO 클래스 코드 열을 선택할 수 있으면 MDM Hub에서 생성되는 ROWID 대신 미리 정의된 코드를 기반으로 외래 키 관계를 간단하게 정의할 수 있습니다.
6. 사용할 기존 BO 열에서 기존 열을 선택하거나 새 열 생성 옵션을 선택합니다.
BO 열이 없을 경우 새 열 생성 옵션만 사용할 수 있습니다.
7. 표시 이름 및 실제 이름 필드에서 열에 대한 표시 이름과 실제 이름을 생성하고 **확인**을 클릭합니다.
이제 기본 개체에 계층 관리자에서 필요한 열이 포함됩니다. 열을 더 추가하려면 스키마 관리자를 사용합니다.
중요: 스키마 관리자 도구를 사용하여 기본 개체를 수정하는 경우 계층 도구를 사용하여 추가한 열은 변경하지 마십시오. 이러한 열을 수정하면 예상치 않은 동작이 발생하고 데이터가 유실될 수 있습니다.

항목 유형

항목 유형은 항목을 분류합니다. 항목 유형을 통해 항목 기본 개체 내에서 항목을 분류할 수 있습니다.

항목 유형은 특정 항목에 속합니다. 항목 기본 개체에는 두 개 이상의 항목 유형이 있을 수 있지만 기본 개체의 각 개별 항목 및 하나의 항목 유형에만 속할 수 있습니다. 예를 들어 제품 항목 기본 개체에는 제품 항목 유형 또는 제품 그룹 항목 유형이 있을 수 있습니다. 제품 기본 개체의 항목에는 제품 항목 유형 또는 제품 그룹 항목 유형 중 하나가 있을 수 있습니다.

예를 들어 의사, 당좌 예금 계좌, 은행 등이 여기에 해당됩니다. 항목 기본 개체에는 항목 유형 테이블(Rbo BO 클래스)에 대한 외래 키가 있어야 합니다. ROWID 또는 사전 정의된 코드 값으로 외래 키를 정의할 수 있습니다.

올바르게 정의된 항목 유형은 다음과 같은 특성을 갖습니다.

- 항목 유형을 사용하여 실제 항목 조직을 반영하는 방식으로 레코드를 구성합니다. 항목 유형은 항목의 실제 특성을 반영하는 방식으로 데이터를 구분합니다.
- 종합적으로 볼 때 항목 유형은 전체 항목 집합을 포괄합니다. 즉, 각 항목에는 하나의 항목 유형만 있습니다.
- 항목 유형은 각 항목 유형이 가질 수 있는 관계의 유형을 쉽게 정의할 수 있을 만큼 충분히 세분화되어 있습니다. 예를 들어 "의사"라는 항목 유형은 의료 그룹과 "멤버" 관계에 있을 수 있으며 병원과는 "직원"(또는 "입원 승인 권한이 있는 비직원") 관계에 있을 수 있습니다.
- 간호사, 전문 간호사, 의사 등을 포함하는 보다 일반적인 항목 유형인 "의료 공급자"는 충분히 세분화되어 있지 않습니다. 이 경우 일반적인 항목 유형이 갖는 관계 유형은 해당 관계 유형 외 다른 항목에 따라 달라지게 됩니다. 따라서 보다 세분화된 항목 유형을 정의해야 합니다.

계층 유형을 구성하는 경우 계층 관리자 도구에서 해당 유형의 항목 모양에 대한 아이콘 및 색상을 선택합니다.







항목 유형을 생성하는 경우 다음 속성을 구성할 수 있습니다.

필드	설명
코드	항목 유형의 고유 코드 이름입니다. 항목 기본 개체에서 외래 키로 사용할 수 있습니다. 각 항목 유형에는 고유한 코드 값이 있어야 합니다.
이름 표시	계층 도구에 표시되는 이 항목 유형의 이름입니다.
설명	항목 유형에 대한 설명입니다. 선택 사항입니다.
컬러	이 항목 유형에 연결된 항목의 색상으로, Hub 콘솔의 계층 관리자 도구와 Informatica Data Director 계층 보기에 표시됩니다.
작은 아이콘	이 항목 유형에 연결된 작은 아이콘으로, Hub 콘솔의 계층 관리자 도구와 Informatica Data Director 계층 보기에 표시됩니다. 계층에 표시할 항목의 수가 많으면 작은 아이콘이 나타납니다.
큰 아이콘	이 항목 유형에 연결된 큰 아이콘으로, Hub 콘솔의 계층 관리자 도구와 Informatica Data Director 계층 보기에 표시됩니다.







항목 유형 예제

예제의 항목 유형은 제품 및 제품 그룹입니다.

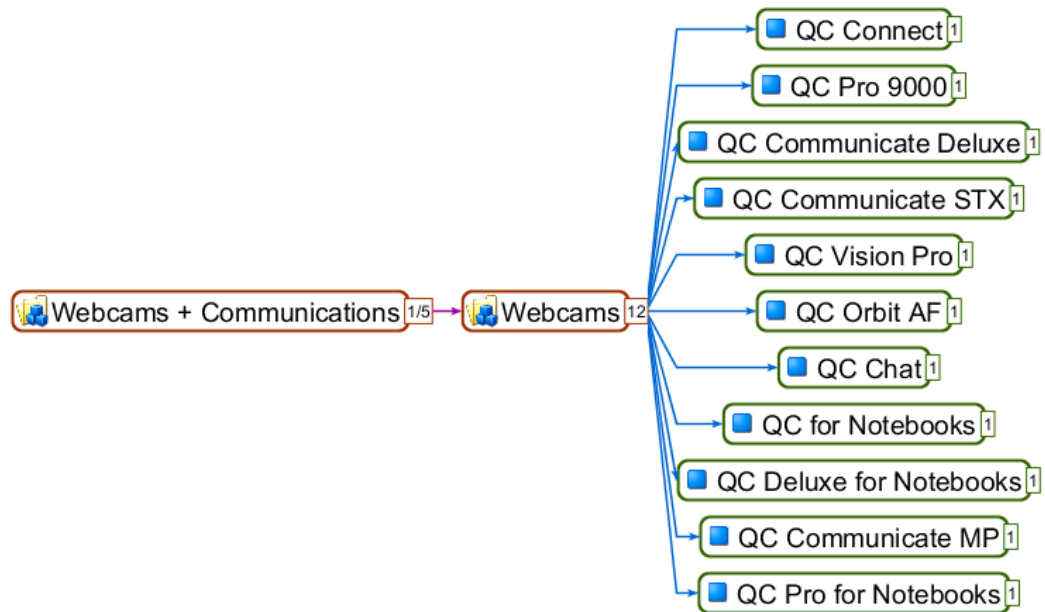
다음 이미지는 제품 항목 유형에 대한 속성을 보여 줍니다.

Entity Type	
Rowid	4
Code	Product
Display Name	Product
Description	
Color	 0x336600 
Small Icon	 HM_Icons/BulletSquareBlue/bullet_square_blue... 
Large Icon	 HM_Icons/BulletSquareBlue/bullet_square... 

다음 이미지는 제품 그룹 항목 유형에 대한 속성을 보여 줍니다.

Entity Type	
Rowid	5
Code	Product Group
Display Name	Product Group
Description	
Color	 0x993300 
Small Icon	 hierarchymanager/Group/Group_Small.png 
Large Icon	 hierarchymanager/Group/Group.png 

다음 이미지는 제품 및 제품 그룹 항목 유형과 함께 계층을 보여 줍니다.



항목 유형 생성

항목 유형을 생성하려면 **계층** 도구에서 항목 개체를 선택한 다음 항목 유형을 추가합니다.

1. 쓰기 잠금을 획득합니다.
2. **모델** 작업 영역에서 **계층** 도구를 선택합니다.
3. 계층 도구 탐색 패널의 **항목 개체** 노드 아래에서 항목 유형을 생성할 항목 기본 개체를 선택합니다.

먼저 항목 기본 개체를 선택하기 전까지는 항목 유형을 추가할 수 없습니다.

4. **계층 > 항목 유형 추가**를 선택합니다.

계층 도구에 이름이 **새 항목 유형**인 항목 유형이 표시됩니다.

5. 항목 유형의 속성을 편집한 다음 **저장**을 클릭합니다.

항목 유형 편집

항목 유형을 편집하려면

1. 계층 도구의 탐색 트리에서 편집할 항목 유형을 클릭합니다.
2. 편집할 각 필드에 대해 **편집** 단추를 클릭하고 필요한 사항을 변경합니다.
3. 변경을 마쳤으면 **저장**을 클릭하여 변경 내용을 저장합니다.

참고: 항목 개체가 코드 열을 사용하는 경우 해당 항목 유형에 대한 레코드가 이미 있다면 항목 유형 코드를 수정하지 않는 것이 좋습니다.

항목 유형 삭제

관계 유형에서 사용되지 않는 항목 유형을 삭제할 수 있습니다.

하나 이상의 관계 유형에서 항목 유형을 사용 중인 경우 해당 항목을 삭제하려고 하면 오류가 나타납니다.

항목 유형을 삭제하려면

1. 쓰기 잠금을 획득합니다.
2. 계층 도구의 탐색 트리에서 삭제할 항목 유형을 마우스 오른쪽 단추로 클릭하고 **항목 유형 삭제**를 선택합니다.
항목 유형이 관계 유형에서 사용되고 있지 않을 경우 계층 도구에서 삭제를 확인하는 메시지가 표시됩니다.
3. **예**를 선택합니다.
목록에서 선택한 항목 유형이 제거됩니다.

참고: 항목 유형을 사용하는 항목 레코드가 이미 있는 경우 해당 항목 유형을 삭제하지 않는 것이 좋습니다. 항목 개체에서 rowid 열 대신 코드 열을 사용하고 해당 항목 유형의 항목 개체에 레코드가 있을 경우 삭제하려고 하면 오류가 발생합니다.

항목에 대한 옵션 표시

항목의 색상과 아이콘을 구성할 수 있을 뿐 아니라 글꼴 크기와 최대 너비도 구성할 수 있습니다.

색상과 아이콘은 각 항목 유형별로 지정할 수 있는 반면 글꼴 크기와 너비는 모든 유형의 항목에 적용됩니다.

HM에서 글꼴 크기를 변경하려면 HM 글꼴 크기 및 항목 상자 크기를 사용합니다. 기본 항목 글꼴 크기(38pt)와 최대 항목 상자 너비(600픽셀)는 cmxserver.properties 파일의 설정을 통해 재정의할 수 있습니다. 사용할 설정은 다음과 같습니다.

```
sip.hm.entity.font.size=fontSize  
sip.hm.entity.max.width=maxWidth
```

fontSize의 값은 6 ~ 100 사이여야 하고 maxWidth의 값은 20 ~ 5000 사이여야 합니다. 지정한 값이 범위를 벗어나면 최소값 또는 최대값이 사용됩니다. 지정한 값이 숫자가 아니면 기본값이 사용됩니다.

항목 기본 개체를 기본 개체로 변환

실수로 기본 개체를 항목 개체로 변환했거나 더 이상 계층 관리자에서 항목 개체를 사용하지 않으려면 항목 개체를 기본 개체로 되돌리면 됩니다.

그렇게 하면 개체에서 **HM** 메타데이터가 제거됩니다.
항목 기본 개체를 기본 개체로 되돌리려면

1. 계층 도구에서 쓰기 잠금을 획득합니다.
2. 항목 기본 개체를 마우스 오른쪽 단추로 클릭하고 **항목/관계 개체를 BO로 되돌리기**를 선택합니다.
3. 연결된 관계 개체를 되돌린다는 메시지가 표시되면 **확인**을 클릭합니다.

참고: 항목 개체를 되돌리면 해당 관계 개체도 되돌려집니다.

4. [항목/관계 개체 되돌리기] 대화 상자에서 **확인**을 클릭합니다.
항목이 되돌려지면 대화 상자가 표시됩니다.

계층 유형

관계 유형을 구성하는 경우 하나 이상의 계층 유형을 관계 유형과 연결해야 합니다.

이러한 관계 유형은 순위가 지정되는 것도 아니고 서로 관련성이 있어야 할 필요도 없습니다. 단지 쉽게 분류하고 식별할 수 있도록 함께 그룹화된 관계 유형일 뿐입니다. 동일한 관계 유형이 여러 계층과 연결될 수 있습니다. 계층 유형은 계층을 논리적으로 분류한 것입니다.

계층 유형을 생성하는 경우 다음 속성을 구성합니다.

필드	설명
코드	계층의 고유 코드 이름입니다. 계층 관리자 관계 기본 개체에서 외래 키로 사용할 수 있습니다. 이 코드는 계층 유형을 생성한 후에는 변경할 수 없습니다.
표시 이름	이 계층의 이름으로, Hub 콘솔에 표시됩니다.
설명	해당 계층에 대한 설명입니다.

처음 계층 유형을 생성하면 이러한 계층 유형에는 참조가 없습니다. 관계 유형을 생성한 계층 유형과 연결하면 계층 도구가 계층 유형 참조를 항목 유형, 관계 유형 및 프로필로 채웁니다.

계층 추가

새 계층을 추가하려면

1. 계층 도구에서 쓰기 잠금을 획득합니다.
2. 탐색 창에서 항목 개체를 마우스 오른쪽 단추로 클릭하고 **계층 추가**를 선택합니다.

계층 도구의 탐색 트리에서 계층 노드 아래에 새 계층(이름: 새 계층)이 표시됩니다. 속성 창에는 기본 속성이 표시됩니다.

3. 이 새 계층에 대해 다음 속성을 지정합니다.
4. **저장**을 클릭하여 새 계층을 저장합니다.

계층 삭제

특정 계층을 사용하는 관계 레코드가 있는 경우 해당 계층을 삭제해서는 안 됩니다.

관계 개체가 rowid 열 대신 계층 코드 열을 사용하고 해당 관계 개체에 삭제하려는 계층에 대한 레코드가 있는 경우 오류가 발생합니다.

계층을 삭제하려면

1. 계층 도구에서 쓰기 잠금을 획득합니다.
2. 탐색 트리에서 삭제할 계층을 마우스 오른쪽 단추로 클릭하고 **계층 삭제**를 선택합니다.

계층 도구에 삭제를 확인하는 메시지가 나타납니다.

3. **예**를 선택합니다.

선택한 계층이 목록에서 제거됩니다.

참고: 관계 유형이 연결되어 있는 계층을 삭제할 수 있습니다. 이 경우 연결된 관계 유형 목록이 경고와 함께 나타납니다. 계층을 삭제하도록 선택한 경우 계층에 대한 모든 참조가 자동으로 제거됩니다.

관계 개체

관계 개체는 특정 두 항목 간의 소속을 설명합니다.

관계 개체는 다음 개체 중 하나일 수 있습니다.

관계 기본 개체

두 항목 기본 개체 간의 관계를 유지 관리합니다. 관계 기본 개체를 사용하여 다대다 관계를 설정합니다.

외래 키 기본 개체

외래 키 기본 개체는 외래 키 열이 있는 항목 기본 개체입니다. 외래 키 기본 개체를 사용하여 일대일 또는 일대다 관계를 설정합니다. 예를 들어 여러 제품이 단일 제품 그룹과 관련될 수도 있습니다. 항목 기본 개체에는 둘 이상의 외래 키 열이 있을 수 있습니다. 각 외래 키 열은 관계를 유지 관리합니다.

계층 관계는 관계 유형, 계층 유형, 관계 특성 및 관계가 활성 상태인 날짜를 지정하는 방식으로 정의됩니다.

스키마 도구의 관계 노드에 계층 관계 및 데이터 모델 관계가 표시됩니다.

관계 기본 개체

관계 기본 개체는 계층 관계를 저장하는 기본 개체입니다. 외래 키 관계 기본 개체는 다른 항목 기본 개체에 대한 외래 키가 포함된 항목 기본 개체입니다. 관계 기본 개체는 두 개의 항목 기본 개체를 연결하는 테이블입니다. 외래 키 관계 유형은 단일 계층에만 연결할 수 있습니다.

두 개의 특정 항목 간에는 하나의 관계만 있을 수 있습니다. 두 항목 간에 관계를 더 추가하려고 하면 새 관계가 추가되는 것이 아니고 기존 관계가 업데이트됩니다.

스키마 관리자를 사용하여 기본 개체를 수정하는 경우 계층 관리자가 추가한 열은 변경하지 마십시오. 이러한 열을 수정하면 예상치 않은 동작이 발생하고 데이터가 유실될 수 있습니다.

다음 테이블에는 관계 기본 개체를 생성할 때 구성할 수 있는 속성이 설명되어 있습니다.

속성	설명
항목 유형	읽기 전용. 이미 지정되어 있습니다.
표시 이름	Hub 콘솔에 표시되는 기본 개체 이름입니다.
실제 이름	데이터베이스에서 사용되는 실제 테이블 이름입니다. Informatica MDM Hub에서 입력한 표시 이름을 기준으로 하여 실제 테이블 이름을 제안합니다.
데이터 테이블스페이스	데이터 테이블스페이스의 이름입니다. 자세한 내용은 <i>Multidomain MDM 설치 가이드</i> 를 참조하십시오.
인덱스 테이블스페이스	인덱스 테이블스페이스의 이름입니다. 자세한 내용은 <i>Multidomain MDM 설치 가이드</i> 를 참조하십시오.
설명	이 기본 개체에 대한 설명입니다.
항목 기본 개체 1	이 관계 기본 개체를 통해 연결할 항목 기본 개체입니다.
표시 이름	항목 기본 개체 1에 대한 FK인 열의 이름입니다.
실제 이름	데이터베이스 열의 실제 이름입니다. Informatica MDM Hub에서는 사용자가 입력한 표시 이름을 기반으로 열의 실제 이름이 제안됩니다.
항목 기본 개체 2	이 관계 기본 개체를 통해 연결할 항목 기본 개체입니다.
표시 이름	항목 기본 개체 2에 대한 FK인 열의 이름입니다.
실제 이름	데이터베이스 열의 실제 이름입니다. Informatica MDM Hub에서는 사용자가 입력한 표시 이름을 기반으로 열의 실제 이름이 제안됩니다.
계층 FK 열	계층의 외래 키로 사용되는 열로, ROWID 또는 CODE일 수 있습니다. BO 클래스 CODE 열을 선택할 수 있으면 Informatica MDM Hub에서 생성되는 ROWID 대신 미리 정의된 코드를 기반으로 외래 키 관계를 간단하게 정의할 수 있습니다.
계층 FK 표시 이름	Hub 콘솔에 표시되는 FK 열의 이름입니다.
계층 FK 실제 이름	테이블에서 계층 외래 키 열의 실제 이름입니다. Informatica MDM Hub에서는 사용자가 입력한 표시 이름을 기반으로 열의 실제 이름이 제안됩니다.
관계 유형 FK 열	관계에 대한 외래 키로 사용되는 열로, ROWID 또는 CODE 중 하나일 수 있습니다.
관계 유형 표시 이름	관계 유형 CODE 또는 ROWID를 저장하는 데 사용되는 열의 이름입니다.
관계 유형 실제 이름	테이블에서 관계 유형 FK 열의 실제 이름입니다. Informatica MDM Hub에서는 사용자가 입력한 표시 이름을 기반으로 열의 실제 이름이 제안됩니다.

관계 기본 개체 생성

관계 기본 개체를 생성하려면 **계층** 도구를 사용하여 외래 키 제약 조건을 구성합니다.

1. 쓰기 잠금을 획득합니다.

2. 모델 작업 영역에서 계층 도구를 선택합니다.
3. 계층 > 새 항목/관계 개체 생성을 선택합니다.
4. 새 항목/관계 기본 개체 생성 대화 상자에서 새 관계 기본 개체 생성을 선택합니다. 확인을 클릭합니다.
5. 새 관계 기본 개체 생성 대화 상자에서 관계 기본 개체 속성을 지정합니다. 확인을 클릭합니다.

기본 개체를 관계 기본 개체로 변환

관계 기본 개체는 두 항목 기본 개체에 대한 정보를 포함하는 테이블입니다.

기본 개체에는 관계 정보에 대해 계층 관리자에서 필요로 하는 메타데이터가 없습니다. 기본 개체를 사용하려면 먼저 기본 개체를 항목 개체로 변환한 다음 관계 개체에서 연결할 항목 개체를 선택합니다.

1. 계층 도구에서 쓰기 잠금을 획득합니다.
2. 탐색 창에서 마우스 오른쪽 단추를 클릭하고 **BO를 항목/관계 개체로 변환**을 선택합니다.
3. **확인**을 클릭합니다.
관계 기본 개체로 변환 화면이 표시됩니다.
4. 이 기본 개체에 대해 다음 속성을 지정합니다.

필드	설명
항목 기본 개체 1	이 관계 기본 개체를 통해 연결할 항목 기본 개체입니다.
표시 이름	항목 기본 개체 1에 대한 FK인 열의 이름입니다.
실제 이름	데이터베이스 열의 실제 이름입니다. Informatica MDM Hub에서는 사용자가 입력한 표시 이름을 기반으로 열의 실제 이름이 제안됩니다.
항목 기본 개체 2	이 관계 기본 개체를 통해 연결할 항목 기본 개체입니다.
표시 이름	항목 기본 개체 2에 대한 FK인 열의 이름입니다.
실제 이름	데이터베이스 열의 실제 이름입니다. Informatica MDM Hub에서는 사용자가 입력한 표시 이름을 기반으로 열의 실제 이름이 제안됩니다.
계층 FK 열	계층의 외래 키로 사용되는 열로, ROWID 또는 코드일 수 있습니다. BO 클래스 코드 열을 선택할 수 있으면 Informatica MDM Hub에서 생성되는 ROWID 대신 미리 정의된 코드를 기반으로 외래 키 관계를 간단하게 정의할 수 있습니다.
사용할 기존 BO 열	기존 BO에서 사용할 실제 열입니다.
계층 FK 표시 이름	Hub 콘솔에 표시되는 FK 열의 이름입니다.
계층 FK 실제 이름	테이블에서 계층 외래 키 열의 실제 이름입니다. Informatica MDM Hub에서는 사용자가 입력한 표시 이름을 기반으로 열의 실제 이름이 제안됩니다.
관계 유형 FK 열	관계에 대한 외래 키로 사용되는 열로, ROWID 또는 CODE 중 하나일 수 있습니다.
사용할 기존 BO 열	기존 BO에서 사용할 실제 열입니다.

필드	설명
관계 유형 FK 표시 이름	관계 유형 코드 또는 ROWID를 저장하는 데 사용되는 FK 열의 이름입니다.
관계 유형 FK 실제 이름	테이블에서 관계 유형 FK 열의 실제 이름입니다. Informatica MDM Hub에서는 사용자가 입력한 표시 이름을 기반으로 열의 실제 이름이 제안됩니다.

5. **확인**을 클릭합니다.

참고: 스키마 관리자 도구를 사용하여 기본 개체를 수정하는 경우 계층 관리자에서 추가한 열은 변경하지 마십시오. 이러한 열을 수정하면 예상치 않은 동작이 발생하고 데이터가 유실될 수 있습니다.

관계 기본 개체를 기본 개체로 변환

관계 기본 개체를 기본 개체로 되돌리면 관계 개체에서 계층 관리자 메타데이터가 제거됩니다. 관계 개체는 기본 개체로 남아 있지만 계층 관리자에는 기본 개체가 표시되지 않습니다.

되돌리려는 관계-유형 열이 조화를 위한 준비 테이블에 있는 경우 관계 기본 개체를 되돌리려면 준비 테이블 열이 비어 있어야 합니다.

1. 계층 도구에서 쓰기 잠금을 획득합니다.
2. 관계 기본 개체를 마우스 오른쪽 단추로 클릭하고 **항목/관계 개체를 BO로 되돌리기**를 선택합니다.
3. **항목/관계 개체 되돌리기** 대화 상자에서 **확인**을 클릭합니다.
항목이 되돌려지면 대화 상자가 표시됩니다.

외래 키 관계 기본 개체

외래 키 관계 기본 개체는 다른 항목 기본 개체에 대한 외래 키가 있는 항목 기본 개체입니다.

항목 기본 개체가 있어야 하며 외래 키 열을 추가하여 외래 키 관계 기본 개체로 변환해야 합니다.

외래 키 관계 기본 개체를 작성할 때 외래 키 열을 추가할 항목 기본 개체를 선택합니다.

필드	설명
FK 제약 조건 항목 BO 1	목록에서 FK 항목 기본 개체를 선택합니다.
사용할 기존 BO 열	외래 키에 사용하거나 새 열을 작성하기 위해 선택하는 기존 기본 개체 열의 이름입니다.
FK 열 표시 이름 1	Hub 콘솔에 표시될 FK 열의 이름입니다.
FK 열 실제 이름 1	데이터베이스에 표시될 FK 열의 실제 이름입니다. Hub 콘솔에서는 입력한 표시 이름을 기준으로 실제 테이블 이름을 제안합니다.
FK 열이 나타내는 항목	관계에서 FK 열이 나타내는 항목에 따라 항목 1 또는 항목 2를 선택합니다.

외래 키 관계를 사용하려면 연결할 항목 기본 개체를 작성합니다. 항목 간에 일대다 관계가 있는 경우 외래 키 관계를 사용합니다.

외래 키 관계를 작성하는 경우 연결할 각 항목에 대해 외래 키 열을 추가해야 합니다.

외래 키 관계 기본 개체 생성

외래 키 관계 기본 개체를 생성하려면 **계층** 도구를 사용합니다.

1. 쓰기 잠금을 획득합니다.
2. **모델** 작업 영역에서 **계층** 도구를 선택합니다.
3. **계층 > 외래 키 관계 개체 생성**을 선택합니다.
4. **기존 기본 개체 수정** 대화 상자에서 외래 키를 추가할 항목 기본 개체를 선택하고 외래 키 열의 수를 지정합니다. **확인**을 클릭합니다.
5. 외래 키 열 속성을 선택한 다음 **확인**을 클릭합니다.

관계 유형

관계 유형은 관계 클래스를 설명하고 이 유형의 관계에 포함될 수 있는 항목의 유형, 관계의 방향(있는 경우) 및 관계가 **Hub** 콘솔에 표시되는 방식을 정의합니다. 관계 기본 개체 또는 외래 키 관계에 대한 관계 유형을 생성할 수 있습니다.

참고: 계층 유형은 논리적 구성에 가깝고 일반적으로 구성 양이 적은 반면 관계 유형은 실제 구성으로, 구성 양이 많을 수 있습니다. 따라서 여러 관계 유형을 포함하는 것보다 여러 계층 유형을 포함하는 것이 더 간편한 경우가 많습니다. **MDM Hub** 내에서 계층 유형 및 관계 유형을 정의하기 전에 데이터와 계층 관리 요구 사항을 이해해야 합니다.

제대로 정의된 계층 관계 유형 집합에는 다음 특성이 있습니다.

- 항목 유형 간 실제 관계를 나타냅니다.
- 각 관계에 대해 여러 관계 유형이 지원됩니다.

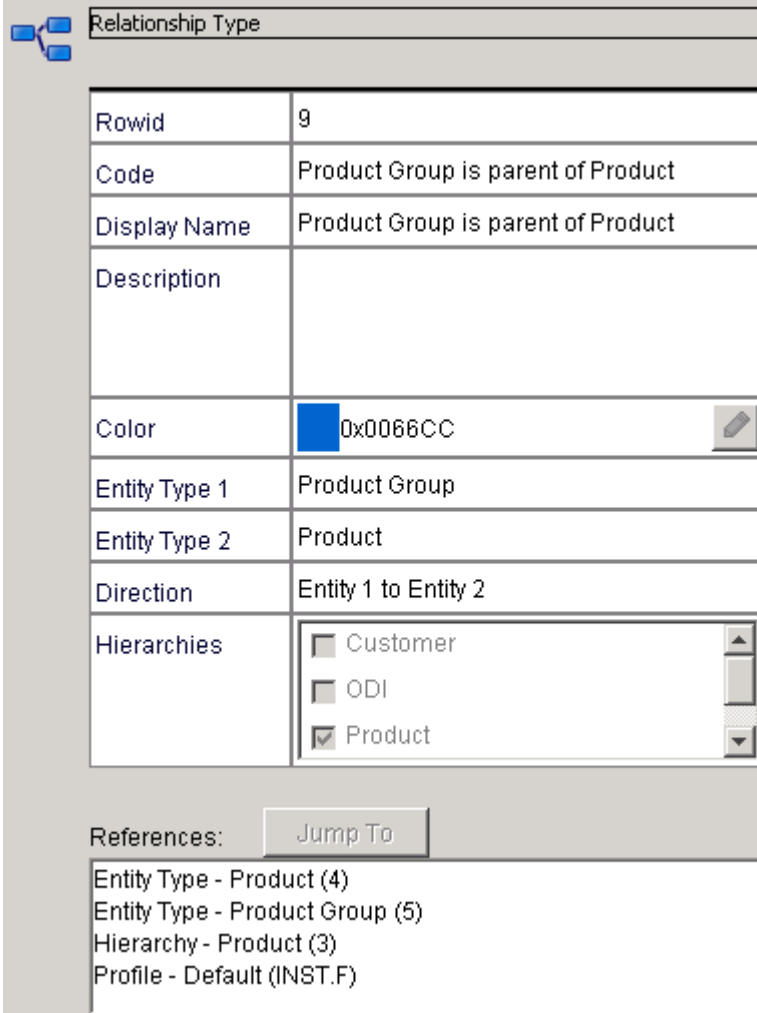
필드	설명
코드	관계 유형의 고유 코드 이름입니다. 계층 관계 기본 개체에서 외래 키로 사용할 수 있습니다.
표시 이름	Hub 콘솔에 표시되는 관계 유형 이름입니다. 고유한 설명 이름을 지정합니다.
설명	해당 관계 유형에 대한 설명입니다.
색상	해당 관계 유형과 연결된 관계를 Hub 콘솔의 계층 관리자 콘솔과 Informatica Data Director에 표시하는 데 사용되는 색상입니다.
항목 유형 1	이 새 관계 유형과 연결되는 첫 번째 항목 유형입니다. 이 유형의 모든 항목은 이 관계 유형의 관계를 가질 수 있습니다.
항목 유형 2	이 새 관계 유형과 연결되는 두 번째 항목 유형입니다. 이 유형의 모든 항목은 이 관계 유형의 관계를 가질 수 있습니다.



필드	설명
방향	<p>새 관계 유형에서 방향이 지정된 계층을 허용하기 위한 방향을 선택합니다. 가능한 방향은 다음과 같습니다.</p> <ul style="list-style-type: none"> - 항목 1에서 항목 2로 - 항목 2에서 항목 1로 - 방향 지정 안 됨 - 양방향 - 알 수 없음 <p>방향이 지정된 계층의 예로는 직원에서 상급 직원을 거쳐 조직의 최고직까지 이어지는 보고 관계를 사용하는 조직도를 들 수 있습니다.</p>
FK 관계 시작 날짜	외래 키 관계의 시작 날짜입니다.
FK 관계 종료 날짜	외래 키 관계의 종료 날짜입니다.
계층	이 새 관계 유형과 연결할 계층 옆의 확인란을 선택합니다. 선택한 계층에는 이 관계 유형의 관계가 포함될 수 있습니다.

관계 유형 예제

예제에서는 두 개의 관계 유형이 있습니다. 첫 번째 관계에서 제품 그룹 항목 유형은 제품 항목 유형의 상위입니다. 두 번째 관계에서 제품 그룹 항목 유형은 제품 그룹 항목 유형의 상위입니다.

다음 그림에서는 제품 그룹의 속성이 제품 관계 관계 개체에 대한 제품 관계 유형의 상위임을 보여 줍니다.

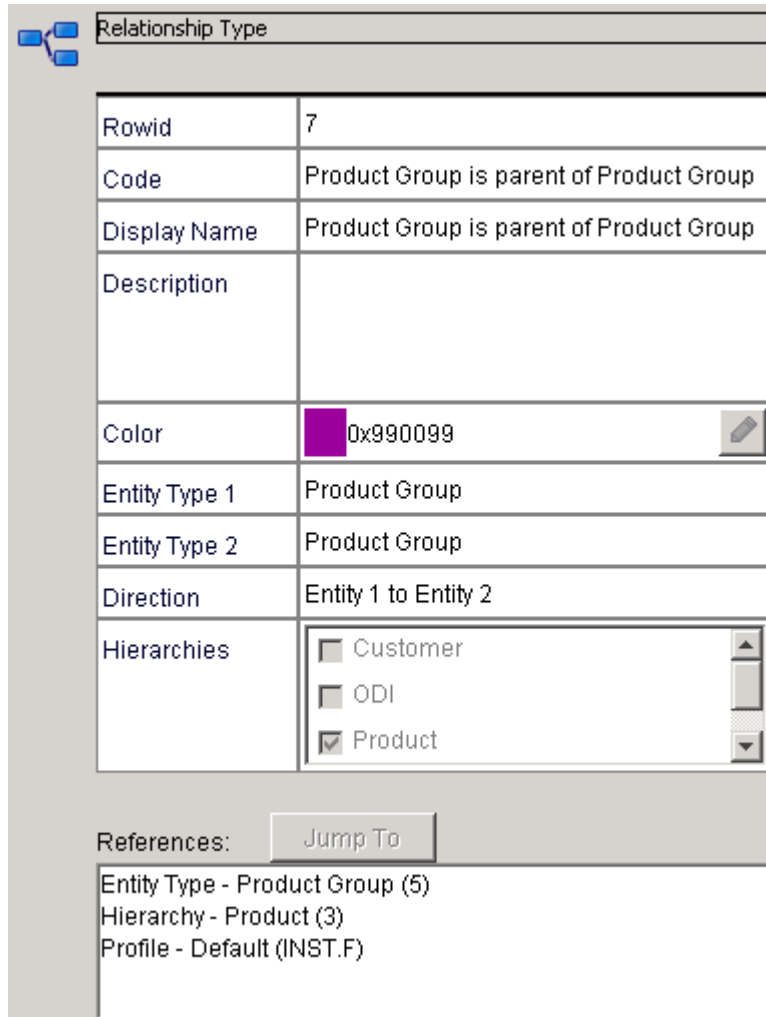


Relationship Type	
Rowid	9
Code	Product Group is parent of Product
Display Name	Product Group is parent of Product
Description	
Color	 0x0066CC 
Entity Type 1	Product Group
Entity Type 2	Product
Direction	Entity 1 to Entity 2
Hierarchies	<input type="checkbox"/> Customer <input type="checkbox"/> ODI <input checked="" type="checkbox"/> Product

References: [Jump To](#)

- Entity Type - Product (4)
- Entity Type - Product Group (5)
- Hierarchy - Product (3)
- Profile - Default (INST.F)

다음 그림에서는 제품 그룹의 속성이 제품 관계 관계 개체에 대한 제품 그룹 관계 유형의 상위임을 보여 줍니다.



Relationship Type

Rowid	7
Code	Product Group is parent of Product Group
Display Name	Product Group is parent of Product Group
Description	
Color	<div style="display: flex; align-items: center;"> <div style="width: 20px; height: 20px; background-color: #990099; margin-right: 5px;"></div> <div>0x990099</div> <div style="margin-left: 10px; border: 1px solid #ccc; padding: 2px;">✎</div> </div>
Entity Type 1	Product Group
Entity Type 2	Product Group
Direction	Entity 1 to Entity 2
Hierarchies	<div style="border: 1px solid #ccc; padding: 5px;"> <div><input type="checkbox"/> Customer</div> <div><input type="checkbox"/> ODI</div> <div><input checked="" type="checkbox"/> Product</div> </div>

References:

Jump To

Entity Type - Product Group (5)
Hierarchy - Product (3)
Profile - Default (INST.F)

관계 유형 생성

관계 유형을 생성하려면 **계층** 도구를 사용하여 관계 기본 개체에 관계 유형을 추가합니다.

1. 쓰기 잠금을 획득합니다.
2. **모델** 작업 영역에서 **계층** 도구를 선택합니다.
3. **계층** 도구 탐색 패널에서 관계 유형을 생성할 관계 기본 개체를 선택합니다.
먼저 관계 기본 개체를 선택하기 전까지는 관계 유형을 추가할 수 없습니다.
4. **계층 > 관계 유형 추가**를 선택합니다.
계층 도구에 이름이 **새 관계 유형**인 관계 유형이 표시됩니다.
5. 관계 유형의 속성을 편집한 다음 **저장**을 클릭합니다.

관계 유형 편집

관계 유형을 편집하려면

1. 계층 도구에서 쓰기 잠금을 획득합니다.
2. 탐색 트리에서 편집할 관계 유형을 클릭합니다.
3. 편집할 각 필드에 대해 **편집**을 클릭하고 필요한 사항을 변경합니다.
4. 변경을 마쳤으면 **저장**을 클릭하여 변경 내용을 저장합니다.

참고: 관계 개체가 코드 열을 사용하는 경우 해당 관계 유형에 대한 레코드가 이미 있다면 관계 유형 코드를 수정하지 않는 것이 좋습니다.

FK 관계 유형에는 이 경고가 적용되지 않습니다.

관계 유형 삭제

참고: 관계 유형을 사용하는 관계 레코드가 이미 있는 경우 해당 관계 유형을 삭제하지 않는 것이 좋습니다. 관계 개체가 rowid 열 대신 관계 유형 코드 열을 사용하고 해당 관계 개체에 삭제하려는 관계 유형에 대한 레코드가 있는 경우 오류가 발생합니다.

위 경고는 FK 관계 유형에는 적용되지 않습니다. 계층과 연결된 관계 유형은 삭제할 수 있습니다. 관계 유형과 연결된 계층을 삭제할지 묻는 확인 대화 상자가 표시됩니다.

관계 유형을 삭제하려면

1. 계층 도구에서 쓰기 잠금을 획득합니다.
2. 탐색 트리에서 삭제할 관계 유형을 마우스 오른쪽 단추로 클릭하고 **관계 유형 삭제**를 선택합니다.
계층 도구에 삭제를 확인하는 메시지가 나타납니다.
3. **예**를 선택합니다.
선택한 관계 유형이 목록에서 제거됩니다.

패키지

이 섹션에서는 계층 도구를 사용하여 스키마에 패키지를 추가하는 방법에 대해 설명합니다. 항목 기본 개체, 관계 기본 개체 및 외래 키 관계 기본 개체에 대한 패키지를 생성할 수 있습니다. 패키지에서 레코드가 삽입 또는 변경되는 경우 놓기 옵션을 활성화해야 합니다.

패키지는 Informatica MDM Hub에서 하나 이상의 기본 테이블에 대한 공용 보기입니다. 패키지는 이러한 테이블과 이러한 테이블에 조인된 다른 테이블에 있는 열의 하위 집합을 나타냅니다. 패키지는 쿼리를 기반으로 합니다. 기본 쿼리에서는 테이블 또는 다른 패키지의 레코드 하위 집합을 선택할 수 있습니다. 패키지는 기본 데이터의 사용자 뷰를 구성하는 데 사용됩니다.

먼저 패키지를 생성한 다음 패키지를 항목 유형 또는 관계 유형과 연결해야 합니다. 외래 키 관계 개체에 대한 표시 패키지의 경우에는 REL_START_DATE 및 REL_END_DATE 열을 패키지에 포함해야 합니다.

참고: 로드 작업을 실행하기 전에 패키지를 구성하고 연결된 프로필의 유효성을 검사해야 합니다.

계층 관리자 데이터 구성

계층 관리자 도구에서 사용자가 액세스할 수 있는 데이터를 구성할 수 있습니다.

계층 패키지를 구성하는 경우 사용자 인터페이스에서 표시할 항목 필드 및 표시 순서를 구성합니다. 패키지의 각 필드에 대해 0부터 숫자를 선택할 수 있습니다. 필드에 값 0을 할당한 경우 해당 필드는 사용자 인터페이스에서 필드 목록 맨 위에 표시됩니다.

다음 계층 관리자 사용자 인터페이스 요소에 대해 표시할 필드 및 필드 표시 순서를 구성할 수 있습니다.

레이블

숫자를 사용하여 계층 관리자 도구에 표시할 항목 필드를 지정합니다.

도구 설명

가장 낮은 값을 할당한 열이 계층 관리자 도구의 항목에 마우스를 올려 놓으면 나타나는 도구 설명에 필드로 표시됩니다.

공통

다양한 유형의 항목 및 관계가 동일한 목록에 나타날 때 표시되는 필드입니다. 선택한 필드는 모든 계층 관리자 패키지에 있어야 합니다.

검색

레코드를 검색하는 데 사용할 수 있는 필드입니다.

목록

검색 결과 목록에 표시되는 필드입니다.

세부 정보

특정 항목의 세부 정보를 보기 위해 선택할 때 표시되는 필드입니다.

놓기

특정 항목을 수정할 때 변경할 수 있는 값에 대한 필드입니다.

추가

계층 관리자에서 항목을 추가할 때 표시되는 필드입니다.

항목, 관계 및 FK 관계 개체 패키지 생성

HM 패키지를 생성하려면

1. 계층 도구에서 탐색 창 아무 곳이나 마우스 오른쪽 단추로 클릭하고 **새 패키지 생성**을 선택합니다.
2. 쓰기 잠금을 획득합니다.

계층 도구에서 새 패키지 생성 마법사가 시작되고 첫 번째 대화 상자가 표시됩니다.

3. 새 패키지에 대해 다음 정보를 지정합니다.

필드	설명
패키지 유형	다음 유형 중 하나입니다. 항목 개체 관계 개체 FK 관계 개체
쿼리 그룹	기존 쿼리 그룹을 선택하거나 새 쿼리 그룹 생성을 선택합니다. Informatica MDM Hub에서 쿼리 그룹은 쿼리의 논리적 그룹입니다. 쿼리가 구성된 쿼리 그룹을 선택합니다.
쿼리 그룹 이름	새 쿼리 그룹의 이름입니다. 위에서 새 그룹을 생성하도록 선택한 경우에만 필요합니다.
설명	생성하는 새 쿼리 그룹에 대한 선택적 설명입니다.

4. 다음을 클릭합니다.

새 패키지 생성 마법사에 다음 대화 상자가 표시됩니다.

5. 새 패키지에 대해 다음 정보를 지정합니다.

필드	설명
쿼리 이름	쿼리의 이름입니다. Informatica MDM Hub에서 쿼리는 Hub 저장소에서 데이터를 검색하는 요청입니다.
설명	선택적 설명입니다.
기본 테이블 선택	이 쿼리의 기본 테이블입니다.

6. 다음을 클릭합니다.

새 패키지 생성 마법사에 다음 대화 상자가 표시됩니다.

7. 새 패키지에 대해 다음 정보를 지정합니다.

필드	설명
표시 이름	이 패키지에 대한 표시 이름으로, 패키지 도구에서 이 패키지를 표시하는 데 사용됩니다.
실제 이름	이 패키지의 실제 이름입니다. 입력한 표시 이름을 기반으로 Hub 콘솔에 실제 이름이 제시됩니다.
설명	선택적 설명입니다.

필드	설명
PUT 활성화	레코드를 삽입하거나 변경할 수 있게 하려면 선택합니다. (선택 사항) 이 항목을 선택하지 않으면 패키지가 읽기 전용이 됩니다. 외래 키 관계 개체 패키지를 생성하는 경우 이 절차의 9단계에서 추가적인 단계를 수행해야 합니다. 참고: 모든 외래 키 관계에 대해 PUT 및 비PUT 패키지가 모두 있어야 합니다. 동일한 외래 키 관계 개체에 대해 생성한 Put 및 비Put 패키지에는 동일한 열이 있어야 합니다.
보안 리소스	보안 리소스를 생성하려면 선택합니다. (선택 사항)

8. 다음을 클릭합니다.

새 패키지 생성 마법사에 마지막 대화 상자가 표시됩니다. 표시되는 대화 상자는 생성하는 패키지 유형에 따라 달라집니다.

항목 또는 관계에 대한 패키지나 FK 관계에 대한 PUT 패키지를 생성하도록 선택한 경우에는 다음 대화 상자와 유사한 대화 상자가 표시됩니다. 필요한 열(회색으로 표시됨)은 자동으로 선택되며 선택을 취소할 수 없습니다.

패키지와 관련이 없는 열을 선택 취소합니다.

참고: 모든 외래 키 관계에 대해 PUT 및 비PUT 패키지가 모두 있어야 합니다. 동일한 외래 키 관계 개체에 대해 생성한 Put 및 비Put 패키지에는 동일한 열이 있어야 합니다.

외래 키 관계에 대해 비Put 패키지를 생성하도록 선택한 경우(이 절차의 7단계 참조 - Put 확인란 선택 안 함) 다음 대화 상자가 표시됩니다.

9. 외래 키 관계에 대해 비Put 활성화 패키지를 생성하는 경우에는 이 새 패키지에 대해 다음 정보를 지정합니다.

필드	설명
계층	이 패키지와 연결된 계층입니다.
관계 유형	이 패키지와 연결된 관계 유형입니다.

참고: 모든 외래 키 관계에 대해 PUT 및 비PUT 패키지가 모두 있어야 합니다. 동일한 외래 키 관계 개체에 대해 생성한 Put 및 비Put 패키지에는 동일한 열이 있어야 합니다.

10. 이 새 패키지에 대한 열을 선택합니다.

11. 마침을 클릭하여 패키지를 생성합니다.

새로 생성한 패키지를 보거나 편집 또는 삭제하려면 패키지 도구를 사용합니다.

계층 관리자에 필요한 열을 제거해서는 안 됩니다. 이러한 열은 사용자가 계층 도구를 사용하여 패키지를 생성할 때 자동으로 선택(회색으로 표시)됩니다.

항목 또는 관계 유형에 패키지 할당

프로필과 프로필의 각 항목/관계 유형에 대한 패키지를 생성한 후에는 항목 또는 관계 유형에 패키지를 할당해야 합니다. 이렇게 하면 항목이 계층 관리자에 표시될 때 함께 표시되는 필드가 정의됩니다. 관계 유형 및 항목 유형에 대한 패키지를 할당할 수도 있습니다.

항목/관계 유형에 패키지를 할당하려면

1. 쓰기 잠금을 획득합니다.
2. 계층 도구에서 항목/관계 유형을 선택합니다.

계층 관리자에 해당 유형의 패키지에 대한 속성(있는 경우)이 표시되거나, 빈 필드가 포함된 동일한 속성 창이 표시됩니다. 표시 및 Put 패키지를 선택한 경우 아래쪽 패널에 계층 관리자 패키지 열 정보가 표시됩니다.

셀 안의 수는 특성이 표시되는 순서를 정의합니다.

3. 항목 또는 관계 유형에 대한 패키지를 구성합니다.

필드	설명
레이블	계층 관리자 그래픽 콘솔에서 보려는 항목/관계의 레이블을 표시하는 데 사용되는 열입니다. 이러한 열은 계층 관리자 콘솔과 Informatica Data Director에서 레이블 패턴을 생성하는 데 사용됩니다. 레이블을 편집하려면 레이블 오른쪽의 레이블 값을 클릭합니다. 패턴 편집 대화 상자에서는 새 레이블을 입력하거나, 열을 두 번 클릭하여 패턴에 사용할 수 있습니다.
도구 설명	항목/관계 위로 스크롤할 때 나타나는 설명을 표시하는 데 사용되는 열입니다. 계층 관리자 콘솔과 Informatica Data Director의 도구 설명 패턴을 생성하는 데 사용됩니다. 도구 설명을 편집하려면 "도구 설명 패턴" 레이블 오른쪽의 도구 설명 패턴 값을 클릭합니다. 패턴 편집 대화 상자에서는 새 도구 설명 패턴을 입력하거나, 열을 두 번 클릭하여 패턴에 사용할 수 있습니다.
공통	서로 다른 유형의 항목/관계가 동일한 목록에 표시될 때 사용되는 열입니다. 선택한 열은 프로필의 모든 항목/관계 유형과 연결된 패키지에 포함되어야 합니다.
검색	검색 도구와 함께 사용할 수 있는 열입니다.
목록	검색 결과에 표시되는 열입니다.
세부 정보	화면 아래쪽에 표시되는 항목/관계의 세부 정보 보기에 사용되는 열입니다.
Put	레코드를 편집하려고 할 때 표시되는 열입니다.
추가	새 레코드를 생성하려고 할 때 표시되는 열입니다.

4. 변경을 마쳤으면 **저장**을 클릭하여 변경 내용을 저장합니다.

프로필 정보

계층 프로필은 계층 개체에 대한 사용자 액세스를 정의합니다.

프로필은 계층 관리자에서 사용자가 표시, 편집 또는 추가할 수 있는 필드 및 레코드를 정의합니다. 예를 들어 전체 항목 및 관계에 대한 모든 읽기/쓰기 권한을 허용하는 프로필과 추가 또는 편집 작업은 허용하지 않고 읽기만 허용하는 프로필이 있을 수 있습니다. 프로필을 정의한 후에는 프로필을 보안 리소스로 구성할 수 있습니다.

프로필 추가

HM에 액세스하기 전에 새 프로필(이름: "기본값")이 자동으로 생성되어 있습니다. 기본 프로필을 유지 관리할 수 있을 뿐 아니라 다른 프로필을 추가할 수도 있습니다.

참고: Informatica Data Director에서는 기본 프로필을 사용하여 항목 레이블과 관계 및 항목 도구 설명이 표시되는 방식을 정의합니다. 추가 프로필과 프로필 내에 정의된 추가 정보는 계층 관리자 콘솔에서만 사용되고 Informatica Data Director에서는 사용되지 않습니다.

새 프로필을 추가하려면

1. 쓰기 잠금을 획득합니다.

2. 계층 도구에서 탐색 창 내부를 마우스 오른쪽 단추로 클릭하고 **프로필 추가**를 선택합니다.

계층 도구의 탐색 트리에서 프로필 노드 아래에 새 프로필(이름: **새 프로필**)이 표시됩니다. 속성 창에는 기본 속성이 표시됩니다.

이러한 관계 유형을 선택하고 "저장"을 클릭하면 트리의 프로필 아래에 항목 개체, 항목 유형, 관계 개체 및 관계 유형이 채워집니다. 한 관계 유형의 선택을 취소하면 해당 관계 유형만 트리에서 제거되고 항목 유형은 제거되지 않습니다.

3. 이 새 프로필에 대해 다음 정보를 지정합니다.

필드	설명
이름	해당 프로필의 고유한 설명 이름입니다.
설명	해당 프로필에 대한 설명입니다.
관계 유형	해당 프로필과 연결된 하나 이상의 관계 유형을 선택합니다.

4. **저장**을 클릭하여 새 프로필을 저장합니다.

선택한 관계 유형에 대한 정보가 계층 도구 화면의 참조 섹션에 표시됩니다. 항목 유형도 표시됩니다. 이 정보는 선택한 관계 유형에서 파생됩니다.

프로필 편집

프로필을 편집하려면

1. 쓰기 잠금을 획득합니다.

2. 계층 도구의 탐색 트리에서 편집할 프로필을 클릭합니다.

3. 필요에 따라 프로필을 구성합니다(적절한 프로필 이름, 설명 및 관계 유형을 지정하고 패키지 할당).

4. **저장**을 클릭하여 변경 내용을 저장합니다.

프로필 유효성 검사

프로필의 유효성을 검사하려면

1. 쓰기 잠금을 획득합니다.

2. 계층 도구의 탐색 창에서 유효성을 검사할 프로필을 선택합니다.

3. 속성 창에서 유효성 검사 탭을 클릭합니다.

참고: 항목 유형 및 관계 유형에 패키지가 할당된 후에만 프로필의 유효성을 검사할 수 있습니다.

계층 도구에 유효성 검사 탭이 표시됩니다.

4. 사용할 샌드박스를 선택합니다.

샌드박스를 생성하고 구성하는 방법에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

5. 데이터의 유효성을 검사하려면 **데이터 유효성 검사**를 선택합니다. 레코드가 많은 경우에는 이 작업에 오랜 시간이 걸릴 수 있습니다.

6. 유효성 검사 프로세스를 시작하려면 **HM 구성 유효성 검사**를 클릭합니다.

유효성 검사 프로세스가 진행되는 동안 계층 도구에는 진행률 창이 표시됩니다. 유효성 검사 결과가 단추 아래의 창에 표시됩니다.

- 유효성 검사가 완료되면 **저장**을 클릭합니다.
- 유효성 검사 보고서를 저장할 디렉터리를 선택합니다.
- 지우기**를 클릭하여 유효성 검사 결과에 대한 설명이 포함된 상자를 지웁니다.

프로필 복사

프로필을 복사하려면

- 쓰기 잠금을 획득합니다.
- 계층 도구에서 복사할 프로필을 마우스 오른쪽 단추로 클릭하고 **프로필 복사**를 선택합니다.
계층 도구의 탐색 트리에서 프로필 노드 아래에 새 프로필(이름: 새 프로필)이 표시됩니다. 이 새 프로필은 복사하려고 선택한 프로필과 정확히 동일한 다른 이름의 복사본입니다. 속성 창에는 기본 속성이 표시됩니다.
- 필요한 경우 프로필을 구성합니다. 즉, 적절한 프로필 이름, 설명, 관계 유형 및 할당 패키지를 지정합니다.
- 저장**을 클릭하여 새 프로필을 저장합니다.

프로필 삭제

프로필을 삭제하려면

- 쓰기 잠금을 획득합니다.
- 계층 도구에서 삭제할 프로필을 마우스 오른쪽 단추로 클릭하고 **프로필 삭제**를 선택합니다.
계층 도구에서 이 프로필을 삭제하면 패키지가 제거된다고 경고하는 창이 표시됩니다.
- 예**를 클릭합니다.
삭제하는 프로필이 제거됩니다.

프로필에서 관계 유형 삭제

관계 유형을 삭제하려면

- 쓰기 잠금을 획득합니다.
- 계층 도구에서 관계 유형을 마우스 오른쪽 단추로 클릭하고 **프로필에서 항목 유형/관계 유형 삭제**를 선택합니다.
프로필에 삭제하려는 항목/관계 유형을 사용하는 관계 유형이 있을 경우 먼저 프로필에서 이 관계 유형을 삭제해야 해당 항목/관계 유형을 삭제할 수 있습니다.

프로필에서 항목 유형 삭제

항목 유형을 삭제하려면

- 쓰기 잠금을 획득합니다.
- 계층 도구에서 항목 유형을 마우스 오른쪽 단추로 클릭하고 **프로필에서 항목 유형/관계 유형 삭제**를 선택합니다.
프로필에 삭제하려는 항목 유형을 사용하는 관계 유형이 있을 경우 먼저 프로필에서 이 관계 유형을 삭제해야 해당 항목 유형을 삭제할 수 있습니다.

항목 및 관계 유형에 패키지 할당

프로필을 생성한 후에는 다음을 수행해야 합니다.

- 프로필과 연결된 항목 유형 및 관계 유형에 패키지를 할당합니다.
- 패키지를 보안 리소스로 구성합니다.

제 14 장

계층 관리자 자습서

이 장에 포함된 항목:

- [계층 구성 개요, 219](#)
- [자습서 예제 정보, 220](#)
- [1단계. 제품 항목 기본 개체 작성, 221](#)
- [2단계. 항목 유형 작성, 222](#)
- [3단계. 제품 관계 개체 작성, 225](#)
- [4단계. 관계 유형 작성, 226](#)
- [5단계. 계층 유형 작성, 229](#)
- [6단계. 계층 프로필에 관계 유형 추가, 229](#)
- [7단계. 패키지 작성, 230](#)
- [8단계. 패키지 할당, 231](#)
- [9단계. 계층 관리자의 데이터 표시 구성, 235](#)
- [계층 관리, 252](#)

계층 구성 개요

MDM 계층에는 MDM Hub에 있는 레코드 간의 관계가 표시됩니다. 관계는 동일한 항목 기반 개체의 레코드 간에 지정되거나 다른 항목 기반 개체의 레코드 간에 지정될 수 있습니다. 계층을 데이터로 채우기 전에 먼저 계층을 구성해야 합니다.

계층을 구성하기 전에 먼저 데이터를 잘 파악해야 하고 설정할 관계를 결정해야 합니다. 이 자습서에는 다른 제품 그룹으로 분류하려는 제품 레코드가 있습니다. 이 자습서에는 제품 계층을 구성하는 단계가 설명됩니다.

제품 계층을 구성하려면 다음 단계를 수행합니다.

1. 제품 항목 기본 개체를 작성합니다. 제품 항목 기본 개체에서 제품 그룹 항목 유형 및 제품 항목 유형에 대한 데이터를 저장합니다.
2. 항목 유형을 작성합니다.
 - a. 제품 항목 유형을 작성합니다. 각 제품에 대한 데이터를 포함한 레코드는 제품 항목 유형으로 지정됩니다.
 - b. 제품 그룹 항목 유형을 작성합니다. 계층의 제품 범주는 제품 그룹 항목 유형으로 지정됩니다.
3. 제품 관계 관계 기본 개체를 작성합니다. 제품 관계 관계 기본 개체는 제품 그룹이 제품 그룹의 상위 항목 관계 유형 및 제품 그룹이 제품의 상위 항목 관계 유형에 대한 데이터를 저장합니다.

4. 관계 유형을 작성합니다.
 - a. 제품 그룹이 제품 그룹의 상위 항목 관계 유형을 작성합니다.
 - b. 제품 그룹이 제품의 상위 항목 관계 유형을 작성합니다.
5. 계층 구성을 작성합니다. 계층 구성은 항목 유형, 관계 유형 및 계층 프로필을 계층과 연결합니다.
6. 관계 유형을 기본 계층 프로필에 추가합니다. 관계 유형을 계층 프로필에 추가하면 관계 유형과 연결된 항목 유형도 계층 프로필에 추가됩니다.
7. 패키지를 작성합니다. 패키지와 관련 쿼리는 사용자가 계층 관리자에서 액세스할 수 있는 데이터를 정의합니다.
 - a. 제품 항목 개체 패키지를 작성합니다.
 - b. 제품 관계 관계 개체 패키지를 작성합니다.
8. 패키지를 할당합니다.
 - a. 제품 항목 유형에 **PKG** 제품 패키지를 할당합니다.
 - b. 제품 그룹 항목 유형에 **PKG** 제품 패키지를 할당합니다.
 - c. 제품 그룹이 제품 그룹의 상위 항목 관계 유형에 **PKG** 제품 관계 패키지를 할당합니다.
 - d. 제품 그룹이 제품의 상위 항목 관계 유형에 **PKG** 제품 관계 패키지를 할당합니다.
9. 계층 관리자의 데이터 표시를 구성합니다.
 - a. 각 항목 유형에 대해 항목 개체 레이블을 구성합니다.
 - b. 각 항목 유형에 대해 항목 개체 도구 설명 텍스트를 구성합니다.
 - c. 각 관계 유형에 대해 관계 개체 도구 설명 텍스트를 구성합니다.
 - d. 각 항목 유형에 대해 항목 목록을 구성합니다.
 - e. 각 관계 유형에 대해 관계 목록을 구성합니다.
 - f. 각 항목 유형에 대해 항목 검색 필드를 구성합니다.
 - g. 각 관계 유형에 대해 관계 검색 필드를 구성합니다.
 - h. 각 항목 유형에 대해 항목 검색 결과를 구성합니다.
 - i. 각 항목 유형에 대해 관계 검색 결과를 구성합니다.
 - j. 각 항목 유형에 대해 항목 세부 정보를 구성합니다.
 - k. 각 관계 유형에 대해 관계 세부 정보를 구성합니다.
 - l. 편집 가능한 항목 필드를 구성합니다.
 - m. 각 항목 유형에 대해 항목 작성 필드를 구성합니다.
 - n. 각 관계 유형에 대해 관계 작성 필드를 구성합니다.

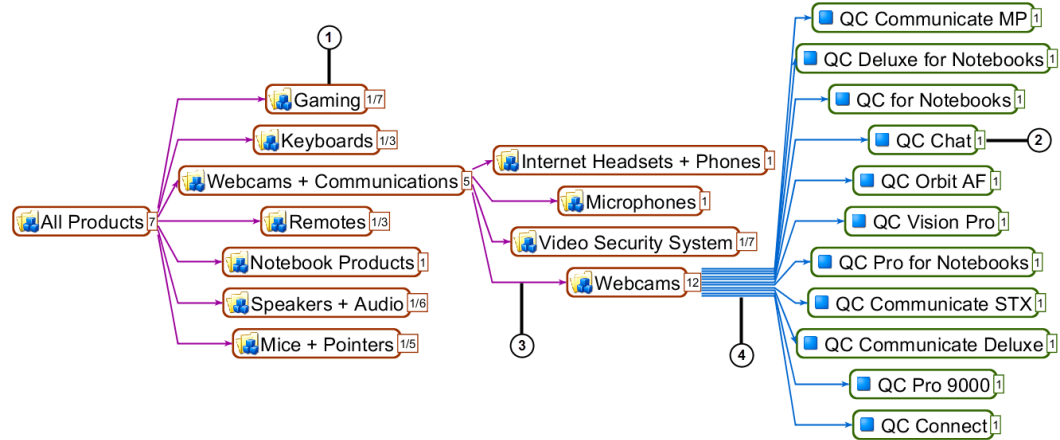
자습서 예제 정보

이 자습서에는 제품 계층을 작성하는 단계가 설명됩니다. 계층은 제품 그룹을 구성하는 제품으로 구성되어 있습니다.

자습서에서 구성할 계층은 사전 구성되어 있고 리소스 키트에 제공되는 예제 연산 참조 저장소의 데이터로 채워져 있습니다. 완료된 계층을 참조하려면 예제 연산 참조 저장소를 가져오십시오. 예제 연산 참조 저장소에 대한 자세한 내용은 *Multidomain MDM 샘플 ORS 가이드*를 참조하십시오.

자습서의 계층에는 두 개의 항목 유형과 두 개의 관계 유형이 있습니다. 항목 유형은 제품 그룹과 제품입니다. 두 개의 관계 유형은 제품 그룹이 제품 그룹의 상위 항목 및 제품 그룹이 제품의 상위 항목입니다. 항목 유형에 대한 데이터는 제품 항목 기본 개체에 저장됩니다. 관계 유형에 대한 데이터는 제품 관계 관계 기본 개체에 저장됩니다.

다음 이미지는 Hub 콘솔 계층 관리자의 계층 보기에 표시되는 제품 계층의 일부를 보여 줍니다.



1. 제품 그룹 항목
2. 제품 항목
3. 제품 그룹이 제품 그룹의 상위 항목 관계
4. 제품 그룹이 제품의 상위 항목 관계

외래 키 관계

외래 키 관계는 외래 키 필드를 사용하여 두 개의 항목 기본 개체 간 관계를 유지합니다.

이 자습서에서는 계층의 관계를 제품 관계 관계 기본 개체에서 유지 관리하므로 외래 키 필드를 기반으로 하는 관계는 필요하지 않습니다.

1단계. 제품 항목 기본 개체 작성

제품 및 제품 그룹 항목 유형의 레코드를 저장할 제품 항목 기본 개체를 작성해야 합니다. 항목 기본 개체로 변환할 수 있고 상태 관리가 활성화된 빈 제품 기본 개체가 있다고 가정합니다. 기본 개체를 항목 기본 개체로 변환하면 Hub 콘솔에서 필요한 계층 열을 기본 개체에 추가합니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. **계층 > BO를 항목/관계 개체로 변환**을 선택합니다.
4. 기존 기본 개체 수정 대화 상자에 있는 목록에서 제품 기본 개체를 선택합니다. **항목 기본 개체로 변환**을 선택한 후 **확인**을 클릭합니다.
5. **확인**을 클릭합니다.

6. 항목 유형 필드 매개 변수에 대해 다음 외래 키 열을 선택합니다.

항목 유형에 대한 외래 키 열

기본 개체 클래스 코드를 선택하여 선택한 값을 기반으로 관계를 설정합니다.

사용할 기존 기본 개체 열

새 열 작성을 선택합니다.

표시 이름

외래 키 열의 표시 이름입니다. Product Type을 입력합니다.

실제 이름

외래 키 열의 이름입니다. PRODUCT_TYPE을 입력합니다.

2단계. 항목 유형 작성

제품 항목 유형 및 제품 그룹 항목 유형을 작성합니다. 제품 항목은 제품 데이터를 포함한 제품 레코드를 의미합니다. 제품 그룹 항목은 제품을 그룹으로 분류할 때 사용합니다. 예를 들어, Webcams 제품 그룹에는 모든 웹캠 제품 레코드가 포함됩니다. 항목 유형을 작성할 때 외부 키 및 항목 유형이 계층 관리자에 표시되는 방식을 구성할 수 있습니다.

다음 항목 유형의 매개 변수를 구성할 수 있습니다.

코드

항목 유형의 고유 코드 이름입니다.

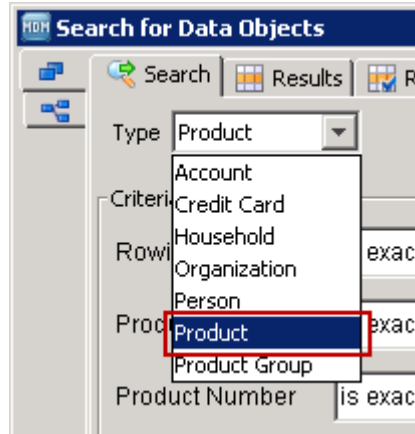
다음 이미지는 웹캠 제품 레코드의 셀에 표시될 제품 외래 키 코드를 보여 줍니다.

3. Cell data:		
Column Name	...	Base Object Cell
Rowid Object		118
Name		QC Deluxe for Notebooks
Number		960-000043
Description		Stylish design with glass-element lens performance to match.
Product Type Cd		Webcam
Product Type		Product
Hub State Indicator		Active

표시 이름

계층 도구에 표시되는 이 항목 유형의 이름입니다.

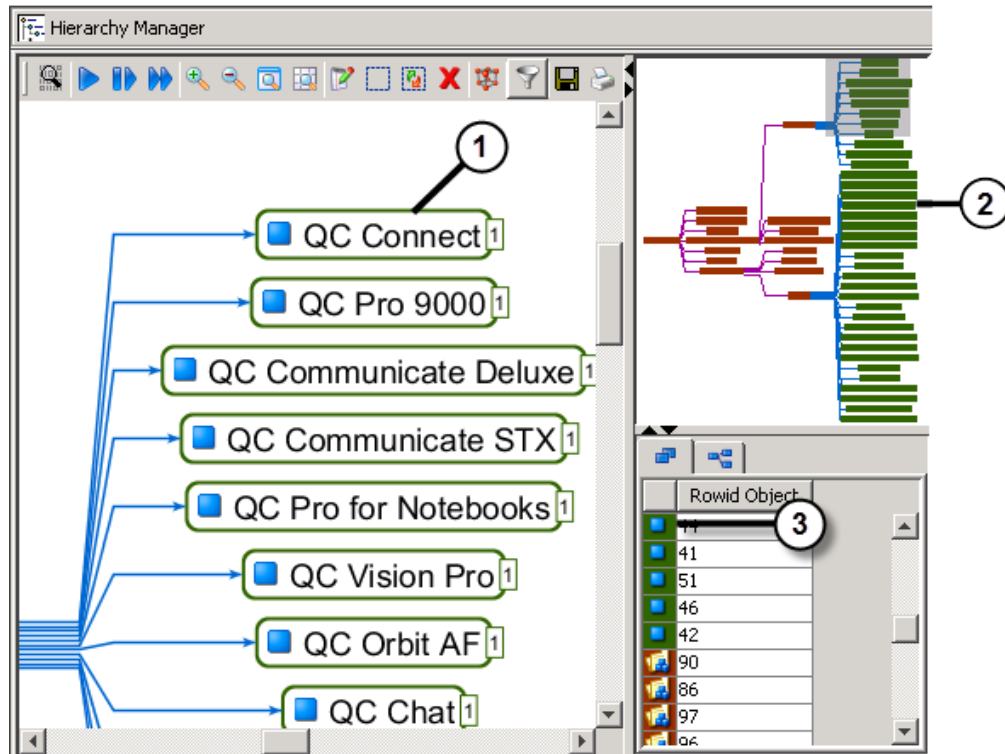
다음 이미지는 계층 관리자 검색 창에 표시될 제품 표시 이름을 보여 줍니다.



컬러

계층 관리자 도구에 나타나는 해당 항목 유형과 연결된 항목 컬러입니다.

다음 이미지는 계층 관리자 도구에 영향을 미치는 항목 유형 컬러를 보여 줍니다.



1. 계층 보기의 항목 아웃라인 컬러입니다.
2. 계층 개요에 있는 항목 컬러입니다.
3. 항목 테이블에 있는 항목의 백그라운드 컬러입니다.

작은 아이콘

계층 관리자 테이블에 항목 유형과 연결된 항목을 표시하는 데 사용하는 아이콘입니다.

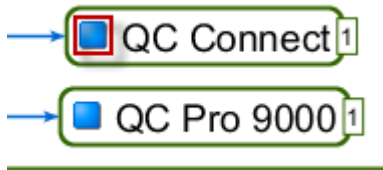
다음 이미지는 계층 관리자 테이블에 나타나는 작은 제품 항목 유형 아이콘을 보여 줍니다.

Rowid	Object
44	
41	
51	
46	
42	
90	
86	
97	
96	

큰 아이콘

계층 관리자의 계층 보기 창에 해당 항목 유형과 연결된 항목을 표시하는 데 사용하는 아이콘입니다.

다음 이미지는 계층 보기에 제품 항목 유형을 표시하는 데 사용하는 큰 아이콘을 보여 줍니다.



제품 항목 유형을 작성합니다.

제품 데이터를 포함한 제품 레코드는 제품 항목 유형입니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 **항목 개체** 노드 아래에서 제품 항목 기본 개체를 선택합니다.
4. **계층 > 항목 유형 추가**를 선택합니다.
5. 다음 항목 유형 속성을 입력합니다.

코드

Product을 입력합니다.

표시 이름

Product을 입력합니다.

컬러

컬러 선택 대화 상자에서 **RGB** 탭을 선택한 후 컬러 코드 336600을 입력합니다.

작은 아이콘

작고 파란 네모 이미지를 선택합니다.

큰 아이콘

크고 파란 네모 이미지를 선택합니다.

6. **저장**을 클릭합니다.

제품 그룹 항목 유형 작성

제품 그룹 항목은 제품을 그룹으로 분류할 때 사용합니다. 예를 들어, Webcams 제품 그룹에는 모든 웹캠 제품 레코드가 포함됩니다.

1. 모델 작업 영역에서 **계층**을 클릭합니다.
2. 계층 도구의 탐색 창에 있는 **항목 개체** 노드 아래에서 제품 항목 기본 개체를 선택합니다.
3. **계층 > 항목 유형 추가**를 선택합니다.
4. 다음 항목 유형 속성을 입력합니다.

코드

Product Group을 입력합니다.

표시 이름

Product Group을 입력합니다.

컬러

컬러 선택 대화 상자에서 **RGB** 탭을 선택한 후 컬러 코드 993300을 입력합니다.

작은 아이콘

작고 파란 네모 그룹 이미지를 선택합니다.

큰 아이콘

크고 파란 네모 그룹 이미지를 선택합니다.

5. **저장**을 클릭합니다.

3단계. 제품 관계 개체 작성

제품 관계 개체는 제품 항목 개체와 연결된 계층 관계를 유지 관리합니다. 기존 기본 개체를 변환하거나 제품 관계 개체를 작성하여 제품 관계 개체를 작성할 수 있습니다. 이 자습서에서는 제품 관계 개체를 작성합니다.

1. 모델 작업 영역에서 **계층**을 클릭합니다.
2. **계층 > 새 항목/관계 개체 작성**을 선택합니다.
3. **새 관계 기본 개체 작성**을 선택합니다. **확인**을 클릭합니다.
4. 관계 개체에 대해 다음 매개 변수를 입력합니다.

매개 변수	매개 변수 값
표시 이름	제품 관계
실제 이름	C_PRODUCT_REL
데이터 테이블스페이스	CMX_DATA
인덱스 테이블스페이스	CMX_INDX
설명	선택 사항입니다.

매개 변수	매개 변수 값
보안 리소스	활성화
항목 기본 개체 1	제품
표시 이름	제품 ID1
실제 이름	PRODUCT_ID1
항목 기본 개체 2	제품
표시 이름	제품 ID2
실제 이름	PRODUCT_ID2
계층 FK 열	행 ID 개체
계층 FK 표시 이름	행 ID 계층
계층 FK 실제 이름	ROWID_HIERARCHY
관계 유형 FK 열	행 ID 개체
관계 유형 FK 표시 이름	행 ID 관계 유형
관계 유형 FK 실제 이름	ROWID_REL_TYPE

5. **확인**을 클릭합니다.

4단계. 관계 유형 작성

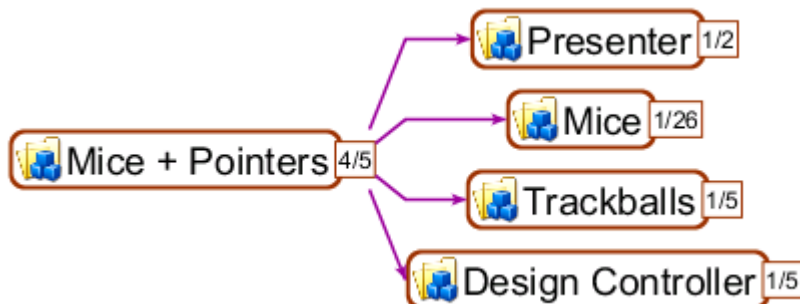
관계 유형은 항목 유형 간의 관계를 정의하고 계층을 볼 때 관계의 모양도 정의합니다.

다음 관계 유형을 작성합니다.

제품 그룹이 제품 그룹의 상위 항목

제품 그룹 항목이 다른 제품 그룹 항목의 상위 항목인 관계 유형입니다. 예를 들어, Mice + Pointers 제품 그룹은 Presenter, Mice, Trackballs 및 Design Controller 제품 그룹의 상위 항목입니다.

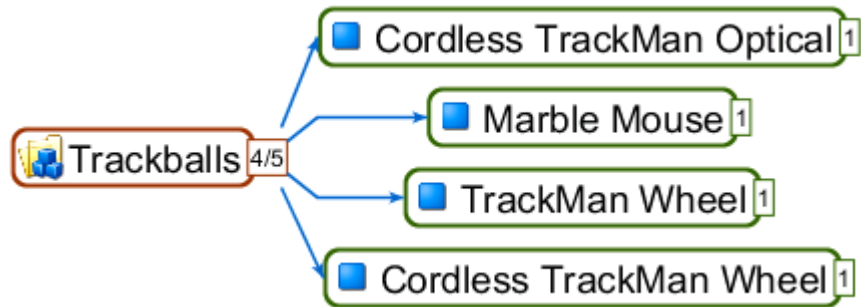
다음 이미지는 제품 그룹이 다른 제품 그룹의 상위 항목인 관계를 보여 줍니다.



제품 그룹이 제품 그룹의 상위 항목

제품 그룹 항목이 제품 항목의 상위 항목인 관계 유형입니다. 예를 들어, Trackballs 제품 그룹은 Cordless TrackMan Optical, Marble Mouse, TrackMan Wheel 및 Cordless TrackMan Wheel 제품의 상위 항목입니다.

다음 이미지는 제품 그룹이 제품의 상위 항목인 관계를 보여 줍니다.



다음 관계 유형의 매개 변수를 구성할 수 있습니다.

코드

계층 구성 관계에 대한 코드입니다.

표시 이름

계층 도구에 표시되는 관계 이름입니다.

설명

선택 사항입니다.

컬러

두 항목 간의 관계 화살표에 표시된 컬러입니다.

항목 유형 1

관계에 있는 두 개의 항목 유형 중 첫 번째 항목 유형입니다.

항목 유형 2

관계에 있는 나머지 다른 항목 유형입니다.

방향

관계 방향입니다. 예를 들어 **항목 1**에서 **항목 2**를 선택하면 항목 1은 항목 2의 상위 항목이 됩니다.

계층

관계가 속한 계층입니다.

관계 유형을 작성할 때 관계, 계층 관리자에 관계 유형이 표시되는 방식, 관계가 속한 계층을 구성합니다.

제품 그룹이 제품 그룹의 상위 항목 관계 유형 작성

제품 그룹이 제품 그룹의 상위 항목 관계 유형은 제품 그룹 항목 간에 관계를 유지 관리합니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 **관계 개체** 아래에서 제품 관계 관계 개체를 선택합니다. **계층 > 관계 추가**를 선택합니다.
4. 다음 관계 유형 매개 변수를 입력합니다.

코드

Product Group is parent of Product Group을 입력합니다.

표시 이름

Product Group is parent of Product Group을 입력합니다.

설명

선택 사항입니다.

컬러

계층 관리자의 관계를 나타내는 화살표 컬러입니다. **컬러 선택** 대화 상자에서 **RGB** 탭을 선택한 후 컬러 코드 0066CC를 입력합니다.

항목 유형 1

제품 그룹이 제품 그룹의 상위 항목을 선택합니다.

항목 유형 2

제품 그룹이 제품 그룹의 상위 항목을 선택합니다.

방향

항목 1에서 항목 2를 선택합니다.

계층

제품 계층을 선택합니다.

5. **저장**을 클릭합니다.

제품 그룹이 제품의 상위 항목 관계 유형 작성

제품 그룹이 제품의 상위 항목 관계 유형은 제품 그룹 항목과 제품 항목 간에 관계를 유지 관리합니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 **관계 개체** 아래에서 제품 관계 관계 개체를 선택합니다. **계층 > 관계 추가**를 선택합니다.
4. 다음 관계 유형 매개 변수를 입력합니다.

코드

Product Group is parent of Product을 입력합니다.

표시 이름

Product Group is parent of Product을 입력합니다.

설명

선택 사항입니다.

컬러

계층 관리자의 관계를 나타내는 화살표 컬러입니다. **컬러 선택** 대화 상자에서 **RGB** 탭을 선택한 후 컬러 코드 990099를 입력합니다.

항목 유형 1

제품 그룹을 선택합니다.

항목 유형 2

제품을 선택합니다.

방향

항목 1에서 항목 2를 선택합니다.

계층

제품 계층을 선택합니다.

5. **저장**을 클릭합니다.

5단계. 계층 유형 작성

항목 유형, 관계 유형 및 프로필을 계층과 연결할 계층 유형을 작성합니다. 관계 유형을 작성하기 전에 먼저 계층 유형을 작성해야 합니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. **계층 > 계층 추가**를 선택합니다.
4. 다음 계층 속성을 입력합니다.

코드

계층의 고유 코드 이름입니다. **Product**을 입력합니다.

표시 이름

Hub 콘솔에 표시될 계층 이름입니다. **Product**을 입력합니다.

설명

선택 사항입니다.

5. **저장**을 클릭합니다.

6단계. 계층 프로필에 관계 유형 추가

관계 유형을 계층 프로필에 추가하면 해당 관계 유형과 관련된 항목 유형도 계층 프로필에 추가됩니다. 이 자습서에서는 관계 유형을 기본 프로필에 추가합니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.

3. 계층 도구의 탐색 창에서 기본 프로필을 선택합니다.



4. 프로필 창의 일반 탭에 있는 관계 유형 섹션에서 **제품 그룹이 제품 그룹의 상위 항목** 및 **제품 그룹이 제품의 상위 항목** 관계 유형을 활성화합니다.
5. **저장**을 클릭합니다.

7단계. 패키지 작성

항목 개체 및 관계에 대한 패키지를 작성합니다. 패키지는 사용자가 계층 관리자에서 액세스할 수 있는 데이터를 정의합니다.

계층 도구를 사용하여 계층 패키지 및 관련 쿼리를 작성할 수 있습니다. 패키지를 작성할 때 액세스를 구성할 수 있는 열을 선택합니다. 열이 패키지의 일부가 아닌 경우 계층 관리자에서 열을 사용 가능하게 구성할 수 없습니다.

계층 패키지를 작성하면 해당 패키지가 패키지 도구에 표시되고 쿼리가 쿼리 도구에 표시됩니다. 패키지 도구나 쿼리 도구를 사용하여 계층 패키지나 쿼리를 작성할 수 없습니다.

패키지 및 쿼리에 대한 자세한 내용은 *Multidomain MDM 구성 가이드*를 참조하십시오.

제품 항목 개체 패키지 작성

PKG 제품 패키지를 작성하면 해당 패키지를 제품 항목 개체에 할당할 수 있습니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. **계층 > 새 패키지 작성**을 선택합니다.
4. 새 패키지 작성 대화 상자의 1단계에서 **항목 개체에 대한 패키지 작성**을 선택합니다.
5. 쿼리 그룹 섹션에서 쿼리 도구에 패키지를 구성할 쿼리 그룹을 선택하거나 작성합니다. **다음**을 클릭합니다.
6. 새 패키지 작성 대화 상자의 2단계에서 **쿼리 이름** 필드에 **Product**를 입력합니다. **기본 테이블 선택** 필드에서 제품 항목 기본 개체를 선택합니다. **다음**을 클릭합니다.
7. 새 패키지 작성 대화 상자의 3단계에서 PKG **Product**를 입력합니다. **논리 활성화** 확인란을 선택하면 계층 관리자 도구의 항목 레코드 필드의 값을 변경할 수 있습니다. **다음**을 클릭합니다.
8. 새 패키지 작성 대화 상자의 4단계에서 쿼리에 사용할 다음 열을 선택합니다.
 - Rowid 개체
 - 제품 이름

- 제품 번호
- 제품 설명
- 제품 유형 코드
- 제품 유형
- Hub 상태 표시기

참고: Rowid 개체, Hub 상태 표시기 또는 제품 유형을 비활성화할 수 없습니다.

9. **마침**을 클릭합니다.

제품 관계 패키지 작성

PKG 제품 관계 패키지를 작성하면 해당 패키지를 제품 관계 관계 개체에 할당할 수 있습니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. **계층 > 새 패키지 작성**을 선택합니다.
4. 새 패키지 작성 대화 상자의 1단계에서 **관계 개체에 대한 패키지 작성**을 선택합니다.
5. 쿼리 그룹 섹션에서 쿼리 도구에 패키지를 구성할 쿼리 그룹(예: RL 쿼리)을 선택하거나 작성합니다. **다음**을 클릭합니다.
6. 새 패키지 작성 대화 상자의 2단계에서 **쿼리 이름** 필드에 RL Product을 입력합니다. **기본 테이블 선택** 필드에서 제품 관계 항목 기본 개체를 선택합니다. **다음**을 클릭합니다.
7. 새 패키지 작성 대화 상자의 3단계에서 PKG Product Rel를 입력합니다. **논리 활성화** 확인란을 선택합니다. **다음**을 클릭합니다.
8. 새 패키지 작성 대화 상자의 4단계에서 쿼리에 사용할 다음 열을 선택합니다.
 - Rowid 개체
 - Rowid 계층
 - Rowid 관계 유형
 - 제품 ID1
 - 제품 ID2
 - 계층 수준
 - Hub 상태 표시기

참고: Rowid 개체, Rowid 계층, Rowid 관계 유형, 제품 ID1, 제품 ID2 또는 Hub 상태 표시기는 비활성화할 수 없습니다.

9. **마침**을 클릭합니다.

8단계. 패키지 할당

패키지를 항목 유형이나 관계 유형에 할당하면 항목 유형이나 관계 유형에 대한 데이터 표시를 구성할 수 있습니다.

다음 유형의 패키지를 할당할 수 있습니다.

표시 패키지

데이터 읽기 액세스를 위한 패키지입니다. 필수 사항입니다.

놓기 패키지

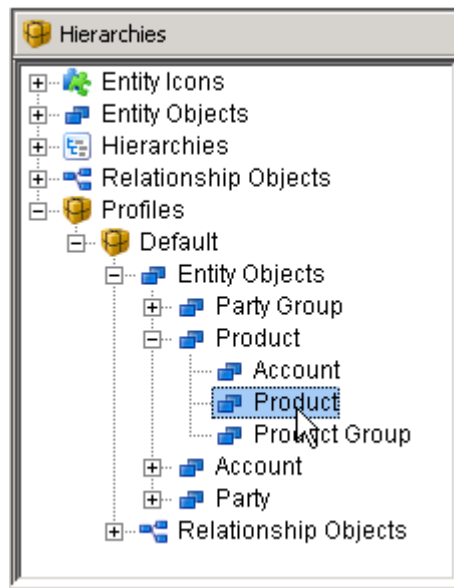
데이터 쓰기 액세스를 위한 패키지입니다. 선택 사항입니다.

놓기 패키지를 항목이나 관계 유형에 할당하지 않으면 해당 유형을 작성하거나 편집할 수 없습니다. 예를 들어 놓기 패키지를 제품 항목 유형에 할당하지 않으면 새 제품 레코드를 작성하거나 계층 관리자의 기존 제품 레코드를 편집할 수 없습니다. 놓기가 가능한 놓기 패키지만 할당할 수 있습니다.

제품 항목 유형에 PKG 제품 패키지 할당

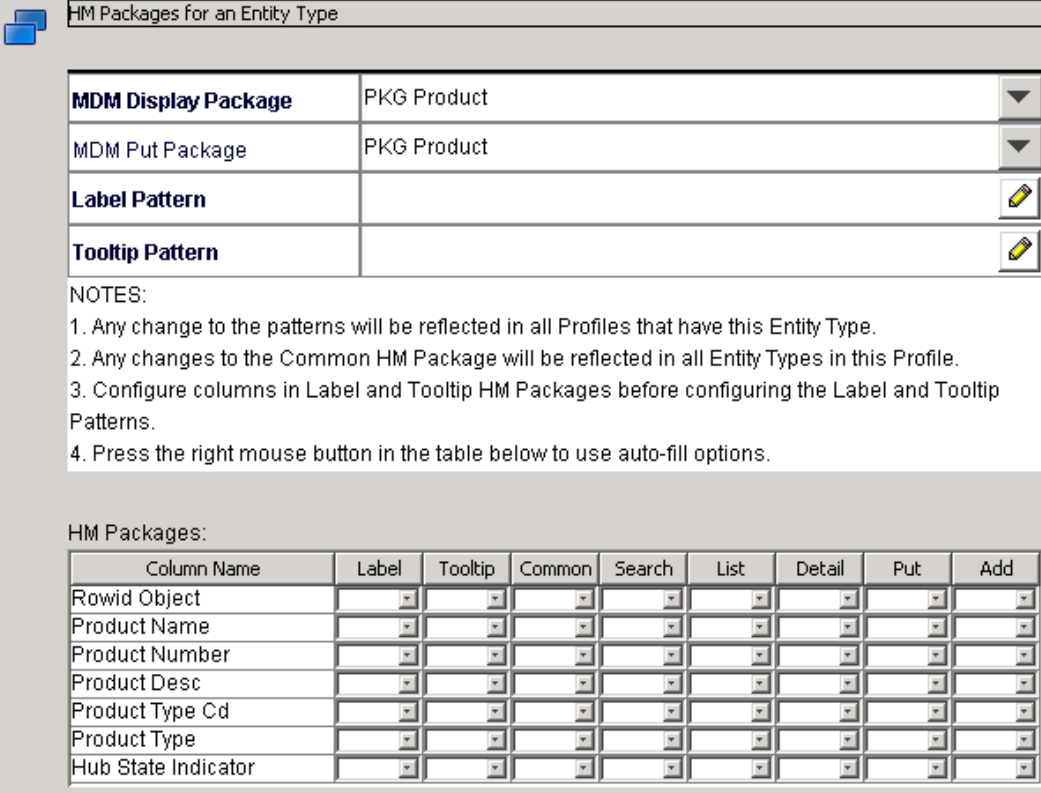
제품 항목 유형에 PKG 제품 패키지를 할당하면 계층 관리자에 제품 항목 데이터가 표시되는 방식을 구성할 수 있습니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 항목 유형 노드를 선택합니다.



4. **MDM 표시 패키지** 목록에서 **PKG 제품** 패키지를 선택합니다.
5. **MDM 놓기 패키지** 목록에서 **PKG 제품** 패키지를 선택합니다.

이제 **HM 패키지** 섹션의 제품 항목 유형에 대해 데이터 표시를 구성할 수 있습니다.



The screenshot shows a software window titled "HM Packages for an Entity Type". It contains a table with four rows: "MDM Display Package" and "MDM Put Package" both set to "PKG Product"; "Label Pattern" and "Tooltip Pattern" are empty with edit icons. Below this is a "NOTES" section with four instructions. At the bottom is a table titled "HM Packages:" with columns for "Column Name", "Label", "Tooltip", "Common", "Search", "List", "Detail", "Put", and "Add". The rows list various attributes like "Rowid Object", "Product Name", "Product Number", etc., each with a dropdown menu for configuration.

제품 그룹 항목 유형에 PKG 제품 패키지 할당

제품 그룹 항목 유형에 **PKG** 제품 패키지를 할당하면 계층 관리자에 제품 그룹 항목 데이터가 표시되는 방식을 구성할 수 있습니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 그룹 항목 유형 노드를 선택합니다.
4. **MDM 표시 패키지** 목록에서 **PKG 제품** 패키지를 선택합니다.
5. **MDM 놓기 패키지** 목록에서 **PKG 제품** 패키지를 선택합니다.

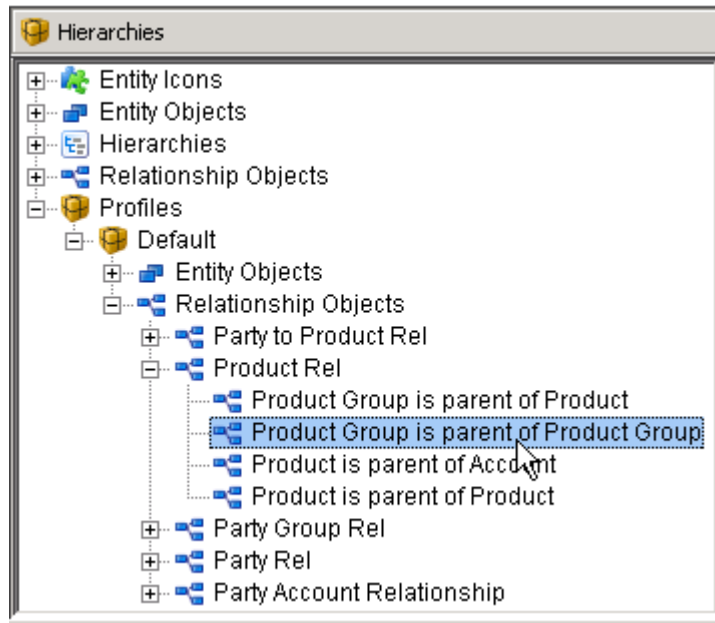
이제 **HM 패키지** 섹션의 제품 그룹 항목 유형에 대해 데이터 표시를 구성할 수 있습니다.

제품 그룹이 제품 그룹의 상위 항목 관계 유형에 PKG 제품 관계 패키지 할당

제품 그룹이 제품 그룹의 상위 항목 관계 유형에 **PKG** 제품 관계 패키지를 할당하면 관계 데이터가 계층 관리자에 표시되는 방식을 구성할 수 있습니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.

- 계층 도구의 탐색 창에 있는 기본 계층 프로파일 아래에서 제품 그룹이 제품 그룹의 상위 항목 관계 유형 노드를 선택합니다.



- MDM 표시 패키지 목록에서 **PKG 제품 관계** 패키지를 선택합니다.
- MDM 놓기 패키지 목록에서 **PKG 제품 관계** 패키지를 선택합니다.

이제 **HM 패키지** 섹션에서 제품 그룹이 제품 그룹의 상위 항목 관계 유형에 대해 데이터 표시를 구성할 수 있습니다.

HM Packages for a Relationship Type

MDM Display Package	PKG Product Rel	▼
MDM Put Package	PKG Product Rel	▼
Tooltip Pattern		

NOTES:

- Any change to the pattern will be reflected in all Profiles that have this Relationship Type.
- Any change to the Common HM Package will be reflected in all Relationship Types in this Profile.
- Configure columns in Tooltip HM Package before configuring the Tooltip Pattern.
- Press the right mouse button in the table below to use auto-fill options.

HM Packages:

Column Name	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	▼	▼	▼	▼	▼	▼	▼
Rowid Hierarchy	▼	▼	▼	▼	▼	▼	▼
Rowid Rel Type	▼	▼	▼	▼	▼	▼	▼
Product ID1	▼	▼	▼	▼	▼	▼	▼
Product ID2	▼	▼	▼	▼	▼	▼	▼
Hub State Indicator	▼	▼	▼	▼	▼	▼	▼
Hierarchy Level	▼	▼	▼	▼	▼	▼	▼

제품 그룹이 제품의 상위 항목 관계에 PKG 제품 관계 패키지 할당

PKG 제품 관계 패키지를 제품 그룹이 제품의 상위 항목 관계 유형에 할당하면 계층 관리자에 관계 데이터가 표시되는 방식을 구성할 수 있습니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 그룹이 제품의 상위 항목 관계 유형 노드를 선택합니다.
4. **MDM 표시 패키지** 목록에서 **PKG 제품 관계** 패키지를 선택합니다.
5. **MDM 놓기 패키지** 목록에서 **PKG 제품 관계** 패키지를 선택합니다.

이제 **HM 패키지** 섹션의 제품 그룹이 제품의 상위 항목 관계 유형에 대해 데이터 표시를 구성할 수 있습니다.

9단계. 계층 관리자의 데이터 표시 구성

사용자가 레코드 검색, 보기, 편집 및 추가를 수행할 때 계층 관리자에 표시할 항목 필드 및 관계 필드를 구성할 수 있습니다. 또한 계층 관리자 레이블 및 도구 설명에 표시할 필드를 구성할 수 있습니다.

계층 관리자의 데이터 표시를 구성하려면 각 관계 유형 및 항목 유형에 대해 계층 관리자 패키지를 구성하십시오. 계층 프로필이 여러 개 있는 경우 프로필마다 데이터 표시를 구성해야 합니다.

다음 계층 관리자 사용자 인터페이스 요소에 대해 표시될 필드와 표시 순서를 구성할 수 있습니다.

레이블

계층 관리자 도구에 있는 항목에 대한 레이블 텍스트입니다.

도구 설명

계층 관리자 도구에서 항목 또는 관계에 커서를 잠시 올려 놓으면 표시되는 텍스트입니다.

공통

다양한 유형의 항목 및 관계가 동일한 목록에 나타날 때 표시되는 필드입니다. 선택한 필드는 모든 계층 관리자 패키지에 있어야 합니다.

검색

항목 레코드나 관계 레코드를 검색하기 위해 사용하는 필드입니다.

목록

항목 레코드나 관계 레코드에 대한 검색 결과 목록에 표시되는 필드입니다.

세부 정보

선택한 항목이나 관계의 세부 정보를 보기 위해 선택할 때 표시되는 필드입니다.

놓기

선택한 항목이나 관계를 편집할 때 변경할 수 있는 값을 가진 필드입니다.

추가

계층 관리자에서 항목 또는 관계를 추가할 때 표시되는 필드입니다.

제품 항목 유형 레이블 구성

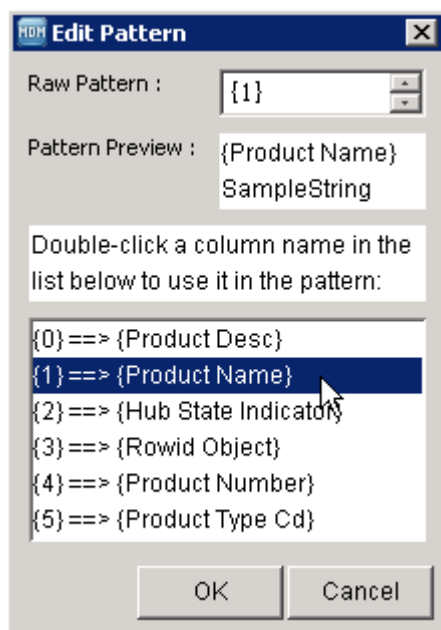
이 자습서에서는 제품 항목에 대한 레이블 텍스트로 표시될 제품 이름을 구성합니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 항목 유형 노드를 선택합니다.
4. **HM 패키지** 섹션의 **레이블** 열에서 제품 이름 행에 번호를 할당해야 합니다.

HM Packages:

Column Name	Label
Rowid Object	3
Product Name	1
Product Number	4
Product Desc	0
Product Type Cd	5
Product Type	
Hub State Indicator	2

5. 항목 유형에 대한 **HM 패키지** 섹션에서 **레이블 패턴** 필드에 대한 편집 단추를 클릭합니다.
6. **패턴 편집** 대화 상자에서 **{#} ==> {Product Name}**을 두 번 클릭합니다.



행 패턴 필드에서 열을 연결하거나 사용자 지정 텍스트를 추가할 수 있습니다. 사용자 지정 텍스트 없이 제품 이름 열을 사용합니다.

7. **저장**을 클릭합니다.

다음 이미지는 계층 관리자에 **Media Desktop Laser** 제품 이름이 표시된 제품 그룹 항목을 보여 줍니다.



제품 그룹 항목 유형에 대해 레이블 구성

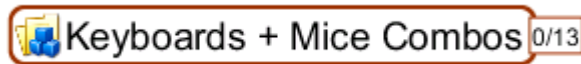
이 자습서에서는 제품 그룹 항목에 대한 레이블 텍스트로 표시될 제품 이름을 구성합니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로파일 아래에서 제품 그룹 항목 유형 노드를 선택합니다.
4. **HM 패키지** 섹션의 레이블 열에서 제품 이름 행에 번호를 할당해야 합니다.
5. **항목 유형에 대한 HM 패키지** 섹션에서 **레이블 패턴** 필드에 대한 편집 단추를 클릭합니다.
6. **패턴 편집** 대화 상자에서 **{#} ==> {Product Name}**을 두 번 클릭합니다.

행 패턴 필드에서 열을 연결하거나 사용자 지정 텍스트를 추가할 수 있습니다. 사용자 지정 텍스트 없이 제품 이름 열을 사용합니다.

7. **저장**을 클릭합니다.

다음 이미지는 계층 관리자에 **Keyboards + Mice Combos**로 제품 이름이 표시된 제품 그룹 항목을 보여 줍니다.



제품 항목 유형에 대한 도구 설명 텍스트 구성

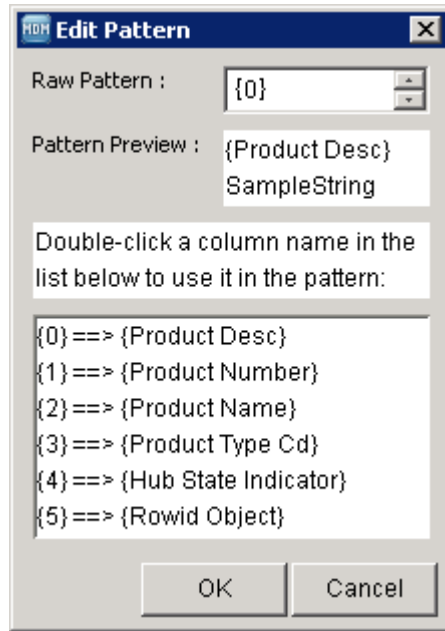
이 자습서에서는 제품 항목에 대한 도구 설명 텍스트로 표시될 제품 설명을 구성합니다. 계층 관리자에서 제품 항목에 커서를 잠시 올려 놓으면 제품 설명이 도구 설명으로 표시됩니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로파일 아래에서 제품 항목 유형 노드를 선택합니다.
4. **HM 패키지** 섹션의 **도구 설명** 열에서 제품 설명 행에 번호를 할당해야 합니다.

Column Name	Label	Tooltip
Rowid Object	3	5
Product Name	1	2
Product Number	4	1
Product Desc	0	0
Product Type Cd	5	3
Product Type		
Hub State Indicator	2	4

5. **항목 유형에 대한 HM 패키지** 섹션에서 **도구 설명 패턴** 필드에 대한 편집 단추를 클릭합니다.

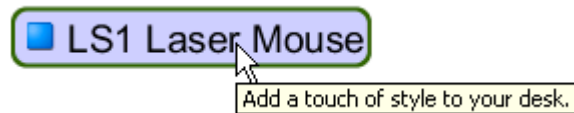
- 패턴 편집 대화 상자에서 **{#}==>{Product Desc}**을 두 번 클릭합니다.



행 패턴 필드에서 열을 연결하거나 사용자 지정 텍스트를 추가할 수 있습니다. 사용자 지정 텍스트 없이 제품 설명 열을 사용합니다.

- 저장을 클릭합니다.

다음 이미지는 도구 설명으로 표시될 제품 설명이 있는 LS1 Laser Mouse 제품 항목을 보여 줍니다.



제품 그룹 항목 유형에 대한 도구 설명 텍스트 구성

이 자습서에서는 제품 그룹 항목 유형에 대한 도구 설명 텍스트로 표시될 제품 설명을 구성합니다. 계층 관리자에서 제품 그룹 항목에 커서를 잠시 올려 놓으면 제품 설명이 도구 설명으로 표시됩니다.

- 쓰기 잠금을 획득합니다.
- 모델 작업 영역에서 **계층**을 클릭합니다.
- 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 그룹 항목 유형 노드를 선택합니다.
- HM 패키지** 섹션의 **도구 설명** 열에서 제품 설명 행에 번호를 할당해야 합니다.
- 항목 유형에 대한 HM 패키지** 섹션에서 **도구 설명 패턴** 필드에 대한 편집 단추를 클릭합니다.
- 패턴 편집 대화 상자에서 **{#}==>{Product Desc}**을 두 번 클릭합니다.

행 패턴 필드에서 열을 연결하거나 사용자 지정 텍스트를 추가할 수 있습니다. 사용자 지정 텍스트 없이 제품 설명 열을 사용합니다.

- 저장을 클릭합니다.

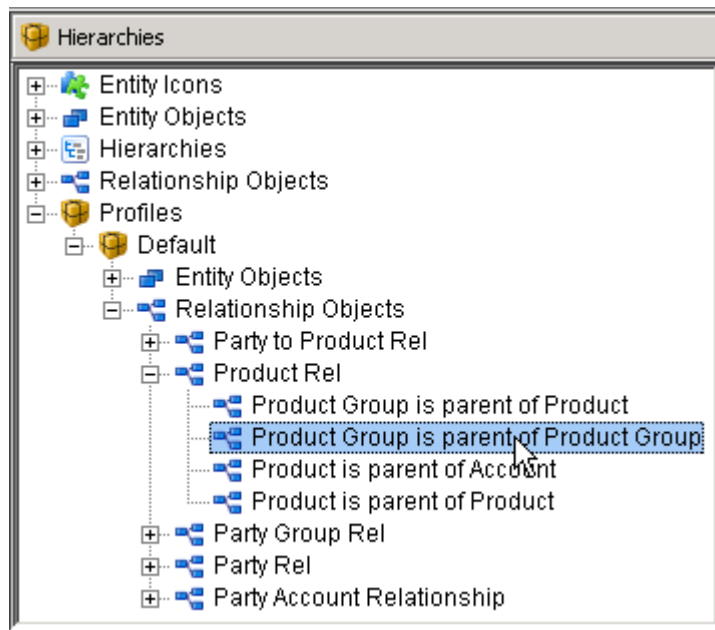
다음 이미지는 도구 설명으로 표시될 제품 설명이 있는 Webcams 제품 그룹 항목을 보여 줍니다.



제품 그룹이 제품 그룹의 상위 항목 관계 유형에 대한 도구 설명 텍스트 구성

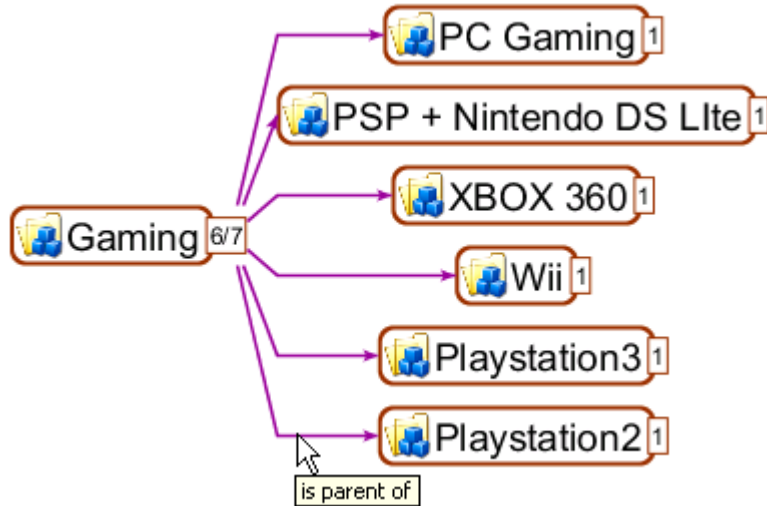
이 자습서에서 제품 그룹이 제품 그룹의 상위 항목 관계에 대한 도구 설명 텍스트로 표시될 **상위 항목** 텍스트를 구성합니다. 계층 관리자의 관계를 나타내는 화살표에 커서를 잠시 올려 놓으면 도구 설명 텍스트가 표시됩니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 그룹이 제품 그룹의 상위 항목 관계 유형 노드를 선택합니다.



4. 항목 유형에 대한 **HM 패키지** 섹션에서 **도구 설명 패턴** 필드에 대한 편집 단추를 클릭합니다.
5. **패턴 편집** 대화 상자의 **행 패턴** 필드에서 is parent of를 입력합니다.
6. **저장**을 클릭합니다.

다음 이미지는 게임 제품 그룹 및 Playstation 2 제품 그룹 간의 관계에 대한 도구 설명을 보여 줍니다.

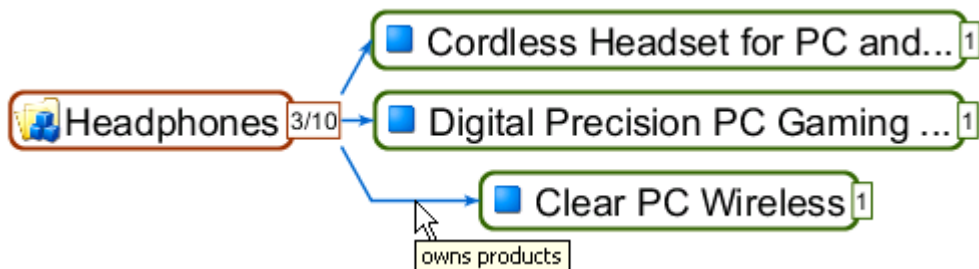


제품 그룹이 제품의 상위 항목 관계 유형에 대한 도구 설명 텍스트 구성

이 자습서에서는 제품 그룹이 제품의 상위 항목 관계에 대한 도구 설명 텍스트로 표시될 **제품 소유** 텍스트를 구성합니다. 계층 관리자의 관계를 나타내는 화살표에 커서를 잠시 올려 놓으면 도구 설명 텍스트가 표시됩니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 그룹이 제품의 상위 항목 관계 유형 노드를 선택합니다.
4. **항목 유형에 대한 HM 패키지** 섹션에서 **도구 설명 패턴** 필드에 대한 편집 단추를 클릭합니다.
5. **패턴 편집** 대화 상자의 행 패턴 필드에서 owns products를 입력합니다.
6. **저장**을 클릭합니다.

다음 이미지는 헤드폰 제품 그룹 및 PC 무선 제품 간의 관계에 대한 도구 설명을 보여 줍니다.



항목 목록 구성

이 자습서에서는 계층 관리자에 표시될 항목 목록에 나타날 행 ID 개체 필드를 구성합니다. 항목 목록에 다양한 항목 유형이 포함될 수 있습니다. 모든 항목 유형 패키지에 대한 모든 쿼리에 포함되는 필드만 선택할 수 있습니다.

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 항목 유형 노드를 선택합니다.

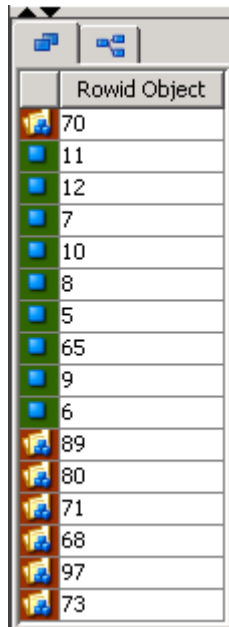
4. **HM 패키지** 섹션의 **공통** 열에서 Rowid 개체 행에 0을 할당합니다.

Column Name	Label	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	0	0	0	4	0	4		
Product Name	1	1		0	1	0	0	0
Product Number	2	2		1	4	2	1	1
Product Desc	3	3		2	3	1		2
Product Type Cd	4	4		3	2	3		3
Product Type					5	6		
Hub State Ind	5	5				5		

5. **저장**을 클릭합니다.

6. 제품 그룹 항목 유형에 대해 해당 단계를 반복해서 수행합니다.

다음 이미지는 계층 관리자에 표시되는 제품 및 제품 그룹 항목이 포함된 목록을 보여 줍니다.



관계 목록 구성

이 자습서에서는 계층 관리자에 표시될 관계 목록에 나타날 행 ID 개체 필드를 구성합니다. 관계 목록에는 다른 항목 유형의 관계가 포함될 수 있습니다. 모든 관계 유형 패키지에 대한 모든 쿼리에 포함되는 필드만 선택할 수 있습니다.

- 쓰기 잠금을 획득합니다.
- 모델 작업 영역에서 **계층**을 클릭합니다.
- 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 그룹이 제품 그룹의 상위 항목 관계 유형 노드를 선택합니다.

4. **HM 패키지** 섹션의 **공통** 열에서 Rowid 개체 행에 0을 할당합니다.

HM Packages:							
Column Name	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	U	U	U	U	U	U	U
Rowid Hierarchy			1	1	1	1	1
Rowid Rel Type							
Product ID1			2	2	2	2	2
Product ID2			3	3	3	3	3
Hub State Indicator			4	4	4	4	4
Hierarchy Level							

5. **저장**을 클릭합니다.
6. 제품 그룹이 제품의 상위 항목 관계 유형에 대해 해당 단계를 반복해서 수행하십시오.
- 다음 이미지는 계층 관리자에 표시된 관계가 포함된 목록을 보여 줍니다.

Rowid Object
24
25
30
32
29
28
105
26
31
27
103
106
102
104
112

항목 검색 필드 구성

이 자습서에서는 제품 항목을 검색할 때 **데이터 개체 검색** 대화 상자의 **검색** 탭에 다음 필드가 내림차순으로 표시되도록 구성하려고 합니다.

- 제품 이름
- 제품 번호
- 제품 설명
- 제품 유형 코드
- 행 ID 개체

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 항목 유형 노드를 선택합니다.

4. **HM 패키지** 섹션의 **검색** 열에서 검색 필드로 표시할 열에 번호를 할당합니다. 가장 낮은 값을 할당하는 열이 필드 목록의 맨 위에 표시됩니다.

Column Name	Label	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	0	0	0	4	0	0	0	0
Product Name	1	1		0	1	1	1	1
Product Number	2	2		1	4	2	2	2
Product Desc	3	3		2	3	3		3
Product Type Cd	4	4		3	2	4		4
Product Type					5			5
Hub State Ind	5	5				5		6

5. **저장**을 클릭합니다.
6. 제품 그룹 항목 유형에 대해 해당 단계를 반복해서 수행합니다.

예를 들어, 제품 유형 필드와 Hub 상태 표시기 필드에는 값이 할당되지 않았기 때문에 **데이터 개체 검색** 대화 상자에 표시되지 않습니다. 제품 이름에 가장 낮은 값을 할당했기 때문에 제품 이름 필드가 필드 목록의 맨 위에 표시됩니다. Rowid 개체에 가장 높은 값을 할당했기 때문에 Rowid 개체 필드가 목록 맨 아래에 표시됩니다.

다음 이미지는 계층 관리자에 표시되는 **검색** 탭을 보여 줍니다.

관계 검색 필드 구성

이 자습서에서는 관계를 검색할 때 **데이터 개체 검색** 대화 상자의 **검색** 탭에 다음 필드가 내림차순으로 표시되도록 구성합니다.

- Rowid 개체
- Rowid 계층
- 제품 ID1
- 제품 ID2

- Hub 상태 표시기

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 계층을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 그룹이 제품의 상위 항목 관계 유형 노드를 선택합니다.
4. **HM 패키지** 섹션의 **검색** 열에서 검색 필드로 표시할 열에 번호를 할당합니다. 가장 낮은 값을 할당하는 열이 필드 목록의 맨 위에 표시됩니다.

HM Packages:

Column Name	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	U	U	U	U	U	U	U
Rowid Hierarchy			1	1	1	1	1
Rowid Rel Type							
Product ID1			2	2	2	2	2
Product ID2			3	3	3	3	3
Hub State Indicator			4	4	4	4	4
Hierarchy Level							

5. **저장**을 클릭합니다.
 6. 제품 그룹이 제품 그룹의 상위 항목 관계 유형에 대해 해당 단계를 반복해서 수행하십시오.
- 다음 이미지는 계층 관리자에 표시되는 **검색** 탭을 보여 줍니다.

항목 검색 결과 구성

이 자습서에서는 항목을 검색할 때 **데이터 개체 검색** 대화 상자의 **결과** 탭에 다음 제품 필드가 왼쪽에서 오른쪽으로 표시되도록 구성합니다.

- 행 ID 개체

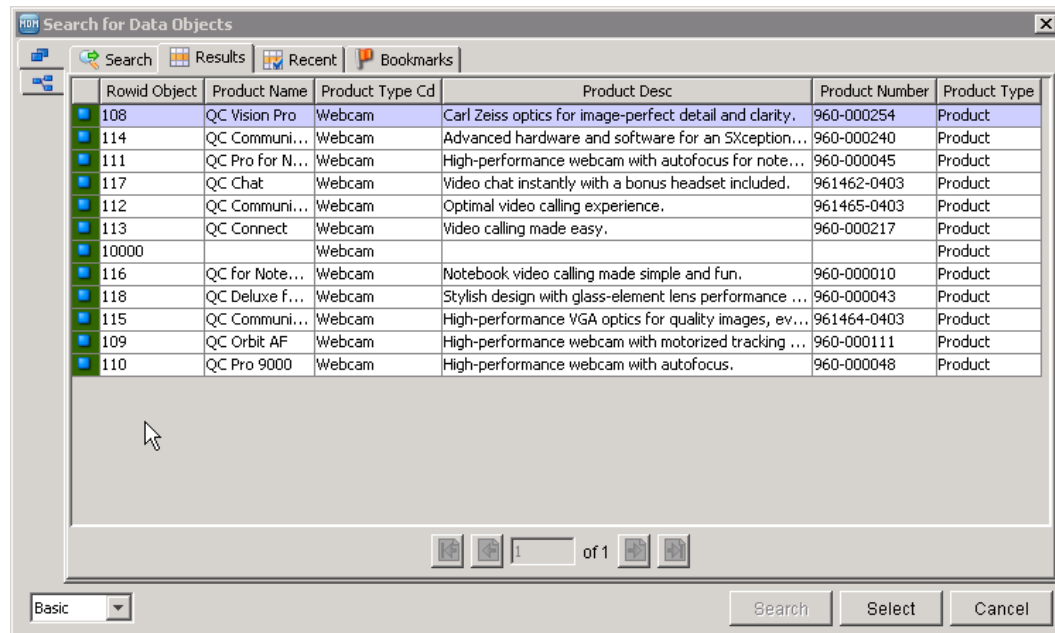
- 제품 이름
- 제품 유형 코드
- 제품 설명
- 제품 번호
- 제품 유형

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 항목 유형 노드를 선택합니다.
4. **HM 패키지** 섹션의 **목록** 열에서 검색 결과로 표시할 열에 번호를 할당합니다. 가장 낮은 값을 할당하는 열이 필드 목록의 맨 위에 표시됩니다.

Column Name	Label	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	0	0	0	4	0	0	0	0
Product Name	1	1		0	1	1	1	1
Product Number	2	2		1	4	2	2	2
Product Desc	3	3		2	3	3		3
Product Type Cd	4	4		3	2	4		4
Product Type					5			5
Hub State Ind	5	5				5		6

5. **저장**을 클릭합니다.
6. 제품 그룹 항목 유형에 대해 해당 단계를 반복해서 수행합니다.

이 예제의 경우 **Hub** 상태 표시기 필드에는 값이 할당되지 않았기 때문에 **결과** 탭에 해당 필드가 표시되지 않습니다. **Rowid** 개체에 가장 낮은 값을 할당했기 때문에 **Rowid** 개체 필드가 오른쪽에 표시됩니다. 제품 유형에 가장 높은 값을 할당했기 때문에 제품 유형 필드가 왼쪽에 표시됩니다.
다음 이미지는 이 예제의 구성 설정을 사용할 때 표시되는 **결과** 탭을 보여 줍니다.



관계 검색 결과 구성

이 자습서에서는 항목을 검색할 때 **데이터 개체 검색** 대화 상자의 **결과** 탭에 제품 그룹이 제품의 상위 항목 관계 유형에 대한 다음 필드가 왼쪽에서 오른쪽으로 표시되도록 구성합니다.

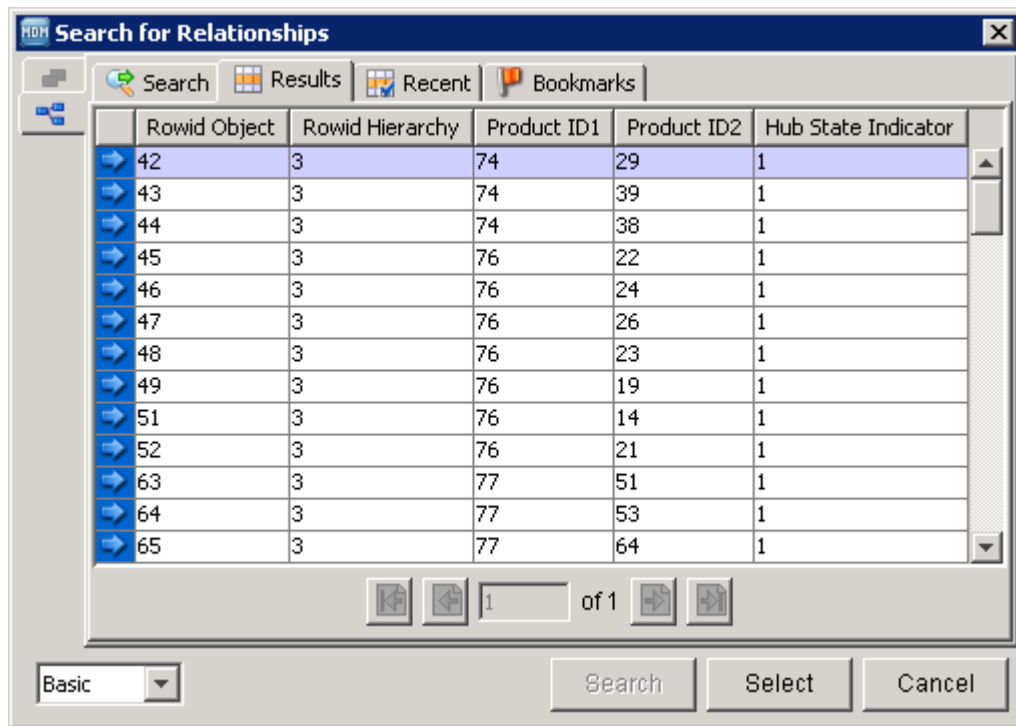
- Rowid 개체
 - Rowid 계층
 - 제품 ID1
 - 제품 ID2
 - Hub 상태 표시기
1. 쓰기 잠금을 획득합니다.
 2. 모델 작업 영역에서 **계층**을 클릭합니다.
 3. 계층 도구의 탐색 창에 있는 기본 계층 프로파일 아래에서 제품 그룹이 제품의 상위 항목 관계 유형 노드를 선택합니다.
 4. **HM 패키지** 섹션의 **목록** 열에서 검색 결과로 표시할 열에 번호를 할당합니다. 가장 낮은 값을 할당한 열이 검색 결과의 제일 왼쪽에 표시됩니다.

HM Packages:

Column Name	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	U	U	U	U	U	U	U
Rowid Hierarchy			1	1	1	1	1
Rowid Rel Type							
Product ID1			2	2	2	2	2
Product ID2			3	3	3	3	3
Hub State Indicator			4	4	4	4	4
Hierarchy Level							

5. **저장**을 클릭합니다.
6. 제품 그룹이 제품 그룹의 상위 항목 관계 유형에 대해 해당 단계를 반복해서 수행하십시오.

다음 이미지는 이 예제의 구성 설정을 사용할 때 표시되는 **결과** 탭을 보여 줍니다.



Hub 콘솔 계층 관리자의 항목 세부 정보 구성

이 자습서에서는 사용자가 Hub 콘솔 계층 관리자에서 항목 세부 정보를 볼 때 **세부 정보** 대화 상자에 다음 필드가 내림차순으로 표시되도록 구성하려고 합니다.

- 제품 이름
- 제품 설명
- 제품 번호
- 제품 유형 코드
- 행 ID 개체
- Hub 상태 표시기

1. 쓰기 잠금을 획득합니다.
2. 모델 작업 영역에서 **계층**을 클릭합니다.
3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 항목 유형 노드를 선택합니다.
4. **HM 패키지** 섹션의 **세부 정보** 열에서 세부 정보 대화 상자에 표시할 열에 번호를 할당합니다. 가장 낮은 값을 할당하는 열이 필드 목록의 맨 위에 표시됩니다.

Column Name	Label	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	0	0	0	4	0	4	0	0
Product Name	1	1		0	1	0	1	1
Product Number	2	2		1	4	2	2	2
Product Desc	3	3		2	3	1		3
Product Type Cd	4	4		3	2	3		4
Product Type					5			5
Hub State Ind	5	5				5		6

5. **저장**을 클릭합니다.
 6. 제품 그룹 항목 유형에 대해 해당 단계를 반복해서 수행합니다.
- 다음 이미지는 계층 관리자에 표시되는 **세부 정보** 대화 상자를 보여 줍니다.

Name	Value
Product Name	QC for Notebooks
Product Desc	Notebook video calling made simple and fun.
Product Number	960-000010
Product Type Cd	Webcam
Rowid Object	116
Hub State Ind	1

IDD 계층 관리자의 항목 세부 정보 구성

이 자습서에서는 IDD 계층 관리자에서 **Person** 개체의 세부 정보를 보는 사용자에게 표시할 고객 이름을 구성합니다.

1. Informatica Data Director 구성 관리자에 로그인합니다.
`http://<호스트>:<포트>/bdd/config`
2. IDD 응용 프로그램을 선택하고 **편집**을 클릭합니다.
3. 응용 프로그램 편집 화면의 제목 영역 탭에서 **제목 영역 그룹 > 고객 > 개인 > 이름**을 선택합니다.
4. **제목 영역 하위 항목 편집**을 클릭합니다.
5. 제목 영역 하위 항목 대화 상자, 레이아웃 탭에서 테이블의 이름 열 이름을 선택하고 **레이아웃 편집**을 클릭합니다.
6. **HM에 표시**를 활성화합니다. **확인**을 클릭합니다.
7. **확인**을 클릭한 다음 **저장**을 클릭합니다.

관계 세부 정보 구성

이 자습서에서는 관계 세부 정보를 볼 때 **세부 정보** 대화 상자에 다음 필드가 내림차순으로 표시되도록 구성합니다.

- Rowid 개체
 - Rowid 계층
 - 제품 ID1
 - 제품 ID2
 - Hub 상태 표시기
1. 쓰기 잠금을 획득합니다.
 2. 모델 작업 영역에서 **계층**을 클릭합니다.
 3. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 그룹이 제품의 상위 항목 관계 유형 노드를 선택합니다.

4. **HM 패키지** 섹션의 **세부 정보** 열에서 세부 정보 대화 상자에 표시할 열에 번호를 할당합니다. 가장 낮은 값을 할당하는 열이 필드 목록의 맨 위에 표시됩니다.

HM Packages:

Column Name	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	U	U	U	U	U	U	U
Rowid Hierarchy			1	1	1	1	1
Rowid Rel Type							
Product ID1			2	2	2	2	2
Product ID2			3	3	3	3	3
Hub State Indicator			4	4	4	4	4
Hierarchy Level							

5. **저장**을 클릭합니다.
6. 제품 그룹이 제품 그룹의 상위 항목 관계 유형에 대해 해당 단계를 반복해서 수행하십시오.
- 다음 이미지는 계층 관리자에 표시되는 **세부 정보** 대화 상자를 보여 줍니다.

Details

Relationship properties

Name	Value
Rowid Object	26
Rowid Hierarchy	3
Product ID1	70
Product ID2	9
Hub State Indicator	1

OK

편집 가능한 항목 필드 구성

사용자가 편집할 수 있는 항목 필드를 구성하려면 **농기** 패키지를 항목 유형에 할당해야 합니다. 이 자습서에서는 항목을 편집할 때 **항목 레코드 편집기** 대화 상자에 다음 필드가 내림차순으로 표시되도록 구성하려고 합니다.

- 제품 이름
- 제품 번호

- 쓰기 잠금을 획득합니다.
- 모델 작업 영역에서 **계층**을 클릭합니다.
- 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 항목 유형 노드를 선택합니다.
- HM 패키지** 섹션의 **농기** 열에서 편집 가능한 필드로 표시할 열에 번호를 할당합니다. 가장 낮은 값을 할당하는 열이 필드 목록의 맨 위에 표시됩니다.

Column Name	Label	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	0	0	0	4	0	4		
Product Name	1	1		0	1	0	0	0
Product Number	2	2		1	4	2	1	1
Product Desc	3	3		2	3	1		2
Product Type Cd	4	4		3	2	3		3
Product Type					5			4
Hub State Ind	5	5				5		

5. **저장**을 클릭합니다.

6. 제품 그룹 항목 유형에 대해 해당 단계를 반복해서 수행합니다.

다음 이미지는 계층 관리자에 표시되는 **항목 레코드 편집기** 대화 상자를 보여 줍니다.

The image shows a dialog box titled "Entity Record Editor". It contains the following fields and controls:

- Type:** A dropdown menu with "Product" selected.
- Product Name:** A text input field containing "QC Pro for Notebooks".
- Product Number:** A text input field containing "960-000045".
- Show only editable cells:** A checkbox that is currently unchecked.
- Buttons:** "OK", "Cancel", and "Set Default Tru..." (partially visible).

관계 필드 편집 가능성

PKG 제품 관계 관계 패키지의 관계 필드는 계층 관리자에 편집할 수 없습니다. 관계 패키지에 대해 놓기 열을 구성하지 않아도 됩니다. 편집할 열에 번호를 할당해도 관계 패키지의 관계 필드를 편집할 수 없습니다.

계층 관리자의 관계 레코드 편집기에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

항목 작성 필드 구성

이 자습서에서는 사용자가 계층 관리자에 항목을 작성할 때 **항목 레코드 편집기** 대화 상자에 다음 필드가 내림차순으로 표시되도록 구성합니다.

- 제품 이름
- 제품 번호
- 제품 설명
- 제품 유형 코드

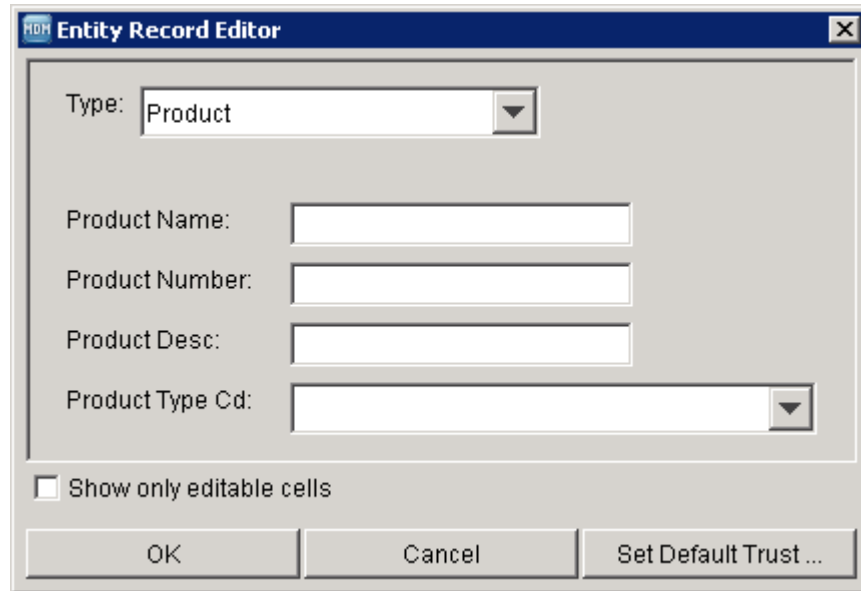
1. 쓰기 잠금을 획득합니다.
2. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 항목 유형 노드를 선택합니다.
3. **HM 패키지** 섹션의 **추가** 열에서 **항목 레코드 편집기** 대화 상자에 필드로 표시할 열에 번호를 할당합니다. 가장 낮은 값을 할당하는 열이 필드 목록의 맨 위에 표시됩니다.

Column Name	Label	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	0	0	0	4	0	4		
Product Name	1	1		0	1	0	0	0
Product Number	2	2		1	4	2	1	1
Product Desc	3	3		2	3	1		2
Product Type Cd	4	4		3	2	3		3
Product Type					5			
Hub State Ind	5	5				5		

4. **저장**을 클릭합니다.

5. 제품 그룹 항목 유형에 대해 해당 단계를 반복해서 수행합니다.

다음 이미지는 계층 관리자에서 제품 항목을 작성할 때 표시되는 **항목 레코드 편집기** 대화 상자를 보여 줍니다.



The image shows a dialog box titled "Entity Record Editor". It contains the following fields:

- Type: Product (dropdown menu)
- Product Name: (text input field)
- Product Number: (text input field)
- Product Desc: (text input field)
- Product Type Cd: (dropdown menu)
- ☐ Show only editable cells
- Buttons: OK, Cancel, Set Default Trust ...

관계 작성 필드 구성

이 자습서에서는 사용자가 계층 관리자에 관계를 작성할 때 **관계 레코드 편집기** 대화 상자에 다음 필드가 내림차순으로 표시되도록 구성합니다.

- Hub 상태 표시기
- 계층 수준

1. 쓰기 잠금을 획득합니다.
2. 계층 도구의 탐색 창에 있는 기본 계층 프로필 아래에서 제품 그룹이 제품의 상위 항목 관계 유형 노드를 선택합니다.
3. **HM 패키지** 섹션의 **추가** 열에서 **관계 레코드 편집기** 대화 상자에 필드로 표시할 열에 번호를 할당합니다. 가장 낮은 값을 할당하는 열이 필드 목록의 맨 위에 표시됩니다.

HM Packages:							
Column Name	Tooltip	Common	Search	List	Detail	Put	Add
Rowid Object	U	U	U	U	U	U	
Rowid Hierarchy			1	1	1	1	
Rowid Rel Type						5	
Product ID1			2	2	2	2	
Product ID2			3	3	3	3	
Hub State Indicator			4	4	4	4	U
Hierarchy Level						6	1

4. **저장**을 클릭합니다.
5. 제품 그룹이 제품 그룹의 상위 항목 관계 유형에 대해 해당 단계를 반복해서 수행하십시오.

다음 이미지는 계층 관리자에서 두 개의 항목 간에 관계를 작성할 때 표시되는 **관계 레코드 편집기** 대화 상자를 보여 줍니다.

MDM Relationship Record Editor

77 Mice

61 S150 Laser Mouse for Notebooks

Hierarchy: Product

Relationship Type: Product Group is parent of Pr...

Start Date: [Red X] [Calendar Icon]

End Date: [Red X] [Calendar Icon]

Hub State Indicator: [Dropdown]

Hierarchy Level: [Text Box]

☐ Show only editable cells

OK Cancel Set Default ...

계층 관리

계층을 구성한 후 레코드와 레코드 간의 관계를 MDM Hub의 계층 관리 도구에 추가하거나 Informatica Data Director의 계층 관리 도구에 추가할 수 있습니다. 이 자습서에는 계층 관리자 도구를 사용하여 구성된 계층을 데이터로 채우는 방법에 대해 설명되지 않습니다.

MDM Hub 콘솔의 계층 관리자 도구에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

Informatica Data Director의 계층 관리자에 대한 자세한 내용은 *Multidomain MDM Data Director 사용자 가이드*를 참조하십시오.

파트 IV: 데이터 흐름 구성

이 파트에 포함된 장:

- [MDM Hub 프로세스, 254](#)
- [랜드 프로세스 구성, 283](#)
- [MDM Hub 준비, 289](#)
- [영구 삭제 검색, 311](#)
- [데이터 정리 구성, 327](#)
- [로드 프로세스 구성, 352](#)
- [일치 프로세스 구성, 373](#)
- [일치 규칙 구성 예제, 431](#)
- [Elasticsearch를 사용한 검색, 454](#)
- [통합 프로세스 구성, 475](#)
- [보류 중인 제어 테이블, 480](#)
- [게시 프로세스 구성, 481](#)

제 15 장

MDM Hub 프로세스

이 장에 포함된 항목:

- [MDM Hub 프로세스 개요, 254](#)
- [Informatica MDM Hub 프로세스 정보, 254](#)
- [랜드 프로세스, 257](#)
- [준비 프로세스, 258](#)
- [로드 프로세스, 260](#)
- [토큰화 프로세스, 267](#)
- [일치 프로세스, 273](#)
- [통합 프로세스, 278](#)
- [게시 프로세스, 280](#)

MDM Hub 프로세스 개요

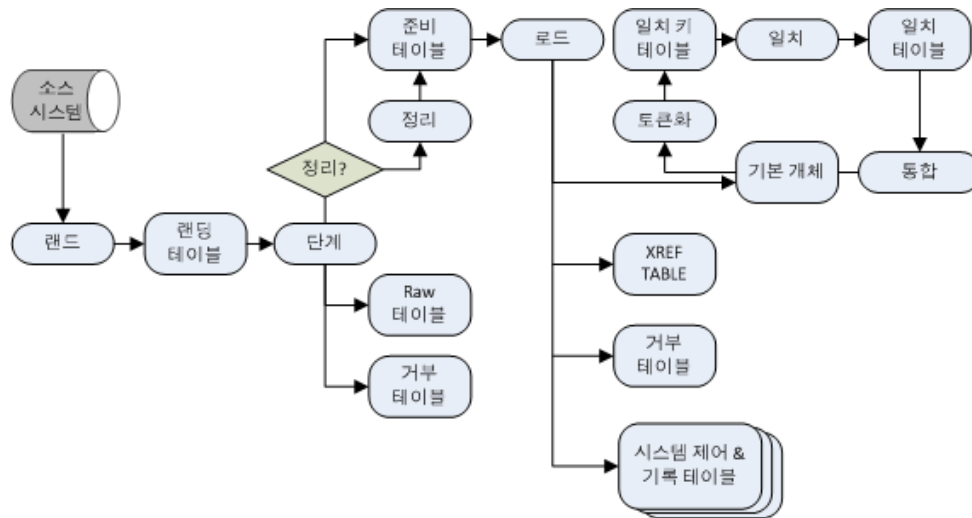
MDM Hub에 데이터를 로드하고 여러 일괄 처리를 통해 이러한 데이터를 처리할 수 있습니다. 랜드 프로세스를 통해 데이터를 MDM Hub으로 이동할 수 있습니다. 그러면 준비 프로세스를 사용하여 로드 프로세스에 대해 데이터를 정리 및 준비하여 데이터를 기본 개체에 로드할 수 있습니다. 로드 프로세스 후 토큰화 및 일치 프로세스를 실행하여 중복 레코드를 식별하고 통합 프로세스를 통해 해당 레코드를 통합할 수 있습니다.

Informatica MDM Hub 프로세스 정보

Informatica MDM Hub의 일괄 처리 기능을 사용하면 Informatica MDM Hub에서 데이터가 개별 프로세스를 순서대로 거치며 처리됩니다.

일괄 처리의 전체 데이터 흐름

다음 그림은 개별 프로세스, 소스 시스템, 기본 개체 및 지원 테이블 등을 비롯하여 일괄 처리를 사용한 Informatica MDM Hub에서의 전반적인 데이터 흐름을 상세하게 보여 줍니다.



참고: 게시 프로세스는 일괄 처리가 아니므로 이 그림에 나열되어 있지 않습니다.

기본 개체 레코드의 통합 상태

이 섹션에서는 기본 개체의 레코드 통합 상태에 대해 설명합니다.

통합 표시기

모든 기본 개체에는 **CONSOLIDATION_IND**라는 시스템 열이 있습니다.

이 **통합 표시기**는 개별 레코드가 **MDM Hub**의 여러 프로세스를 거칠 때 해당 레코드의 통합 상태를 나타냅니다.

다음 테이블에서는 통합 표시기 값에 대해 설명합니다.

값	상태 이름	설명
1	CONSOLIDATED	레코드가 고유한 것으로 확인되어 통합되었으며 BVT(최선의 진실)를 나타냄을 나타냅니다.
2	QUEUED_FOR_MERGE	레코드가 일치 프로세스를 거쳤을 수 있으며 병합에 대한 준비가 완료되었음을 나타냅니다. 또한 레코드가 병합 프로세스를 거쳤을 수 있으며 다른 병합에 대한 준비가 완료되었습니다.
3	Not MERGED 또는 MATCHED	이 레코드는 일치 프로세스를 거쳤으며 통합 준비가 완료되었습니다. 또한 레코드가 수동 병합 프로세스를 거친 경우(여기서 일치하는 데이터 스튜어드에 의해 수행됨) 통합 표시기가 2입니다.
4	NEWLY_LOADED	레코드가 새로 삽입되었고 레코드가 일치 프로세스를 거치도록 플래그 지정되었음을 나타냅니다.
9	ON_HOLD	데이터 스튜어드가 추가 알림이 있을 때까지 해당 레코드를 대기 상태로 전환했음을 나타냅니다. 통합 표시기 값에 관계없이 모든 레코드를 대기 상태로 전환할 수 있습니다. 일치 및 통합 프로세스는 대기 중인 레코드를 무시합니다.

통합 표시기가 변경되는 방식

Informatica MDM Hub에서는 다음과 같은 순서로 기본 개체 레코드의 통합 표시기를 업데이트합니다.

1. 로드 프로세스 중에 기본 개체에 새 레코드가 로드될 때 Informatica MDM Hub는 해당 레코드에 통합 표시기 4를 할당합니다. 이 상태는 레코드를 일치시켜야 한다는 의미입니다.
2. 일치 프로세스가 시작되기 직전에 레코드가 일치 후보로 선택될 때 일치 프로세스가 레코드의 통합 표시기를 3으로 변경합니다.

참고: 일치 또는 병합 구성 설정이 변경되면 기본 개체의 레코드를 재설정(통합 표시기를 일치 준비 완료 상태인 4로 변경)할지 묻는 일치 재설정 대화 상자가 나타납니다.

3. 일치 프로세스는 일치 후보 레코드의 통합 표시기를 2(통합 준비 완료)로 변경한 후 완료됩니다. 수동 병합을 수행하는 경우 데이터 스튜어드가 수동 일치를 수행하고 병합된 레코드의 통합 표시기가 2가 됩니다.

참고: 일치 프로세스에서 레코드의 일치 항목을 찾을 수도 있고 그렇지 않을 수도 있습니다.

병합 관리자에 통합 표시기가 2인 레코드가 표시됩니다.

4. [일치되지 않은 모든 행을 고유 행으로 허용]이 활성화되어 있는 상태에서 레코드가 일치 프로세스를 거쳤지만 일치 항목을 찾지 못한 경우 Informatica MDM Hub가 자동으로 레코드의 통합 표시기를 1(고유)로 변경합니다.
5. "일치되지 않은 모든 행을 고유 행으로 허용"이 활성화되어 있는 경우 통합 프로세스를 거친 레코드에 더 이상 병합할 중복 항목이 없으면 Informatica MDM Hub가 레코드의 통합 표시기를 1로 변경합니다. 이 상태는 이 레코드가 기본 개체 내에서 고유하며 기본 개체에서 해당 항목에 대한 마스터 레코드(BVT)를 나타낸다는 의미입니다.

참고: 레코드의 통합 표시기가 1로 설정되어 있으면 Informatica MDM Hub가 이 레코드를 다른 레코드와 직접 일치시키지 않습니다. 새로운 레코드나 업데이트된 레코드(통합 표시기가 4인 레코드)는 통합된 레코드와 일치시킬 수 있습니다.

셀 데이터 존속 및 우선 순위 순서

두 레코드에서 병합할 셀을 평가하는 동안 MDM Hub는 존속할 셀 데이터와 삭제할 데이터를 결정합니다. MDM Hub는 존속 셀 데이터 또는 우선 셀이 두 셀 중에서 BVT(최선의 진실, Best Version of the Truth)를 보다 잘 나타내는 것으로 간주합니다. 최종적으로 통합된 단일 레코드는 최상의 존속 셀 데이터를 포함하며 BVT를 나타냅니다.

존속은 트러스트가 활성화된 열과 트러스트가 활성화되지 않은 열 모두에 적용됩니다. MDM Hub에서 서로 다른 두 레코드의 셀을 비교할 때 다음 요소(우선 순위순으로 나열됨)를 기준으로 존속 여부를 결정합니다.

1. 트러스트 점수. 트러스트 점수는 트러스트가 활성화된 열에만 적용됩니다. ROWID_OBJECT가 가장 높은 데이터가 우선합니다. 트러스트 점수가 동일하거나 한 열에 대해 트러스트가 활성화되지 않은 경우 MDM Hub는 다음 비교로 진행합니다.
2. 교차 참조 레코드의 SRC_LUD. 교차 참조 SRC_LUD 값이 가장 최신인 데이터가 우선합니다. SRC_LUD 값이 같으면 MDM Hub는 다음 비교로 진행합니다.
3. 기본 개체의 ROWID_OBJECT. ROWID_OBJECT 값은 수(내림차순)에 따라 평가됩니다. ROWID_OBJECT 값이 같으면 MDM Hub는 다음 비교로 진행합니다.
4. 교차 참조의 ROWID_XREF. ROWID_XREF 값은 수(내림차순)에 따라 평가됩니다. ROWID_XREF가 가장 높은 데이터가 우선합니다.

ROWID_OBJECT 존속

레코드가 병합될 때 MDM Hub는 병합된 레코드의 ROWID_OBJECT가 되기 위해 어떤 ROWID_OBJECT가 존속되는지를 결정합니다. ROWID_OBJECT의 존속은 레코드가 병합되는 방법과 위치에 따라 다릅니다. 예를 들어

일괄 병합 작업 후 레코드가 존속하는 방법은 레코드가 IDD(Informatica Data Director)에서 존속하는 방법과는 다릅니다.

MDM Hub는 ROWID_OBJECT의 존속을 다음 각 시나리오에서 다르게 처리합니다.

일괄 병합 작업

일괄 병합 작업 중에 MDM Hub는 모든 레코드에 대한 통합 표시기를 고려합니다. 새 레코드 (CONSOLIDATION_IND = 4)가 통합된 레코드(CONSOLIDATION_IND = 1)와 일치 및 병합되는 경우 통합된 ROWID_OBJECT가 존속합니다. 두 개의 새 레코드(CONSOLIDATION_IND = 4)가 일치하는 경우 가장 낮은 ROWID_OBJECT가 있는 레코드가 존속합니다.

SIF API 병합

레코드를 병합하는 데 SIF API 병합을 사용할 경우 대상 ROWID_OBJECT가 존속합니다. 대상은 SIF 요청에 나열된 첫 번째 레코드입니다.

Informatica Data Director

IDD에서 실시간으로 레코드를 병합할 경우 대상 ROWID_OBJECT가 존속합니다. 대상은 IDD의 병합 일치 비교 뷰에서 병합 미리보기 열에 표시되는 레코드입니다.

병합할 레코드를 대기열에 넣는 경우 IDD에 표시되는 현재 레코드의 ROWID_OBJECT가 존속합니다. 그러나 IDD에서 존속은 기본 개체 레코드에 대한 통합 표시기에 따라 다릅니다. 예를 들어 레코드 (CONSOLIDATION_IND = 4)가 통합된 레코드(CONSOLIDATION_IND = 1)와 일치 및 병합되는 경우 통합된 ROWID_OBJECT가 존속합니다.

참고: 레코드가 병합하기 위해 대기열에 있는 경우 IDD는 기본 _MTCH 테이블의 항목을 사용합니다. 따라서 존속하는 ROWID_OBJECT를 정확히 알려면 _MTCH 테이블을 확인합니다. ROWID_OBJECT_MATCHED 열에 존속하는 ROWID_OBJECT가 나열되어 있습니다.

랜드 프로세스

이 섹션에서는 Informatica MDM Hub의 랜드 프로세스와 관련된 개념 및 태스크에 대해 설명합니다.

랜드 프로세스 정보

데이터 랜딩은 데이터를 Informatica MDM Hub로 로드하기 위한 초기 단계입니다.

소스 시스템 및 랜딩 테이블

데이터 랜딩에는 하나 이상의 소스 시스템의 데이터를 Informatica MDM Hub 랜딩 테이블로 전송하는 작업이 포함됩니다.

- 소스 시스템은 Informatica MDM Hub에 데이터를 제공하는 외부 시스템입니다. 소스 시스템은 조직 내부의 응용 프로그램, 데이터 저장소 및 기타 시스템이 될 수 있으며, 외부 소스로부터 획득하거나 구입할 수도 있습니다.
- 랜딩 테이블은 처음에 소스 시스템에서 로드된 데이터를 포함하는, Hub 저장소의 테이블입니다.

Informatica MDM Hub 외부에서 실행되는 랜드 프로세스

랜드 프로세스는 Informatica MDM Hub 외부 프로세스로, Hub 저장소의 랜딩 테이블을 직접적으로 채우는 외부 일괄 처리 또는 외부 응용 프로그램을 사용하여 실행됩니다. 이후에 데이터를 관리하는 프로세스는 Informatica MDM Hub 내부 프로세스입니다.

랜딩 테이블을 채우는 방식

랜딩 테이블은 다음과 같은 방법으로 채울 수 있습니다.

로드 방법	설명
외부 일괄 처리	ETL(추출-변환-로드) 도구 또는 다른 외부 프로세스는 소스 시스템의 데이터를 Informatica MDM Hub에 복사합니다. 일괄 로드는 Informatica MDM Hub 외부에서 실행됩니다. Informatica MDM Hub에서는 일괄 로드의 결과만을 채워진 랜딩 테이블의 형태로 볼 수 있습니다. 참고: 이 프로세스는 사용자가 선택한 별도의 ETL 도구에 의해 처리됩니다. 이 ETL 도구는 Informatica MDM Hub 제품군에 포함되어 있지 않습니다.
실시간 처리	외부 응용 프로그램에서는 온라인 실시간 모드로 랜딩 테이블을 채울 수 있습니다. 이러한 응용 프로그램은 Informatica MDM Hub 제품군에 포함되어 있지 않습니다.

특정 소스 시스템에 사용되는 방법은 해당 소스 시스템의 데이터에 가장 효율적인 방법(또는 유일한 방법)이 무엇인지에 따라 결정됩니다. 또한 초기 데이터 로드(Hub 저장소에 비즈니스 데이터가 처음 로드되는 경우)에는 일괄 처리가 사용되는 경우가 많은데, 이는 이 방법이 랜딩 테이블에 대량의 레코드를 채우는 데 가장 효율적인 방법이기 때문입니다.

참고: 랜딩 테이블의 데이터는 기본 개체에 대한 로드 프로세스가 실행되어 완료되기 전까지는 삭제할 수 없습니다.

랜드 프로세스 관리

랜드 프로세스를 관리하려면 이 설명서의 다음 항목을 참조하십시오.

태스크	항목
구성	장 16, “랜드 프로세스 구성” 페이지 283 : - “소스 시스템 구성” 페이지 283 - “랜딩 테이블 구성” 페이지 286
실행	랜드 프로세스 실행은 Informatica MDM Hub 외부에서 발생하며 “랜딩 테이블을 채우는 방식” 페이지 258 에 설명된 대로 랜딩 테이블을 채우기 위해 사용하는 방식에 따라 달라집니다.
응용 프로그램 개발	외부 응용 프로그램을 사용하여 랜딩 테이블을 채우는 경우 응용 프로그램에서 사용하는 API에 대한 개발자 설명서를 참조하십시오.

준비 프로세스

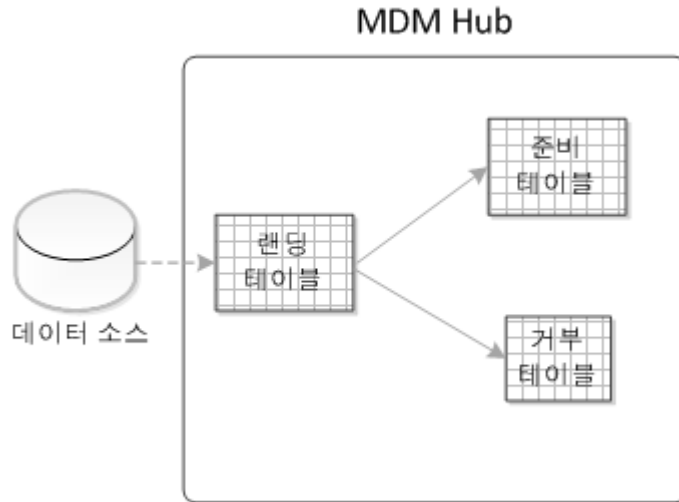
준비 프로세스는 소스 데이터를 특정 기본 개체와 연결된 준비 테이블로 전송합니다. MDM Hub 또는 Informatica 플랫폼에서 전체 준비 프로세스를 실행할 수 있습니다.

준비 프로세스 동안 소스 데이터는 기본 개체와 연결된 준비 테이블로 이동됩니다. 준비 프로세스를 실행하여 소스 시스템에서 준비 테이블로 데이터를 바로 로드하거나 MDM Hub 랜딩 테이블을 통해 데이터를 로드할 수 있습니다. MDM Hub를 Informatica 플랫폼과 통합한 경우 소스 시스템에서 MDM Hub 준비 테이블로 데이터를 바로 로드할 수 있습니다. MDM Hub를 Informatica 플랫폼과 통합하지 않은 경우 준비 프로세스에서 데이터를 준비 테이블로 이동할 수 있도록 소스 데이터를 랜딩 테이블로 로드합니다.

MDM Hub 준비

MDM Hub 준비 프로세스에서는 랜딩 테이블의 소스 데이터를 특정 기본 개체와 연결된 준비 테이블로 전송합니다. 전체 준비 프로세스는 MDM Hub에서 수행됩니다.

다음 이미지는 소스 데이터가 랜딩 테이블에서 준비 테이블 및 거부 테이블로 전송되는 MDM Hub 준비 프로세스를 보여 줍니다.



MDM Hub 준비 프로세스를 수행하기 전에 외부 데이터 소스의 데이터를 랜딩 테이블에 로드합니다. 랜딩 테이블과 준비 테이블 간 매핑을 정의합니다. 매핑은 랜딩 테이블의 소스 열을 준비 테이블의 대상 열과 연결합니다. MDM Hub에서 준비 테이블로 데이터를 이동하기 전에 데이터를 정리해야 하는 경우 매핑에서 데이터 정리를 구성합니다. 준비 작업을 실행하면 MDM Hub에서 매핑을 기준으로 한 데이터를 랜딩 테이블 열에서 준비 테이블 열로 전송합니다.

준비 프로세스 중에 MDM Hub는 한 번에 250개 레코드가 포함된 단일 블록을 처리합니다. 블록의 레코드에 문제가 있는 경우 MDM Hub는 레코드를 거부 테이블로 이동합니다. 셀 값이 너무 길거나 레코드 업데이트 날짜가 오늘 날짜보다 이후이기 때문에 레코드가 거부될 수 있습니다. MDM Hub에서 거부된 레코드가 이동한 후 MDM Hub는 블록의 나머지 레코드 처리를 중지하고 다른 블록으로 이동합니다. 준비 프로세스가 완료되면 작업을 다시 실행하십시오. 처리되지 않은 레코드가 다시 선택되고 처리됩니다.

랜딩 테이블의 데이터 기록을 유지할 수 있습니다. 준비 테이블에 대해 감사 내역을 활성화한 경우 랜딩 테이블 데이터는 Raw 테이블에 보관됩니다. MDM Hub은 구성된 준비 작업 실행 횟수 또는 보존 기간 동안 Raw 테이블에서 랜딩 테이블 데이터를 유지합니다. 지정한 준비 작업 실행 횟수 또는 보존 기간에 MDM Hub이 도달하면 MDM Hub은 Raw 테이블에서 소스 개체의 각 기본 키에 대해 하나의 레코드를 유지합니다.

MDM Hub을 구성하여 랜딩 테이블에서 새로 추가되고 업데이트된 레코드를 식별할 수 있습니다. 준비 테이블에 대해 델타 검색을 활성화한 경우 MDM Hub에서는 새로 추가되고 업데이트된 레코드를 처리하고 변경되지 않은 레코드는 무시합니다.

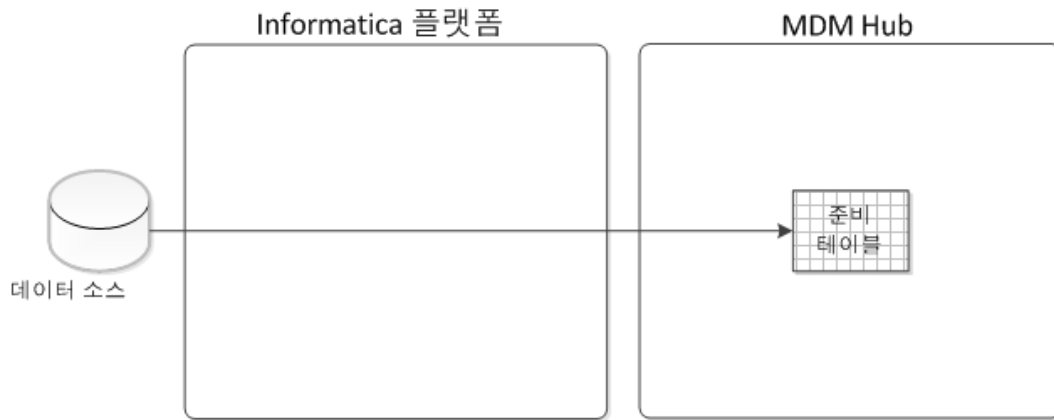
MDM Hub에서는 데이터를 한 랜딩 테이블에서 여러 준비 테이블로 전송할 수 있습니다. 그러나 각 준비 테이블은 하나의 랜딩 테이블에서만 데이터를 수신합니다.

준비 프로세스에서는 로드 프로세스를 진행할 수 있도록 데이터를 준비하여 준비 테이블의 데이터를 대상 기본 개체로 로드합니다.

Informatica 플랫폼 준비

Informatica 플랫폼 준비 프로세스에서는 프로세스 중에 랜딩 테이블을 사용하지 않고 소스 데이터를 특정 기본 개체와 연결된 MDM Hub 준비 테이블로 전송합니다. 준비 프로세스는 모델 리포지토리에서 논리 데이터 개체를 생성하여 데이터를 처리한 다음 이 데이터를 MDM Hub 준비 테이블로 전송합니다.

다음 이미지는 소스 데이터가 데이터 소스에서 MDM Hub 준비 테이블로 바로 전송되는 Informatica 플랫폼 준비 프로세스를 보여 줍니다.



Informatica 플랫폼 준비를 실행하려면 MDM Hub을 Informatica 플랫폼과 통합해야 합니다. Hub 콘솔을 사용하여 MDM Hub을 Informatica 플랫폼과 통합합니다. 모델 리포지토리 서비스에 대한 연결 매개 변수를 구성한 후 Informatica 플랫폼 준비를 활성화하고 MDM Hub을 모델 리포지토리와 동기화해야 합니다. 동기화에서는 개발자 도구를 사용하여 편집할 수 있는 데이터 개체를 생성합니다.

MDM Hub를 모델 리포지토리 서비스와 동기화한 후 동기화 프로세스에서 생성한 개체를 사용하여 매핑을 생성합니다. 준비 프로세스를 실행하면 데이터 통합 서비스에서 소스 데이터를 처리하고 MDM Hub 준비 테이블로 데이터를 전송합니다.

준비 프로세스 동안 데이터 통합 서비스에서 거부할 수도 있는 데이터는 모델 리포지토리에서 관리됩니다.

Informatica 플랫폼 준비를 수행할 때는 탭 검색, 영구 삭제 검색 및 감사 내역을 설정할 수 없습니다.

준비 프로세스에서는 로드 프로세스를 진행할 수 있도록 데이터를 준비하여 준비 테이블의 데이터를 대상 기본 개체로 이후에 로드합니다.

로드 프로세스

이 섹션에서는 Informatica MDM Hub의 로드 프로세스와 관련된 개념 및 태스크에 대해 설명합니다.

로드 프로세스 정보

Informatica MDM Hub에서 로드 프로세스는 준비 테이블의 데이터를 Hub 저장소의 해당하는 대상 테이블(기본 개체)로 이동합니다.

로드 프로세스에서는 다음을 기준으로 준비 테이블의 데이터에 대해 수행할 작업을 결정합니다.

- 대상 테이블에 해당하는 레코드가 이미 있는지 여부와, 이미 있을 경우 로드 프로세스를 마지막으로 실행한 이후 준비 테이블의 레코드가 업데이트되었는지 여부
- 일부 열에 대한 트러스트가 활성화되어 있는지 여부(기본 개체만 해당). 활성화된 경우 로드 프로세스에서는 셀 데이터에 대한 트러스트 점수를 계산합니다.

- 데이터가 로드하기에 올바른지 여부. 올바르지 않은 경우 로드 프로세스에서는 해당 레코드를 거부합니다.
- 기타 구성 설정

로드 프로세스와 연결된 테이블

기본 개체 외에 **Hub** 저장소의 준비 테이블, 교차 참조 테이블, 기록 테이블 및 거부 테이블도 로드 프로세스와 연결되어 있습니다.

다음 테이블은 로드 프로세스와 연결되어 있습니다.

준비 테이블

준비 프로세스 중 랜딩 테이블에서 허용 및 복사된 데이터를 포함합니다.

교차 참조 테이블

데이터 연계, 즉 기본 개체의 각 레코드에 대한 소스 시스템을 추적하는 데 사용됩니다. **Informatica MDM Hub**는 기본 개체로 로드된 각 소스 시스템 레코드에 대해 다음을 포함한 레코드를 교차 참조 테이블에서 유지 관리합니다.

- 레코드를 제공한 시스템의 식별자
- 소스 시스템에 있는 해당 레코드의 기본 키 값
- 해당 시스템에서 제공한 가장 최신 셀 값

각 기본 개체 레코드에는 하나 이상의 교차 참조 레코드가 포함됩니다.

기록 테이블

기본 개체에 대해 기록이 활성화되어 있는 경우 레코드가 업데이트되거나 삽입되면 로드 프로세스는 이 정보를 다음 두 개의 테이블에 기록합니다.

- 기본 개체 기록 테이블
- 교차 참조 기록 테이블

거부 테이블

로드 프로세스가 특정 이유로 거부한 준비 테이블의 레코드를 포함합니다. 거부된 레코드는 기본 개체에 로드되지 않습니다. 거부 테이블은 준비 테이블과 연결됩니다(이름: `stagingTableName_REJ`). 로드 작업 실행 후 거부된 레코드를 검사할 수 있습니다.

Informatica MDM Hub는 레코드를 거부할 이유를 처음 발견하면 레코드를 거부하여 성능을 높입니다. 거부 테이블은 **Informatica MDM Hub**가 레코드를 거부한 한 가지 이유를 설명합니다. **Informatica MDM Hub**가 레코드를 거부한 이유가 두 가지 이상인 경우 거부 테이블은 **Informatica MDM Hub**가 발견한 첫 번째 이유를 설명합니다.

초기 데이터 로드 및 증분 로드

초기 데이터 로드(IDL)란 데이터가 새로 생성된 빈 기본 개체에 처음 로드되는 시점을 나타냅니다.

초기 데이터 로드 중에는 준비 테이블의 모든 레코드가 기본 개체에 새 레코드로 삽입됩니다.

기본 개체에 대한 초기 데이터 로드가 실행된 후에는 새 데이터나 업데이트된 데이터만 기본 개체에 로드되기 때문에 모든 후속 로드 프로세스는 증분 로드라고 합니다.

중복 데이터는 무시됩니다.

트러스트 설정 및 유효성 검사 규칙

Informatica MDM Hub에서는 트러스트 및 유효성 검사 규칙을 사용하여 가장 신뢰도가 높은 데이터를 손쉽게 확인할 수 있습니다.

트러스트 설정

기본 개체의 열에 있는 데이터가 여러 소스 시스템에서 파생된 경우 Informatica MDM Hub에서는 트러스트를 사용하여 여러 소스 시스템에서 가져온 열 데이터의 상대적 신뢰도를 비교하도록 도와 줍니다. 예를 들어 Orders 시스템은 Direct Marketing 시스템보다 청구 주소의 신뢰도가 더 높은 소스일 수 있습니다.

트러스트는 열 수준에서 활성화되고 구성됩니다. 예를 들어 Orders 시스템의 Customer Name과 Billing 시스템의 Phone Number에 더 높은 트러스트 수준을 지정할 수 있습니다.

다음 테이블에서는 통합할 두 개의 기본 개체 레코드를 보여 줍니다.

ROWID_OBJECT	이름	전화
100	Doug McDougal Grp	1-555-901-4670
200	The Doug McDougal Group	201-10810

다음 테이블에서는 각 열에 대해 계산된 트러스트 점수를 보여 줍니다.

ROWID_OBJECT	이름	전화
100	62	56('전화' 열에서 트러스트 점수가 더 높음)
200	71('이름' 열에서 트러스트 점수가 더 높음)	37

통합된 레코드에는 트러스트 점수가 가장 높은 데이터가 존속됩니다. 다음 테이블에서는 통합된 레코드를 보여 줍니다.

ROWID_OBJECT	이름	전화
100	The Doug McDougal Group	1-555-901-4670

트러스트는 소스 시스템, 변경 기록 및 기타 비즈니스 규칙에 따라 각 셀과 연관된 상대적 신뢰도를 측정하는 메커니즘을 제공합니다. 트러스트에는 셀 데이터의 품질 및 보존 기간과 시간에 따른 신뢰도 붕괴(감소) 정도가 반영됩니다. 트러스트는 두 레코드가 통합될 때 존속되는 레코드를 결정하고 소스 시스템의 업데이트가 마스터 레코드를 업데이트하기에 충분히 신뢰할 수 있는지 여부를 확인하는 데 사용됩니다.

데이터 스튜어드는 특정 값이 정확하다고 직접적으로 알고 있는 경우 계산된 트러스트 설정을 수동으로 재정의할 수 있습니다. 또한 데이터 스튜어드는 기본 개체의 레코드에 값을 바로 입력할 수도 있습니다. 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

유효성 검사 규칙

트러스트는 종종 유효성 검사 규칙과 함께 사용되며, 그 결과 구성된 조건 및 작업에 따라 트러스트 점수가 다운 그레이드(감소)될 수 있습니다.

데이터가 유효성 검사 규칙에 지정된 조건을 충족하면 해당 데이터의 트러스트 값이 유효성 검사 규칙에 지정된 백분율만큼 다운그레이드됩니다. (예:

```
Downgrade trust on First_Name by 50% if Length < 3
Downgrade trust on Address Line 1, City, State, Zip and Valid_address_ind if Valid_address_ind= 'False'
```

열에 대해 최소 트러스트 보유 플래그가 활성화(선택)되어 있는 경우에는 트러스트가 열의 최소 트러스트 설정 아래로 다운그레이드될 수 없습니다.

로드 프로세스의 런타임 실행 흐름

이 섹션에서는 구성된 설정을 바탕으로 로드 프로세스 동안 수행되는 작업 및 처리 중인 데이터 특성에 대해 자세히 설명합니다. 또한 Informatica MDM Hub 로드 프로세스의 기본 동작에 대해서도 설명합니다. 또는 증분 로드인 경우 RowID별로 로드하여 처리 중인 로드, 일치 및 병합을 간소화할 수 있습니다.

레코드 존재 여부 확인

로드 프로세스 동안 Informatica MDM Hub는 먼저 동일한 소스 시스템의 기존 레코드와 동일한 기본 키를 가진 레코드가 있는지 확인합니다. 준비 테이블의 각 레코드를 대상 테이블의 레코드와 비교하여 해당 레코드가 대상 테이블에 이미 있는지 확인합니다.

이 비교 결과에 따라 다음 작업이 결정됩니다.

로드 작업	설명
삽입형 로드	준비 테이블의 레코드가 대상 테이블에 없을 경우 Informatica MDM Hub는 해당 새 레코드를 대상 테이블에 삽입 합니다.
업데이트형 로드	준비 테이블의 레코드가 대상 테이블에 이미 있을 경우 Informatica MDM Hub는 적절한 작업을 수행합니다. 대상 기본 개체가 준비 테이블의 레코드에 있는 데이터로 업데이트되는 경우 업데이트형 로드가 수행됩니다. 로드 프로세스에서는 레코드가 소스 시스템에서 마지막으로 제공된 이후 변경된 경우에만 레코드를 업데이트합니다. 업데이트형 로드는 현재 Informatica MDM Hub 구성 설정과 준비 테이블의 각 레코드에 있는 데이터의 특성에 따라 제어됩니다. 예를 들어 강제 업데이트가 활성화된 경우 레코드는 이미 로드되었는지 여부에 관계없이 업데이트됩니다.

로드 프로세스 동안 업데이트형 로드가 먼저 실행된 후 삽입형 로드가 실행됩니다.

삽입형 로드

MDM Hub는 삽입형 로드 프로세스 중에 다음 단계를 수행합니다.

1. 셀 데이터의 트러스트를 계산합니다.
2. 유효성 검사 규칙을 실행합니다.
3. 외래 키 조화를 수행합니다.
4. ROWID_OBJECT 값을 생성합니다.
5. 레코드를 대상 기본 개체와 기타 테이블에 삽입합니다.

삽입형 로드 중에 수행되는 작업은 대상 기본 개체 및 기타 요인에 따라 달라집니다.

삽입형 로드와 대상 기본 개체

준비 테이블에서 레코드에 대한 삽입형 로드를 수행하려면

- 로드 프로세스에서는 새 레코드에 대한 고유한 ROWID_OBJECT 값을 생성합니다.

- 로드 프로세스에서 외래 키 조회를 수행하여 참조 무결성을 유지 관리하는 데 필요한 모든 외래 키 값을 대체합니다.
- 로드 프로세스에서는 레코드를 기본 개체에 삽입하고, 생성된 ROWID_OBJECT 값(기본 개체에서 이 레코드의 기본 키), 외래 키 조회 값 및 null 값을 비롯한 준비 테이블의 모든 열 데이터(PKEY_SRC_OBJECT 제외)를 이 새 레코드에 복사합니다.
기본 개체에는 동일한 개체의 레코드가 여러 개 포함될 수 있습니다. 예를 들어 소스 시스템 A의 레코드 하나와 소스 시스템 B의 레코드 하나가 있을 수 있습니다. Informatica MDM Hub는 두 개의 새 레코드를 모두 새로운 항목으로 플래그 지정합니다.
- 기본 개체의 새로운 각 레코드에 대해 토큰화 프로세스 중 일치 키가 다시 생성될 수 있도록 로드 프로세스를 통해 ROWID_OBJECT 값이 연결된 더티 테이블에 삽입됩니다.
- 기본 개체의 새로운 각 레코드에 대해 로드 프로세스는 새 레코드를 기본 개체의 다른 레코드에 일치시킬 수 있도록 해당 CONSOLIDATION_IND를 4(일치 준비 완료)로 설정합니다.
- 로드 프로세스에서는 레코드를 기본 개체와 연결된 교차 참조 테이블에 삽입합니다. 로드 프로세스에서는 교차 참조 테이블에 대한 기본 키 값을 생성한 후 생성된 키와 소스 시스템에 대한 식별자 및 준비 테이블의 열(PKEY_SRC_OBJECT 포함)을 이 새 레코드에 복사합니다.
참고: 기본 개체에는 소스 시스템의 기본 키 값이 포함되지 않습니다. 대신, 생성된 ROWID_OBJECT 값이 기본 개체의 기본 키가 됩니다. 소스 시스템의 기본 키(PKEY_SRC_OBJECT)는 교차 참조 테이블에 저장됩니다.
- 기본 개체에 대해 기록이 활성화된 경우 로드 프로세스는 레코드를 기록 및 교차 참조 기록 테이블에 삽입합니다.
- 기본 개체에서 하나 이상의 열에 대해 트러스트가 활성화된 경우 로드 프로세스는 트러스트 알고리즘을 지원하는 제어 테이블에도 레코드를 삽입하여 트러스트된 각 셀의 트러스트 및 유효성 검사 규칙의 요소를 트러스트 계산에 사용되는 값으로 채웁니다. 이러한 정보는 나중에 필요할 때 트러스트를 계산하는 데 사용될 수 있습니다.
- 기본 개체에 대해 로드 시 일치 토큰 생성이 활성화된 경우 로드 프로세스가 완료된 후 토큰화 프로세스가 자동으로 시작됩니다.

업데이트형 로드

업데이트형 로드 프로세스 중에 MDM Hub는 다음과 같은 단계를 수행합니다.

1. 레코드가 변경되었는지 확인합니다.
2. 셀 데이터의 트러스트를 계산합니다.
3. 유효성 검사 규칙을 실행합니다.
4. 외래 키 조회를 수행합니다.
5. 대상 기본 개체 및 기타 테이블에서 대상 레코드를 업데이트합니다.

업데이트형 로드 중에 수행되는 작업은 대상 기본 개체와 기타 요인에 따라 달라집니다.

업데이트형 로드와 대상 기본 개체

대상 기본 개체의 업데이트형 로드 중에 다음과 같은 변경이 발생합니다.

1. 기본적으로 로드 프로세스에서는 준비 테이블의 각 레코드에 대해 LAST_UPDATE_DATE 열의 값을 연결된 교차 참조 테이블의 최근 소스 업데이트 날짜(SRC_LUD)와 비교합니다.
 - 소스 시스템에서 최근에 레코드를 제공한 시간 이후에 준비 테이블의 레코드가 업데이트된 경우 로드 프로세스에서 업데이트형 로드를 진행합니다.

- 소스 시스템에서 최근에 레코드를 제공한 시간 이후에 준비 테이블의 레코드가 변경되지 않았다면 날짜가 동일하고 트러스트가 활성화되지 않은 경우 로드 프로세스에서 레코드를 무시하거나(아무 작업도 수행하지 않음) 레코드가 중복된 경우 레코드를 거부합니다.
관리자는 로드 프로세스에서 이 **LAST_UPDATE_DATE** 확인을 건너뛰고 레코드가 이미 로드되었는지 여부와 관계없이 레코드를 강제로 업데이트하도록 기본 동작을 변경할 수 있습니다.
2. 로드 프로세스에서 외래 키 조화를 수행하여 참조 무결성을 유지 관리하는 데 필요한 모든 외래 키 값을 대체합니다.
 3. 대상 기본 개체에 트러스트가 활성화된 열이 있는 경우 로드 프로세스에서 다음이 수행됩니다.
 - 이 트러스트된 열에 대해 구성된 트러스트 설정을 기반으로 업데이트할 레코드의 트러스트가 활성화된 각 열에 대해 트러스트 점수를 계산합니다.
 - 유효성 검사 규칙이 정의되어 있다면 규칙을 적용하여 해당하는 경우 트러스트 점수를 다운그레이드합니다.
 4. 로드 프로세스에서 기본 개체의 레코드를 업데이트하고 교차 참조 테이블, 기록 테이블 및 기타 제어 테이블(해당하는 경우)의 연결된 레코드를 업데이트합니다. 또한 로드 프로세스에서는 토큰화 프로세스를 통해 일치 키가 다시 생성될 수 있도록 기본 개체와 연결된 더티 테이블에 레코드의 **ROWID_OBJECT** 값이 삽입됩니다. 기본 개체에서는 통합 표시기 값을 유지합니다. 로드 프로세스에서 다음 규칙에 따라 기본 개체의 대상 레코드를 업데이트합니다.
 - 준비 테이블 레코드에 있는 셀의 트러스트 점수가 대상 기본 개체 레코드에 있는 해당 셀의 트러스트 점수보다 높은 경우 로드 프로세스에서 대상 레코드의 셀을 업데이트합니다.
 - 준비 테이블 레코드에 있는 셀의 트러스트 점수가 대상 기본 개체 레코드에 있는 해당 셀의 트러스트 점수보다 낮은 경우 로드 프로세스에서 대상 레코드의 셀을 업데이트하지 않습니다.
 - 준비 테이블 레코드에 있는 셀의 트러스트 점수가 대상 기본 개체 레코드에 있는 해당 셀의 트러스트 점수와 같거나 열에 대해 트러스트가 활성화되지 않은 경우에는 두 레코드 중에서 **LAST_UPDATE_DATE**가 더 최신인 레코드의 셀 값이 사용됩니다.
 - 준비 테이블 레코드의 **LAST_UPDATE_DATE**가 더 최신인 경우 대상 기본 개체 레코드에서 해당 셀이 업데이트됩니다.
 - 대상 기본 개체 레코드의 **LAST_UPDATE_DATE**가 더 최신인 경우에는 셀이 업데이트되지 않습니다.
 5. 기본 개체에 대해 로드 시 일치 토큰 생성이 활성화되어 있다면 로드 프로세스가 완료된 후 자동으로 토큰화 프로세스가 시작됩니다.

외래 키 조회

일괄 로드 또는 **Put API**에서 레코드를 삽입하거나 업데이트하면 **Informatica MDM Hub**에서는 준비 테이블 조회 구성을 사용하여 소스 시스템 외래 키를 **Informatica MDM Hub** 외래 키로 변환합니다.

참조 무결성 제약 조건 비활성화

초기 로드/업데이트 중이나 실시간 동시 액세스가 없는 경우 성능 향상을 위해 기본 개체에 대한 참조 무결성 제약 조건을 비활성화할 수 있습니다.

정의되지 않은 조회

조회 테이블 및 조회 열을 채워 하위 개체에 대한 조회를 정의하지 않은 경우 로드 프로세스를 실행하기 전에 하위 개체에 대해 준비 프로세스를 반복해야만 데이터를 로드할 수 있습니다.

Null 외래 키 허용

준비 테이블에 대해 열을 구성할 때 대상 기본 개체에 대해 **Null** 외래 키를 허용할지 여부를 지정할 수 있습니다. **Null** 외래 키 허용 준비 테이블 열 속성은 **null** 외래 키 값이 허용되는지 여부를 결정합니다.

참고: MDM Hub에서 빈 문자열은 빈 문자열을 적용하는 데이터베이스 유형에 상관없이 **null** 값에 해당합니다.

기본적으로 Null 외래 키 허용 확인란은 선택 취소되어 있습니다. 즉 Null 외래 키가 허용되지 않습니다. 로드 프로세스는

- 올바른 조회 값을 포함한 레코드를 수락합니다.
- Null 외래 키를 포함한 레코드를 거부합니다.
- 잘못된 외래 키 값을 포함한 레코드를 거부합니다.

Null 외래 키 허용이 활성화(선택)된 경우, 로드 프로세스:

- 올바른 조회 값을 포함한 레코드를 수락합니다.
- Null 외래 키를 포함한 레코드를 수락하고 이 레코드에 대해 삽입형 로드 및 업데이트형 로드를 허용합니다.
- 잘못된 외래 키 값을 포함한 레코드를 거부합니다.

로드 프로세스는 수락된 레코드에 대해서만 삽입형 로드 및 업데이트형 로드를 허용합니다. 거부된 레코드는 대상 테이블에 로드되지 않고 거부 테이블에 삽입됩니다.

참고: 대상 기본 개체가 비어 있으면 초기 데이터 로드와 한해 로드 프로세스에서 Null 외래 키가 허용됩니다.

로드 작업에서 거부된 레코드

로드 프로세스 동안 준비 테이블의 레코드는 다음과 같은 이유 등으로 거부될 수 있습니다.

- LAST_UPDATE_DATE 열의 미래 날짜나 NULL 날짜
- 1900년도 이전의 LAST_UPDATE_DATE
- 준비 테이블의 PKEY_SRC_OBJECT에 매핑된 NULL 값
- PKEY_SRC_OBJECT에 중복되는 값이 있음
- HUB_STATE_IND 필드의 값이 잘못됨(상태 사용 기본 개체에만 해당)
- 잘못된 외래 키나 NULL 외래 키

거부된 레코드는 기본 개체에 로드되지 않습니다. 로드 작업 실행 후 거부된 레코드를 검사할 수 있습니다.

참고: 레코드를 거부하려면 로드 프로세스에서 랜딩 테이블을 역추적해야 합니다. 준비 테이블의 레코드를 로드하고 있고 연결된 랜딩 테이블의 해당 레코드가 삭제된 경우 로드 프로세스는 이 레코드를 거부 테이블에 삽입하지 않습니다.

로드 프로세스의 기타 고려 사항

이 섹션에서는 로드 프로세스의 기타 고려 사항에 대해 설명합니다.

로드 프로세스에서 상위-하위 레코드를 처리하는 방법

상위 테이블에서 생성된 키가 하위 테이블에 포함된 경우 로드 프로세스에서는 적절한 기본 키 값을 상위 테이블에서 하위 테이블로 복사합니다. 예를 들어 다음과 같은 데이터가 있다고 가정합니다.

상위 테이블:

PARENT_ID	FNAME	LNAME
101	Joe	Smith
102	Jane	Smith

하위 테이블: PARENTS PKEY_SRC_OBJECT와의 관계가 있습니다.

ADDRESS	CITY	STATE	FKEY_PARENT
1893	my city	CA	101
1893	my city	CA	102

이 예에는 ROWID_OBJECT, PKEY_SRC_OBJECT 또는 테이블 조회를 위한 고유 열을 가리키는 관계가 있을 수 있습니다.

상태 사용 기본 개체 로드

로드 프로세스에서 상태 사용 기본 개체에 대한 레코드를 처리할 때 고려해야 할 특별한 사항이 있습니다.

참고: 로드 프로세스는 준비 테이블에서 HUB_STATE_IND 열의 값이 잘못된 레코드를 거부합니다.

일치 토큰 생성(선택 사항)

일치 프로세스를 실행하기 전에 일치 토큰을 생성해야 합니다. 스키마 관리자에서 기본 개체를 구성할 때 로드 작업이 완료된 후 즉시 일치 토큰을 생성할지, 아니면 일치 작업이 실행될 때까지 데이터 토큰화를 지연할지를 지정할 수 있습니다. 로드 시 일치 토큰 생성 확인란의 설정에 따라 토큰화 프로세스가 수행되는 시간이 결정됩니다.

토큰화 프로세스

토큰화 프로세스에서는 일치 토큰을 생성하고 기본 개체와 연결된 일치 키 테이블에 저장합니다. 일치 토큰은 이후에 일치 프로세스에서 일치 후보를 식별하는 데 사용됩니다.

참고: MDM Hub는 암호화된 데이터가 있는 열에 대한 일치 키를 생성할 수 없습니다.

일치 토큰 및 일치 키

일치 토큰은 기본 개체 레코드의 데이터에 대한 인코딩된 표현 및 인코딩되지 않은 표현입니다. 일치 토큰에 포함되는 항목은 다음과 같습니다.

- 일치 키 - 유사 항목 일치 기본 개체에 대한 유사 항목 일치 키의 모든 열에서 작성된 인코딩된 값으로 구성된 고정 길이의 압축된 문자열. 일치 키에는 관련 변형이 동일한 일치 키 값을 가지도록 이름이나 주소에서 문자와 숫자의 조합이 포함됩니다.
- 일치 열의 플랫폼 데이터로 구성된 인코딩되지 않은 문자열(유사 항목 일치 키, 모든 유사 항목 일치 열 및 정확히 일치 열)

일치 키 테이블

일치 키 테이블에는 MDM Hub가 기본 개체 레코드에 대해 생성하는 일치 토큰과 일치 키가 포함됩니다. 일치 키 테이블은 각 기본 개체와 연결됩니다.

일치 키 테이블 이름의 형식은 C_<기본 개체 이름>_STRP입니다. 예를 들어 기본 개체의 이름이 PARTY인 경우 연결된 일치 키 테이블의 이름은 C_PARTY_STRP입니다. 토큰화 프로세스에서는 기본 개체의 각 레코드에 대한 일치 키 테이블에 하나 이상의 레코드가 생성됩니다.

일치 및 병합 프로세스가 완료된 후에는 유효하지 않은 일치 토큰을 일치 키 테이블에서 삭제해야 합니다. 또한 재토큰화 중에 유효하지 않은 일치 토큰이 삭제되고 유효한 일치 토큰으로 바뀝니다. 유효하지 않은 일치 토큰을 삭제하는 작업은 MDM Hub의 성능에 영향을 미칠 수 있습니다. IBM DB2 환경의 성능을 개선하려면 유효하지 않은 일치 토큰을 삭제하는 대신 해당 토큰을 유효하지 않은 것으로 표시하는 `cmx.server.stripDML.useUpdate=true` 속성을 구성합니다.

다음 테이블에는 일치 키 테이블의 주요 열이 설명되어 있습니다.

열 이름	데이터 유형(크기)	설명
ROWID_OBJECT	CHAR(14)	일치 토큰을 생성한 레코드를 식별합니다.
SSA_KEY	CHAR(8)	레코드의 일치 키입니다. 이름, 주소, 조직 이름 등 연결된 기본 개체 레코드에 대한 유사 항목 일치 키 열 값의 인코딩된 표현입니다. 문자열은 이름이나 주소의 단어 및 숫자 조합으로 생성된 고정 길이의 압축되고 인코딩된 값으로 구성됩니다.
SSA_DATA	VARCHAR2(500)	연결된 기본 개체 레코드에 정의된 일치 열 연결의 인코딩되지 않은 일반 텍스트 문자열 표현입니다. 일치 열 연결에는 유사 항목 일치 키, 모든 유사 항목 일치 열 및 정확히 일치 열이 포함됩니다.

일치 키 테이블의 각 레코드에는 일치 토큰(SSA_KEY 및 SSA_DATA 모두에 속한 데이터)이 포함됩니다.

일치 키 예

생성되는 일치 키는 구성된 일치 설정과 기본 개체의 데이터 특성에 따라 다릅니다.

다음 예에서는 유사 항목 일치/검색 전략을 사용하여 문자열에서 생성된 일치 키를 보여 줍니다.

레코드의 문자열	생성된 일치 키
BETH O'BRIEN	MMU\$?/\$-
BETH O'BRIEN	PCOG\$\$\$\$
BETH O'BRIEN	VL/IEFLM
LIZ O'BRIEN	PCOG\$\$\$\$
LIZ O'BRIEN	SXOG\$\$\$\$-
LIZ O'BRIEN	VL/IEFLM

이 예에서 문자열 **BETH O'BRIEN** 및 **LIZ O'BRIEN**의 일치 키 값(PCOG\$\$\$\$)은 동일합니다. 일치 프로세스 중에 일치 후보를 검색하는 동안 일치 프로세스에서 이러한 문자열을 일치 후보로 간주할 수 있습니다.

토큰화 프로세스는 유사 항목 일치 기본 개체에만 적용됨

토큰화 프로세스는 유사 항목 일치 기본 개체에만 적용되고 정확히 일치 기본 개체에는 적용되지 않습니다. 유사 항목 일치 기본 개체의 경우 토큰화 프로세스는 Informatica MDM Hub가 일치하는 것으로 간주할 수 있을 만큼의 유사점을 갖는 행(반드시 동일하지 않아도 됨)을 일치시킬 수 있도록 합니다.

토큰화 프로세스의 주요 개념

이 섹션에서는 토큰화 프로세스에 적용되는 주요 개념에 대해 설명합니다.

일치 토큰 생성

MDM Hub는 일괄 작업 또는 SIF API가 실행될 때 일치 토큰을 생성하거나 업데이트합니다. 일치 토큰은 일치 키 테이블에 저장되며 일치 프로세스를 위해 최신 상태로 유지되어야 합니다. MDM Hub는 일치 프로세스와 관계없이 일치 토큰을 유지 관리합니다.

일괄 작업 또는 SIF API가 실행되면 기본 개체 레코드가 더티 상태로 표시될 수 있습니다. MDM Hub는 더티로 표시된 기본 개체 레코드에 대한 일치 토큰을 생성하거나 업데이트합니다.

기본 개체 레코드는 다음 모든 조건이 충족될 경우 더티로 표시됩니다.

- 업데이트로 인해 기본 개체의 일치 열이 영향을 받습니다.
- 업데이트 후 일치 열의 BVT(최선의 진실)가 이전 값과 다릅니다.

더티 테이블

모든 기본 개체에는 시스템 테이블인 <기본 개체 이름>_DRTY라는 더티 테이블이 연결되어 있습니다. 더티 테이블에는 ROWID_OBJECT 열이 있으며 이 열은 일치 토큰을 생성해야 하는 기본 개체 레코드를 식별합니다. MDM Hub는 일치 토큰을 일치 키 테이블에 저장합니다.

더티 테이블의 고유한 각 ROWID_OBJECT에 대해 토큰화 프로세스를 통해 일치 토큰이 생성된 후 해당 더티 테이블이 정리됩니다.

MDM Hub는 다음 일괄 프로세스 시퀀스 중 더티 테이블을 업데이트합니다.

1. 로드 - MDM Hub가 새 레코드를 로드하거나 기존 레코드를 업데이트합니다. MDM Hub는 새 레코드 또는 일치 열 값이 변경된 업데이트된 레코드의 ROWID_OBJECT 값으로 더티 테이블을 채웁니다.
2. 토큰화 - MDM Hub가 일치 키를 생성합니다. MDM Hub는 더티 테이블에서 토큰화된 레코드의 ROWID_OBJECT 값을 제거합니다.
3. 일치 - MDM Hub가 일치를 식별합니다. 더티 테이블은 변경되지 않은 상태로 유지됩니다.
4. 통합 - MDM Hub가 일치된 레코드를 통합합니다. MDM Hub는 새 레코드 또는 일치 열 값이 변경된 업데이트된 레코드의 ROWID_OBJECT 값으로 더티 테이블을 채웁니다.
5. 토큰화 - MDM Hub가 일치 키를 생성합니다. MDM Hub는 더티 테이블에서 토큰화된 레코드의 ROWID_OBJECT 값을 제거합니다.

유사 항목 일치 기본 개체의 키 유형 및 키 너비

유사 항목 일치 기본 개체의 경우 다음 설정을 기반으로 일치 키가 생성됩니다.

속성	설명
키 유형	이 기본 개체에 대해 토큰화되는 정보의 기본 유형(Person_Name, Organization_Name 또는 Address_Part1)을 식별합니다. 일치 프로세스에서는 이름 및 주소 특성에 대한 정보를 사용하여 일치 키를 생성하고 검색을 수행합니다. 사용 가능한 키 유형은 사용하는 인구집단 집합에 따라 달라집니다.
키 너비	유사 항목 일치 키의 분석 정확도, 반환되는 가능한 일치 후보의 수 및 키가 사용하는 디스크 공간의 양을 결정합니다. 사용 가능한 키 너비로는 제한됨, 표준, 확장됨 및 기본 설정이 있습니다.

일치 키는 데이터의 오류, 변형 및 단어 치환을 처리할 수 있어야 하므로 Informatica MDM Hub에서는 각 이름, 주소 또는 조직에 대해 여러 개의 일치 토큰을 생성합니다. 기본 개체 레코드당 생성되는 키 수는 데이터와 일치 키 너비에 따라 달라집니다.

일치 키 분포 및 핫스팟

스키마 관리자의 일치/병합 설정 세부 정보 창에 있는 일치 키 분포 탭을 사용하여 일치 키 테이블의 일치 키 분포를 조사할 수 있습니다. 이 도구는 데이터에서 잠재적 핫스팟(과도 일치를 초래할 수 있는 일치 키의 과도한 집중)을 식별하는 데 도움을 줍니다. 핫스팟이 있는 경우 일치 프로세스에서 관련 없는 일치를 포함하여 지나치게 많은 일치를 생성하게 됩니다.

토큰화 비율

변경된 레코드의 백분율이 기본 개체에 고급 속성으로 구성된 일정 비율을 초과할 때마다 토큰화 프로세스를 반복하도록 일치 프로세스를 구성할 수 있습니다.

토큰화 및 병합 프로세스 성능 최적화

MDM Hub 환경의 토큰화 및 병합 프로세스 성능을 최적화하려면 `cmxcleanse.properties` 파일에 최적화 속성을 추가합니다.

- 다음 디렉터리에서 `cmxcleanse.properties` 파일을 엽니다.
UNIX의 경우, `<infamdm 설치 디렉터리>/hub/cleanse/resources`
Windows의 경우, `<infamdm 설치 디렉터리>\hub\cleanse\resources`
- 토큰화 및 병합 프로세스 성능을 최적화하려면 `cmxcleanse.properties` 파일에 최적화 속성을 추가합니다.
다음 테이블에는 구성할 수 있는 최적화 속성이 설명되어 있습니다.

속성	설명
<code>cmx.server.stripDML.useUpdate</code>	IBM DB2에만 해당합니다. <code>true</code> 로 설정하면 MDM Hub가 일치 키 테이블에서 필요하지 않은 일치 토큰을 삭제하지 않고 대신 해당 토큰의 상태를 유효하지 않은 것으로 업데이트하도록 구성됩니다. 기본값은 <code>false</code> 입니다. 유효하지 않은 일치 토큰의 상태를 업데이트하는 것이 유효하지 않은 일치 토큰을 제거하는 삭제 작업보다 효율적입니다. 참고: 이 속성을 <code>true</code> 로 설정하여 유효하지 않은 레코드를 제거하려면 일치 키 테이블을 주기적으로 정리해야 합니다.
<code>cmx.server.stripDML.blockSize</code>	MDM Hub가 각 블록에서 처리하는 레코드 수입니다. 기본값은 100입니다.
<code>cmx.server.stripDML.noOfThreadsForInsert</code>	MDM Hub가 레코드를 일치 키 테이블에 삽입할 때 사용하는 스레드 수입니다. 기본값은 50입니다.
<code>cmx.server.stripDML.noOfThreadsForUpdate</code>	MDM Hub가 일치 키 테이블의 레코드를 업데이트할 때 사용하는 스레드 수입니다. 기본값은 30입니다.
<code>cmx.server.stripDML.noOfThreadsForDelete</code>	MDM Hub가 일치 키 테이블에서 레코드를 삭제할 때 사용하는 스레드 수입니다. 기본값은 30입니다.

참고: MDM Hub 환경의 요구 사항에 따라 다양한 작업의 블록 크기 및 스레드 수를 구성하십시오.

일치 키 테이블 정리

일치 키 테이블에 유효하지 않은 일치 토큰이 포함되는 경우 일치 키 테이블을 정리해야 합니다.

일치 키 테이블을 정리하려면 다음 태스크 중 하나를 수행합니다.

- 모든 일치 토큰을 다시 생성합니다.
- CleanStrp 일괄 작업을 실행합니다.
- 백그라운드 정리 프로세스를 실행합니다.

일치 토큰 다시 생성

일치 키 테이블에서 유효하지 않은 모든 일치 토큰을 삭제하려면 기본 개체에 대한 모든 일치 토큰을 다시 생성할 수 있습니다.

1. Hub 콘솔에서 일괄 처리 뷰어 도구를 시작합니다.
2. 모든 일치 토큰을 다시 생성할 기본 개체를 확장합니다.
3. **일치 토큰 생성**을 확장합니다.

일치 토큰을 생성할 때 사용할 수 있는 기본 개체에 연결된 일괄 작업이 나타납니다.

4. 일치 토큰을 생성할 때 사용할 일괄 작업을 선택합니다.
일치 토큰 생성 일괄 작업에 대한 속성이 나타납니다.
5. **모든 일치 토큰 다시 생성** 옵션을 활성화합니다.
6. **일괄 실행**을 클릭합니다.

MDM Hub가 전체 기본 개체에 대한 일치 토큰을 다시 생성합니다.

CleanStrp 일괄 작업 실행

CleanStrp 일괄 작업을 실행하여 일치 키 테이블에서 유효하지 않은 일치 토큰을 삭제할 수 있습니다.

CleanStrp 일괄 작업은 `invalid_ind=1`인 일치 토큰을 식별하고 이러한 일치 토큰을 삭제합니다. 또한 이 일괄 작업은 상태가 `invalid_ind=1`인 일치 토큰을 삭제한 후 일치 키 테이블의 인덱스를 재작성합니다.

1. Hub 콘솔에서 일괄 처리 뷰어 도구를 시작합니다.
2. 유효하지 않은 모든 일치 토큰을 정리할 기본 개체를 확장합니다.
3. **CleanStrp**를 확장합니다.
일치 키 테이블에서 유효하지 않은 일치 토큰을 정리할 때 사용할 수 있는 일괄 작업이 나타납니다.
4. 일치 토큰을 정리할 때 사용할 일괄 작업을 선택합니다.
CleanStrp 일괄 작업에 대한 속성이 나타납니다.
5. **일괄 실행**을 클릭합니다.

MDM Hub가 기본 개체에 연결된 일치 키 테이블에서 유효하지 않은 일치 토큰을 정리합니다.

백그라운드 정리 프로세스 실행

백그라운드 정리 프로세스를 실행하여 일치 키 테이블에서 유효하지 않은 일치 토큰을 삭제할 수 있습니다. 백그라운드 정리 일괄 프로세스는 `invalid_ind=1`인 일치 토큰을 식별하고 이러한 일치 토큰을 삭제합니다. 백그라운드 정리 프로세스는 일치 토큰을 삭제한 후 일치 키 테이블의 인덱스를 재작성하지 않습니다.

- ▶ 백그라운드 정리 프로세스의 실행을 활성화하려면 관련된 Hub 서버 속성을 속성 파일에 추가합니다.
 - a. 다음 디렉터리에서 `cmxserver.properties` 파일을 엽니다.
 UNIX의 경우. <infadm 설치 디렉터리>/hub/server/resources
 Windows의 경우. <infadm 설치 디렉터리>\hub\server\resources
 - b. 다음 속성을 `cmxserver.properties` 파일에 추가합니다.

속성	설명
<code>cmx.server.strp_clean.execution_mode</code>	<p>일치 키 테이블에 대한 백그라운드 정리 프로세스의 작업 범위를 구성합니다.</p> <p>작업 범위의 경우 다음 값 중 하나를 사용합니다.</p> <ul style="list-style-type: none"> - ALL. 등록된 모든 연산 참조 저장소의 모든 일치 키 테이블에서 <code>invalid_ind=1</code>인 일치 토큰을 검색합니다. - CONFIGURED_ORS. 지정한 모든 연산 참조 저장소의 모든 일치 키 테이블에서 <code>invalid_ind=1</code>인 일치 토큰을 검색합니다. 작업 범위를 CONFIGURED_ORS로 설정한 경우 <code>cmx.server.strp_clean.ors</code> 속성을 <code>cmxserver.properties</code> 파일에 추가합니다. - CONFIGURED_STRP. 특정 연산 참조 저장소의 특정 기본 개체에 대한 일치 키 테이블에서 <code>invalid_ind=1</code>인 일치 토큰을 검색합니다. 작업 범위를 CONFIGURED_STRP로 설정한 경우 <code>cmx.server.strp_clean.strp</code> 속성을 <code>cmxserver.properties</code> 파일에 추가합니다.
<code>cmx.server.strp_clean.ors</code>	<p>백그라운드 정리 프로세스를 실행하여 유효하지 않은 일치 토큰을 삭제해야 하는 연산 참조 저장소의 이름을 지정합니다. 예를 들어 <code>cmx_ors1</code> 및 <code>cmx_ors2</code>의 모든 일치 키 테이블에서 <code>invalid_ind=1</code>인 일치 토큰을 삭제하려면 <code>cmx.server.strp_clean.ors=cmx_ors1,cmx_ors2</code>를 추가합니다.</p>
<code>cmx.server.strp_clean.strp</code>	<p>백그라운드 정리 프로세스를 실행하여 일치 키 테이블을 정리해야 하는 연산 참조 저장소 및 기본 개체 조합을 지정합니다. 예를 들어 <code>cmx_ors1</code>의 B01 및 <code>cmx_ors2</code>의 B02에 대한 일치 키 테이블에서 <code>invalid_ind=1</code>인 일치 토큰을 삭제하려면 <code>cmx.server.strp_clean.strp=cmx_ors1.C_B01,cmx_ors2.C_B02</code>를 추가합니다.</p>
<code>cmx.server.strp_clean.delete_records_count</code>	<p>일치 키 테이블에서 정리할 레코드 수를 지정합니다.</p>

속성	설명
cmx.server.strp_clean.retry_sec	MDM Hub가 일치 키 테이블에서 유효하지 않은 일치 토큰이 있는 레코드를 검색할 기간(초)을 지정합니다. 기본값은 60입니다.
cmx.server.strp_clean.threads_count	MDM Hub가 일치 키 테이블에서 유효하지 않은 일치 토큰이 있는 레코드를 검색할 때 사용할 스레드 수를 지정합니다. 기본값은 20입니다.

일치 프로세스

기본 개체의 레코드를 통합하려면 먼저 **Informatica MDM Hub**에서 중복(일치) 가능성이 있는 레코드를 확인해야 합니다.

일치 프로세스에서는 일치 규칙을 사용하여 다음과 같은 작업을 수행합니다.

- 기본 개체에서 중복(동일 또는 유사) 가능성이 있는 레코드를 식별합니다.
- 자동으로 통합할 수 있을 만큼 유사점이 많은 레코드와 통합 전에 데이터 스튜어드가 수동으로 검토해야 하는 레코드를 확인합니다.

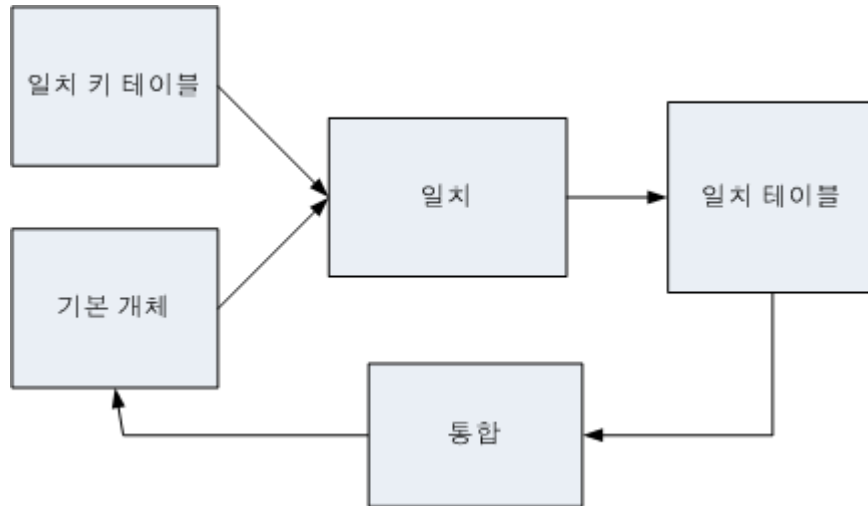
Informatica MDM Hub에서 일치 프로세스는 레코드를 비교하고 중복 항목을 확인하는 두 가지 용도로 사용됩니다.

- 유사 항목 일치는 **Informatica MDM Hub**에서 기본 개체의 레코드를 일치시키는 데 사용되는 가장 일반적인 방법입니다. 유사 항목 일치 방법에서는 레코드 간에 충분한 유사점이 있는지 찾고, 데이터 패턴에 있을 수 있는 맞춤법 오류, 전치, 단어 결합 및 분리, 누락, 생략, 음운 변이 등의 변이를 고려한 확률적 일치 항목을 확인합니다.
- 정확히 일치 방법은 일치 열의 값이 동일한 레코드를 일치시키므로 일반적으로 자주 사용되지 않습니다. 정확히 일치 전략은 속도는 빠르지만 데이터가 불완전한 경우 일부 일치 항목이 누락될 수 있다는 단점이 있습니다.

최적의 선택 옵션은 데이터의 특성, 데이터에 대한 지식 및 특정 일치 및 통합 요구 사항에 따라 달라집니다.

일치 프로세스 중 Informatica MDM Hub에서는 기본 개체의 레코드를 비교하여 유사점을 확인합니다. 두 레코드 간에 충분한 유사점(동일 또는 유사 일치 항목)이 있어 두 레코드가 서로 중복된 레코드일 수 있음을 나타내는 경우 일치 프로세스에서는 다음을 수행합니다.

- 일치 테이블에 일치된 레코드 쌍에 대한 ROWID_OBJECT 참조와 함께 일치 항목을 식별한 일치 규칙과 일치된 레코드를 자동으로 통합할 수 있는지 여부를 채웁니다.



- 통합 표시기를 2(통합 준비 완료)로 변경하여 이러한 레코드에 통합 플래그를 지정합니다.

참고: 데이터 암호화를 구성하는 경우 암호화된 열이 일치 규칙에 포함되어 있으면 MDM Hub가 일치 키를 생성할 수 없으므로 일치 작업을 수행할 수 없습니다.

일치 규칙

일치 규칙은 Informatica MDM Hub가 기본 개체의 두 개의 레코드가 중복되는지 여부를 결정하는 기준을 정의합니다. 일치 규칙은 일치 열이나 기본 키를 기준으로 할 수 있습니다.

유형	설명
일치 열 규칙	일치 열로 정의한 열의 값을 바탕으로 기본 개체 레코드를 일치시키는 데 사용되며(예: 이름, 성, 주소 1, 주소 2), 일치 항목을 식별하는 데 있어 가장 일반적으로 사용되는 방법입니다.
기본 키 일치 규칙	레코드에 대해 같은 기본 키를 사용하는 두 시스템의 레코드를 일치시키는 데 사용됩니다. 두 개의 서로 다른 소스 시스템이 같은 기본 키를 사용하는 것은 드문 경우입니다. 하지만 이러한 상황이 발생하면 기본 키 일치가 신속하고 매우 정확합니다.

같은 기본 개체에 대해 두 유형의 일치 규칙을 같이 사용할 수 있습니다.

지정한 일치 규칙은 다음 유형 중 하나가 될 수 있습니다.

규칙 유형	설명
정확히 일치	값이 정확히 일치해야 하거나 특별한 경우가 정확히 일치해야 합니다(예: null이 null과 일치). 정확히 일치에 대한 처리는 데이터베이스 서버에서 주로 수행됩니다.
유사 항목 일치	값이 정확히 일치하지는 않지만 일치되는 값과 비슷합니다. 일치 항목은 일부 값에서 공유되는 일치 토큰에 따라 결정되며 인구집단에 종속됩니다. 예를 들어 이름 일치 목적으로 영어로 된 인구집단에서 Robert, Rob, Bob에는 같은 일치 토큰 값이 포함될 수 있습니다. 유사 항목 일치는 주로 응용 프로그램 서버에서 처리됩니다.
필터링된 일치	정확히 일치와 비슷합니다. 정확히 일치 규칙이 [필터링됨] 유형으로 바뀌며, 정확히 일치 규칙에서와 같은 결과를 얻기 위해 유사 항목 일치 엔진이 사용됩니다. 필터링된 일치 규칙에서는 [한정됨] 검색 수준이 사용됩니다. 유사 항목 일치에 따른 결과가 정확히 일치 결과와 같도록 많이 한정되어야 하기 때문입니다. 필터링된 항목 일치에 대한 처리는 응용 프로그램 서버에서 주로 수행됩니다.

정확히 일치하는 기본 개체 및 유사 항목 일치 기본 개체

기본 개체는 다음과 같은 일치 유형 중 하나를 사용하도록 구성됩니다.

기본 개체 유형	설명
정확히 일치하는 기본 개체	정확히 일치 열만 포함할 수 있습니다.
유사 항목 일치 기본 개체	유사 항목 일치 열과 정확히 일치 열을 모두 포함할 수 있습니다. <ul style="list-style-type: none"> - 유사 항목 일치 열만 - 정확히 일치 열만 또는 - 유사 항목 일치 열과 정확히 일치 열의 조합

기본 개체의 유형에 따라 정의할 수 있는 일치 유형과 일치 열 유형이 결정됩니다. 기본 개체 유형은 기본 개체에 대해 선택한 일치/검색 전략에 따라 결정됩니다.

일치 프로세스에 사용되는 지원 테이블

일치 프로세스에서는 다음과 같은 지원 테이블을 사용합니다.

테이블	설명
일치 키 테이블	모든 기본 개체 레코드에 대해 생성된 일치 키가 포함됩니다. 일치 키 테이블에는 다음과 같은 명명 규칙이 사용됩니다. <i>C_baseObjectName_STRP</i> 여기서 <i>baseObjectName</i> 은 기본 개체의 루트 이름입니다. 예: C_PARTY_STRP
일치 테이블	이 기본 개체에 대해 일치 프로세스를 실행하여 얻은 기본 개체의 일치된 레코드 쌍이 포함됩니다. 일치 테이블에는 다음과 같은 명명 규칙이 사용됩니다. <i>C_baseObjectName_MTCH</i> 여기서 <i>baseObjectName</i> 은 기본 개체의 루트 이름입니다. 예: C_PARTY_MTCH
일치 플래그 감사 테이블	병합 관리자에서 자동 병합을 위해 수동 일치 레코드를 대기열에 넣은 사용자의 사용자 ID가 포함됩니다. 일치 플래그 감사 테이블에는 다음과 같은 명명 규칙이 사용됩니다. <i>C_baseObjectName_FHMA</i> 여기서 <i>baseObjectName</i> 은 기본 개체의 루트 이름입니다. 이 기본 개체에 대해 일치 플래그 감사 테이블이 활성화되어 있는 경우에만 사용됩니다.

인구집단 집합

유사 항목 일치/검색 전략을 포함한 기본 개체에 대해 일치 프로세스는 표준 인구집단 설정을 사용하여 국가별, 지역별 및 언어 차이를 처리합니다. 인구집단 설정은 일치 프로세스가 토근화, 일치/검색 전략 및 일치 목적을 처리하는 방법에 영향을 줍니다.

인구집단 설정은 지정된 인구집단에 대해 일반적인 이름, 주소 및 기타 식별 정보에 대한 인텔리전스를 캡슐화합니다. 예를 들어 국가마다 번지 및 거리 이름 배치, 우편 번호 위치 등과 같은 서로 다른 주소 형식이 사용됩니다. 마찬가지로 지역마다 분포되어 있는 성도 다릅니다. "Smith"라는 성은 미국의 인구집단에서는 매우 일반적이지만 다른 국가에서는 일반적이지 않습니다.

인구집단 설정은 특정 인구집단에 대해 데이터에서 발생할 수 있는 변형 및 오류를 수용하여 일치 정확도를 개선합니다.

중복 데이터에 대한 일치

중복 데이터에 대한 일치 기능은 시스템 이외의 모든 기본 개체 열의 중복 항목에 대한 일치 항목을 생성하는 데 사용됩니다. 이 일치 항목은 기본 개체 열의 완전한 중복 항목에 대한 집합 수보다 많을 경우 생성됩니다. 대부분의 데이터의 경우 최적의 값은 2입니다.

일치 항목이 생성되었더라도 표준 일치 규칙을 사용하여 나중에 일치시킬 수 있도록 해당 레코드에 대해 통합 표시기는 4(통합되지 않음)로 표시됩니다.

참고: 중복 데이터에 대한 일치 작업은 임계값이 1 이상으로 설정되고 해당 기본 개체에 대해 정의된 NON_EQUAL 일치 규칙이 없을 경우 표시됩니다.

일치 그룹 빌드 및 전이적 일치

BMG(일치 그룹 빌드) 프로세스에서는 통합 프로세스 전에 불필요한 일치 항목을 제거합니다. 예를 들어 기본 개체에 다음 일치 쌍이 있다고 가정합니다.

- 레코드 1이 레코드 2와 일치
- 레코드 2가 레코드 3과 일치
- 레코드 3이 레코드 4와 일치

일치 프로세스를 실행하고 일치 그룹 빌드를 생성한 후 통합 프로세스를 실행하기 전에 다음 레코드를 확인할 수 있습니다.

- 레코드 2가 레코드 1과 일치
- 레코드 3가 레코드 1과 일치
- 레코드 4가 레코드 1과 일치

이 예에서 레코드 4를 레코드 1과 일치시키는 명시적인 규칙은 없습니다. 대신 다른 일치 동작(레코드 1이 2와 일치되고, 2가 3과 일치하며, 3이 4와 일치)으로 인해 간접적인 일치가 수행되었습니다. 간접적 일치를 *전이적 일치*라고도 합니다. 병합 관리자와 데이터 관리자에서는 전체 일치 기록을 표시하여 전이적 일치의 세부 정보를 제공할 수 있습니다.

BMG(일치 그룹 빌드) 프로세스를 사용하는 경우 기본 개체에 대한 일치 속성에서 **일치되지 않은 모든 행을 고유 행으로 허용**을 활성화합니다. **일치되지 않은 모든 행을 고유 행으로 허용**을 활성화하지 않으면 전이적 일치 작업 중 HMRG 테이블에 rowid_match_rule에 대한 잘못된 값이 표시됩니다.

수동 통합의 최대 일치 항목 수

일괄 작업 동안 처리할 최대 수동 일치 항목 수를 구성할 수 있습니다.

이 제한을 설정함으로써 데이터 스튜어드가 처리해야 하는 수동 통합 수를 제한할 수 있습니다. 이 제한에 도달하면 일치 프로세스가 수동 통합을 위해 준비된 레코드 수가 줄어들 때까지 실행을 중지합니다.

외부 일치 작업

Informatica MDM Hub에서는 새 데이터를 실제로 기존의 기본 개체에 로드하지 않고 기본 개체와 일치시키는 방법을 제공합니다. 전체 일치 작업을 실행하는 대신 외부 일치 작업을 실행하여 일치 항목을 테스트하고 결과를 검토할 수 있습니다. 외부 일치 작업은 유사 항목 일치 규칙과 정확히 일치 규칙을 모두 처리할 수 있으며, 유사 항목 일치 기본 개체 및 정확히 일치하는 기본 개체와 함께 사용할 수 있습니다.

분산 처리 서버

Informatica MDM Hub 구현에서는 여러 처리 서버를 병렬로 실행하여 일치 프로세스의 처리량을 늘릴 수 있습니다.

응용 프로그램 서버 또는 데이터베이스 서버 실패 처리

일치 일괄 처리 크기가 큰 매우 큰 일치 작업을 실행할 때 응용 프로그램 서버나 데이터베이스가 실패하는 경우 전체 일괄 처리를 다시 실행해야 합니다. 일치 일괄 처리는 한 단위이며, 중분 검사점이 없습니다. 데이터베이스 또는 응용 프로그램 서버가 실패할 수 있다고 판단되는 경우 이 문제를 해결하려면 일치 일괄 처리 크기를 더 작은 크기로 설정하여 일치 일괄 처리를 다시 실행하는 데 걸리는 시간을 줄이십시오.

통합 프로세스

이 섹션에서는 Informatica MDM Hub의 통합 프로세스와 관련된 개념 및 태스크에 대해 설명합니다.

통합 프로세스 정보

일치 프로세스에서 일치 쌍을 식별한 후 통합 프로세스에서는 일치된 레코드의 데이터를 단일 마스터 레코드에 통합합니다.

다음 그림에서는 서로 다른 세 개의 소스 시스템에서 단일 마스터 레코드로 통합되는 레코드의 셀 데이터를 보여줍니다.

Informatica MDM Hub	마스터 ID	이름	MN	성	주소	도시	상태	우편 번호
	M-0001	Abel	Noel	Willan	161 Washington Ave.	Buffalo	NY	14263
영업	SFA_ID	이름	MN	성	주소	도시	상태	우편 번호
	12345	Abel		Willan	161 Washington Ave.	Buffalo	NY	14263
계정	Cust_ID	이름	MN	성	주소	도시	상태	우편 번호
	502068	Abel	Noel	Willan	161 Washington Ave.	Buffalo	NY	14263
마케팅	Target_ID	이름	MN	성	주소	도시	상태	우편 번호
	willan05	Abel	N	Willan	Elm & Carliston Streets	Buffalo	NY	14263

레코드 자동 또는 수동 통합

일치 규칙은 일치 테이블의 AUTOMERGE_IND 열을 설정하여 일치된 레코드를 자동으로 통합할지 수동으로 통합할지를 지정합니다.

- 수동 통합 플래그가 지정된 레코드는 데이터 스튜어드가 병합 관리자 도구를 사용하여 검토합니다. 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.
- 자동 통합 플래그가 지정된 레코드는 자동으로 병합됩니다. 또는 사용자가 기본 개체에 대해 자동 일치 및 병합 작업을 실행할 수 있습니다. 이 작업에서는 기본 개체의 모든 레코드에 대해 일치 여부가 확인되거나 수동으로 통합할 수 있는 최대 레코드 수에 도달할 때까지 일치 후 자동 병합 작업을 반복적으로 호출합니다.

추적 가능성

Informatica MDM Hub의 목적은 모든 중복 데이터를 확인하여 제거하고, 이를 통합된 단일 레코드로 병합하거나 연결하면서 추적 가능성을 완전하게 유지 관리하는 것입니다.

추적 가능성은 통합된 레코드에 데이터를 제공하는 시스템 및 해당 시스템의 레코드에 대한 정보를 유지 관리하는 Informatica MDM Hub 기능입니다. Informatica MDM Hub에서는 교차 참조 및 기록 테이블을 사용하여 추적 가능성을 유지 관리합니다.

통합 프로세스를 위한 키 구성 설정

다음과 같은 구성 가능한 설정이 통합 프로세스에 영향을 미칩니다.

옵션	설명
기본 개체 스타일	통합 프로세스가 병합을 사용하는지, 아니면 연결을 사용하는지 결정합니다.
변경할 수 없는 소스	소스 시스템을 변경할 수 없는 시스템으로 지정할 수 있습니다. 이 경우 해당 소스 시스템의 레코드가 고유한 레코드로 인식되며, 해당 소스의 레코드는 완전히 통합된 후 더 이상 변경되지 않습니다.
고유 시스템	소스 시스템을 고유 시스템으로 지정할 수 있습니다. 이 경우 해당 시스템의 데이터는 통합되지 않은 상태로 기본 개체에 삽입됩니다.
하위 기본 개체에 대한 계단식 병합 해제	하위 기본 개체에 대한 계단식 병합 해제를 활성화할 수 있으며 상위 기본 개체의 레코드가 병합 해제될 때 수행될 작업을 지정할 수 있습니다.
상위 항목 병합 시 하위 기본 개체 레코드	상위-하위 관계인 두 기본 개체에서 하위 기본 개체에 이 옵션이 활성화되어 있으면 상위 레코드가 통합될 경우 하위 레코드가 일치 프로세스를 위해 다시 제출됩니다.

통합 옵션

일치된 레코드를 병합하여 통합할 수 있습니다. 병합(실제 통합) - 일치된 레코드를 조합하여 기본 개체를 업데이트합니다. 병합은 병합 스타일 기본 개체에 대해 수행됩니다.

기본적으로 기본 개체 통합은 실제로 저장되므로 병합이 기본 동작입니다.

병합 시 기본 개체 테이블에 있는 둘 이상의 레코드가 조합됩니다. 병합은 두 레코드 간의 유사점에 따라 자동으로 수행되거나 수동으로 수행됩니다.

- 명확하게 일치하는 레코드는 자동으로 병합됩니다(자동 병합 프로세스).
- 유사하지만 명확하게 일치하지는 않는 레코드는 데이터 스튜어드가 병합 관리자 도구에서 수동으로 검토할 수 있도록 대기됩니다(수동 병합 프로세스). 데이터 스튜어드는 일치 후보를 검사하고 병합해야 하는 일치 항목을 선택합니다. 유사한 일치 항목을 식별하기 위해 수동 병합 일치 규칙이 구성됩니다.
- Informatica MDM Hub에서 다른 모든 레코드는 데이터 스튜어드가 병합 관리자 도구에서 수동으로 검토할 수 있도록 대기됩니다.

일치 규칙은 자동 병합을 위해 명확하게 일치하는 항목을 식별하고 수동 병합을 위해 유사하게 일치하는 항목을 식별하도록 구성됩니다.

Informatica MDM Hub에서 이러한 레코드의 상태를 자동으로 "통합됨"으로 변경하여 데이터 스튜어드의 대기열에서 제거할 수 있도록 하려면 **일치되지 않은 모든 행을 고유 행으로 허용** 확인란을 선택하면 됩니다.

BVT(최선의 진실, Best Version of the Truth)

기본 개체에서 BVT(최선의 진실, Best Version of the Truth)는 소스 레코드에서 제공된 최적의 데이터 셀과 통합된 레코드입니다.

기본 개체 레코드가 BVT 레코드이며, 이 레코드는 해당하는 소스 레코드에서 가장 신뢰할 수 있는 셀 값과 통합하는 방법으로 만들어집니다.

통합 및 워크플로우 통합

상태 사용 기본 개체의 경우 통합 동작은 기본 개체에 있는 레코드의 현재 시스템 상태에 따라 영향을 받습니다. 예를 들어 **ACTIVE** 상태의 레코드만 자동으로 통합될 수 있으며 **PENDING** 또는 **DELETED** 시스템 상태의 레코드는 통합될 수 없습니다. 통합 중 시스템 상태의 의미를 이해하려면 다음 항목을 참조하십시오.

- 장 11, “상태 관리 및 BPM 워크플로우 도구” 페이지 166, 특히 “상태 전환” 페이지 169 및 “레코드 상태와 기본 개체 레코드 값의 존속” 페이지 170
- *Multidomain MDM 데이터 스투어드 가이드*의 “데이터 통합”

게시 프로세스

Informatica MDM Hub는 Hub 저장소의 데이터 변경에 대한 XML 메시지를 생성하고 이러한 메시지를 아웃바운드 JMS(Java Messaging System) 메시지 대기열에 게시하여 외부 시스템과 통합됩니다.

Informatica MDM Hub 구현에서는 정해진 비즈니스 및 기술 요구 사항을 지원하기 위해 게시 프로세스를 사용합니다. 다른 외부 시스템, 프로세스 또는 응용 프로그램에서는 JMS 메시지 대기열을 수신하고 XML 메시지를 검색하여 적절히 처리합니다.

모든 조직에서 이 기능을 사용하는 것은 아니며, Informatica MDM Hub 구현에서 이 기능을 사용하는 것은 선택적입니다.

Informatica MDM Hub가 지원하는 JMS 모델

Informatica MDM Hub는 다음과 같은 JMS 모델을 지원합니다.

점 대 점

대상 외부 시스템의 특정 대상입니다.

게시/구독

ESB(Enterprise Service Bus)까지는 점 대 점 모델이고, ESB에서 다른 시스템까지는 게시/구독 모델입니다.

조정된 데이터의 아웃바운드 분포에 대한 게시 프로세스

게시 프로세스는 Informatica MDM Hub의 기본 아웃바운드 흐름입니다. 랜드, 준비, 로드, 일치 및 통합 프로세스는 모두 조정과 연결되며 조정은 Informatica MDM Hub의 기본 인바운드 흐름입니다.

게시 프로세스는 분포에 대한 기본 Informatica MDM Hub 아웃바운드 흐름에 속합니다. 조정 후 Informatica MDM Hub는 마스터 레코드 데이터를 다른 응용 프로그램이나 다른 데이터베이스에 배포할 수 있습니다.

게시 프로세스 메시지 트리거

Informatica MDM Hub 메시지 트리거가 게시 프로세스를 실행합니다.

MDM Hub 저장소에서 데이터가 변경되면 Informatica MDM Hub에서 메시지 트리거를 생성합니다. 메시지 트리거는 Informatica MDM Hub가 Java Message Service 메시지 대기열에 게시하는 XML 메시지를 생성합니다. XML 메시지가 수신되면 게시 프로세스가 실행됩니다.

아웃바운드 JMS 메시지 대기열

Informatica MDM Hub에서는 아웃바운드 메시지 대기열을 통신 채널로 사용하여 데이터 변경 내용을 다시 외부 시스템에 제공합니다.

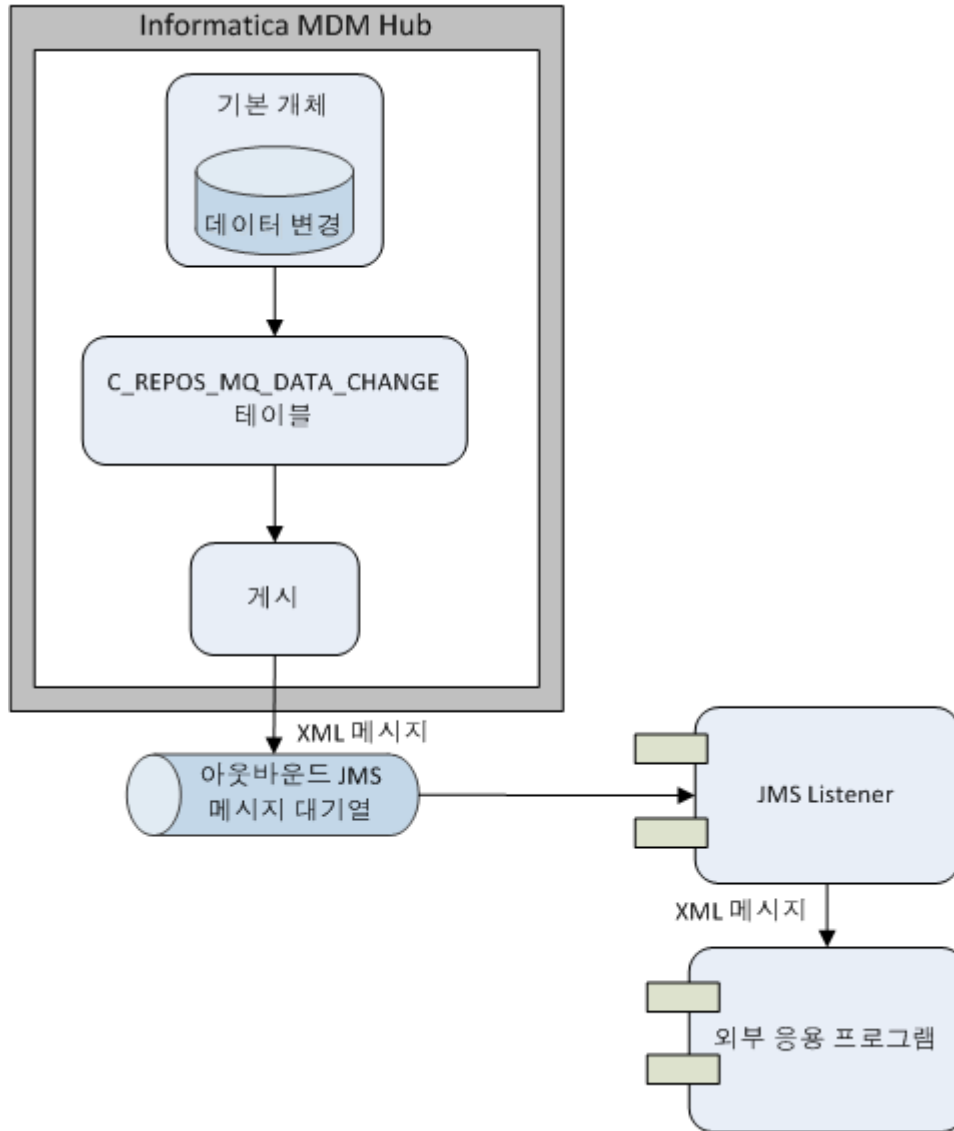
Informatica는 포함 메시지 대기열을 지원하며 이 대기열에서는 응용 프로그램 서버와 함께 제공되는 JMS 공급자가 사용됩니다. 포함 메시지 대기열은 JMS 대기열의 이름과 **ConnectionFactory**의 JNDI 이름을 사용하여 연결합니다. 이때 JNDI 이름은 응용 프로그램 서버에 의해 설정된 이름이어야 합니다. Hub 콘솔을 통해 응용 프로그램 서버 환경에서 이미 구성된 메시지 대기열과 메시지 대기열 서버를 등록할 수 있습니다.

ORS 관련 XML 메시지 스키마

XML 메시지는 공통 XML 스키마(**siperian-mrm-events.xsd**)를 기준으로 하는 ORS 관련 스키마 파일(<ors-name>-siperian-mrm-event.xsd)을 사용하여 생성됩니다. 이 ORS 관련 스키마를 생성하려면 JMS 이벤트 스키마 관리자를 사용합니다. 이 태스크는 게시 프로세스를 설정하는 데 있어 필수 태스크입니다.

게시 프로세스의 런타임 흐름

다음 그림은 게시 프로세스의 런타임 흐름을 보여 줍니다.



이 시나리오에서:

1. 일괄 로드나 실시간 SIF API 요청(SIF Put 또는 `cleanse_put` 요청)으로 인해 기본 개체에 대해 삽입이나 업데이트가 수행될 수 있습니다.
C_REPOS_MQ_DATA_CHANGE 테이블로 이동되는 데이터를 제어하도록 메시지 규칙을 구성할 수 있습니다.
2. Hub 서버는 정기적으로 C_REPOS_MQ_DATA_CHANGE 테이블의 데이터를 폴링합니다.
3. 전송되지 않은 데이터의 경우 Hub 서버는 데이터를 바탕으로 XML 메시지를 생성하고 메시지 대기열에 대해 구성된 아웃바운드 대기열에 이 메시지를 전송합니다.
4. 아웃바운드 대기열에서 메시지를 검색하고 처리하는 작업은 외부 응용 프로그램에 의해 수행됩니다.

제 16 장

랜드 프로세스 구성

이 장에 포함된 항목:

- [랜드 프로세스 구성 개요, 283](#)
- [소스 시스템 구성, 283](#)
- [랜딩 테이블 구성, 286](#)

랜드 프로세스 구성 개요

랜드 프로세스를 구성하려면 Hub 콘솔에서 다음 태스크를 수행합니다.

- [“소스 시스템 구성” 페이지 283](#)
- [“랜딩 테이블 구성” 페이지 286](#)

소스 시스템 구성

이 섹션에서는 Informatica MDM Hub 구현의 소스 시스템을 정의하는 방법에 대해 설명합니다.

소스 시스템 정보

소스 시스템은 Informatica MDM Hub에 데이터를 제공하는 외부 응용 프로그램 또는 시스템입니다. 다양한 소스 시스템에서의 입력을 관리하려면 Informatica MDM Hub에 각 소스 시스템의 고유한 내부 이름이 있어야 합니다. 모델 작업 영역의 시스템 및 트러스트 도구를 사용하여 Informatica MDM Hub 구현의 소스 시스템을 정의할 수 있습니다.

소스 시스템에 대한 트러스트 구성

여러 소스 시스템에서 기본 개체의 동일한 열에 데이터를 제공하는 경우 열 단위로 *트러스트*를 구성하면 해당 열에 대해 다른 소스 시스템과 비교하여 상대적으로 신뢰도가 더 높은 데이터 공급자인 소스 시스템을 지정할 수 있습니다. 트러스트는 두 레코드가 통합될 때 존속되는 레코드를 결정하고 소스 시스템의 업데이트가 "BVT(최선의 진실, Best Version of the Truth)" 레코드를 업데이트하기에 충분히 신뢰할 수 있는지 여부를 확인하는 데 사용됩니다.

관리 소스 시스템

*Multidomain MDM 데이터 스튜어드 가이드*에 설명된 대로 **Informatica MDM Hub**는 데이터 관리자 또는 병합 관리자 도구의 수동 트러스트 재정의 및 데이터 편집에 관리 소스 시스템을 사용합니다.

이 관리 소스 시스템은 트러스트가 활성화된 열에 데이터를 제공할 수 있습니다. 관리 소스 시스템의 이름은 기본적으로 "관리"로 지정되지만 필요한 경우 이름을 변경할 수 있습니다.

상태 관리 재정의 시스템

상태 관리 재정의 시스템은 다른 모든 소스 시스템의 레코드 상태를 재정의하고 레코드 상태를 "삭제됨"으로 표시할 수 있는 소스 시스템입니다. 일부 교차 참조에서 "활성" 상태로 나타나는 레코드도 "삭제됨" 상태로 지정할 수 있습니다.

여러 소스 시스템에서 기본 개체 레코드에 데이터를 제공하며 데이터를 제공하는 하나 이상의 레코드가 "활성" 상태에 있는 경우 **MDM Hub**에서는 상태 관리 재정의 시스템에서 "삭제됨" 상태의 레코드를 삽입합니다. 이때 레코드의 전체 상태가 "삭제됨"으로 설정됩니다.

하나의 소스 시스템만 상태 관리 재정의 시스템으로 설정할 수 있습니다. 모델 작업 영역의 시스템 및 트러스트 도구를 사용하여 소스 시스템을 상태 관리 재정의 시스템으로 활성화할 수 있습니다.

참고: 상태 관리 재정의 시스템은 일괄 작업에 적용할 수 없습니다.

Informatica 시스템 리포지토리 테이블

시스템 및 트러스트 도구에서 정의하는 소스 시스템은 특수 공용 **Informatica MDM Hub** 리포지토리 테이블 (**C_REPOS_SYSTEM**, MDM 시스템의 표시 이름 사용)에 저장됩니다. 이 테이블은 시스템 테이블 표시 옵션이 선택된 경우 스키마 관리자에 표시됩니다. **C_REPOS_SYSTEM**도 패키지에 사용할 수 있습니다.

참고: **C_REPOS_SYSTEM** 테이블에는 **Informatica MDM Hub** 메타데이터가 포함되어 있습니다. 다른 **Informatica MDM Hub** 시스템 테이블과 마찬가지로 **C_REPOS_SYSTEM** 테이블의 구조나 데이터를 변경해서는 안 됩니다. 변경할 경우 **Informatica MDM Hub**가 예기치 않게 동작하고 데이터가 손실될 수 있습니다.

시스템 및 트러스트 도구 시작

시스템 및 트러스트 도구를 시작하려면

- Hub 콘솔에서 모델 작업 영역을 확장한 후 **시스템 및 트러스트**를 클릭합니다.

Hub 콘솔에 시스템 및 트러스트 도구가 표시됩니다.

시스템 및 트러스트 도구에는 다음과 같은 창이 표시됩니다.

창	설명
탐색	시스템 관리 소스 시스템을 비롯하여 Informatica MDM Hub에 데이터를 제공하는 모든 소스 시스템 목록입니다. 트러스트 표시할 트리를 확장합니다. <ul style="list-style-type: none">- 트러스트가 활성화된 하나 이상의 열을 포함한 기본 개체- 트러스트가 활성화된 열만
속성	선택한 소스 시스템의 속성입니다. 기본 개체 열이 선택된 경우 기본 개체 열의 트러스트 설정입니다.

소스 시스템 속성

MDM Hub에 제공해야 하는 소스 시스템을 정의합니다. 소스 시스템 정의는 MDM Hub 외부에서 이루어집니다. 모델 작업 영역의 시스템 및 트러스트 도구에서 MDM Hub에 대한 소스 시스템을 정의합니다.

다음 테이블에서는 MDM Hub의 소스 시스템 정의의 속성을 설명합니다.

속성	설명
이름	소스 시스템에 대해 고유한 설명 이름입니다.
기본 키	소스 시스템에 대해 고유한 식별자로, MDM Hub이 소스 시스템에 대한 기본 키 값에 접두사로 추가합니다. 이 값은 읽기 전용입니다.
상태 관리 재정의의 시스템	MDM Hub에 제공하는 다른 모든 소스 시스템의 레코드 상태를 재정의할지 지정합니다. 속성을 활성화하여 다른 모든 소스 시스템의 레코드 상태를 재정의합니다. MDM Hub에 제공하는 다른 모든 소스 시스템의 레코드 상태를 재정의하지 않으려는 경우 속성을 비활성화합니다. 기본적으로 이 속성은 비활성화되어 있습니다.
설명	선택 사항입니다. 소스 시스템에 대한 설명입니다.

소스 시스템 추가

시스템 및 트러스트 도구를 사용하려면 Informatica MDM Hub 구현에 데이터를 제공할 각 소스 시스템을 정의합니다.

참고: 소스 시스템의 기본 키는 대문자로 된 소스 시스템 이름의 처음 14자입니다. 기본 키는 C_REPOS_SYSTEM 리포지토리 테이블의 ROWID_SYSTEM 필드에 저장됩니다.

1. 시스템 및 트러스트 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 소스 시스템 목록을 마우스 오른쪽 단추로 클릭하고 **시스템 추가**를 선택합니다.
시스템 및 트러스트 도구에 새 시스템 대화 상자가 표시됩니다.
4. 소스 시스템 속성을 지정합니다.
5. **확인**을 클릭합니다.
시스템 및 트러스트 도구에 소스 시스템이 목록으로 표시됩니다.

소스 시스템 속성 편집

관리 시스템을 비롯한 모든 소스 시스템의 이름을 바꿀 수 있습니다. 소스 시스템의 이름을 바꾸면 이름이 Hub 콘솔의 컨텍스트 내에서 변경됩니다.

참고: 이 소스 시스템이 Informatica MDM Hub 구현에 데이터를 제공한 경우 Informatica MDM Hub에서 이 소스 시스템의 데이터에 대한 연계를 계속 추적합니다.

1. 시스템 및 트러스트 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 소스 시스템 목록에서 소스 시스템을 선택합니다.
화면이 새로 고쳐지고 소스 시스템 속성이 표시됩니다. 편집 가능한 필드 옆에 **편집** 아이콘이 나타납니다.
4. 속성을 편집하려면 **편집** 아이콘을 클릭하고 편집합니다.
5. 필요한 경우 트러스트 설정을 편집합니다.
6. **저장**을 클릭합니다.

소스 시스템 제거

소스 시스템이 준비 테이블에 데이터를 제공하기 전에 소스 시스템을 제거할 수 있습니다. 소스 시스템을 제거하면 소스 시스템 정의 및 연결된 모든 메타데이터가 삭제됩니다. Informatica MDM Hub 외부에 미치는 영향은 없습니다.

참고: 다음 소스 시스템은 제거할 수 없습니다.

- 관리 시스템.
 - 기본 개체의 소스로 구성된 모든 소스 시스템. 기본 개체에 연결된 준비 테이블은 소스 시스템을 가리킵니다.
 - 준비 테이블에 데이터를 제공한 모든 소스 시스템. 즉, 준비 프로세스에 따라 준비 테이블에 데이터가 채워집니다.
1. 시스템 및 트러스트 도구를 시작합니다.
 2. 쓰기 잠금을 획득합니다.
 3. 소스 시스템 목록에서 소스 시스템을 마우스 오른쪽 단추로 클릭하고 **시스템 제거**를 선택합니다.
 4. 작업을 확인하는 메시지가 표시되면 **예**를 클릭합니다.
- 시스템 및 트러스트 도구가 목록에서 소스 시스템을 제거합니다.

랜딩 테이블 구성

이 섹션에서는 Informatica MDM Hub 구현에서 랜딩 테이블을 구성하는 방법에 대해 설명합니다.

랜딩 테이블 정보

랜딩 테이블은 소스 시스템에서 Informatica MDM Hub로 데이터를 이동하는 과정에서 중간 저장소의 역할을 합니다. 실제로 랜딩 테이블은 소스 시스템에서 Hub 저장소로 "데이터가 랜딩되는 곳"입니다. 모델 작업 영역의 스키마 관리자를 사용하여 랜딩 테이블을 정의할 수 있습니다.

소스 시스템이 랜딩 테이블에 데이터를 채우는 방식은 Informatica MDM Hub에 대해 완전히 외부적입니다. 다양한 소스 시스템에서 랜딩 테이블에 데이터를 수집하는 데 사용하는 데이터 모델도 Informatica MDM Hub에 대해 완전히 외부적입니다. 한 개의 소스 시스템에서 여러 랜딩 테이블을 채울 수 있습니다. 단일 랜딩 테이블은 여러 소스 시스템에서 데이터를 수신할 수 있습니다. 사용하는 데이터 모델은 전적으로 특정 구현 요구 사항에 따라 달라집니다.

그러나 Informatica MDM Hub 내에서는 랜딩 테이블이 준비 테이블에 매핑됩니다. 기본 개체에 데이터를 제공하는 소스 시스템은 랜딩 테이블에 매핑된 준비 테이블에서 식별됩니다. 로드 프로세스 중 Informatica MDM Hub에서는 랜딩 테이블의 데이터를 대상 준비 테이블에 복사하고, 소스 시스템 ID로 데이터에 태그를 지정하며, 필요할 경우 프로세스 중에 데이터를 정리합니다. 한 개의 랜딩 테이블이 한 개 이상의 준비 테이블에 매핑될 수 있습니다. 한 개의 준비 테이블은 한 개의 랜딩 테이블에만 매핑됩니다.

랜딩 테이블은 Informatica MDM Hub 외부에서 일괄 처리 또는 실시간 방법을 사용하여 채워집니다. 랜딩 테이블이 채워진 후 준비 프로세스에서는 랜딩 테이블의 데이터를 가져오고, 필요한 경우 데이터를 추가로 정리한 다음, 적절한 준비 테이블을 채웁니다.

랜딩 테이블 열

랜딩 테이블은 사용자가 추가하는 열인 사용자 정의 열로 구성됩니다. 또한 랜딩 테이블에는 랜딩 테이블 레코드를 고유하게 식별하는 SRC_ROWID 시스템 열도 포함되어 있습니다.

레코드가 준비 프로세스에 의해 로드되어 있는 경우 SRC_ROWID 값으로 편리하게 추적할 수 있습니다. SRC_ROWID 열의 값은 고유해야 합니다. SRC_ROWID 열에 중복 값이 있으면 준비 작업에 실패합니다. SRC_ROWID 열에 중복 값이 있는 경우 Informatica 글로벌 고객 지원 센터에 문의하십시오.

참고: 소스 시스템 테이블에 다중 열 키가 있을 경우 이들 열을 연결하여 기본 키 열의 단일한 고유 VARCHAR 값을 생성합니다.

랜딩 테이블 속성

랜딩 테이블에는 다음과 같은 속성이 있습니다.

속성	설명
항목 유형	추가하려는 테이블의 유형입니다. 랜딩 테이블 을 선택합니다.
표시 이름	이 랜딩 테이블의 이름으로, Hub 콘솔에 표시됩니다.
실제 이름	데이터베이스에서 랜딩 테이블의 실제 이름입니다. Informatica MDM Hub에서는 사용자가 입력한 표시 이름을 기준으로 랜딩 테이블의 실제 이름을 제안합니다.
데이터 테이블 스페이스	이 랜딩 테이블의 데이터 테이블스페이스 이름입니다. 자세한 내용은 <i>Multidomain MDM 설치 가이드</i> 를 참조하십시오.
인덱스 테이블 스페이스	이 랜딩 테이블의 인덱스 테이블스페이스 이름입니다. 자세한 내용은 <i>Multidomain MDM 설치 가이드</i> 를 참조하십시오.
설명	이 랜딩 테이블에 대한 설명입니다.
생성 날짜	이 랜딩 테이블이 생성된 날짜와 시간입니다.
전체 데이터 집합 포함	이 랜딩 테이블이 소스 시스템의 전체 데이터 집합을 포함하는지, 아니면 업데이트만 포함하는지를 지정합니다. <ul style="list-style-type: none"> - 선택되어 있으면(기본값) 이 랜딩 테이블에 소스 시스템의 전체 데이터 집합이 포함된다는 의미입니다(예: 초기 데이터 로드의 경우). 이 확인란이 활성화되어 있으면 Informatica MDM Hub의 델타 검색 기능을 구성할 수 있습니다. 즉, 준비 프로세스 중에 변경된 레코드만 준비 테이블에 복사됩니다. - 선택되어 있지 않으면 이 랜딩 테이블에 소스 시스템의 변경된 데이터만 포함된다는 의미입니다(예: 증분 로드의 경우). 이 경우 Informatica MDM Hub에서는 랜딩 테이블을 채우기 전에 변경되지 않은 레코드가 필터링되었다고 가정합니다. 따라서 준비 프로세스에서 랜딩 테이블의 모든 레코드를 준비 테이블에 바로 삽입합니다. 이 확인란이 비활성화되어 있으면 Informatica MDM Hub의 델타 검색 기능을 사용할 수 없습니다. 참고: 소스 시스템 속성을 편집할 때만 이 속성을 변경할 수 있습니다.

랜딩 테이블 추가

랜딩 테이블을 추가할 수 있습니다.

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. **랜딩 테이블** 노드를 선택합니다.
4. 랜딩 테이블 노드를 마우스 오른쪽 단추로 클릭하고 **항목 추가**를 선택합니다.
스키마 관리자에 테이블 추가 대화 상자가 표시됩니다.
5. 이 새 랜딩 테이블의 속성을 지정합니다.

6. **확인**을 클릭합니다.

연산 참조 저장소(연산 참조 저장소)에 새 랜딩 테이블과 지원 테이블이 생성되고 스키마 트리에서 새 랜딩 테이블이 추가됩니다.

7. 랜딩 테이블의 열을 구성합니다.

8. 소스 시스템에서 변경된 데이터만 포함하도록 랜딩 테이블을 구성하려면 랜딩 테이블 속성("전체 데이터 집합 포함")을 편집합니다.

랜딩 테이블 속성 편집

랜딩 테이블의 속성을 편집하려면

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 편집할 랜딩 테이블을 선택합니다.
스키마 관리자에 선택한 테이블의 랜딩 테이블 ID 창이 표시됩니다.
4. 원하는 랜딩 테이블 속성을 변경합니다.
5. **저장** 단추를 클릭하여 변경 내용을 저장합니다.
6. 원하는 경우 랜딩 테이블의 열 구성을 변경합니다.

랜딩 테이블 제거

랜딩 테이블을 제거하려면

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 **랜딩 테이블** 노드를 확장합니다.
4. 제거할 랜딩 테이블을 마우스 오른쪽 단추로 클릭한 후 **제거**를 선택합니다.
스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.
5. **예**를 선택합니다.
스키마 관리자가 데이터베이스에서 랜딩 테이블을 삭제하고 이 랜딩 테이블과 모든 준비 테이블 간 매핑을 삭제하며(준비 테이블은 삭제되지 않음) 스키마 트리에서 삭제된 랜딩 테이블을 제거합니다.

제 17 장

MDM Hub 준비

이 장에 포함된 항목:

- [MDM Hub 준비 개요, 289](#)
- [MDM Hub 준비 테이블, 290](#)
- [MDM Hub 준비 선행 조건, 292](#)
- [준비 테이블 추가, 292](#)
- [랜딩 테이블과 준비 테이블 간의 열 매핑, 295](#)
- [감사 내역 및 델타 검색 구성, 304](#)
- [준비 테이블 관리, 307](#)
- [매핑 관리, 309](#)

MDM Hub 준비 개요

MDM Hub 준비에서는 랜딩 테이블의 소스 데이터를 기본 개체와 연결된 준비 테이블로 옮깁니다. MDM Hub 준비를 수행하려면 랜딩 테이블 및 준비 테이블을 생성해야 합니다. MDM Hub 준비를 수행하기 전에 랜딩 테이블에 데이터를 로드해야 합니다.

참고: MDM Hub에서는 단일 랜딩 테이블에서 여러 준비 테이블로 데이터를 옮길 수 있습니다. 그러나 각 준비 테이블은 하나의 랜딩 테이블에서만 데이터를 수신합니다.

준비 작업을 실행하기 전에 랜딩 테이블 및 준비 테이블 간 매핑을 정의합니다. 매핑은 랜딩 테이블의 소스 열을 준비 테이블의 대상 열과 연결합니다. 준비 작업을 실행하면 MDM Hub에서는 랜딩 테이블 열에서 준비 테이블 열로의 매핑을 기준으로 데이터를 이동합니다.

준비 프로세스 중에 MDM Hub는 한 번에 250개 레코드가 포함된 단일 블록을 처리합니다. 블록의 레코드에 문제가 있는 경우 MDM Hub는 레코드를 거부 테이블로 이동합니다. 셀 값이 너무 길거나 레코드 업데이트 날짜가 오늘 날짜보다 이후이기 때문에 레코드가 거부될 수 있습니다. MDM Hub에서 거부된 레코드가 이동한 후 MDM Hub는 블록의 나머지 레코드 처리를 중지하고 다른 블록으로 이동합니다. 준비 프로세스가 완료되면 작업을 다시 실행하십시오. 처리되지 않은 레코드가 다시 선택되고 처리됩니다.

준비 프로세스 동안 정리 작업을 수행하려는 경우 MDM Hub 정리 함수를 사용합니다. 매핑에서 데이터 정리를 구성합니다. MDM Hub 준비를 수행할 때 델타 검색 및 감사 내역을 설정할 수 있습니다. 준비 프로세스에서 로드 프로세스를 진행할 수 있도록 데이터를 준비합니다.

MDM Hub 준비 테이블

MDM Hub 준비 작업을 실행하면 MDM Hub에서 랜딩 테이블의 데이터를 MDM Hub 준비 테이블로 로드합니다. 준비 테이블의 구조는 통합된 데이터를 포함할 대상 기본 개체의 구조를 기반으로 합니다. 모델 작업 영역의 스키마 도구를 사용하여 준비 테이블을 구성합니다.

다음 작업을 수행하여 MDM Hub 준비 테이블을 구성합니다.

1. MDM Hub 준비 선행 조건을 완료합니다.
2. 준비 테이블을 추가합니다.
3. 랜딩 테이블과 준비 테이블 간의 열을 매핑합니다.
4. 감사 내역 및 델타 검색을 구성합니다.

MDM Hub 준비 프로세스 테이블

MDM Hub 준비 프로세스에서는 랜딩 테이블, 준비 테이블, Raw 테이블 및 거부 테이블을 사용하여 소스 시스템의 데이터를 MDM Hub으로 옮깁니다.

MDM Hub 준비 프로세스에서는 다음 MDM Hub 테이블을 사용합니다.

랜딩 테이블

처음에 소스 시스템에서 로드된 데이터를 포함하는 MDM Hub 테이블입니다. MDM Hub은 MDM Hub 준비 프로세스에 대해 랜딩 테이블의 데이터를 사용합니다.

준비 테이블

준비 프로세스 동안 MDM Hub에서 허용하는 데이터가 포함되는 MDM Hub 테이블입니다. MDM Hub 준비 프로세스 동안 데이터는 랜딩 테이블에서 준비 테이블로 옮겨집니다.

Raw 테이블

MDM Hub이 랜딩 테이블에서 보관한 Raw 데이터가 포함되는 MDM Hub 테이블입니다. 각 Raw 테이블은 준비 테이블과 연결되며 <staging table name>_RAW로 이름이 지정됩니다. 특정 횟수의 데이터 로드 후 또는 특정 시간 간격 후에 Raw 데이터를 보관하도록 MDM Hub을 구성할 수 있습니다. MDM Hub에서는 랜딩 테이블이 변경되지 않은 경우에도 null 기본 키를 삽입합니다.

거부 테이블

MDM Hub에서 거부한 레코드 및 각 거부에 대한 이유 설명이 포함되는 MDM Hub 테이블입니다. 각 거부 테이블은 준비 테이블과 연결되며 <staging table name>_REJ로 이름이 지정됩니다. 생성한 각 준비 테이블에 대한 거부 테이블을 MDM Hub에서 생성합니다. MDM Hub에서는 랜딩 테이블이 변경되지 않은 경우에도 null 기본 키를 삽입합니다.

MDM Hub은 준비 프로세스 동안 다음 이유로 레코드를 거부합니다.

- LAST_UPDATE_DATE 열에 미래 날짜 또는 null 날짜가 포함되어 있습니다.
- LAST_UPDATE_DATE 열 값이 1900보다 작습니다.
- 준비 테이블의 PKEY_SRC_OBJECT 열에 null 값이 매핑되어 있습니다.
- PKEY_SRC_OBJECT 열에 중복 항목이 포함되어 있습니다. MDM Hub에서 동일한 PKEY_SRC_OBJECT 값으로 된 레코드를 여러 개 발견할 경우 MDM Hub은 가장 큰 SRC_ROWID 값으로 된 레코드를 로드합니다. MDM Hub에서 나머지 레코드를 거부 테이블로 옮깁니다.
- HUB_STATE_IND 필드에 상태 관리가 활성화된 기본 개체에 대해 올바르지 않은 값이 포함되어 있습니다.
- 고유 열에 중복 항목이 포함되어 있습니다.

준비 테이블 속성

Hub 콘솔을 통해 준비 테이블을 생성하고 관리할 수 있습니다. 준비 테이블을 생성할 때 일부 준비 테이블 속성을 구성합니다.

다음 표에서는 준비 테이블을 생성할 때 구성하는 준비 테이블 속성을 설명합니다.

속성	설명
표시 이름	Hub 콘솔에 표시되는 준비 테이블의 이름입니다.
실제 이름	데이터베이스에 있는 준비 테이블의 이름입니다. MDM Hub에서는 사용자가 입력한 표시 이름을 기준으로 준비 테이블에 대한 실제 이름을 제안합니다.
데이터 테이블스페이스	준비 테이블에 대한 데이터 테이블스페이스의 이름입니다.
인덱스 테이블스페이스	준비 테이블에 대한 인덱스 테이블스페이스의 이름입니다.
설명	준비 테이블에 대한 설명입니다.
테이블 유형	테이블 유형입니다. 기본값은 Staging입니다.
작성 날짜	준비 테이블이 작성된 날짜입니다.
시스템	준비 테이블 데이터의 소스 시스템입니다.
소스 시스템 키 유지	MDM Hub이 소스 시스템의 키 값을 사용해야 하는지 또는 MDM Hub이 생성한 키 값을 사용해야 하는지 지정합니다. 소스 시스템의 키 값을 사용하려면 활성화합니다. MDM Hub이 생성한 키 값을 사용하려면 비활성화합니다. 기본적으로 비활성화되어 있습니다. 참고: 준비 프로세스 동안 여러 레코드에 동일한 PKEY_SRC_OBJECT가 포함된 경우 존속 레코드는 LAST_UPDATE_DATE가 최근인 레코드가 됩니다. 다른 레코드는 거부 테이블로 보내집니다.
공백 채우기	새 레코드 버전을 추가할 경우 레코드 버전의 유효 날짜 간 연속성이 유지되는지 여부를 지정합니다. 활성화할 경우 새 레코드 버전을 기본 개체에 추가할 수 있을 때 MDM Hub에서 레코드 버전의 유효 날짜 간 연속성이 유지됩니다. 활성화하지 않을 경우 MDM Hub에서는 레코드 버전의 유효 기간 간 연속성을 깨뜨리는 레코드 버전의 추가를 거부합니다. 기본적으로 비활성화되어 있습니다.
최상위 예약된 키	첫 번째 로드 후 키를 증가시킬 수입니다. 소스 시스템 키 유지 확인란을 활성화한 경우 속성이 표시됩니다.
Informatica Platform 준비	MDM Hub가 Informatica Platform 준비를 사용하는지 여부를 지정합니다. 기본적으로 비활성화되어 있습니다.
모델 리포지토리 서비스와 동기화	MDM Hub 메타데이터가 모델 리포지토리와 동기화되는지 여부를 지정합니다. 기본적으로 비활성화되어 있습니다.
셀 업데이트	준비 테이블에서 오는 수신 레코드의 값이 같을 경우 MDM Hub에서 대상 테이블의 셀을 업데이트할 수 있도록 지정합니다.
열	준비 테이블의 열입니다.

매핑 속성

랜딩 테이블과 준비 테이블 간에 매핑을 작성할 때 매핑의 속성을 구성하십시오.

다음 테이블에는 매핑 속성이 설명되어 있습니다.

필드	설명
이름	Hub 콘솔에 표시되는 매핑 이름입니다.
설명	매핑의 설명입니다.
랜딩 테이블	매핑의 소스가 되는 랜딩 테이블을 선택합니다.
준비 테이블	매핑의 대상이 되는 준비 테이블을 선택합니다.
보안 리소스	매핑에 대한 액세스를 제어할 수 있도록 매핑을 보안 리소스로 지정하려면 활성화합니다. 매핑이 보안 리소스로 지정되면 보안 리소스 도구에서 이 매핑에 대한 권한을 할당할 수 있습니다.

MDM Hub 준비 선행 조건

시작하기 전에 다음 설치 및 구성 작업을 수행합니다.

1. 데이터를 정리하는 데 필요한 외부 정리 엔진을 구성합니다.
2. 랜드 프로세스를 완료하여 데이터를 랜딩 테이블에 로드합니다.

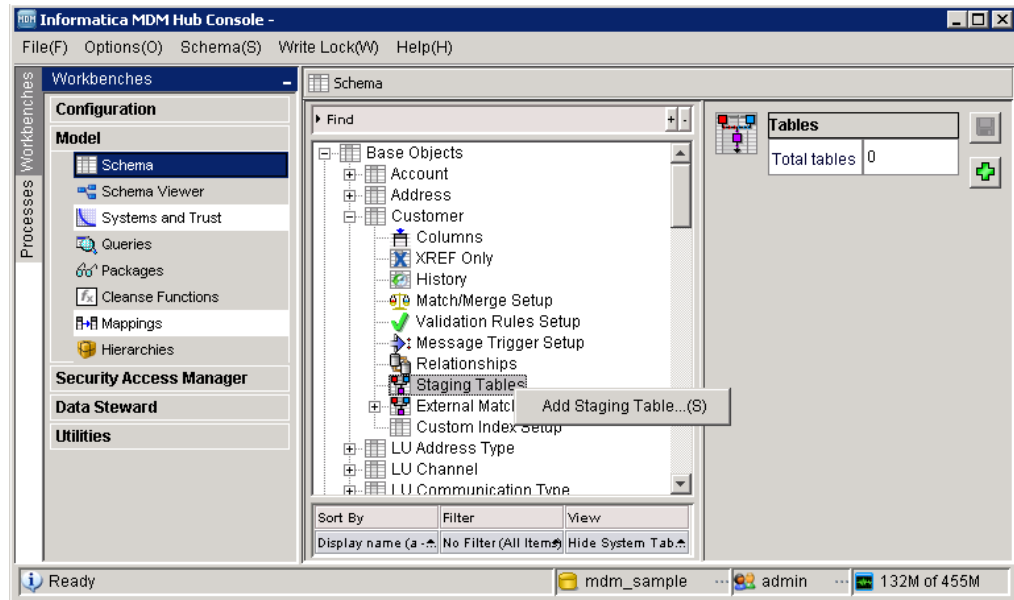
준비 테이블 추가

랜딩 테이블의 데이터를 이동할 준비 테이블을 추가합니다.

1. Hub 콘솔에서 스키마 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서, 준비 테이블을 추가할 기본 개체의 노드를 확장합니다.
4. 준비 테이블 노드를 마우스 오른쪽 단추로 클릭합니다.

준비 테이블 추가 옵션이 표시됩니다.

다음 이미지는 준비 테이블을 고객 기본 개체에 추가하는 **준비 테이블 추가** 옵션을 보여 줍니다.



5. 준비 테이블 추가를 클릭합니다.
기본 개체에 준비 추가 대화 상자가 표시됩니다.
6. 준비 테이블 속성을 지정합니다.

다음 이미지는 기본 개체 고객에 준비 추가 대화 상자(대화 상자의 표시 이름 필드가 S_CRM_CUST로 설정되어 있음)를 보여 줍니다.

Table Identity	
Display name	S_CRM_CUST
Physical name	C_S_CRM_CUST
System	SFA
Preserve Source System Keys	<input type="checkbox"/>
Data Tablespace	CMX_DATA
Index Tablespace	CMX_INDEX
Description	

Columns							
		Column	Lookup Sy...	Lookup Table	Lookup Col...	Allow Null ...	Allow Null ...
	<input type="checkbox"/>	First Name				<input type="checkbox"/>	<input type="checkbox"/>
	<input type="checkbox"/>	Last Name				<input type="checkbox"/>	<input type="checkbox"/>
	<input type="checkbox"/>	Gender Cd				<input type="checkbox"/>	<input type="checkbox"/>

☐ Cell update

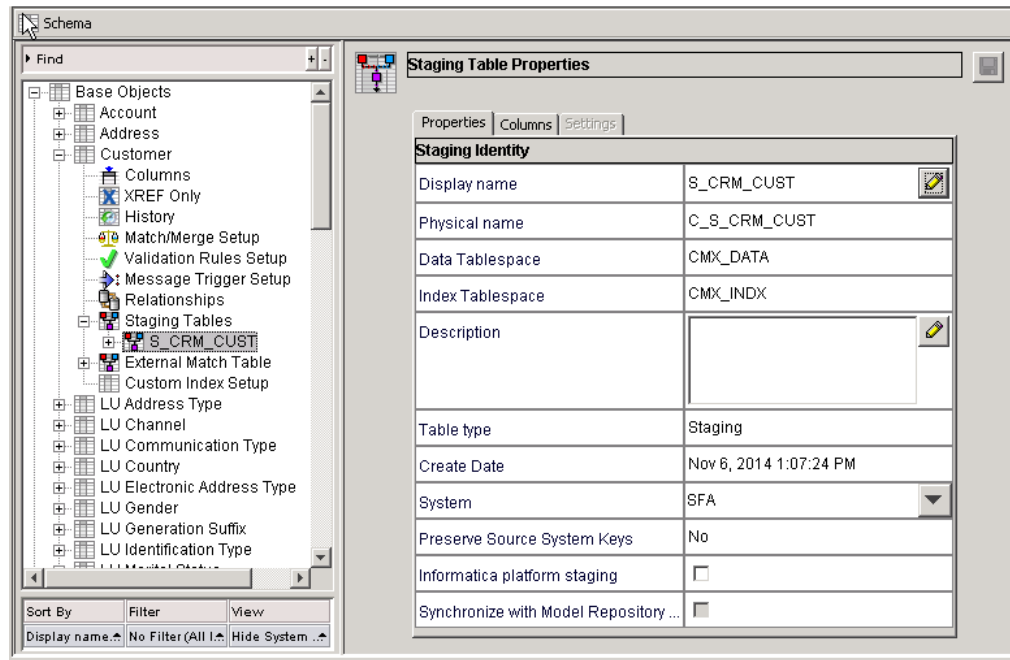
OK Cancel

7. 기본 개체의 열 목록에서 소스 시스템이 제공하는 열을 선택합니다.
모든 열 선택을 클릭하면 모든 기본 개체 열을 선택할 수 있습니다.
8. 열 속성을 지정합니다.

9. **확인**을 클릭합니다.

스키마 도구가 연산 참조 저장소에 지원 테이블과 함께 준비 테이블을 생성한 다음 탐색 트리에서 준비 테이블을 추가합니다.

다음 이미지는 탐색 트리의 S_CRM_CUST 준비 테이블을 보여 줍니다.



랜딩 테이블과 준비 테이블 간의 열 매핑

하나의 랜딩 테이블에서 하나 이상의 준비 테이블로 열을 매핑합니다. 열 매핑은 랜딩 테이블의 열에서 준비 테이블의 열로 데이터를 전송하는 방법을 정의합니다. 최소 하나 이상의 고유 기본 키를 매핑하여 소스 데이터에서 마스터 데이터까지 추적 가능성을 보장해야 합니다. 다른 소스 데이터를 마스터 데이터로 매핑할 수도 있습니다.

각각의 열 매핑은 입력과 출력을 갖습니다. 랜딩 테이블의 열이 입력입니다. 준비 테이블의 열이 출력입니다. 데이터에 대해 동작하는 정리 함수를 통해 입력을 전달할 수 있습니다. 예를 들어 데이터를 표준화하거나 확인하거나 집계할 수 있습니다. 또한 보안 리소스 또는 개인 리소스로 열 매핑을 정의할 수 있습니다.

열 매핑을 구성한 후 준비 테이블을 소스 데이터로 채우려면 준비 작업을 실행합니다.

데이터는 정리되거나, 변경되지 않은 채로 통과됨

준비 테이블에서 데이터의 각 열에 대해 다음 두 방법 중 하나로 랜딩 열의 데이터를 가져옵니다.

복사 방법	설명
통과	Informatica MDM Hub에서 데이터를 변경하지 않고 있는 그대로 복사합니다. 랜딩 테이블의 열에서 직접 데이터를 가져옵니다.
정리	Informatica MDM Hub에서 정리 함수를 사용하여 데이터를 표준화하고 확인합니다. 정리 함수의 출력은 준비 테이블의 대상 열에 대한 입력으로 사용됩니다.

참고: 준비 테이블에서 랜딩 테이블의 모든 열을 사용하거나 정리 함수의 모든 출력 문자열을 사용할 필요는 없습니다. 동일한 랜딩 테이블에서 여러 준비 테이블에 대한 입력을 제공할 수 있으며, 동일한 정리 함수를 여러 랜딩 테이블의 여러 열에서 재사용할 수 있습니다.

분해 및 집계

정리 함수를 사용하여 데이터를 분해하고 집계할 수도 있습니다.

데이터를 분해하는 정리 함수

분해 정리 함수는 필드 하나의 데이터를 더 작은 단위로 나눈 후 각 단위를 준비 테이블의 여러 열에 할당합니다.

예를 들어 인사말 필드를 개별 **name-type** 필드 5개로 나누려고 합니다. 이 경우 분해 정리 함수에는 입력 문자열 하나와 출력 문자열 다섯 개가 있습니다. 매핑에서는 입력 문자열을 정리 함수에 매핑하고 출력 문자열 각각을 준비 테이블의 대상 열에 매핑합니다.

데이터를 집계하는 정리 함수

집계 정리 함수는 여러 필드에서 데이터를 집계한 후, 집계된 데이터를 준비 테이블의 단일 열에 할당합니다.

예를 들어 **name-type** 필드 5개를 인사말 필드 하나로 집계하려고 합니다. 이 경우 집계 정리 함수에는 입력 문자열 다섯 개와 출력 문자열 한 개가 있습니다. 매핑에서는 입력 문자열을 정리 함수에 매핑하고 출력 문자열 각각을 준비 테이블의 대상 열에 매핑합니다.

Informatica Data Quality 일괄 작업 및 열 매핑

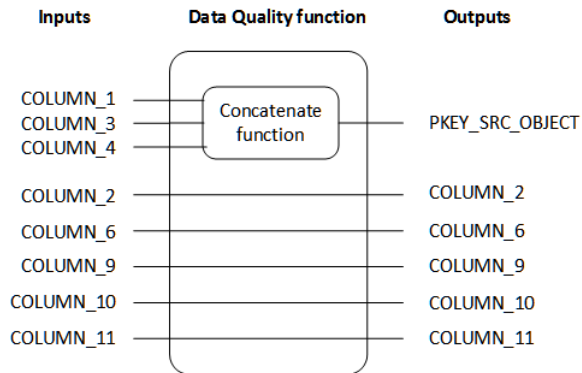
Informatica Data Quality 일괄 작업을 사용하여 Informatica MDM 마스터 레코드를 처리할 수 있습니다.

Informatica Data Quality 일괄 작업을 사용하려면 랜딩 테이블에서 매핑할 모든 열이 같은 **Data Quality** 함수를 수행해야 합니다. 열은 함수 내에서 정리되거나 변경되지 않은 함수를 통해 통과할 수 있습니다.

참고: 매핑이 단일 **Data Quality** 함수를 통해 모든 입력을 통과하지 않는 경우 Informatica Data Quality 일괄 작업을 사용할 수 없습니다. 대신 Informatica Data Quality는 레코드를 한 번에 하나씩 처리합니다.

예를 들어 랜딩 테이블의 여러 열에서 기본 키를 생성하려고 합니다. 매핑을 생성하고 **Data Quality** 함수를 추가한 다음 랜딩 테이블에서 준비 테이블로 매핑하려는 모든 열을 추가합니다. **Data Quality** 함수 내에 기본 키에 대한 열을 연결할 함수를 추가하고 **PKEY_SRC_OBJECT** 열에 연결 함수의 출력을 매핑합니다. 나머지 열에 대해서는 다른 함수를 사용하거나 변경하지 않은 채 통과할 수 있습니다.

다음 다이어그램은 필요한 모든 입력 및 매핑된 출력과 함께 Data Quality 함수를 보여 줍니다.



준비 테이블에서 영구 삭제 검색 기능을 사용할 경우 영구 삭제 검색 테이블에 기본 키를 생성하는 데 사용되는 열을 지정합니다.

매핑 도구 시작

매핑 도구를 시작하려면 Hub 콘솔에서 모델 작업 영역을 확장한 후 **매핑**을 클릭합니다.

Hub 콘솔에 다음 패널을 포함한 매핑 도구가 표시됩니다.

열	설명
매핑 목록	정의된 모든 랜딩-준비 매핑 목록입니다.
속성	선택한 매핑의 속성입니다.

매핑 목록에서 매핑을 선택하면 해당 속성이 표시됩니다.

매핑 도구의 탭

매핑이 선택되면 매핑 도구에 다음과 같은 탭이 표시됩니다.

탭	설명
일반	이 매핑의 일반 속성입니다.
다이어그램	랜딩 테이블과 준비 테이블의 열 간 매핑을 정의할 수 있도록 해주는 대화형 다이어그램입니다.
쿼리 매개 변수	이 매핑의 쿼리 매개 변수를 지정할 수 있습니다.
테스트	매핑을 테스트할 수 있습니다.

매핑 다이어그램

매핑에 대한 [다이어그램] 탭을 클릭하면 매핑 도구에 현재 열 매핑이 표시됩니다.

매핑 행에는 랜딩 테이블의 소스 열-준비 테이블의 대상 열 매핑이 표시됩니다. 매핑 행의 양쪽 끝의 원 색은 데이터 유형을 나타냅니다.

매핑 작성

랜딩 테이블과 준비 테이블 간 매핑을 생성합니다.

1. **모델** 작업 영역에서 **매핑**을 클릭합니다.
2. 쓰기 잠금을 획득합니다.
3. 매핑 영역에서 마우스 오른쪽 단추를 클릭하고 **매핑 추가**를 선택합니다.
매핑 도구에 매핑 대화 상자가 표시됩니다.
4. 매핑 속성을 지정합니다.

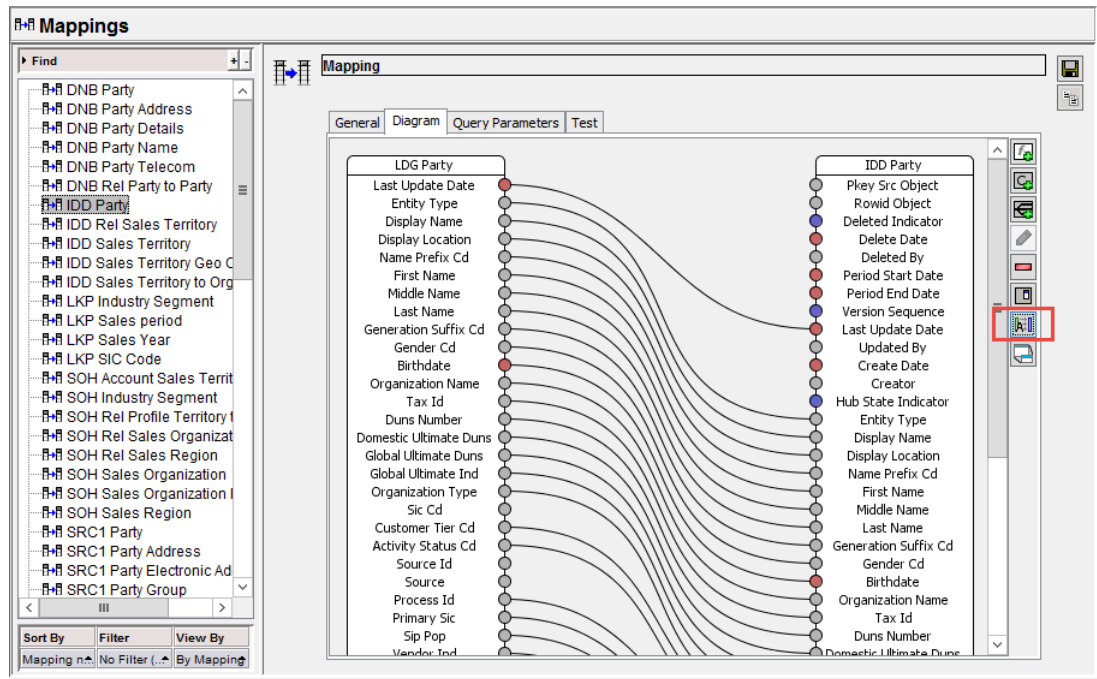
필드	설명
이름	Hub 콘솔에 표시되는 매핑 이름입니다.
설명	매핑의 설명입니다.
랜딩 테이블	매핑의 소스가 되는 랜딩 테이블을 선택합니다.
준비 테이블	매핑의 대상이 되는 준비 테이블을 선택합니다.
보안 리소스	매핑에 대한 액세스를 제어할 수 있도록 매핑을 보안 리소스로 지정하려면 활성화합니다. 매핑이 보안 리소스로 지정되면 보안 리소스 도구에서 이 매핑에 대한 권한을 할당할 수 있습니다.

아래 이미지는 샘플 매핑을 보여 줍니다.

Mapping	
Name	IDD Party
Description	Example mapping
Landing table	LDG Party
Staging table	IDD Party
Secure Resource	<input checked="" type="checkbox"/>

OK Cancel

5. **확인**을 클릭합니다.
6. **다이어그램** 탭을 클릭합니다.
매핑 도구의 작업 공간에 랜딩 테이블과 준비 테이블이 표시됩니다.
7. 랜딩 테이블에서 준비 테이블로 이름이 동일한 열을 매핑하려면 **자동 매핑** 단추를 클릭합니다.
아래 이미지는 샘플 매핑에서의 자동 매핑 결과를 보여 줍니다. 연결을 변경할 수 있습니다.



8. 저장을 클릭합니다.

기본 매핑 상태입니다. 다음으로 기본 키를 매핑합니다.

기본 키 매핑

소스 시스템의 레코드를 식별하려면 MDM Hub는 기본 키인 고유 키가 필요합니다. 소스 시스템에 고유 키가 있는 열이 포함된 경우 랜딩 테이블의 해당 열을 준비 테이블의 PKEY_SRC_OBJECT 열에 매핑합니다. 소스 시스템에 고유 키가 포함되지 않은 경우 여러 열에서 값을 연결하여 고유 키를 생성할 수 있습니다.

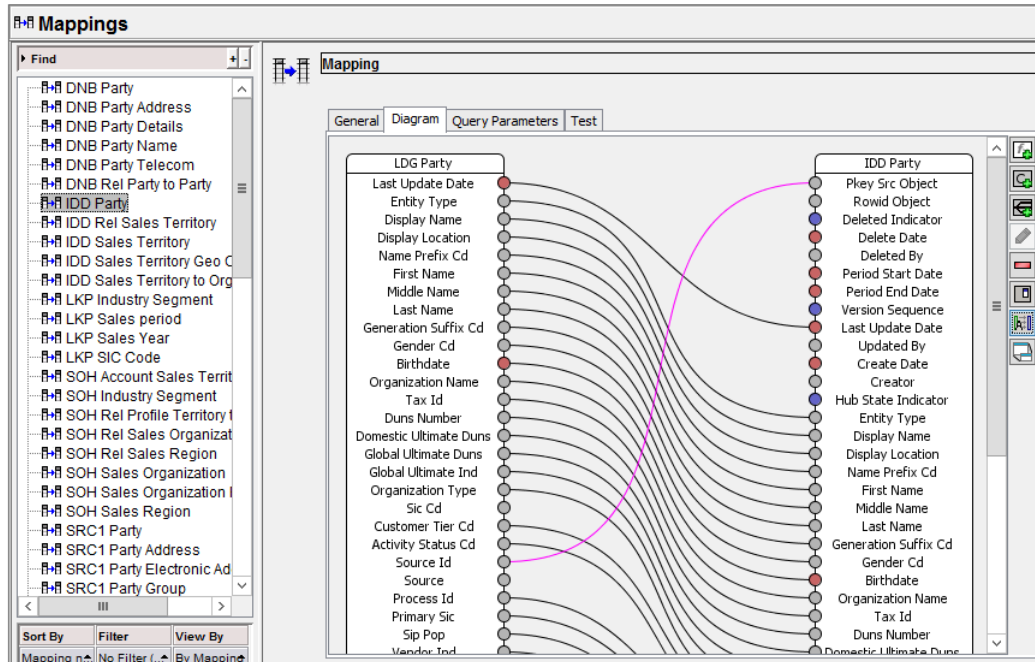
기본 키로 단일 열 사용

소스 시스템에 고유 키가 있는 열이 포함된 경우 랜딩 테이블의 해당 열을 준비 테이블의 PKEY_SRC_OBJECT 열에 매핑합니다.

이 단계에서는 다이어그램 작업 공간에 매핑이 열려 있다고 가정합니다.

1. **다이어그램** 작업 공간에서 랜딩 테이블의 고유 열 이름을 찾습니다. 해당 출력 커넥터를 준비 테이블에서 PKEY_SRC_OBJECT 열의 입력 커넥터로 끌어옵니다.

선이 열을 연결합니다.



2. **저장**을 클릭합니다.

기본 키로 여러 열 사용

소스 데이터에 고유 값이 있는 열이 없는 경우 여러 열에서 값을 연결하여 고유 기본 키를 형성합니다. MDM Hub 또는 Informatica Data Quality에서 문자열 연결 함수를 사용하거나 사용자 지정 함수를 생성합니다.

다이어그램 작업 공간에 매핑이 열려 있는지 확인합니다.

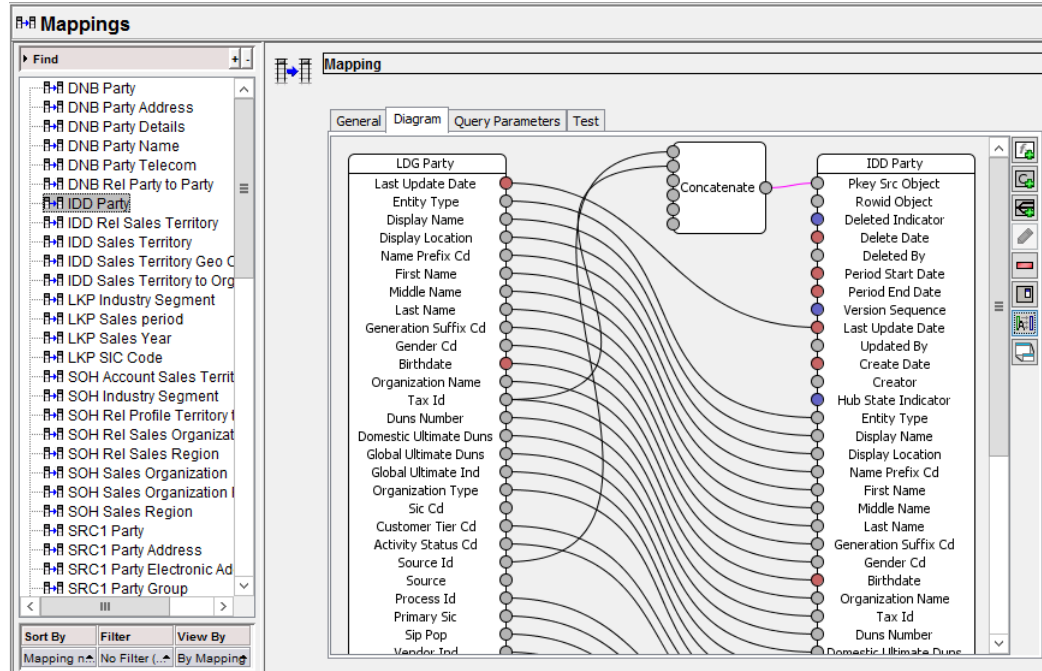
참고: 랜딩 테이블의 열 중 네 개 이상이 준비 테이블의 PKEY_SRC_OBJECT 열에 매핑될 수 있지만 인덱스 생성은 최대 세 개의 열로 제한됩니다.

1. 기본 키와 연결할 열을 식별합니다.
2. 연결 함수를 추가합니다.
 - a. **다이어그램** 작업 공간에서 작업 공간을 마우스 오른쪽 단추로 클릭하고 **함수 추가**를 선택합니다.

- b. 문자열 함수를 확장합니다.
- c. 연결을 클릭합니다.
- d. 확인을 클릭합니다.

연결 함수가 작업 공간에 나타납니다.

3. 랜딩 테이블에서 연결할 각 열에 대해 해당 출력 커넥터를 연결 함수의 입력 커넥터로 끌어옵니다.
 4. 함수에서 출력 커넥터를 준비 테이블에 있는 PKEY_SRC_OBJECT 열의 입력 커넥터로 끌어옵니다.
- 기본 키 연결이 완료되었습니다.



5. 저장을 클릭합니다.

열 매핑

직접 연결의 경우 랜딩 테이블의 열을 준비 테이블의 열에 연결합니다. 랜딩 테이블에서 들어오는 데이터를 정리하려면 함수를 추가합니다. 함수에 대한 입력은 랜딩 테이블의 열에서 들어옵니다. 함수의 출력은 준비 테이블의 열로 이동합니다.

이 단계에서는 다이어그램 탭에 매핑이 열려 있다고 가정합니다. 자동 매핑 프로세스를 사용한 경우 같은 이름의 열이 직접 연결됩니다.

1. **다이어그램** 작업 공간에서 자동으로 생성된 연결을 검토합니다. 자동 연결이 올바르지 않으면 연결을 삭제합니다.
 - a. 연결선을 클릭합니다.
 - b. 마우스 오른쪽 단추를 클릭하고 **링크 삭제**를 선택합니다.
2. 연결되지 않은 랜딩 테이블의 열을 준비 테이블 열에 연결해야 할지 결정합니다.

연결을 계획할 때 다음 요구 사항을 고려합니다.

 - 데이터 유형. 열은 같은 데이터 유형이거나 암시적으로 변환할 수 있는 데이터 유형이어야 합니다.

- 정리. 데이터를 준비 테이블에 로드하기 전에 정리하거나 변환하려고 합니까? 그렇다면 사용할 함수를 결정해야 합니다.
 - 문자열 길이. 준비 테이블에서 **CHAR** 또는 **VARCHAR** 데이터 유형의 열은 길이가 랜딩 테이블의 열 길이보다 크거나 같아야 합니다.
경고: 로드 시 준비 테이블 열의 문자열이 너무 긴 경우 로드 프로세스는 전체 레코드를 거부 테이블로 보냅니다.
3. 데이터를 정리할 필요가 없는 경우 바로 열을 연결합니다. 열을 연결하려면 랜딩 테이블 열에서 출력 커넥터를 준비 테이블 열의 입력 커넥터로 끌어옵니다.
 4. 데이터를 정리해야 할 경우 작업 공간에 함수를 추가한 다음 열을 연결합니다.
 - a. 작업 공간에서 마우스 오른쪽 단추를 클릭하고 **함수 추가**를 선택합니다.
 - b. 함수를 선택하고 **확인**을 클릭합니다.
다이어그램 작업 공간에 함수가 표시됩니다.
 - c. 랜딩 테이블에서 열을 찾습니다. 해당 출력 커넥터를 함수에 대한 입력 커넥터로 끌어옵니다. 함수에 대한 입력으로 열을 반복해서 추가합니다.
 - d. 함수에서 해당 출력 커넥터를 준비 테이블 열의 입력 커넥터로 끌어옵니다.
 5. 연결 추가를 마친 후에 **저장**을 클릭합니다.

매핑의 레코드 필터링

기본적으로 모든 레코드는 랜딩 테이블에서 검색됩니다.

필요할 경우 랜딩 테이블에서 레코드를 필터링하는 매핑을 구성할 수 있습니다. 필터 유형은 고유 및 조건부의 두 가지가 있습니다. 매핑 도구의 쿼리 매개 변수 탭에서 이러한 설정을 구성할 수 있습니다.

고유 매핑

쿼리 매개 변수 탭에서 고유 활성화 확인란을 클릭하면 준비 작업이 랜딩 테이블에서 고유 레코드만 선택합니다. Informatica MDM Hub는 다음 **SELECT** 문을 사용하여 준비 테이블을 채웁니다.

```
select distinct * from landing_table
```

여러 개의 준비 테이블에 공급하는 단일 랜딩 테이블이 있으며 해당 랜딩 테이블이 정규화되지 않은 상태(예: 고객 데이터와 주소 데이터 모두 포함)인 경우 고유 매핑을 유용하게 사용할 수 있습니다. 한 고객에게 3개의 주소가 있을 수 있습니다. 이 경우 고유 매핑을 사용하면 두 개의 추가 고객 레코드가 거부 테이블에 기록되지 않도록 할 수 있습니다.

또 다른 예로, 랜딩 테이블에 다음 데이터가 포함되어 있다고 가정합니다.

LUD	CUST_ID	NAME	ADDR_ID	ADDR
7/24	1	JOHN	1	1 MAIN ST
7/24	1	JOHN	2	1 MAPLE ST

고객 테이블에 대한 매핑에서 고유 활성화를 선택하면 **LUD**, **CUST_ID** 및 **NAME**만 고객 준비 테이블에 매핑되므로 중복 레코드를 방지할 수 있습니다. 고유 매핑을 활성화한 상태에서는 하나의 레코드만 고객 테이블을 채우므로 거부가 발생하지 않습니다.

또는 주소 매핑의 경우 고유 매핑을 비활성화한 상태로 **ADDR_ID**와 **ADDR**를 매핑하면 거부 없이 두 개의 레코드를 얻을 수 있습니다.

조건부 매핑

조건 활성화 확인란을 선택할 경우 SQL WHERE 절을 적용하여 정리에서 데이터를 언로드할 수 있습니다. 예를 들어 랜딩 테이블의 데이터가 미국의 모든 주를 기반으로 한다고 가정해 봅시다. WHERE 절을 통해 준비 테이블에 기록되는 데이터를 필터링하여 특정 주(예: 캘리포니아)의 데이터만 포함할 수 있습니다. 이렇게 하려면 WHERE 절에 WHERE 키워드는 생략하고 STATE = 'CA'를 입력합니다. 정리 작업이 실행되면 SELECT * FROM LANDING WHERE STATE = 'CA'에 따라 레코드를 언로드하고 처리합니다. 조건부 매핑을 지정할 경우 유효성 검사 단추를 클릭하여 SQL 문의 유효성을 검사합니다.

매핑에 대한 쿼리 매개 변수 구성

매핑에 대한 쿼리 매개 변수를 구성하려면

1. 매핑 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 구성할 매핑을 선택합니다.
4. 쿼리 매개 변수 탭을 클릭합니다.
매핑 도구에 이 매핑에 대한 쿼리 매개 변수 탭이 표시됩니다.
5. 필요한 경우 **고유 활성화** 확인란을 적절하게 선택하거나 선택 취소하여 고유 매핑을 구성합니다.
6. 필요한 경우 **조건 활성화** 확인란을 적절하게 선택하거나 선택 취소하여 조건부 매핑을 구성합니다.
이 확인란을 활성화한 경우 SQL WHERE 절(WHERE 키워드 생략)을 입력하고 **유효성 검사**를 클릭하여 해당 절의 유효성을 검사합니다.
7. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

정리 함수를 사용하는 매핑 관리

정리 함수를 포함하는 매핑은 정리 엔진의 상태에 영향을 받을 수 있습니다.

다음과 같은 경우에는 MDM Hub가 매핑에서 함수를 제거합니다.

- 사용자 환경에서 정리 엔진을 변경하고 매핑에 사용하는 함수는 새 정리 엔진에서 사용할 수 없습니다.
- 처리 서버를 위한 응용 프로그램 서버를 다시 시작했지만 정리 엔진 초기화에 실패했습니다.

매핑에 대한 함수를 복원하려면 매핑을 편집하고 활성화 정리 엔진에서 사용할 수 있는 함수를 추가합니다.

행 ID별 로드

Informatica MDM Hub가 로드, 일치 및 병합 처리를 간소화할 수 있도록 행 ID별 로드를 구성할 수 있습니다.

준비 프로세스를 구성하는 경우 랜딩 테이블 열을 준비 테이블 열에 매핑해야 합니다. 행 ID별 로드를 구성하려면 ROWID_OBJECT 값이 포함된 랜딩 테이블 열을 준비 테이블의 ROWID_OBJECT 열에 매핑합니다. 예를 들어, 랜딩 테이블의 INPUT_ROWID_OBJECT 열을 준비 테이블의 ROWID_OBJECT 열에 매핑하여 ROWID_OBJECT 값을 MDM Hub에 제공할 수 있습니다.

로드 프로세스에서는 준비 테이블의 ROWID_OBJECT 열 값을 기본 개체와 연결된 교차 참조 테이블의 ORIG_ROWID_OBJECT 열 값과 비교합니다. 값이 일치하면 로드 프로세스에서는 교차 참조 테이블의 레코드를 업데이트합니다.

PKEY_SRC_OBJECT 및 ROWID_SYSTEM 열의 값이 일치하는 경우 로드 프로세스에서는 기존 교차 참조 레코드를 수신 레코드로 업데이트합니다. 그렇지 않은 경우 로드 프로세스에서는 수신 레코드의 값을 가진 새 교차 참조 레코드를 삽입한 후 기존 교차 참조 레코드와 병합합니다. 로드 프로세스에서는 기본 개체 레코드를 교차 참조 레코드의 상위 셀 값으로 업데이트합니다.

지정된 소스 및 지정된 행 ID에 대해 -1의 HUB_STATE_IND 값을 로드한 경우 소스에 상관없이 지정된 행 ID의 모든 교차 참조 레코드에 대해 HUB_STATE_IND 값이 -1로 설정됩니다. 모든 교차 참조 레코드의 HUB_STATE_IND 값이 -1이기 때문에 MDM Hub은 기본 개체 레코드가 삭제될 것으로 간주합니다.

기본 개체의 초기 데이터 로드에서 모든 레코드를 대상 기본 개체에 삽입합니다. 따라서 초기 데이터 로드 후에 발생하는 증분 로드에는 행 ID별 로드를 활성화합니다.

감사 내역 및 델타 검색 구성

준비 테이블에 열 매핑을 완료한 후에는 준비 테이블 데이터에 대한 감사 내역 및 델타 검색을 구성할 수 있습니다.

감사 내역을 활성화한 경우 MDM Hub에서 Raw 테이블의 레코드를 유지합니다. MDM Hub은 구성된 준비 작업 실행 횟수 또는 보존 기간 동안 레코드를 유지합니다. 랜딩 테이블의 모든 중복 레코드는 해당하는 Raw 테이블에 있습니다. 각 준비 작업 실행에 대해 MDM Hub이 준비 작업 실행 횟수 또는 보존 기간에 도달하면 MDM Hub은 소스 개체의 각 기본 키에 대해 하나의 레코드를 유지합니다. 예를 들어 구성된 보존이 2번의 준비 작업 실행인 경우 처음 두 번 준비 작업이 실행되는 동안에는 MDM Hub이 소스 개체의 기본 키에 대한 모든 중복 항목을 Raw 테이블에 유지합니다. 준비 작업을 두 번 실행하고 나면 Raw 테이블은 각 준비 작업 실행에 대해 소스 개체의 각 준비 키마다 하나의 레코드를 유지합니다.

델타 검색을 활성화한 경우 MDM Hub은 랜딩 테이블에서 새로 추가되거나 업데이트된 레코드를 검색하여 변경 없이 해당 Raw 테이블에 복사합니다. 그렇지 않은 경우에는 MDM Hub이 레코드를 대상 테이블에 복사합니다.

감사 내역 및 델타 검색을 구성하려면 **설정** 탭을 클릭합니다.

준비 테이블에 대한 감사 내역 구성

Informatica MDM Hub에서는 로드 횟수 및 타임스탬프에 따라 RAW 테이블에 데이터 기록을 보존하는 감사 내역을 구성할 수 있습니다.

이 감사 내역은 예를 들어 HDD(영구 삭제 검색)를 사용하는 경우에 유용합니다. 기본적으로 감사 내역은 활성화되어 있지 않으며 RAW 테이블은 비어 있습니다. 감사 내역이 활성화되어 있으면 준비 작업을 구성된 횟수만큼 실행하는 동안 또는 지정된 보존 기간 동안 레코드가 RAW 테이블에 보관됩니다.

참고: 감사 내역은 감사 관리자 도구와는 매우 다른 기능을 하므로 혼동해서는 안 됩니다.

준비 테이블에 대한 감사 내역을 구성하려면

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 준비 테이블에 대한 매핑을 아직 추가하지 않았으면 추가합니다.
4. 구성할 준비 테이블을 선택합니다.
5. 속성 패널 아래쪽의 **다음 기간 동안 Raw 테이블에 감사 내역 유지**를 클릭하여 Raw 데이터 감사 내역을 활성화합니다.

감사 테이블에 대한 보존 기간을 선택하라는 메시지가 표시됩니다.

6. 감사 보존 기간에 대한 다음 옵션 중 하나를 선택합니다.

옵션	설명
로드	데이터를 유지할 일괄 로드 횟수입니다.
기간	데이터를 유지할 기간입니다.

7. **저장**을 클릭하여 변경 내용을 저장합니다.

감사 내역이 구성되어 있으면 지정한 보존 기간 동안 데이터가 보관됩니다. 예를 들어 두 번의 로드(준비 작업 실행)에 대한 감사 내역을 구성했다고 가정합니다. 이 경우 감사 내역에는 최근 두 번의 로드를 통해 준비 테이블에 저장된 데이터가 유지됩니다. 랜딩 테이블에 로드당 10개의 레코드가 있었다면 RAW 테이블의 총 레코드 수는 20개가 됩니다.

준비 작업을 여러 번 실행할 경우 RAW 테이블에는 ROWID_JOB을 기준으로 최근 두 개의 집합에 해당하는 데이터가 유지됩니다. 오래된 ROWID_JOB의 데이터는 삭제됩니다. 예를 들어 첫 번째 준비 작업의 ROWID_JOB 값이 1이고 두 번째 준비 작업의 ROWID_JOB 값은 2라고 가정하면 세 번째 준비 작업을 실행할 때 ROWID_JOB이 1인 레코드는 삭제됩니다.

참고: 프로세스를 처음 실행한 후 일괄 처리 뷰어에서 "기록 지우기" 단추를 사용할 경우: 준비 테이블에 대한 감사 내역이 활성화되어 있는 경우 연결된 준비 작업이 선택된 상태에서 일괄 처리 뷰어의 기록 지우기 단추를 선택하면 다음에 준비 작업을 실행할 때 RAW 및 REJ 테이블의 레코드가 지워집니다.

준비 테이블에 대한 델타 검색 구성

준비 테이블에 대한 델타 검색을 활성화한 경우 Informatica MDM Hub에서는 새로 추가되거나 변경된 레코드만 처리하고 변경되지 않은 레코드는 무시합니다. 델타 검색 대상으로 선택한 열의 값이 null이거나 미래 날짜인 레코드는 준비 프로세스에서 거부됩니다. 시간 표시 막대가 사용되는 기본 개체의 경우 PERIOD_START_DATE or PERIOD_END_DATE 열을 기반으로 델타 검색을 실행하면 PERIOD_START_DATE 또는 PERIOD_END_DATE 열의 값이 null이거나 미래 날짜인 레코드는 거부됩니다.

특정 열에 대한 델타 검색 활성화

특정 열에 대해 델타 검색을 활성화하려면

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 구성할 준비 테이블을 선택합니다.
준비 테이블 속성 창이 표시됩니다.
4. **델타 검색 활성화** 확인란을 선택합니다.
5. **특정 열을 사용하여 델타를 검색합니다** 옵션을 선택합니다.
사용 가능한 열 목록이 표시됩니다.
6. 델타 검색을 사용할 열을 선택하고 **추가 >**를 클릭하거나, **모두 추가 >>**를 클릭하고 델타 검색을 사용할 열을 모두 추가합니다.

데이터를 준비 테이블로 로드할 때 정의된 집합의 열에 사용 가능한 이전 로드 값과 다른 값이 있을 경우 행이 변경된 것으로 처리됩니다. 정의된 집합의 모든 열이 동일하면 행이 변경되지 않은 것으로 처리됩니다. 매핑되지 않은 열은 무시됩니다.

준비 테이블에 대한 델타 검색 활성화

준비 테이블에 대한 델타 검색을 활성화하려면

1. 스키마 관리자를 시작합니다.
2. 구성할 준비 테이블을 선택합니다.
3. 쓰기 잠금을 획득합니다.
4. **델타 검색 활성화** 확인란을 선택하여 테이블에 대한 델타 검색을 활성화합니다. 이 옵션을 보려면 아래로 스크롤해야 할 수 있습니다.
5. 델타 검색 방식을 지정합니다.

옵션	설명
매핑의 모든 열을 비교하여 델타 검색	마지막 업데이트 날짜를 비롯한 모든 열이 델타 비교를 위해 선택됩니다.
날짜 열을 통해 델타를 검색합니다.	스키마에 적용 가능한 날짜 열이 있는 경우 이 옵션을 선택하고 델타 비교에 사용할 날짜 열을 선택합니다. 적용 가능한 날짜 열이 있는 경우 이 옵션이 기본 설정 옵션입니다.

6. 준비 프로세스 또는 로드 프로세스 중에 이전 중복 항목이 거부된 경우 준비를 허용할지 여부를 지정합니다.
 - 이전의 준비된 중복 항목이 거부된 경우 다음 번 준비 프로세스가 실행되는 동안 준비되는 중복 레코드가 델타 검색을 건너뛰는 것을 허용하려면 이 옵션을 선택합니다.
참고: 이 옵션이 활성화되어 있는 경우 연결된 준비 작업이 선택된 상태에서 사용자가 일괄 처리 뷰어에서 기록 지우기 단추를 클릭하면 다음 번 준비 작업이 실행될 때 REJ 테이블의 레코드가 지워지기 때문에 이 기능에서 사용하는 이전 거부 기록이 무시됩니다.
 - 이전의 준비된 중복 항목이 거부된 경우 다음 번 준비 프로세스가 실행되는 동안 준비되는 중복 레코드가 델타 검색을 건너뛰지 못하도록 하려면 이 옵션을 선택 취소합니다. 델타 검색이 다음 번 준비 프로세스 실행에서 후속 처리되는 모든 중복 랜딩 레코드를 제외합니다.

Informatica MDM Hub에서 델타 검색이 처리되는 방식

델타 검색이 활성화된 경우 준비 작업에서는 선택한 준비 테이블에 매핑된 랜딩 테이블의 콘텐츠를 이전 준비 작업 실행에서 처리된 데이터 집합과 비교합니다.

이러한 비교를 통해 이전 실행 이후 데이터가 변경되었는지 여부가 확인됩니다. 변경된 레코드, 새로운 레코드 및 거부된 레코드는 준비 테이블에 배치됩니다. 중복 레코드는 무시됩니다.

참고: 거부 레코드는 두 번째 준비 실행 후 정리에서 로드로 이동됩니다.

델타 검색을 사용하기 위한 고려 사항

델타 검색을 사용할 경우 몇 가지 문제를 고려해야 합니다.

다음 문제를 고려하십시오.

- 델타 검색은 전체 레코드 비교나 날짜 열을 통해 수행될 수 있습니다. Informatica MDM Hub에서는 각 수신 레코드의 마지막 업데이트 날짜 열을 해당 레코드의 이전 마지막 업데이트 날짜와 간단하게 비교할 수 있으므로 마지막 업데이트 날짜에 대한 델타 검색이 가장 효율적입니다.
- 델타 검색을 사용할 경우 랜딩 테이블의 열 중 델타 검색용으로 준비 테이블의 last_update_date 열에 매핑된 열은 포함하지 않을 수 있습니다.

참고: 델타 검색을 설정할 때 last_update_date 열을 포함할 경우 변경된 내용이 last_update_date 열에만 있으면 MDM Hub에서는 델타 검색 및 로드 작업 시 불필요하게 많은 작업을 수행하게 됩니다.

- 마지막 업데이트 날짜를 기준으로 레코드를 처리할 때는 예를 들어 소스 레코드의 마지막 업데이트 날짜가 현재 시스템 날짜 이전에 해당하는지 여부를 테스트하는 경우와 같이 마지막 업데이트 값을 비교하는 데 "현재" 정리 함수를 사용하지 마십시오. 이러한 식으로 "현재"를 사용하면 예기치 않은 결과가 발생할 수 있습니다.
- 마지막 업데이트 날짜가 실제로 변경 표시기의 역할을 하지 않는 경우에만 소스의 열에 대해 델타 검색을 수행하십시오. Informatica MDM Hub 준비 작업에서는 전체 소스 레코드를 PRL(이전 로드) 테이블의 해당하는 가장 최근 레코드와 비교합니다. 다른 셀이 있으면 해당 레코드가 준비 테이블에 전달됩니다. 델타 검색은 PRL 테이블에서 수행됩니다.
- 델타 검색이 날짜 열(last_update_date, 시스템 정의 날짜 열 또는 DOB와 같은 사용자 지정 날짜 열)을 기준으로 하는 경우에는 RAW 테이블의 최대 날짜가 아니라 PRL 테이블의 해당하는 레코드와 비교하여 날짜 열의 날짜 값이 더 나중에 해당하는 레코드만 준비 테이블에 삽입됩니다.
- 델타 검색이 DOB 및 last_update_date 같은 날짜 열을 비롯하여 특정 열을 기준으로 하는 경우에는 RAW 테이블의 최대 날짜가 아니라 PRL 테이블의 해당하는 레코드와 비교하여 지정된 열의 날짜 값이 변경된 레코드가 준비 테이블에 삽입됩니다.
- 델타 검색 중 모든 열의 델타를 확인할 때는 null 기본 키가 있는 레코드만 거부됩니다. 이는 정상적인 동작입니다. 델타 프로세스에 실패한 다른 모든 레코드는 후속 준비 프로세스에서 거부됩니다.
- 델타 검색이 마지막 업데이트 날짜를 기준으로 하는 경우 마지막 업데이트 날짜 또는 기본 키에 대한 변경 내용이 검색됩니다. 마지막 업데이트 날짜가 아니거나 연결된 기본 키에 속하지 않는 모든 값에 대한 업데이트 내용은 검색되지 않습니다.
- 매핑된 열을 기준으로 하는 델타 검색을 사용할 경우 후속 준비 프로세스 중에 중복 기본 키는 고려되지 않습니다.
- 거부 처리 기능을 사용하면 다음을 수행할 수 있습니다.
 - 일괄 작업과 관련된 특정 준비 테이블의 모든 거부 레코드를 볼 수 있습니다.
 - 모든 준비 테이블에서 날짜별로 모든 거부 레코드를 볼 수 있습니다.
 - 쿼리 필터를 기준으로 거부 테이블을 쿼리할 수 있습니다.

준비 테이블 관리

MDM Hub 준비 테이블을 구성한 후에는 준비 테이블 속성을 변경할 수도 있습니다. 또한 준비 테이블과 연결된 소스 시스템을 볼 수도 있습니다. 필요하지 않은 준비 테이블을 제거하는 것도 가능합니다.

준비 테이블의 속성 변경

필요한 경우 준비 테이블 속성을 변경할 수 있습니다.

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 **기본 개체** 노드를 확장하고 이 준비 테이블과 연결된 기본 개체의 노드를 확장합니다.
준비 테이블이 기본 개체와 연결되어 있는 경우 **준비 테이블** 노드를 확장하여 기본 개체를 표시합니다.
4. 구성할 준비 테이블을 선택합니다.
스키마 관리자에 선택한 테이블의 속성이 표시됩니다.
5. 준비 테이블 속성을 지정합니다.
편집할 각 속성에서 옆에 있는 **편집** 단추를 클릭하고 새 값을 지정합니다.

참고: 준비 테이블과 관련 지원 테이블(예: 원시 테이블 및 기본 랜딩 테이블)이 비어 있는 경우 소스 시스템을 변경할 수 있습니다.

준비 테이블 또는 관련 테이블에 데이터가 포함되어 있는 경우에는 소스 시스템을 변경하지 마십시오.

6. 기본 개체의 열 목록에서 이 소스 시스템이 제공할 열을 변경합니다.

- 각 열을 개별적으로 클릭할 필요 없이 모든 열을 선택하려면 **모두 선택** 단추를 클릭합니다.
- 선택된 모든 열을 선택 취소하려면 **모두 지우기** 단추를 클릭합니다.

참고: "Rowid 개체"와 "마지막 업데이트 날짜"는 자동으로 선택됩니다. 이러한 열은 선택 취소하거나 속성을 변경할 수 없습니다.

7. 필요한 경우 열 속성을 변경합니다.
8. 필요한 경우 외래 키 열에 대한 조회를 변경합니다. 열을 선택하고 **편집** 단추를 클릭하여 조회 열을 구성합니다.
9. 셀 업데이트를 변경하려면 **셀 업데이트** 확인란을 클릭합니다.
10. 필요한 경우 준비 테이블의 열 구성을 변경합니다.
11. 필요한 경우 이 준비 테이블에 대해 감사 내역 및 델타 검색을 구성합니다.
12. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

준비 테이블의 소스 시스템으로 이동

준비 테이블과 연결된 소스 시스템을 보려면 준비 테이블을 마우스 오른쪽 단추로 클릭하고 **소스 시스템으로 이동**을 선택해야 합니다.

Hub 콘솔에서 시스템 및 트러스트 도구가 시작되고 이 준비 테이블과 연결된 소스 시스템이 표시됩니다.

준비 테이블 제거

준비 테이블을 제거하려면

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 **기본 개체** 노드를 확장하고 이 준비 테이블과 연결된 기본 개체의 노드를 확장합니다.
4. 제거할 준비 테이블을 마우스 오른쪽 단추로 클릭한 후 **제거**를 선택합니다.

스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.

5. **예**를 선택합니다.

스키마 관리자가 연산 참조 저장소(연산 참조 저장소)에서 준비 테이블을 삭제하고 연결된 제어 테이블을 삭제하며 스키마 트리에서도 삭제된 준비 테이블을 제거합니다.

매핑 관리

매핑 속성 변경, 매핑 제거 또는 매핑과 연결된 스키마 보기를 수행할 수도 있습니다. 또한 준비를 실행하기 전에 매핑을 테스트하는 것도 가능합니다.

매핑 속성 편집

기존 매핑을 복사하여 새 매핑을 생성하려면

1. 매핑 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 편집할 매핑을 선택합니다.
4. 필요한 경우 매핑 속성, 다이어그램 및 매핑 설정을 편집합니다.
5. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

매핑 복사

기존 매핑을 복사하여 새 매핑을 생성하려면

1. 매핑 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 복사할 매핑을 마우스 오른쪽 단추로 클릭하고 **매핑 복사**를 선택합니다.
매핑 대화 상자가 표시됩니다.
4. 매핑 속성을 지정합니다.
5. **확인**을 클릭합니다.
6. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

스키마로 이동

매핑 도구를 사용하여 스키마 관리자를 빠르게 실행하고 선택한 매핑과 연결된 스키마를 표시할 수 있습니다.

참고: 스키마로 이동 명령은 프로세스 보기에서 사용할 수 없으며 작업 영역 보기에서만 사용할 수 있습니다.

매핑에 대한 스키마로 이동하려면

1. 매핑 도구를 시작합니다.
2. 스키마를 보려는 매핑을 선택합니다.
3. 탐색 창 아래쪽에 있는 보기 기준 목록에서 다음 옵션 중 하나를 선택합니다.
 - 준비 테이블별
 - 랜딩 테이블별
 - 매핑별
4. 탐색 창에서 아무 곳이나 마우스 오른쪽 단추로 클릭한 다음 **스키마로 이동**을 선택합니다.
매핑 도구에 선택한 매핑의 스키마가 표시됩니다.

매핑 테스트

구성한 매핑을 테스트하려면

1. 매핑 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 구성할 매핑을 선택합니다.
4. **테스트** 탭을 클릭합니다.

매핑 도구에 이 매핑에 대한 테스트 탭이 표시됩니다.

5. 입력 이름 아래의 열에 입력 값을 지정합니다.
6. **테스트**를 클릭합니다.

매핑이 테스트되고 출력 이름 아래의 열에 결과가 채워집니다.

매핑 제거

매핑을 제거하려면

1. 매핑 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 제거할 매핑을 마우스 오른쪽 단추로 클릭한 후 **매핑 삭제**를 선택합니다.

매핑 도구에 삭제를 확인하는 메시지가 표시됩니다.

4. **예**를 클릭합니다.

매핑 도구가 지원 테이블을 삭제하고 메타데이터에서 매핑을 제거하며 매핑 목록을 업데이트합니다.

제 18 장

영구 삭제 검색

이 장에 포함된 항목:

- [영구 삭제 검색 개요, 311](#)
- [영구 삭제 검색 유형, 312](#)
- [기본 개체의 플래그 값 삭제, 312](#)
- [트러스트 및 유효성 검사 규칙, 312](#)
- [영구 삭제 검색 테이블, 313](#)
- [영구 삭제 검색 구성, 314](#)
- [영구 삭제 검색을 위한 기본 키 열 지정, 318](#)
- [직접 삭제, 318](#)
- [합의 삭제, 321](#)
- [사용자 종료 내에서 영구 삭제 검색 사용, 326](#)

영구 삭제 검색 개요

기본 개체를 채우는 소스 시스템은 지속적으로 업데이트되며 이러한 소스 시스템에서 레코드가 영구 삭제될 수도 있습니다. 영구 삭제된 레코드는 소스 시스템에서 제거된 레코드입니다. MDM Hub에서는 소스 시스템에서 영구 삭제된 레코드를 검색하고 연결된 기본 개체에 변경 내용을 반영할 수 있습니다.

참고: Oracle 및 Microsoft SQL Server 환경에서 MDM Hub는 소스 시스템에서 영구 삭제된 레코드를 검색할 수 있습니다.

MDM Hub 준비 작업에서는 최신 소스 시스템 파일의 모든 레코드를 이전 랜딩 테이블의 모든 레코드와 비교합니다. 전체 로드와 경우 MDM Hub에서는 최신 소스 시스템에서 누락된 레코드를 검색하여 해당 레코드에 영구 삭제 플래그를 지정합니다. 그런 다음 MDM Hub에서는 삭제된 레코드의 열 값을 삭제 플래그 및 종료 날짜(선택 사항)와 함께 랜딩 테이블에 다시 삽입합니다. 마지막으로 MDM Hub에서는 연결된 기본 개체를 업데이트합니다.

중분 또는 트랜잭션 로드의 경우 MDM Hub는 최신 소스 시스템 파일에서 누락된 레코드를 검색하지 않습니다. 영구 삭제를 구성하려면 영구 삭제 검색 테이블을 리포지토리 테이블에 작성해야 합니다. MDM Hub에서는 영구 삭제 항목이 검색될 경우 삭제 플래그가 지정된 레코드의 수를 작업 매트릭스 테이블에 삽입합니다. 영구 삭제 검색은 준비 프로세스의 일부인 Java 사용자 종료로 구현합니다.

영구 삭제 검색 유형

선택하는 영구 삭제 검색 유형은 소스 시스템, 그리고 기본 개체 레코드와 연결된 교차 참조 레코드에 따라 달라집니다.

단일 소스 시스템에서 레코드의 삭제 상태를 결정해야 하는 경우 **MDM Hub**는 기본 개체 레코드에 직접 삭제 플래그를 지정합니다. 여러 소스 시스템에서 레코드의 삭제 상태를 결정해야 하는 경우 **MDM Hub**는 합의 삭제를 수행합니다. **MDM Hub**는 모든 소스 시스템의 레코드가 삭제된 상태에 있는지 확인하고 기본 개체 레코드에 삭제 플래그를 할당합니다.

기본 개체의 플래그 값 삭제

기본 개체에서 데이터를 통합하는 방법을 정의하는 경우 삭제 플래그를 식별하고 삭제 플래그 값을 확인해야 합니다.

MDM Hub에는 기본 삭제 플래그 값이 있습니다. 기본 개체 레코드에 대해 삭제 플래그와 함께 종료 날짜 값을 제공하도록 **MDM Hub**를 구성할 수 있습니다.

기본 개체의 기본 삭제 플래그 값은 다음과 같습니다.

활성

모든 소스 시스템에서 레코드가 활성 상태임을 나타냅니다. 값을 변경하려면 `DELETE_FLAG_VAL_ACTIVE` 열의 값을 다른 값으로 설정합니다. 기본값은 **A**입니다.

비활성 또는 완전 삭제

모든 소스 시스템에서 레코드가 영구 삭제됨을 나타냅니다. 값을 변경하려면 `DELETE_FLAG_VAL_INACTIVE` 열의 값을 다른 값으로 설정합니다. 기본값은 **I**입니다.

부분 삭제

일부 소스 시스템에서 레코드가 삭제됨을 나타냅니다. 값을 변경하려면 `DELETE_FLAG_VAL_PARTIAL` 열의 값을 다른 값으로 설정합니다. 기본값은 **P**입니다.

삭제 플래그와 가능한 종료 날짜가 필요한 기본 개체를 확인합니다. 기본 개체와 연결된 랜딩 테이블 및 준비 테이블에 적절한 열을 포함해야 합니다. 또한 준비 테이블에서 영구 삭제를 검색하도록 리포지토리에 영구 삭제 검색 테이블을 구성했는지 확인합니다.

트러스트 및 유효성 검사 규칙

MDM Hub가 소스 시스템에서 삭제된 레코드에 플래그를 지정한 후에는 트러스트 및 유효성 검사 규칙을 사용하여 삭제 플래그 및 종료 날짜의 동작을 제어합니다. 트러스트 및 유효성 검사 규칙은 가장 신뢰도가 높은 데이터를 확인하는 데 유용합니다.

MDM Hub는 기본 개체 레코드의 열 값이 소스에서 영구 삭제된 것으로 검색될 경우 트러스트 및 유효성 검사 규칙을 사용합니다. 트러스트 및 유효성 검사 규칙은 이러한 기본 개체 열을 다른 소스 시스템의 값으로 재정의해야 하는 시기를 결정합니다. **MDM Hub**는 유효성 검사 규칙을 사용하여 삭제된 소스 레코드에 대해 트러스트된 모든 열에서 트러스트 백분율을 다운그레이드합니다. 삭제된 소스 레코드가 제공했던 기본 개체 레코드의 값은 기본 개체 레코드에 유지됩니다. 다른 소스 시스템에서 트러스트가 더 높은 업데이트를 제공한 후에는 **MDM Hub**가 삭제된 값을 바꿔 삭제된 소스 레코드의 값을 재정의합니다.

영구 삭제 검색 테이블

영구 삭제 검색 테이블은 MDM Hub가 소스 시스템에서 영구 삭제 항목을 검색하는 데 필요한 메타데이터를 저장하기 위해 사용자가 생성하는 리포지토리 테이블입니다. 영구 삭제 검색 테이블의 이름은 C_REPOS_EXT_HARD_DEL_DETECT입니다.

다음 표에서는 영구 삭제 검색 테이블의 열에 대해 설명합니다.

열 이름	데이터 유형 및 크기(바이트)	설명
ROWID_TABLE	CHAR(14)	영구 삭제된 레코드를 검색해야 하는 준비 테이블의 ROWID_OBJECT 열 값을 포함합니다. ROWID_TABLE 열은 null 값을 포함할 수 없습니다.
HDD_ENABLED	INTEGER	MDM Hub에서 준비 테이블의 영구 삭제 항목을 검색하는 기능을 활성화하거나 비활성화합니다. HDD_ENABLED 열은 null 값을 포함할 수 없습니다. 다음 값 중 하나를 사용합니다. - 0 - MDM Hub의 영구 삭제 검색 기능을 활성화합니다. - 1 - MDM Hub의 영구 삭제 검색 기능을 비활성화합니다. 기본값은 0입니다.
HDD_TYPE	VARCHAR2 (14)	준비 테이블의 영구 삭제 유형을 나타냅니다. 영구 삭제 유형의 경우 다음 값 중 하나를 사용합니다. - DIRECT. 기본 개체 레코드에 대해 직접 영구 삭제를 수행합니다. - CONSENSUS. 기본 개체 레코드에 대해 합의 기반 영구 삭제를 수행합니다.
DELETE_FLAG_COLUMN_NAME	VARCHAR2 (30)	MDM Hub가 검색하는 모든 영구 삭제된 레코드의 삭제 플래그 값을 포함하는 준비 테이블 열의 이름입니다.
DELETE_FLAG_VAL_ACTIVE	VARCHAR2 (14)	활성 레코드의 삭제 플래그 열 값입니다.
DELETE_FLAG_VAL_INACTIVE	VARCHAR2 (14)	비활성 레코드의 삭제 플래그 열 값입니다.
DELETE_FLAG_VAL_PARTIAL	VARCHAR2 (14)	합의 삭제 시나리오에서 부분 삭제의 삭제 플래그 열 값입니다.
HAS_END_DATE	INTEGER	준비 테이블에 종료 날짜 열이 있는지 여부를 나타냅니다. HAS_END_DATE 열은 null 값을 포함할 수 없습니다. 다음 값 중 하나를 사용합니다. - 0. 준비 테이블에 종료 날짜 열이 없습니다. - 1. 준비 테이블에 종료 날짜 열이 있습니다. 기본값은 0입니다.
END_DATE_COLUMN_NAME	VARCHAR2 (30)	종료 날짜 열 이름입니다. MDM Hub에서는 HAS_END_DATE=1인 경우에 이 열이 사용됩니다.

열 이름	데이터 유형 및 크기(바이트)	설명
END_DATE_VAL	DATE	종료 날짜에 사용할 값입니다. 종료 날짜 값을 하드 코딩해야 구현이 가능한 경우에는 해당 값이 END_DATE_VAL 열에 MM/DD/YYYY 형식으로 저장됩니다. 기본값은 SYSDATE입니다.
TRAN_HDD_ENABLED	INTEGER	영구 삭제 항목을 검색하는 프로세스가 트랜잭션인지 여부를 나타냅니다. TRAN_HDD_ENABLED 열은 null 값을 포함할 수 없습니다. 다음 값 중 하나를 사용합니다. - 0 - 영구 삭제 항목을 검색하는 프로세스가 트랜잭션이 아닙니다. - 1 - 영구 삭제 항목을 검색하는 프로세스가 트랜잭션이며 전체 로드를 이전 랜딩 및 랜딩 테이블과 비교하지 않습니다. 기본값은 0입니다.
HDD_LANDING_PKEY_COLUMNS	VARCHAR	선택 사항입니다. 함수에 대한 입력에 기본 키에 영향을 주지 않는 일부 소스 열이 포함되는 경우 함수를 사용하여 여러 소스 열에서 기본 키를 연결하려면 테이블에 HDD_LANDING_PKEY_COLUMNS 열을 추가합니다. 심표로 구분된 목록에서 기본 키에 영향을 주는 소스 열을 지정합니다. 영구 삭제 검색 프로세스는 함수에 대한 입력이 되는 모든 소스 열이 아닌 지정된 소스 열의 변경 사항을 추적합니다.
EMPTY_LANDING_TABLE_CHECK_OFF	INTEGER	선택 사항입니다. 영구 삭제를 검색하는 프로세스에서 빈 랜딩 테이블을 검사할지 여부를 나타냅니다. 다음 값 중 하나를 사용합니다. - 0. 영구 삭제를 검색하는 프로세스에서 빈 랜딩 테이블을 검사합니다. - 1. 영구 삭제를 검색하는 프로세스에서 빈 랜딩 테이블을 검사하지 않습니다. 기본값은 0입니다. 중요: 이 값은 0으로 설정하는 것이 좋습니다. 1 값은 MDM Hub가 빈 랜딩 테이블을 검사하지 않는 것을 의미하며 이 경우 랜딩 테이블의 소스 시스템과 연결된 모든 레코드가 삭제 대상으로 표시될 수 있습니다.

영구 삭제 검색 구성

소스 시스템에서 영구 삭제 항목을 검색하도록 MDM Hub를 구성할 수 있습니다.

영구 삭제 항목을 검색하도록 MDM Hub를 구성하려면 영구 삭제 검색 테이블을 생성하고 작업 메트릭스 유형을 등록해야 합니다. 또한 랜딩 및 준비 테이블을 설정하고 매핑을 생성해야 합니다. 랜딩 및 준비 테이블을 설정한 후에는 영구 삭제 검색 테이블을 준비 테이블 정보로 채웁니다. 마지막으로, 영구 삭제 검색 기능을 호출하여 영구 삭제된 레코드를 검색하는 사용자 종료를 구현합니다.

참고: 영구 삭제 검색 테이블을 구성하고 이를 단일 SQL 문 집합으로 실행하기 위해 제공된 모든 SQL 문을 조합할 수 있습니다.

1. 영구 삭제 검색 테이블을 구성합니다.

- a. 영구 삭제 검색 테이블을 생성하려면 다음 SQL 문을 실행합니다.

Oracle의 경우.

```
CREATE TABLE C_REPOS_EXT_HARD_DEL_DETECT
(
  ROWID_TABLE          CHAR(14 BYTE),
  HDD_ENABLED          INTEGER          DEFAULT 0          NOT NULL,
  HDD_TYPE             VARCHAR2(14 BYTE),
  DELETE_FLAG_COLUMN_NAME VARCHAR2(30 BYTE),
  DELETE_FLAG_VAL_ACTIVE  VARCHAR2(14 BYTE),
  DELETE_FLAG_VAL_INACTIVE VARCHAR2(14 BYTE),
  DELETE_FLAG_VAL_PARTIAL VARCHAR2(14 BYTE),
  HAS_END_DATE          INTEGER          DEFAULT 0,
  END_DATE_COLUMN_NAME  VARCHAR2(30 BYTE),
  END_DATE_VAL          DATE
DEFAULT 'SYSDATE',
  TRAN_HDD_ENABLED      INTEGER          DEFAULT 0
HDD_LANDING_PKEY_COLUMNS VARCHAR2(<TBD> BYTE),
EMPTY_LANDING_TABLE_CHECK_OFF INTEGER    DEFAULT 0
);
```

Microsoft SQL Server의 경우.

```
CREATE TABLE [dbo].[C_REPOS_EXT_HARD_DEL_DETECT]
(
  [ROWID_TABLE] [nchar](14)
  NOT NULL,
  [HDD_ENABLED] [bigint]          NOT NULL,
  [HDD_TYPE] [nvarchar](14)
  NULL,
  [DELETE_FLAG_COLUMN_NAME] [nvarchar](30) NULL,
  [DELETE_FLAG_VAL_ACTIVE] [nvarchar](14) NULL,
  [DELETE_FLAG_VAL_INACTIVE] [nvarchar](14) NULL,
  [DELETE_FLAG_VAL_PARTIAL] [nvarchar](14) NULL,
  [HAS_END_DATE] [bigint]
  NULL,
  [END_DATE_COLUMN_NAME] [nvarchar](30) NULL,
  [END_DATE_VAL] [datetime2](7) NULL,
  [TRAN_HDD_ENABLED] [bigint]
  NULL,
  [HDD_LANDING_PKEY_COLUMNS] [nvarchar](<TBD>) NULL,
  [EMPTY_LANDING_TABLE_CHECK_OFF] [bigint]
  NULL
)
ON [CMX_DATA]
```

- b. 영구 삭제 검색 테이블에 제약 조건을 추가하려면 다음 SQL 문을 실행합니다.

Oracle의 경우.

```
ALTER TABLE C_REPOS_EXT_HARD_DEL_DETECT ADD (
  CHECK (HDD_TYPE IN ('DIRECT','CONSENSUS')));

ALTER TABLE C_REPOS_EXT_HARD_DEL_DETECT ADD (
  CHECK (HDD_ENABLED IN (0,1)));

ALTER TABLE C_REPOS_EXT_HARD_DEL_DETECT ADD (
  CHECK (HAS_END_DATE IN (0,1)));

ALTER TABLE C_REPOS_EXT_HARD_DEL_DETECT ADD (
  UNIQUE (ROWID_TABLE));
```

Microsoft SQL Server의 경우.

```
ALTER TABLE [dbo].[C_REPOS_EXT_HARD_DEL_DETECT] ADD CONSTRAINT
[CH_C_REPOS_EXT_HARD_DEL_DETECT_HDD_TYPE] CHECK (HDD_TYPE IN ('CONSENSUS','DIRECT'))
GO
ALTER TABLE [dbo].[C_REPOS_EXT_HARD_DEL_DETECT] ADD CONSTRAINT
[CH_C_REPOS_EXT_HARD_DEL_DETECT_HDD_ENABLED] CHECK (HDD_ENABLED IN (0,1))
GO
ALTER TABLE [dbo].[C_REPOS_EXT_HARD_DEL_DETECT] ADD CONSTRAINT
[CH_C_REPOS_EXT_HARD_DEL_DETECT_HAS_END_DATE] CHECK (HAS_END_DATE IN (0,1))
GO
ALTER TABLE [dbo].[C_REPOS_EXT_HARD_DEL_DETECT] ADD CONSTRAINT
[UQ_C_REPOS_EXT_HARD_DEL_DETECT_ROWID_TABLE] UNIQUE (ROWID_TABLE)
GO
```

- c. C_REPOS_TABLE의 ROWID_TABLE 열에 영구 삭제 검색 테이블의 ROWID_TABLE 열에 대한 외래 키 제약 조건을 생성하려면 다음 SQL 문을 실행합니다.

Oracle의 경우.

```
ALTER TABLE C_REPOS_EXT_HARD_DEL_DETECT ADD (
CONSTRAINT FK_ROWID_TABLE_FOR_HDD FOREIGN KEY (ROWID_TABLE)
REFERENCES C_REPOS_TABLE (ROWID_TABLE)
ON DELETE CASCADE);
```

Microsoft SQL Server의 경우.

```
ALTER TABLE [dbo].[C_REPOS_EXT_HARD_DEL_DETECT] ADD CONSTRAINT [FK_ROWID_TABLE_FOR_HDD]
FOREIGN KEY (ROWID_TABLE) REFERENCES [dbo].[C_REPOS_TABLE](ROWID_TABLE) ON DELETE CASCADE
GO
```

C_REPOS_TABLE에서 상위 레코드가 제거되는 경우 MDM Hub가 적절한 메타데이터를 삭제하도록 ON DELETE CASCADE 절로 외래 키를 정의합니다.

2. 작업 매트릭스 유형 테이블에 작업 매트릭스 유형 코드를 등록합니다.

다음 SQL 문을 실행하여 작업 매트릭스 유형 코드를 100으로 등록합니다.

Oracle의 경우.

```
INSERT INTO C_REPOS_JOB_METRIC_TYPE
(METRIC_TYPE_CODE, CREATE_DATE, CREATOR, LAST_UPDATE_DATE, UPDATED_BY, METRIC_TYPE_DESC, SEQ)
VALUES
(100, SYSDATE, 'hdd', SYSDATE, 'hdd', 'Flagged as Deleted', 100);
```

Microsoft SQL Server의 경우.

```
INSERT INTO [dbo].[C_REPOS_JOB_METRIC_TYPE]
([METRIC_TYPE_CODE],[CREATE_DATE],[CREATOR],[LAST_UPDATE_DATE],[UPDATED_BY],
[METRIC_TYPE_DESC],[SEQ])
VALUES
(100,getDate(),'HDD',getDate(),'HDD','Flagged as Deleted',100)
GO
```

3. 랜딩 및 준비 테이블을 구성합니다.

- 삭제 플래그 및 종료 날짜가 필요한 기본 개체 레코드를 확인합니다.
- 기본 개체에 데이터를 제공하는 랜딩 및 준비 테이블 열이 모두 포함되어 있는지 확인합니다.
- Hub 콘솔의 **스키마** 도구에서 영구 삭제 검색 테이블의 DELETE_FLAG_COLUMN_NAME에 지정한 값이 표시되는지 확인합니다. 또한 기본값이 A, I 또는 P인지 확인합니다.
- 레코드가 활성 상태이고 레코드에 종료 날짜가 있는 경우 종료 날짜가 null 또는 시스템 날짜인지 확인합니다.
종료 날짜로 null 값을 지정할 경우 종료 날짜 열에 대한 **Null 업데이트 허용** 확인란을 선택합니다.
- Hub 콘솔의 **매핑** 도구에서 랜딩 테이블과 준비 테이블 간에 열을 매핑합니다.

- f. Hub 콘솔의 **스키마** 도구에서 영구 삭제를 검색해야 하는 준비 테이블에 대한 델타 검색을 활성화합니다.

특정 열을 사용하여 델타를 검색하는 옵션을 활성화할 경우 삭제된 레코드를 검색하도록 열 목록에 삭제 플래그 열 이름을 포함합니다.

4. 영구 삭제 검색 테이블에 메타데이터를 정의하려면 다음 SQL 문을 실행합니다.

Oracle의 경우.

```
INSERT INTO C_REPOS_EXT_HARD_DEL_DETECT (
    ROWID_TABLE, HDD_ENABLED, HDD_TYPE,
    DELETE_FLAG_COLUMN_NAME, DELETE_FLAG_VAL_ACTIVE, DELETE_FLAG_VAL_INACTIVE,
    DELETE_FLAG_VAL_PARTIAL, HAS_END_DATE, END_DATE_COLUMN_NAME,
    END_DATE_VAL, TRAN_HDD_ENABLED, HDD_LANDING_PKEY_COLUMNS)
SELECT
    ROWID_TABLE, 1, <Column to enable hard delete detection>, '<hard delete detection type such as
    DIRECT or CONSENSUS>',
    '<name of the delete flag column>', '<Value of delete flag column for active records>', '<Value
    of delete flag column for inactive records>',
    '<Value of delete flag column for partially active records>', <Indicator whether end date is
    used in hard delete detection or not>,
    <End date column name if staging table has end date column>, '<End date value as MM/DD/YYYY>',
    <Transactional Hard Delete Detection indicator>,
    '<Comma-separated list of column names that contribute to the primary key>',
    FROM C_REPOS_TABLE WHERE table_name = '<Staging table name>'
```

Microsoft SQL Server의 경우.

```
INSERT INTO [dbo].[C_REPOS_EXT_HARD_DEL_DETECT]
(
    [ROWID_TABLE]
    , [HDD_ENABLED]
    , [HDD_TYPE]
    , [DELETE_FLAG_COLUMN_NAME]
    , [DELETE_FLAG_VAL_ACTIVE]
    , [DELETE_FLAG_VAL_INACTIVE]
    , [DELETE_FLAG_VAL_PARTIAL]
    , [HAS_END_DATE]
    , [END_DATE_COLUMN_NAME]
    , [END_DATE_VAL]
    , [TRAN_HDD_ENABLED]
    , [HDD_LANDING_PKEY_COLUMNS])
SELECT
    [ROWID_TABLE]
    , 1--Column for which hard delete detection must be enabled
    , '<hard delete detection type such as DIRECT or CONSENSUS>'
    , '<name of the delete flag column>'
    , '<Value of delete flag column for active records>'
    , '<Value of delete flag column for inactive records>'
    , '<Value of delete flag column for partially active records>'
    , '<Indicator whether end date is used in hard delete detection or not>'
    , '<End date column name if staging table has end date column>'
    , '<End date value>'
    , '<Transactional Hard Delete Detection indicator>'
    , '<Comma-separated list of column names that contribute to the primary key>'
FROM [dbo].[C_REPOS_TABLE] WHERE table_name = '<Staging table name>'
GO
```

5. 영구 삭제 검색에 대한 사용자 종료를 구현합니다.

- a. 사후 랜딩 및 사후 준비 **Java** 사용자 종료를 구현하고 영구 삭제 기능을 호출하기 위한 논리를 구현에 추가합니다.

구현할 사용자 종료 인터페이스는 `mdm-ue.jar`에 있습니다.

- b. 구현된 사후 랜딩 및 사후 준비 사용자 종료 클래스를 **JAR** 파일로 패키징합니다.

- c. 사용자 종료를 등록하여 패키징된 **JAR** 파일을 **MDM Hub**에 업로드합니다.

영구 삭제 검색을 위한 기본 키 열 지정

함수를 사용하여 여러 소스 열에서 기본 키를 생성할 경우 영구 삭제 검색 프로세스는 함수에 대한 입력을 확인하여 소스 열을 식별합니다. 영구 삭제 검색 프로세스는 함수에 대한 입력이 되는 모든 소스 열에서 실행됩니다.

참고: 함수에 대한 모든 입력이 기본 키에 영향을 주는 경우 이 과정을 건너뛸 수 있습니다.

함수에 대한 입력에 기본 키에 사용되지 않는 소스 열이 포함될 경우 기본 키에 영향을 주는 열 이름 목록을 생성합니다. 열 목록을 **C_REPOS_EXT_HARD_DEL_DETECT** 리포지토리 테이블에 추가합니다. 영구 삭제 검색 프로세스가 목록에 있는 열에서 실행됩니다.

1. **MDM Hub**가 기본 키를 생성하는 데 사용하는 랜딩 테이블의 열을 식별합니다. 쉼표로 구분된 열 이름 목록을 생성합니다.
2. **SQL** 도구에서 영구 삭제 검색을 구성할 때 생성한 **C_REPOS_EXT_HARD_DEL_DETECT** 리포지토리 테이블을 엽니다.
3. 리포지토리 테이블을 생성할 때 **HDD_LANDING_PKEY_COLUMNS** 열을 추가하지 않은 경우 열을 추가합니다. 열 유형을 **VARCHAR**로 설정하고 쉼표로 구분된 열 이름 목록을 포함할 수 있는 길이를 입력합니다.
4. 열에 쉼표로 구분된 열 이름 목록을 추가합니다.

준비 작업이 실행되면 유효성 검사 프로세스가 열 이름 목록을 확인합니다. 유효성 검사 프로세스는 공백을 무시하고, 중복되는 열 이름을 제거하고, 랜딩 테이블의 열 이름과 일치하는 열 이름을 확인합니다.

직접 삭제

MDM Hub는 단일 소스 시스템을 기반으로 영구 삭제된 레코드를 검색합니다. **MDM Hub**는 소스 시스템 레코드가 삭제되거나 교차 참조 레코드 중 하나가 삭제된 경우 기본 개체 레코드에 비활성 플래그를 지정합니다.

소스 데이터를 단일 소스 시스템에서 가져온 경우 직접 삭제를 수행합니다. 한 소스 시스템에서 레코드의 삭제 상태를 결정해야 하는 경우에도 직접 삭제를 수행할 수 있습니다. **MDM Hub**는 해당 소스 시스템의 최신 상태를 기반으로 삭제 상태를 결정합니다. 사후 랜딩 사용자 종료로 통해 직접 삭제를 실행할 수 있습니다.

직접 삭제 플래그 지정을 위한 영구 삭제 검색 구성

직접 삭제 플래그 지정을 설정하도록 영구 삭제 검색 테이블을 구성해야 합니다. 직접 영구 삭제 검색을 수행하려면 삭제 플래그 값, 레코드의 종료 날짜(선택 사항) 및 일부 기본 개체 속성이 필요합니다.

직접 삭제 플래그 지정에 대해 다음 메타데이터를 설정할 수 있습니다.

삭제 플래그

직접 삭제를 사용하여 영구 삭제 항목을 검색하려면 다음 삭제 플래그를 사용합니다.

- **DELETE_FLAG_VAL_ACTIVE = A**
- **DELETE_FLAG_VAL_INACTIVE = I**
- **HDD_ENABLED = 1**
- **HDD_TYPE = DIRECT**

종료 날짜

HAS_END_DATE = 1로 설정한 경우 종료 날짜 값 **END_DATE_VAL**을 지정합니다. 레코드의 종료 날짜는 과거 또는 미래의 날짜일 수 있습니다. 종료 날짜의 형식은 **MM/DD/YYYY**입니다.

기본 개체 열

삭제 플래그 열 이름과 종료 날짜 열 이름을 포함하여 기본 개체의 열을 지정합니다.

종료 날짜를 사용한 직접 삭제를 위한 영구 삭제 검색 설정

영구 삭제 항목을 검색하기 위한 일반적인 구성을 완료한 후에는 준비 테이블에 대해 종료 날짜를 사용한 직접 삭제 프로시저를 설정합니다.

1. MDM Hub가 영구 삭제 항목을 검색하도록 구성합니다.
2. MDM Hub가 영구 삭제 항목을 검색해야 하는 준비 작업을 식별합니다.
3. 준비 테이블 세부 정보와 기타 필요한 메타데이터를 영구 삭제 검색 테이블에 삽입합니다.

종료 날짜를 사용한 직접 삭제 코드 예제

조직에서 C_CUSTOMER_STG라는 준비 테이블에 대한 영구 삭제를 검색해야 한다고 가정합니다. 이 조직에서는 기본 개체의 영구 삭제된 레코드에 대한 직접 삭제를 수행하고 삭제된 레코드에 종료 날짜를 할당하려고 합니다.

영구 삭제를 검색하는 데 필요한 메타데이터를 C_CUSTOMER_STG 준비 테이블의 영구 삭제 검색 테이블에 삽입해야 합니다. 영구 삭제를 검색할 때 MDM Hub에서는 삭제된 레코드의 종료 날짜뿐 아니라 삭제 플래그도 제공해야 합니다. 이 예에서 삭제 플래그 열의 이름은 DEL_CODE이고 준비 테이블의 종료 날짜 열 이름은 END_DATE입니다.

다음 샘플 코드에서는 삭제된 레코드의 종료 날짜를 사용하여 직접 삭제를 수행하기 위한 메타데이터를 영구 삭제 검색 테이블에 삽입합니다.

Oracle의 경우.

```
INSERT into c_repos_ext_hard_del_detect
(rowid_table, hdd_enabled, hdd_type, delete_flag_column_name, delete_flag_val_active,
delete_flag_val_inactive, has_end_date, end_date_column_name, end_date_val)
SELECT rowid_table,
1 AS hdd_enabled,
DIRECT AS hdd_type,
'DEL_CODE' AS delete_flag_column_name,
'A' AS delete_flag_val_active,
'I' AS delete_flag_val_inactive,
1 AS has_end_date,
'END_DATE' AS end_date_column_name,
'SYSDATE' AS end_date_val
FROM c_repos_table
WHERE table_name = 'C_CUSTOMER_STG';
```

Microsoft SQL Server의 경우.

```
INSERT INTO [dbo].[C_REPOS_EXT_HARD_DEL_DETECT]
([ROWID_TABLE], [HDD_ENABLED], [HDD_TYPE], [DELETE_FLAG_COLUMN_NAME],
[DELETE_FLAG_VAL_ACTIVE], [DELETE_FLAG_VAL_INACTIVE], [HAS_END_DATE], [END_DATE_COLUMN_NAME],
[END_DATE_VAL],)
SELECT [ROWID_TABLE]
,1
,'DIRECT'
,'DEL_CODE'
,'A'
,'I'
,1
,'END_DATE'
,GETDATE()
FROM [dbo].[C_REPOS_TABLE]
WHERE table_name = 'C_CUSTOMER_STG';
GO
```

직접 삭제를 위한 영구 삭제 검색 예

조직에서 기본 개체 레코드에 데이터를 제공하는 단일 소스 시스템이 있다고 가정합니다. 소스 시스템에서 영구 삭제 항목을 검색하여 기본 개체 레코드에 삭제 플래그를 지정해야 합니다. 소스 시스템은 레코드를 삭제하고 레코드를 다시 활성화한 후 다시 활성화된 레코드를 삭제합니다. MDM Hub는 그에 따라 소스 레코드에 삭제 플래그와 종료 날짜를 지정합니다.

영구 삭제 검색 전의 레코드

MDM Hub가 소스 시스템에서 영구 삭제 항목을 검색하기 전에 기본 개체 레코드와 교차 참조 레코드의 삭제 플래그는 활성 상태입니다.

다음은 소스에서 영구 삭제 항목이 검색되기 전의 교차 참조 레코드입니다.

소스	고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
1	25	Robert	James	Jones	A	null

다음은 소스에서 영구 삭제 항목이 검색되기 전의 기본 개체 레코드입니다.

고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
25	Robert	James	Jones	A	null

소스에서 영구 삭제 항목이 검색될 때의 레코드

소스 시스템에 영구 삭제 항목이 있는 경우 MDM Hub는 다음 태스크를 수행합니다.

1. 랜딩 테이블과 이전 랜딩 테이블을 비교하여 삭제된 레코드를 검색합니다.
2. 삭제된 레코드의 열 값을 삭제 플래그 및 종료 날짜와 함께 랜딩 테이블에 다시 삽입합니다.
3. 교차 참조 레코드를 종료 날짜가 있는 비활성 상태로 업데이트합니다.

종료 날짜에는 시스템 날짜나 사용자가 종료 날짜 값을 결정할 때 제공한 날짜가 포함됩니다.

다음은 소스에서 영구 삭제 항목이 검색될 때의 교차 참조 레코드입니다.

소스	고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
1	25	Robert	James	Jones	I	04/05/13

다음은 소스에서 영구 삭제 항목이 검색될 때의 기본 개체 레코드입니다.

고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
25	Robert	James	Jones	I	04/05/13

소스 레코드가 다시 활성화된 후의 레코드

소스 시스템의 레코드가 다시 활성화된 경우 MDM Hub는 기본 개체 레코드와 교차 참조 레코드에 활성 플래그를 지정합니다. 소스 레코드가 활성 상태일 때 종료 날짜는 null입니다.

다음은 소스 시스템의 영구 삭제된 레코드가 다시 활성화된 후의 교차 참조 레코드입니다.

소스	고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
1	25	Robert	James	Jones	A	null

다음은 소스 시스템의 영구 삭제된 레코드가 다시 활성화된 후의 기본 개체 레코드입니다.

고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
25	Robert	James	Jones	A	null

소스에서 영구 삭제 항목이 다시 검색될 때의 레코드

소스 시스템에 영구 삭제 항목이 있는 경우 MDM Hub는 다음 태스크를 수행합니다.

1. 랜딩 테이블과 이전 랜딩 테이블을 비교하여 삭제된 레코드를 검색합니다.
2. 삭제된 레코드의 열 값을 삭제 플래그 및 종료 날짜와 함께 랜딩 테이블에 다시 삽입합니다.
3. 교차 참조 레코드를 종료 날짜가 있는 비활성 상태로 업데이트합니다.

종료 날짜에는 시스템 날짜나 사용자가 종료 날짜 값을 결정할 때 제공한 날짜가 포함됩니다.

다음은 소스에서 영구 삭제 항목이 검색될 때의 교차 참조 레코드입니다.

소스	고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
1	25	Robert	James	Jones	I	04/05/13

다음은 소스에서 영구 삭제 항목이 검색될 때의 기본 개체 레코드입니다.

고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
25	Robert	James	Jones	I	04/05/13

합의 삭제

MDM Hub는 모든 관련 소스 시스템에 있는 레코드의 삭제 상태를 기반으로 영구 삭제된 레코드를 검색합니다. 합의 삭제는 직접 삭제와 함께 사용됩니다.

MDM Hub는 기본 개체 레코드가 모든 관련 소스 시스템에서 삭제된 경우 해당 레코드에 비활성 플래그를 지정합니다. 기본 개체 레코드와 연결된 모든 교차 참조 레코드가 부분 비활성 상태여야 합니다.

소스 데이터를 여러 소스 시스템에서 가져온 경우 합의 삭제를 수행합니다. MDM Hub는 모든 소스 시스템에서 레코드 삭제가 합의된 경우 해당 레코드에 삭제 플래그를 지정합니다. MDM Hub는 소스 시스템의 레코드에 부분 비활성 플래그를 지정합니다. 그런 다음 MDM Hub는 트러스트 및 유효성 검사 규칙을 사용하여 기본 개체 레코드를 덮어쓸 다른 소스 시스템 레코드를 확인합니다. 삭제 날짜가 있으면 MDM Hub는 삭제 날짜를 사용하여 기본 개체 레코드를 덮어쓸 다른 소스 시스템 레코드를 확인합니다.

모든 소스 시스템에서 레코드의 삭제 상태에 합의해야 하는 경우 사후 랜딩 사용자 종료를 구현하여 직접 삭제를 호출합니다. 또한 사후 준비 사용자 종료를 통해 기본 개체 레코드의 합의 삭제에 대한 합의 삭제 논리를 호출해야 합니다.

합의 삭제 플래그 지정을 위한 영구 삭제 검색 구성

합의 삭제 플래그 지정을 설정하도록 영구 삭제 검색 테이블을 구성해야 합니다. 합의 영구 삭제 검색을 수행하려면 삭제 플래그 값, 레코드의 종료 날짜 및 일부 기본 개체 속성이 필요합니다.

합의 삭제 플래그 지정에 대해 다음 메타데이터를 설정할 수 있습니다.

삭제 플래그

합의 삭제를 수행하려면 다음 삭제 플래그를 사용합니다.

- DELETE_FLAG_VAL_ACTIVE = A
- DELETE_FLAG_VAL_INACTIVE = I
- DELETE_FLAG_VAL_PARTIAL = P
- HDD_ENABLED = 1

- HDD_TYPE = CONSENSUS

종료 날짜

HAS_END_DATE = 1로 설정한 경우 종료 날짜 값 END_DATE_VAL을 지정합니다. 레코드의 종료 날짜는 과거 또는 미래의 날짜일 수 있습니다. 형식은 MM/DD/YYYY입니다.

기본 개체 열

삭제 플래그 열 이름과 종료 날짜 열 이름을 포함하여 기본 개체의 열을 지정합니다.

종료 날짜를 사용한 합의 삭제를 위한 영구 삭제 검색 설정

영구 삭제 항목을 검색하기 위한 일반적인 구성을 완료한 후에는 준비 테이블에 대해 종료 날짜를 사용한 합의 삭제 프로시저를 설정합니다.

1. MDM Hub가 영구 삭제 항목을 검색하도록 구성합니다.
2. MDM Hub가 영구 삭제 항목을 검색해야 하는 준비 작업을 식별합니다.
3. 준비 테이블 세부 정보와 기타 필요한 메타데이터를 영구 삭제 검색 테이블에 삽입합니다.
4. 스키마 도구에서 영구 삭제된 모든 레코드에 대한 삭제 플래그 값이 포함되어 있는 열 이름에 대해 **트러스트** 및 **유효성 검사** 확인란을 선택합니다.
5. 종료 날짜를 사용하는 경우 종료 날짜 열 이름에 대해 **트러스트** 및 **유효성 검사** 확인란을 활성화합니다.
6. 시스템 및 트러스트 도구에서 각 소스 시스템의 종료 날짜 열 이름 및 삭제 플래그 열 이름에 대한 트러스트 설정을 지정합니다.

붕괴 값은 기본 개체의 모든 소스 시스템에 대해 동일해야 합니다.

7. 스키마 도구에서 영구 삭제 검색을 위해 구성된 각 기본 개체의 삭제 플래그 열 이름에 대한 유효성 검사 규칙을 설정합니다. 부분 삭제 플래그의 트러스트 값은 활성 또는 완전 삭제 플래그의 트러스트 값보다 낮게 설정해야 합니다.

- a. 기본 개체에 대해 **유효성 검사 규칙 설정**을 선택합니다.

유효성 검사 규칙 페이지가 나타납니다.

- b. **유효성 검사 규칙 추가** 단추를 클릭합니다.

유효성 검사 규칙 추가 대화 상자가 나타납니다.

- c. 규칙 이름을 입력하고 목록에서 **사용자 지정** 규칙 유형을 선택합니다.

참고: 규칙 이름에 쉼표를 삽입하지 마십시오. 쉼표를 사용할 경우 유효성 검사 규칙의 순서에 영향을 미칠 수 있습니다.

- d. 다운로드 백분율을 선택하고 **규칙 SQL** 섹션에서 다음 형식으로 규칙을 지정합니다.

WHERE S.<Delete flag column name> LIKE '<Value of delete flag column for partial delete>'

열에 대해 **최소 트러스트 보유**를 활성화하면 안 됩니다.

종료 날짜를 사용한 합의 삭제 코드 예제

조직에서 C_CUSTOMER_STG라는 준비 테이블에 대한 영구 삭제를 검색해야 한다고 가정합니다. 이 조직에서는 기본 개체의 영구 삭제된 레코드에 대한 합의 삭제를 수행하고 삭제된 레코드에 종료 날짜를 할당하려고 합니다.

영구 삭제를 검색하는 데 필요한 메타데이터를 C_CUSTOMER_STG 준비 테이블의 영구 삭제 검색 테이블에 삽입해야 합니다. 영구 삭제를 검색할 때 MDM Hub에서는 삭제된 레코드의 종료 날짜뿐 아니라 삭제 플래그도 제공해야 합니다. 또한 이 프로세스에서는 MDM Hub가 기본 개체 레코드에 삭제 플래그를 지정하기 전에 연결된 모든 교차 참조 레코드에 삭제 플래그가 지정되었는지 확인해야 합니다. 삭제 플래그 열의 이름은 DEL_CODE이고 준비 테이블의 종료 날짜 열 이름은 END_DATE입니다.

다음 샘플 코드에서는 삭제된 레코드의 종료 날짜를 사용하여 합의를 삭제 수행하기 위한 메타데이터를 영구 삭제 검색 테이블에 삽입합니다.

Oracle의 경우.

```
INSERT into c_repos_ext_hard_del_detect
(rowid_table, hdd_enabled, hdd_type, delete_flag_column_name, delete_flag_val_active,
delete_flag_val_inactive, delete_flag_val_partial, has_end_date, end_date_column_name, end_date_val)
SELECT rowid_table,
1 AS hdd_enabled,
'CONSENSUS' AS hdd_type,
'DEL_CODE' AS delete_flag_column_name,
'A' AS delete_flag_val_active,
'I' AS delete_flag_val_inactive,
'P' AS delete_flag_val_partial,
1 AS has_end_date,
'END_DATE' AS end_date_column_name,
'SYSDATE' AS end_date_val
FROM c_repos_table
WHERE table_name = 'C_CUSTOMER_STG';
```

Microsoft SQL Server의 경우.

```
INSERT INTO [dbo].[C_REPOS_EXT_HARD_DEL_DETECT]
([ROWID_TABLE], [HDD_ENABLED], [HDD_TYPE], [DELETE_FLAG_COLUMN_NAME],
[DELETE_FLAG_VAL_ACTIVE], [DELETE_FLAG_VAL_INACTIVE], [DELETE_FLAG_VAL_PARTIAL], [HAS_END_DATE],
[END_DATE_COLUMN_NAME], [END_DATE_VAL],)
SELECT [ROWID_TABLE]
,1
,'CONSENSUS'
,'DEL_CODE'
,'A'
,'I'
,'P'
,1
,'END_DATE'
,'SYSDATE'
FROM [dbo].[C_REPOS_TABLE]
WHERE table_name = 'C_CUSTOMER_STG';
GO
```

합의 삭제를 위한 영구 삭제 검색 예

조직에서 기본 개체 레코드에 데이터를 제공하는 소스 시스템이 여러 개 있다고 가정합니다. 각 소스 시스템의 레코드는 차례대로 삭제된 후 소스 시스템 1의 레코드가 다시 활성화되고 다시 영구 삭제됩니다. **MDM Hub**는 소스 시스템에서 영구 삭제 항목을 검색해야 합니다. 레코드가 모든 소스 시스템에서 영구 삭제된 경우 기본 개체 레코드에는 삭제 플래그가 지정됩니다.

MDM Hub는 삭제된 소스 레코드의 트러스트 다운그레이드를 기반으로 기본 개체 레코드의 열 값을 업데이트합니다. 모든 소스 시스템의 레코드가 영구 삭제되어야 하며 **MDM Hub**는 연결된 모든 교차 참조 레코드에 부분 비활성 또는 비활성 플래그를 지정해야 합니다.

영구 삭제 검색 전의 레코드

소스 시스템에서 영구 삭제 항목을 검색하기 전에 기본 개체 레코드와 교차 참조 레코드의 삭제 플래그는 활성 상태입니다. **MDM Hub**가 세 개의 소스 레코드를 하나의 기본 개체 레코드에 병합합니다. 이름은 소스 시스템 2에서 제공되고 중간 이름은 소스 시스템 3에서 제공되며 성은 소스 시스템 1에서 제공됩니다.

다음은 소스에서 영구 삭제 항목이 검색되기 전의 교차 참조 레코드입니다.

소스	고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
1	25	Bob	J	Jones	A	null
2	163	Robert	J	Jones	A	null
3	163	Bob	James	Jones	A	null

다음은 소스에서 영구 삭제 항목이 검색되기 전의 기본 개체 레코드입니다.

고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
25	Robert	James	Jones	A	null

소스 3에서 영구 삭제 항목이 검색될 때의 레코드

소스 시스템 3에 영구 삭제 항목이 있는 경우 MDM Hub는 다음 태스크를 수행합니다.

1. 랜딩 테이블에서 삭제된 소스 레코드를 검색합니다.
2. 삭제된 레코드의 열 값을 비활성 삭제 플래그 및 종료 날짜와 함께 랜딩 테이블에 다시 삽입합니다.
3. 모든 교차 참조 레코드가 비활성 또는 부분 비활성 상태인지 확인하고 연결된 교차 참조 레코드를 부분 비활성 상태로 업데이트합니다.
4. 트러스트 점수가 더 높은 소스 레코드가 해당 열에 대한 업데이트를 제공할 때까지 소스 시스템 3의 레코드에서 제공된 중간 이름을 기본 개체에 유지합니다.

다음은 소스 시스템 3에서 영구 삭제 항목이 검색될 때의 교차 참조 레코드입니다.

소스	고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
1	25	Bob	J	Jones	A	null
2	163	Robert	J	Jones	A	null
3	163	Bob	James	Jones	P	null

다음은 소스 시스템 3에서 영구 삭제 항목이 검색될 때의 기본 개체 레코드입니다.

고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
25	Robert	James	Jones	A	null

소스 1에서 영구 삭제 항목이 검색될 때의 레코드

소스 시스템 1에 영구 삭제 항목이 있는 경우 MDM Hub는 다음 태스크를 수행합니다.

1. 랜딩 테이블에서 삭제된 소스 레코드를 검색합니다.
2. 삭제된 레코드의 열 값을 삭제 플래그 및 종료 날짜와 함께 랜딩 테이블에 다시 삽입합니다.
3. 모든 교차 참조 레코드가 비활성 또는 부분 비활성 상태인지 확인하고 연결된 교차 참조 레코드를 부분 비활성 상태로 업데이트합니다.
4. BVT가 계산되고 중간 이름 John에 대해 소스 1의 값이 기본 개체에 반영됩니다. 이는 해당 레코드가 부분적으로 삭제되었더라도 트러스트 점수는 더 높기 때문입니다.

다음은 소스 시스템 1에서 영구 삭제 항목이 검색될 때의 교차 참조 레코드입니다.

소스	고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
1	25	Bob	John	Jones	P	null
2	163	Robert	J	Jones	A	null
3	163	Bob	James	Jones	P	null

다음은 소스 시스템 1에서 영구 삭제 항목이 검색될 때의 기본 개체 레코드입니다.

고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
25	Robert	John	Jones	A	null

소스 2에서 영구 삭제 항목이 검색될 때의 레코드

소스 시스템 2에 영구 삭제 항목이 있는 경우 MDM Hub는 다음 태스크를 수행합니다.

1. 랜딩 테이블에서 삭제된 소스 레코드를 검색합니다.
2. 삭제된 레코드를 삭제 플래그 및 종료 날짜와 함께 랜딩 테이블에 다시 삽입합니다.

- 모든 교차 참조 레코드가 비활성 또는 부분 비활성 상태인지 확인하고 연결된 교차 참조 레코드를 비활성 상태로 업데이트합니다.
이때 다른 소스 시스템은 활성 상태를 제공할 수 없습니다.
- 다른 소스 시스템이 활성 레코드를 제공할 수 없는 경우 기본 개체 레코드에는 종료 날짜가 있는 비활성 상태가 반영됩니다.

종료 날짜에는 시스템 날짜나 사용자가 종료 날짜 값을 결정할 때 제공한 날짜가 포함됩니다.

다음은 소스 시스템 2에서 영구 삭제 항목이 검색될 때의 교차 참조 레코드입니다.

소스	고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
1	25	Bob	John	Jones	P	null
2	163	Robert	J	Jones	I	04/05/13
3	163	Bob	James	Jones	P	null

다음은 소스 시스템 2에서 영구 삭제 항목이 검색될 때의 기본 개체 레코드입니다.

고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
25	Robert	John	Jones	I	04/05/13

소스 1의 레코드가 다시 활성화된 후의 레코드

소스 시스템의 레코드가 다시 활성화된 경우 MDM Hub는 기본 개체 레코드와 교차 참조 레코드에 활성 플래그를 지정합니다. 소스 레코드가 활성 상태일 때 종료 날짜는 null입니다.

소스 시스템 1의 소스 레코드가 다시 활성화된 경우 MDM Hub는 연결된 교차 참조 레코드에 활성 플래그를 지정합니다. 레코드가 활성 상태일 때 종료 날짜는 null입니다.

다음은 소스 시스템 1의 영구 삭제된 레코드가 다시 활성화된 후의 교차 참조 레코드입니다.

소스	고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
1	25	Bob	John	Jones	A	null
2	163	Robert	J	Jones	I	04/05/13
3	163	Bob	James	Jones	P	null

다음은 소스 시스템 1의 영구 삭제된 레코드가 다시 활성화된 후의 기본 개체 레코드입니다.

고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
25	Robert	John	Jones	A	null

소스 1에서 영구 삭제 항목이 다시 검색될 때의 레코드

소스 시스템 1에 영구 삭제 항목이 있는 경우 MDM Hub는 다음 태스크를 수행합니다.

- 랜딩 테이블에서 삭제된 소스 레코드를 검색합니다.
- 삭제된 레코드의 열 값을 삭제 플래그 및 종료 날짜와 함께 랜딩 테이블에 다시 삽입합니다.
- 모든 교차 참조 레코드가 비활성 또는 부분 비활성 상태인지 확인하고 연결된 교차 참조 레코드를 비활성 상태로 업데이트합니다.
이때 다른 소스 시스템은 활성 상태를 제공할 수 없습니다.
- 기본 개체 레코드에는 종료 날짜가 있는 비활성 상태가 반영됩니다.

종료 날짜에는 시스템 날짜나 사용자가 종료 날짜 값을 결정할 때 제공한 날짜가 포함됩니다.

다음은 소스 시스템 1에서 영구 삭제 항목이 다시 검색될 때의 교차 참조 레코드입니다.

소스	고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
1	25	Bob	John	Jones	I	04/05/13

소스	고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
2	163	Robert	J	Jones	I	04/05/13
3	163	Bob	James	Jones	P	null

다음은 소스 시스템 1에서 영구 삭제 항목이 다시 검색될 때의 기본 개체 레코드입니다.

고객 ID	이름	중간 이름	성	Del_Code	종료 날짜
25	Robert	John	Jones	I	04/05/13

사용자 종료 내에서 영구 삭제 검색 사용

소스 시스템에서 영구 삭제된 레코드를 검색하도록 사후 랜딩 및 사후 준비 사용자 종료를 구현할 수 있습니다. 직접 영구 삭제 검색을 수행하려면 사후 랜딩 사용자 종료와 종료 날짜가 필요합니다. 합의 영구 삭제 검색을 수행하려면 사후 랜딩 사용자 종료와 사후 준비 사용자 종료 날짜가 모두 필요합니다.

1. 사후 랜딩 사용자 종료 구현 내에서 사용할 **HardDeleteDetection** 클래스의 인스턴스를 생성합니다.

HardDeleteDetection 클래스는 다음 디렉터리의 **mdm-ue.jar**에서 사용 가능합니다.

Windows의 경우. <infamdm 설치 디렉터리>\hub\server\lib

UNIX의 경우. <infamdm 설치 디렉터리>/hub/server/lib

2. 사후 랜딩 또는 사후 준비 사용자 종료에 대한 Java 코드에 영구 삭제된 레코드를 검색하기 위한 다음 행을 추가합니다.

- 사후 랜딩 사용자 종료를의 경우.

```
public void processUserExit(UserExitContext userExitContext, String stagingTableName, String
landingTableName, String previousLandingTableName)
    throws Exception
{
    HardDeleteDetection hdd = new HardDeleteDetection(userExitContext.getBatchJobRowid(),
stagingTableName);
    hdd.startHardDeleteDetection(userExitContext.getDBConnection());
}
```

- 사후 준비 사용자 종료를의 경우.

```
public void processUserExit(UserExitContext userExitContext, String stagingTableName, String
landingTableName, String previousLandingTableName)
    throws Exception
{
    ConsensusFlagUpdate consensusProcess = new
ConsensusFlagUpdate(userExitContext.getBatchJobRowid(), stagingTableName);
    consensusProcess.startConsensusFlagUpdate(userExitContext.getDBConnection());
}
```

3. 사용자 종료 JAR을 패키징한 후 MDM Hub에 업로드합니다.

4. 준비 작업을 실행합니다.

MDM Hub에서는 준비 작업을 통해 사용자 종료 날짜가 호출될 때 영구 삭제 항목을 검색하기 위한 입력 매개 변수 값을 제공합니다.

관련 항목:

- [“사용자 종료 JAR 파일 구현” 페이지 567](#)
- [“사용자 종료를 MDM Hub로 업로드” 페이지 567](#)

제 19 장

데이터 정리 구성

이 장에 포함된 항목:

- [데이터 정리 구성 개요, 327](#)
- [MDM Hub에서 데이터 정리 설정, 327](#)
- [데이터 정리에 대해 처리 서버 구성, 328](#)
- [정리 함수 구성, 332](#)
- [함수 테스트, 340](#)
- [정리 함수에 조건 사용, 341](#)
- [정리 목록 구성, 342](#)
- [Informatica 플랫폼에서 데이터 정리 설정, 346](#)
- [맵릿에 변환 추가, 347](#)
- [매핑 구성, 348](#)

데이터 정리 구성 개요

데이터 정리를 구성하여 준비 프로세스 동안 데이터를 정리하고 표준화할 수 있습니다. MDM Hub에서 준비 프로세스를 실행하는 경우 MDM Hub에서 데이터를 정리할 수 있습니다. Informatica 플랫폼에서 준비 프로세스를 실행하는 경우 개발자 도구에서 데이터 정리를 구성할 수 있습니다.

MDM Hub에서 데이터 정리를 수행하기 전에 정리 엔진을 설치하고 구성합니다. 또한 MDM Hub 구현에 대해 처리 서버를 구성해야 합니다. 처리 서버는 데이터를 정리하고 일괄 작업을 처리하는 서블릿입니다. 레코드의 데이터 값을 표준화하거나 확인하는 정리 함수를 구성할 수 있습니다.

Informatica 플랫폼에서 데이터 정리를 수행하려면 데이터 통합 서비스, 모델 리포지토리 서비스, Informatica Developer(개발자 도구)를 설치하고 구성합니다. 소스 데이터를 MDM Hub 준비 테이블로 전송하기 전에 이러한 데이터를 정리하려면 생성한 맵릿 또는 매핑에 변환을 추가합니다.

정리된 데이터에 대해 일치를 실행하는 경우 MDM Hub에서 신뢰할 수 있는 일치 항목을 더 많이 생성합니다.

MDM Hub에서 데이터 정리 설정

MDM Hub에서 데이터를 정리를 설정하려면 다음 태스크를 수행합니다.

- 데이터 정리에 대해 처리 서버를 구성합니다.

- 정리 함수를 구성합니다.
- 정리 목록을 구성합니다.

데이터 정리에 대해 처리 서버 구성

MDM Hub 구현에 대해 처리 서버를 구성할 수 있습니다.

처리 서버는 데이터를 정리하고 일괄 작업을 처리하는 서블릿입니다. 응용 프로그램 서버 환경에서 처리 서버를 배포할 수 있습니다.

처리 서버는 각 인스턴스에서 여러 요청을 동시에 처리할 수 있는 다중 스레드입니다.

여러 처리 서버를 MDM Hub의 각 연산 참조 저장소에 대해 실행할 수 있습니다. 정리 프로세스는 일반적으로 CPU에 바인딩되어 있습니다. 이 확장 가능한 아키텍처를 사용하여 데이터 양이 증가함에 따라 MDM Hub 구현을 확장할 수 있습니다. 여러 호스트에 처리 서버를 배포하여 여러 CPU에 걸쳐 처리 로드를 분산하고 일괄 작업을 병렬로 실행할 수 있습니다. 또한 일부 외부 어댑터는 본래 단일 스레드 방식이므로 MDM Hub 아키텍처는 각 응용 프로그램 서버 인스턴스에 대해 하나의 처리 스레드를 실행하여 다중 스레드 작업을 시뮬레이션할 수 있습니다.

참고: 동일한 호스트에 있거나 다른 호스트에 있을 수 있는 각기 다른 응용 프로그램 서버 인스턴스에 여러 처리 서버 인스턴스를 설치한 후 MDM Hub에서 등록할 수 있습니다. MDM Hub에서 처리 서버의 단일 설치를 각기 다른 포트의 여러 처리 서버 인스턴스로 등록하면 안 됩니다.

정리 작업 모드

다음 모드에 따라 정리 작업을 분류할 수 있습니다.

- 온라인 및 일괄
- 온라인만
- 일괄만

이 모드를 사용하여 특정 처리 서버에서 실행되는 작업의 클래스를 지정할 수 있습니다. 두 개의 처리 서버를 배포한 경우 하나는 일괄 전용으로, 나머지 하나는 온라인 전용으로 설정하거나 둘 모두에서 모든 요청 클래스를 수락하도록 설정할 수 있습니다. 별도로 클래스를 지정하지 않으면 처리 서버가 기본적으로 일괄 요청과 온라인 요청을 둘 다 실행하도록 설정됩니다.

배포 데이터 정리

여러 개의 처리 서버를 병렬로 실행하여 정리 작업 및 유사 항목 일치 프로세스의 처리량을 높일 수 있습니다. 지정된 작업 크기가 충족되거나 초과하는 경우 MDM Hub은 작업을 처리 서버 간에 분배합니다.

각 처리 서버에 대해 분산 데이터 정리를 설정하려면 `cmxcleanse.properties` 파일에서 다음 속성을 설정합니다.

`cmx.server.match.distributed_match`

선택 사항입니다. 수동으로 추가해야 합니다. 처리 서버에서 정리 작업 및 유사 항목 일치 프로세스를 위한 분산 처리 환경에 참여할지 지정합니다. 기본값은 1이며, 이 경우 활성화됩니다. 분산 처리를 비활성화하려면 0으로 설정합니다.

여러 처리 서버 구성에 대한 자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

cmx.server.cleansize.min_size_for_distribution

선택 사항입니다. 수동으로 추가해야 합니다. 처리 서버 간에 작업을 분배할 수 있는 크기를 지정합니다. 기본값은 1000입니다.

Hub 콘솔에서 평소대로 처리 서버 속성을 구성합니다.

정리 요청

모든 정리 요청은 일괄 API에 의해 실행됩니다.

일괄 API는 정리 요청을 XML 페이로드로 패키징화하여 처리 서버로 보냅니다. 처리 서버는 요청을 받으면 XML을 구문 분석하고 적절한 코드를 호출합니다.

모드 유형	설명
온라인 작업	결과가 XML 응답으로 패키징되어 HTTP POST 연결을 통해 다시 보내집니다.
일괄 작업	처리 서버가 플랫폼 파일로 처리할 데이터를 가져와 처리한 후 대량 로더를 사용하여 데이터를 다시 씁니다. <ul style="list-style-type: none">- Oracle용 대량 로더는 SQL*Loader 유틸리티입니다.- IBM DB2용 대량 로더는 IBM DB2 로드 유틸리티입니다.- Microsoft SQL Server용 대량 로더에서는 Java 데이터베이스 연결 드라이버를 사용합니다.

처리 서버는 다중 스레드로 실행되므로 각 인스턴스에서 여러 요청을 동시에 처리할 수 있습니다. 처리 서버에 보낸 일괄 요청의 기본 제한 시간은 1년이고, 온라인 요청의 기본 제한 시간은 1분입니다.

준비 또는 일치 작업을 실행할 때 등록된 처리 서버가 둘 이상이고 준비 또는 일치 대상인 총 레코드 수가 500개를 넘으면 사용 가능한 처리 서버 간에 작업이 병렬로 분산됩니다.

처리 서버 도구 시작

처리 서버 정보(이름, 포트, 서버 유형 및 처리 서버의 온라인/오프라인 여부 등)를 보려면 처리 서버 도구를 시작합니다.

Hub 콘솔에서 처리 서버 도구를 시작하려면 유틸리티 작업 영역을 확장한 다음 **처리 서버**를 클릭합니다. 처리 서버 도구에 구성된 처리 서버 목록이 표시됩니다.

처리 서버 속성

선택한 처리 서버에 대한 서버 및 포트를 식별합니다. 다른 속성은 서버에서 처리하는 프로세스의 유형, 사용할 스레드 수와 같은 처리 서버의 동작을 제어합니다.

처리 서버에서 다음 프로세스 유형을 활성화 또는 비활성화할 수 있습니다.

- 정리 작업
- 유사 항목 일치 및 쿼리 검색 프로세스
- 일괄 프로세스 로드 및 병합
- Elasticsearch 프로세스

도범 사례: 간결성을 유지합니다. 모든 처리 서버에서 작업과 프로세스를 활성화합니다. 성능이 문제가 되는 경우라면 서버 간에 워크로드를 분배하는 방법을 결정할 수 있습니다. 예를 들어 정리 워크로드가 과도한 경우 하나의 처리 서버를 온라인 정리 작업을 위해 구성하고 다른 하나를 일괄 정리 작업을 위해 구성할 수 있습니다. 정리 및 일치를 위한 일괄 작업이 동시에 실행되는 경우 서로 다른 처리 서버에서 작업을 실행하는 것을 고려합니다.

속성을 변경하는 경우 처리 서버를 다시 시작합니다. 스레드 개수는 예외로, 이것은 다시 시작이 필요하지 않습니다. 다음 테이블에는 지정할 수 있는 속성이 설명되어 있습니다.

속성	설명
서버	이 처리 서버가 배포된 응용 프로그램 서버의 IP 주소 또는 정규화된 호스트 이름입니다. 참고: localhost를 호스트 이름으로 사용하지 마십시오.
포트	이 처리 서버를 배포한 응용 프로그램 서버의 HTTP 또는 HTTPS 포트입니다.
정리 작업	이 처리 서버에서 정리 작업을 처리할지 나타냅니다. 기본값은 true입니다. 정리 모드 옵션을 사용하여 이 서버에서 일괄 작업을 처리할지 또는 실시간 정리 요청을 처리할지 아니면 둘 다 처리할지 지정합니다.
정리 및 유사 항목 일치를 위한 일괄 스레드 수	정리, 토큰화 및 일치 일괄 작업에 사용할 스레드의 수입니다. 기본값은 1입니다. 모범 사례: <ul style="list-style-type: none"> - 유사 항목 일치 또는 주소 확인을 사용 중이라면 CPU당 하나의 스레드를 활성화합니다. 예를 들어 쿼드코어 시스템에서는 이 값을 4로 설정합니다. - 유사 항목 일치 또는 주소 확인을 사용 중이 아니고 대부분의 데이터 정리에 문자열이 포함되어 있는 경우 CPU당 4개의 스레드를 활성화합니다. 예를 들어 쿼드코어 시스템에서는 이 값을 16로 설정합니다. - 연산 참조 저장소와 다른 별도의 시스템에서 처리 서버를 실행 중인 경우 원격 데이터베이스에서 발생할 수 있는 대기 시간을 수용하도록 하나의 다른 스레드를 추가합니다. - 메모리 집중 프로세스를 실행하는 경우 JVM의 이러한 모든 스레드에 할당되는 총 메모리를 1GB로 제한합니다.
정리 모드	이 처리 서버에서 처리하는 정리 작업의 유형을 지정합니다. <ul style="list-style-type: none"> - 일괄만. 일괄 작업의 정리 요청만 처리합니다. 정리 함수는 준비 프로세스의 매핑에 의해 호출됩니다. - 온라인만. 실시간 정리 요청만 처리합니다. 요청은 CleansePut SIF API에 의해 암시적으로 호출되거나 IDD 제목 영역 정리 함수에서 명시적으로 구성되는 정리 함수로부터 받습니다. - 둘 다. 일괄 정리 요청 및 온라인 정리 요청을 둘 다 처리합니다.
유사 항목 일치 및 쿼리 검색 처리	이 처리 서버에서 유사 항목 일치를 처리할지 나타냅니다. 기본값은 true입니다. 일치 모드 옵션을 사용하여 이 서버에서 일괄 작업을 처리할지 또는 실시간 일치 요청을 처리할지 아니면 둘 다 처리할지 지정합니다.
일치 모드	이 처리 서버에서 처리하는 유사 항목 일치 처리의 유형을 지정합니다. <ul style="list-style-type: none"> - 일괄만. 요청이 일괄 작업으로부터 온 경우에만 일치에 참여합니다. - 온라인만. 실시간 요청에 대해서만 일치에 참여합니다. searchMatch SIF API 호출, IDD 확장된 검색으로부터 수신한 실시간 일치에 대한 요청입니다. - 둘 다. 일괄 요청 및 온라인 요청에 대해 일치에 참여합니다.
오프라인	이 처리 서버의 상태입니다. 확인란을 선택하거나 지워도 처리 서버의 상태는 변경되지 않습니다.
일괄 처리 로드 및 병합	이 처리 서버에서 데이터 로드 일괄 작업 및 병합 일괄 작업을 처리할지 나타냅니다. 기본값은 false입니다.
로드 및 병합을 위한 스레드	데이터 로드 일괄 작업 및 자동 병합 일괄 작업에 사용할 스레드의 최대 수입니다. 모범 사례: 시스템에서 CPU당 4개의 스레드를 활성화합니다. 예를 들어 쿼드코어 시스템에서는 값을 16으로 설정합니다. 기본값은 20입니다.
CPU 속도	정리 서버 풀에 포함된 시스템의 상대적 CPU 성능을 지정합니다. 모범 사례: 처리 서버를 실행하는 시스템의 성능이 유사한 경우 1의 기본값을 유지합니다. 이 처리 서버 시스템의 성능이 다른 시스템의 2배인 경우 이 값을 2로 설정합니다.

속성	설명
Elasticsearch 처리	이 처리 서버에서 처음에 스마트 검색 데이터 인덱싱 일괄 작업을 처리할지 나타냅니다. 이 일괄 작업은 비즈니스 항목에서 검색 가능한 필드의 모든 값에 대한 인덱스를 생성합니다.
보안 연결(HTTPS)	이 처리 서버에서 HTTPS 프로토콜을 사용할지 나타냅니다. 선택하는 경우 포트 옵션이 HTTPS 포트 번호로 설정되어 있는지 확인합니다.

처리 서버 추가

처리 서버 도구를 사용하여 처리 서버를 추가할 수 있습니다. 각 연산 참조 저장소에 대해 여러 처리 서버를 구성할 수 있습니다.

1. 처리 서버 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 처리 서버 도구의 오른쪽 창에서 **처리 서버 추가** 단추를 클릭하여 처리 서버를 추가합니다.
처리 서버 추가/편집 대화 상자가 표시됩니다.
4. 처리 서버의 속성을 설정합니다.
5. **확인**을 클릭합니다.
6. **저장**을 클릭합니다.

처리 서버에 대한 보안 통신 활성화

각 처리 서버에는 서명된 인증서가 필요합니다. Hub 콘솔을 사용하여 HTTPS 프로토콜을 활성화하고 각 처리 서버에 대한 보안 포트를 지정합니다.

1. 처리 서버에 대한 서명된 인증서를 인증서 저장소에 생성합니다.
2. 응용 프로그램 서버에서 인증서 저장소에 액세스할 수 있는지 확인합니다.
3. Hub 콘솔에 로그인합니다.
4. 연산 참조 저장소 데이터베이스를 선택합니다.
5. 쓰기 잠금을 획득합니다.
6. **유틸리티** 작업 영역에서 **처리 서버**를 선택합니다.
7. 처리 서버를 선택하고 **처리 서버 편집** 아이콘을 클릭합니다.
처리 서버 추가/편집 대화 상자가 열립니다.
8. **포트**가 보안 포트인지 확인합니다.
9. **보안 연결(HTTPS)** 확인란을 선택합니다.
10. **확인**을 클릭합니다.
11. 목록에 나타나는 다른 처리 서버를 확인합니다.

처리 서버 속성 편집

처리 서버의 속성을 편집할 수 있습니다.

1. 처리 서버 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.

3. 편집할 처리 서버를 선택합니다.
4. **처리 서버 편집** 단추를 클릭합니다.
선택한 처리 서버에 대한 **처리 서버 추가/편집** 대화 상자가 표시됩니다.
5. 처리 서버에 대해 필요한 속성을 변경합니다.
6. **확인**을 클릭합니다.
7. **저장**을 클릭합니다.

처리 서버 삭제

처리 서버 도구를 사용하여 처리 서버를 삭제할 수 있습니다.

1. 처리 서버 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 삭제할 처리 서버를 선택합니다.
4. **처리 서버 삭제** 단추를 클릭합니다.
처리 서버 도구에서 삭제를 확인하는 메시지가 표시됩니다.
5. **확인**을 클릭합니다.

처리 서버 구성 테스트

처리 서버 정보를 추가하거나 변경할 때마다 구성을 검사하여 연결이 올바르게 작동하는지 확인하는 것이 좋습니다.

1. 처리 서버 도구를 시작합니다.
2. 테스트할 처리 서버를 선택합니다.
3. **처리 서버 테스트** 단추를 클릭하여 구성을 테스트합니다.
테스트에 성공한 경우 처리 서버 도구에 연결 정보와 성공 메시지를 보여 주는 창이 표시됩니다.
문제가 있는 경우에는 Informatica MDM Hub에 연결 문제에 대한 정보가 들어 있는 창이 표시됩니다.
4. **확인**을 클릭합니다.

정리 함수 구성

MDM Hub에서는 데이터를 정리하는 정리 함수를 작성하여 실행할 수 있습니다.

*정리 함수*는 표준화 또는 확인 목적으로 MDM Hub에서 레코드의 데이터 값에 적용하는 함수입니다. 예를 들어 데이터에 인사말 열이 있는 경우 정리 함수를 사용하여 나타나는 모든 "Doctor" 인스턴스를 "Dr."로 표준화할 수 있습니다. 정리 함수를 연속적으로 적용할 수도 있고, 단순히 준비 테이블의 열에 출력 값을 할당할 수도 있습니다.

정리 함수의 유형

MDM Hub에는 다음 정리 함수 유형 중 하나가 있을 수 있습니다.

- MDM Hub에서 정의된 정리 함수
- 정리 엔진에서 정의된 정리 함수

- 정의한 사용자 지정 정리 함수

미리 정의된 함수는 이름 및 주소 표준화, 주소 분해, 성별 확인 등의 특수화된 정리 기능을 사용할 수 있게 해 줍니다. 정리 함수 도구에 대한 자세한 내용은 콘솔을 참조하십시오.

라이브러리

함수는 **Java** 라이브러리나 사용자 라이브러리 같은 *라이브러리*에 구성됩니다. 라이브러리는 모델 작업 영역의 정리 함수 도구에서 사용할 수 있는 함수를 구성하는 데 사용되는 폴더입니다.

정리 함수는 보안 리소스임

정리 함수를 보안 리소스로 구성하고 **SECURE** 또는 **PRIVATE** 상태로 설정할 수 있습니다.

사용 가능한 함수는 정리 엔진에 따라 달라짐

Hub 콘솔에 표시되는 함수는 사용하는 정리 엔진에 따라 달라집니다. **MDM Hub**에는 현재 사용 중인 정리 엔진에서 사용 가능한 정리 함수가 표시됩니다. 그러나 어떤 정리 엔진을 사용하든 **MDM Hub**에서의 전반적인 데이터 정리 프로세스는 동일합니다.

정리 함수 도구 시작

정리 함수 도구는 데이터 정리 방법을 정의하는 인터페이스를 제공합니다.

정리 함수 도구를 시작하려면

- **Hub** 콘솔에서 모델 작업 영역을 확장한 후 **정리 함수**를 클릭합니다.

Hub 콘솔에 정리 함수 도구가 표시됩니다.

정리 함수 도구는 다음과 같은 창으로 구분됩니다.

창	설명
탐색 창	정리 함수를 트리 보기로 표시합니다. 트리에서 아무 노드나 클릭하면 오른쪽 창에 해당 속성 페이지가 표시됩니다.
속성 창	선택한 함수의 속성을 보여 줍니다. 사용자 지정 정리 함수의 경우 오른쪽 창에서 속성을 편집할 수 있습니다.

왼쪽 창에 표시되는 함수는 사용 중인 정리 엔진에 따라 다르며, 함수가 이전 그림에 표시된 함수와 다를 수 있습니다.

정리 함수 유형

정리 함수는 트리에서 해당 유형에 따라 그룹화됩니다.

정리 함수 유형은 손쉽게 관리 및 액세스할 수 있도록 서로 비슷한 여러 정리 함수를 그룹화하는 데 사용되는 최상위 범주입니다.

정리 함수 속성

탐색 창에서 정리 함수 유형의 목록을 확장하면 정리 함수를 선택하여 해당 속성을 표시할 수 있습니다.

특정 정리 함수 외에도 "기타 함수"에는 데이터를 효율적으로 관리할 수 있도록 해 주는 데이터베이스 읽기 및 거부 함수가 포함되어 있습니다.

필드	설명
데이터베이스 읽기	맵에서 데이터베이스 테이블의 레코드를 직접 조회할 수 있도록 합니다. 참고: 이 함수는 제한된 수의 동일한 데이터 항목에 대한 참조가 여러 개인 경우에 사용됩니다.
거부	맵의 생성자가 잘못된 데이터를 식별하고 거부 이유를 명시하여 해당 레코드를 거부할 수 있도록 합니다.

정리 함수 구성 개요

정리 함수를 정의하려면 다음 태스크를 수행합니다.

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. **새로 고침**을 클릭하여 정리 라이브러리를 새로 고칩니다.
4. 고유한 정리 라이브러리를 생성합니다. 이 라이브러리는 사용자 지정 정리 함수를 보관할 폴더가 됩니다.
5. 해당되는 경우 새 라이브러리에서 정규식 함수를 정의합니다.
6. 해당되는 경우 새 라이브러리에서 그래프 함수를 정의합니다.
7. 정리 함수를 그래프 함수에 추가합니다.
8. 함수를 테스트합니다.

사용자 정리 라이브러리 구성

기존의 내부 또는 외부 정리 함수에서 사용자 지정 정리 함수를 생성하려는 경우 사용자 정리 라이브러리 또는 **Java** 정리 라이브러리를 추가할 수 있습니다. 사용자 정리 라이브러리를 구성하여 그래프 함수, 정규식 함수 및 정리 목록을 추가합니다.

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. **새로 고침**을 클릭하여 정리 라이브러리를 새로 고칩니다.
4. 정리 함수 탐색기에서 마우스 오른쪽 단추로 **정리 함수**를 클릭한 다음 **사용자 라이브러리 추가**를 클릭합니다.

사용자 라이브러리 추가 대화 상자가 나타납니다.

5. 다음 속성을 지정합니다.

이름

라이브러리에 대해 고유한 설명 이름입니다.

설명

라이브러리의 선택적 설명입니다.

6. **확인**을 클릭합니다.

추가한 라이브러리가 정리 함수 탐색기에 표시됩니다.

Java 정리 라이브러리 구성

기존의 내부 또는 외부 정리 함수에서 사용자 지정 정리 함수를 생성하려는 경우 사용자 지정 라이브러리 또는 Java 정리 라이브러리를 추가할 수 있습니다. Java 정리 라이브러리를 구성하여 MDM Hub 외부에서 생성한 사용자 지정 정리 함수를 추가합니다.

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. **새로 고침**을 클릭하여 정리 라이브러리를 새로 고칩니다.
4. 정리 함수 탐색기에서 마우스 오른쪽 단추로 **정리 함수**를 클릭한 다음 **Java 라이브러리 추가**를 선택합니다.
Java 라이브러리 추가 대화 상자가 나타납니다.
5. **찾아보기** 단추를 클릭하여 Java 라이브러리 JAR 파일의 위치로 이동합니다.
6. 다음 속성을 지정합니다.

이름

라이브러리에 대해 고유한 설명 이름입니다.

설명

라이브러리의 선택적 설명입니다.

7. 해당하는 경우 라이브러리에 대한 매개 변수를 구성합니다.
 - a. **매개 변수** 단추를 클릭합니다.
매개 변수 대화 상자가 나타납니다.
 - b. 라이브러리에 필요한 만큼 매개 변수를 추가합니다.
 - 매개 변수를 추가하려면 **매개 변수 추가** 단추를 클릭합니다.
값 추가 대화 상자가 나타납니다.
이름과 값을 입력한 다음 **확인**을 클릭합니다.
 - 매개 변수를 가져오려면 **매개 변수 가져오기** 단추를 클릭합니다.
열기 대화 상자가 나타납니다.
원하는 매개 변수가 포함된 속성 파일을 선택합니다.
 - 이 라이브러리에 필요한 만큼 매개 변수를 추가할 수 있습니다.
 - 매개 변수를 추가하려면 **매개 변수 추가** 단추를 클릭합니다. 값 추가 대화 상자가 표시됩니다.
이름과 값을 입력하고 **확인**을 클릭합니다.
 - 매개 변수를 가져오려면 **매개 변수 가져오기** 단추를 클릭합니다. 열기 대화 상자가 표시되고 원하는 매개 변수가 포함된 속성 파일을 선택하라는 메시지가 표시됩니다.
- 파일에서 가져온 이름, 값 쌍은 런타임 시 사용자 정의 Java 함수를 해당 Java 속성의 요소로 사용할 수 있습니다. 일반 함수에 사용자 ID 또는 대상 URL과 같은 사용자 지정 값을 제공할 수 있습니다.
8. **확인**을 클릭합니다.
추가한 라이브러리가 정리 함수 탐색기에 표시됩니다.

정규식 함수 추가

정리 작업에 대해 정규식 함수를 추가할 수 있습니다. 정규식은 일반적인 구문 규칙 및 기호 패턴에 따라 텍스트 데이터를 일치시키고 조작하는 데 사용할 수 있는 계산 식입니다. 정규식의 구문 및 패턴에 대한 보다 자세한 내용은 `java.util.regex.Pattern`에 대한 Javadoc를 참조하십시오.

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 사용자 정리 라이브러리 이름을 마우스 오른쪽 단추로 클릭한 다음 **정규식 함수 추가**를 클릭합니다.
정규식 함수 추가 대화 상자가 나타납니다.
4. 다음 속성을 지정합니다.

필드	설명
이름	정규식 함수에 대해 고유한 설명 이름입니다.
설명	정규식 함수의 선택적 설명입니다.

5. **확인**을 클릭합니다.
추가한 정규식 함수가 정리 함수 탐색기에서 선택한 사용자 라이브러리 아래에 표시됩니다.
6. 추가한 정규식 함수의 **세부 정보** 탭을 클릭합니다.
7. 입력 또는 출력 식을 입력하려면 **편집**을 클릭한 다음 **편집 허용**을 클릭하여 변경 사항을 적용합니다.
8. **저장**을 클릭합니다.

그래프 함수 구성

그래프 함수는 Hub 콘솔에서 정리 함수 도구를 사용하여 그래픽 방식으로 시각화하고 구성할 수 있는 정리 함수입니다. 그래프 함수에 미리 정의된 함수를 모두 추가할 수 있습니다.

그래프 함수에는 하나 이상의 입력 및 출력이 있습니다. 각 그래프 함수에 대해 모든 필수 입력 및 출력을 구성합니다.

다음 태스크를 수행하여 그래프 함수를 구성합니다.

1. 그래프 함수를 추가합니다.
2. 그래프 함수에 함수를 추가합니다.

입력 및 출력 속성

그래프 함수에는 하나 이상의 입력 및 출력이 있습니다.

모든 그래프 함수의 입력 및 출력에 대해 다음 속성을 구성합니다.

이름

입력 또는 출력에 대해 고유한 설명 이름입니다.

설명

입력 또는 출력의 선택적 설명입니다.

데이터 유형

입력 또는 출력 매개 변수의 데이터 유형입니다.

다음 값 중 하나를 사용합니다.

- 부울. 부울 값을 허용합니다.
- 날짜. 날짜 값을 허용합니다.
- 부동 소수점 수. 부동 소수점 수 값을 허용합니다.
- 정수. 정수 값을 허용합니다.
- 문자열. 모든 데이터를 허용합니다.

그래프 함수 추가

그래프 함수를 추가할 수 있습니다.

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. **사용자 라이브러리** 이름을 마우스 오른쪽 단추로 클릭하고 **그래프 함수 추가**를 선택합니다.
정리 함수 도구에 그래프 함수 추가 대화 상자가 표시됩니다.
4. 다음 속성을 지정합니다.

필드	설명
이름	해당 그래프 함수의 고유한 설명 이름입니다.
설명	해당 그래프 함수에 대한 선택적 설명입니다.

5. **확인**을 클릭합니다.

정리 함수 도구에 그래프 함수가 표시됩니다. 그래프 함수는 비어 있습니다. 그래프 함수를 구성하려면 [“그래프 함수에 함수 추가” 페이지 337](#)를 참조하십시오.

그래프 함수에 함수 추가

그래프 함수에 함수를 필요한 만큼 추가할 수 있습니다.

그래프 함수를 재사용할 수도 있습니다. 그래프 함수를 정의한 다음 정리 라이브러리의 다른 함수처럼 그래프 함수를 사용합니다. 예를 들어 그래프 함수를 다른 그래프 함수 내에 추가할 수 있습니다.

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 그래프 함수를 클릭한 후 **세부 정보** 탭을 클릭합니다.
함수의 그래픽 표현을 작업 공간이라고 합니다. 작업 공간에 입력과 출력을 모두 표시하기 위해 창의 크기를 조정해야 할 수 있습니다.
기본적으로 그래프 함수에는 문자열 유형의 입력과 출력에 각각 하나씩 있습니다. 회색 원은 입력 및 출력을 나타냅니다. 함수에 더 많은 입력 또는 출력과 다른 데이터 유형이 필요할 수 있습니다.
4. 작업 공간을 마우스 오른쪽 단추를 클릭하고 **함수 추가**를 선택합니다.
정리 함수 도구에 **추가할 함수 선택** 대화 상자가 표시됩니다.
5. 추가할 함수가 들어 있는 폴더를 확장하고 함수를 선택한 다음 **확인**을 클릭합니다.

참고: 사용 가능한 함수는 정리 엔진 구성에 따라 달라집니다.

정리 함수 도구의 작업 공간에 함수가 표시됩니다. 함수를 이동하려면 끌어서 새 위치에 놓습니다.

6. 함수를 마우스 오른쪽 단추로 클릭하고 **확장 모드**를 선택합니다.
확장 모드는 모든 입력 및 출력에 대한 레이블을 표시합니다. 원의 색상은 입력 또는 출력의 데이터 유형을 나타냅니다. 데이터 유형은 일치해야 합니다.
7. 입력 커넥터(입력 상자 오른쪽의 작은 원) 위에 마우스를 놓습니다. 사용 가능한 경우 원이 빨간색으로 바뀝니다.
8. 노드를 클릭하고 함수 입력 노드 중 하나까지 선을 그립니다.
9. 함수 출력 노드 중 하나에서 출력 상자 노드까지 선을 그립니다.
10. **저장**을 클릭합니다.
그래프 함수를 테스트하는 방법에 대한 자세한 내용은 [“함수 테스트” 페이지 340](#)를 참조하십시오.

함수 모드


함수 모드는 작업 공간에 함수가 표시되는 방식을 결정합니다. 각 함수에는 다음과 같은 모드가 있으며, 함수를 마우스 오른쪽 단추로 클릭하여 이러한 모드에 액세스할 수 있습니다.

옵션	설명
압축	작은 상자에 이름만 있는 형태로 함수를 표시합니다.
표준	중간 크기 상자에 함수 이름과 입력 및 출력 노드(레이블은 지정되지 않음)가 있는 형태로 함수를 표시합니다. 이것이 기본 모드입니다.
확장	큰 상자에 함수 이름, 입력 및 출력 노드, 노드의 이름이 있는 형태로 함수를 표시합니다.
로깅 활성화됨	디버깅하는 데 사용됩니다. 이 옵션을 선택하면 준비 작업을 실행할 때 이 함수에 대한 로그 파일이 생성됩니다. 준비 작업 중에 함수가 호출될 때마다 로그 파일에 입력 및 출력이 기록됩니다. 각 준비 작업에 대해 새 로그 파일이 생성됩니다. 로그 파일은 <jobID><graph function name>.log 형식으로 이름이 지정되어 다음 위치에 저장됩니다. <code><infamdm_install_dir>\hub\cleanse\tmp\<연산 참조 저장소></code> 참고: 이 옵션은 디스크 공간을 사용하고 디스크 I/O와 관련된 성능 오버헤드를 요구하므로 프로덕션에서는 이 옵션을 사용하지 마십시오. 이 로깅을 비활성화하려면 함수를 마우스 오른쪽 단추로 클릭하고 로깅 활성화를 선택 취소합니다.
개체 삭제	그래프 함수에서 함수를 삭제합니다.

함수를 두 번 클릭하는 방식으로 표시 모드(압축, 표준 및 확장)를 차례대로 전환할 수 있습니다.

작업 공간 단추

작업 공간 오른쪽의 도구 모음에는 다음과 같은 단추가 있습니다.

단추	설명
	변경 내용을 저장합니다.
	함수 입력을 편집합니다.

단추	설명
	함수 출력을 편집합니다.
	함수를 추가합니다.
	상수를 추가합니다.
	조건부 실행 구성 요소를 추가합니다.
	선택한 구성 요소를 편집합니다.
	선택한 구성 요소를 삭제합니다.
	그래프를 확장합니다. 이 단추를 사용하면 왼쪽 창이 숨겨져 화면의 작업 공간이 더 넓어집니다.

상수 사용

상수는 입력을 표준화한 경우에 유용합니다.

예를 들어 전체가 의사로만 구성된 데이터 집합이 있는 경우 상수를 사용하여 직함에 'Dr.'를 입력할 수 있습니다. 그래프 함수에 상수를 사용할 경우 해당 함수는 배경 색상이 회색으로 표시되므로 다른 함수와 구별됩니다.

입력 구성

입력을 추가하려면

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 구성할 정리 함수를 선택합니다.
4. **세부 정보** 탭을 클릭합니다.
5. 입력을 마우스 오른쪽 단추로 클릭하고 **입력 편집**을 선택합니다.

입력 대화 상자가 표시됩니다.

참고: 한 번 생성된 입력은 나중에 편집하여 유형을 변경할 수 없습니다. 입력의 유형을 변경해야 하는 경우에는 올바른 유형의 새 입력을 생성하고 이전 입력을 삭제하십시오.

6. **추가** 단추를 클릭하여 입력을 추가합니다.
매개 변수 추가 대화 상자가 표시됩니다.

7. 다음 속성을 지정합니다.

필드	설명
이름	해당 매개 변수의 고유한 설명 이름입니다.
데이터 유형	해당 매개 변수의 데이터 유형입니다.
설명	해당 매개 변수에 대한 선택적 설명입니다.

8. **확인**을 클릭합니다.

함수에 필요한 만큼 입력을 추가합니다.

출력 구성

출력을 추가하려면

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 구성할 정리 함수를 선택합니다.
4. **세부 정보** 탭을 클릭합니다.
5. 출력을 마우스 오른쪽 단추로 클릭하고 **출력 편집**을 선택합니다.

출력 대화 상자가 표시됩니다.

참고: 한 번 생성된 출력은 나중에 편집하여 유형을 변경할 수 없습니다. 출력의 유형을 변경해야 하는 경우에는 올바른 유형의 새 출력을 생성하고 이전 출력을 삭제하십시오.

6. **추가** 단추를 클릭하여 출력을 추가합니다.

매개 변수 추가 대화 상자가 표시됩니다.

필드	설명
이름	해당 매개 변수의 고유한 설명 이름입니다.
데이터 유형	해당 매개 변수의 데이터 유형입니다.
설명	해당 매개 변수에 대한 선택적 설명입니다.

7. **확인**을 클릭합니다.

함수에 필요한 만큼 출력을 추가합니다.

함수 테스트

그래프 또는 정규식 함수를 추가하고 구성한 후에는 이를 테스트하여 예상한 대로 동작하는지 확인하는 것이 좋습니다.

이 테스트 프로세스에서는 단일 레코드가 함수에 입력되는 경우를 시뮬레이션합니다.

함수를 테스트하려면

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 테스트할 정리 함수를 선택합니다.
4. **테스트** 탭을 클릭합니다.

정리 함수 도구에 테스트 화면이 표시됩니다.

5. 각 입력에 대해 값 열의 셀을 클릭하고 입력 데이터 유형에 맞는 값을 입력하여 테스트할 값을 지정합니다.
 - 부울 입력의 경우 정리 함수 도구에 **true/false** 드롭다운 목록이 표시됩니다.
 - 달력 입력의 경우 정리 함수 도구에 달력 단추가 표시되며, 이 단추를 클릭한 다음 날짜 대화 상자에서 날짜를 선택할 수 있습니다.
6. **테스트**를 클릭합니다.

테스트가 성공적으로 완료되면 출력 섹션에 출력이 표시됩니다.

정리 함수에 조건 사용

이 섹션에서는 그래프 함수에 조건을 추가하는 방법에 대해 설명합니다.

조건부 실행 구성 요소 정보

조건부 실행 구성 요소는 프로그래밍 언어의 **case**(또는 **switch**) 문과 구성이 비슷합니다.

정리 함수는 조건을 평가하고 이 평가를 기반으로 조건과 일치하는 사례에 연결된 적절한 그래프 함수를 적용합니다. 사례가 조건과 일치하지 않으면 기본 사례, 즉 별표(*) 플래그가 지정된 사례가 사용됩니다.

조건부 실행 구성 요소를 사용하는 경우

조건부 실행 구성 요소는 예를 들어 데이터를 세그먼트화한 경우에 유용합니다.

테이블에 고객 및 잠재 고객 등의 개별 데이터 그룹이 여러 개 있는 경우 레코드가 멤버로 속해 있는 그룹을 나타내는 열을 생성할 수 있습니다. 이때 각 그룹을 세그먼트라고 합니다. 이 예의 경우 고객은 이 열에서 **C**로 표시되고 잠재 고객은 **P**로 표시될 수 있습니다. 조건부 실행 구성 요소를 사용하면 데이터를 세그먼트별로 다르게 정리할 수 있습니다. 조건부 값이 지정한 조건을 충족하지 않으면 기본 사례가 실행됩니다.

조건부 실행 구성 요소 추가

조건부 실행 구성 요소를 추가하려면

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 구성할 정리 함수를 선택합니다.
4. 작업 공간을 마우스 오른쪽 단추로 클릭하고 **조건 추가**를 선택합니다.

조건 편집 대화 상자가 표시됩니다.

5. **추가** 단추를 클릭하여 값을 추가합니다.

값 추가 대화 상자가 표시됩니다.

6. 조건에 대한 값을 입력합니다. 예를 들어 고객(customer)과 잠재 고객(prospect)이 있는 경우 C 또는 P를 입력합니다. 그런 다음 **확인**을 클릭합니다.

정리 함수 도구의 왼쪽에 있는 조건 목록과 입력 상자에 새 조건이 표시됩니다.

조건을 필요한 만큼 추가합니다. 기본 조건을 지정해야 합니다. 기본 사례는 새 조건부 실행 구성 요소를 생성할 때 자동으로 생성됩니다. 그러나 별표(*)로 기본 사례를 지정할 수도 있습니다. 기본 사례는 지정한 사례에서 다루지 않는 모든 사례에 대해 실행됩니다.

7. 모든 조건을 처리하는 데 필요한 만큼 함수를 추가합니다.
8. 기본 조건을 포함한 각 조건에 대해 입력 노드와 함수 입력 사이에 링크를 만듭니다. 또한 함수의 출력과 정리 함수의 출력 사이에도 링크를 만듭니다.

참고: 그래프 함수에 중첩된 처리 논리를 지정할 수 있습니다. 예를 들어 중첩된 CASE 문과 같이 조건부 구성 요소를 다른 조건부 구성 요소 내에 중첩할 수 있습니다. 실제로 각 테스트의 복잡성 수준을 달리 하여 여러 조건부 테스트를 포함하는 전체적인 복합 프로세스를 정의할 수 있습니다.

정리 목록 구성

이 섹션에서는 Informatica MDM Hub 구현에서 정리 목록을 구성하는 방법에 대해 설명합니다.

정리 목록 정보

정리 목록은 런타임에 미리 정의된 순서로 실행되는 문자열 함수를 논리적으로 그룹화한 것입니다.

정리 목록을 사용하여 알려진 문자열 값을 표준화하고 입력 문자열에서 구두점 같은 불필요한 문자를 제거할 수 있습니다.

정리 목록 추가

새 정리 목록을 추가하려면

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. **새로 고침**을 클릭하여 정리 라이브러리를 새로 고칩니다. 외부 정리 엔진에 사용됩니다.
중요: 쓰기 잠금을 획득한 후와 레코드를 처리하기 전에는 **새로 고침**을 선택해야 합니다. 그렇지 않으면 외부 정리 엔진에서 오류가 발생합니다.
4. 목록의 정리 함수 아래에 있는 정리 라이브러리를 마우스 오른쪽 단추로 클릭하고 **정리 목록 추가**를 선택합니다.
정리 목록 추가 대화 상자가 표시됩니다.
5. 다음 속성을 지정합니다.

필드	설명
이름	해당 정리 목록의 고유한 설명 이름입니다.
설명	해당 정리 목록에 대한 선택적 설명입니다.

6. **확인**을 클릭합니다.

정리 함수 도구 화면의 오른쪽에 빈 새 정리 목록에 대한 세부 정보 창이 표시됩니다.

정리 목록 속성

이 섹션에서는 정리 목록의 입력 및 출력 속성에 대해 설명합니다.

입력 속성

다음 테이블에는 정리 목록의 입력 속성이 설명되어 있습니다.

속성	설명
Input string	소스 시스템의 문자열 값입니다. 검색의 대상으로 사용됩니다.
searchType	<p>입력 문자열에 대해 실행할 일치의 유형(정리 목록 항목을 입력 문자열과 비교)을 지정합니다. 다음 값 중 하나입니다.</p> <p>ENTIRE</p> <p>정리 목록 항목을 전체 문자열과 비교합니다. 전체 입력 문자열이 정리 목록 항목과 동일한 경우에만 일치가 성공합니다. 이 매개 변수를 지정하지 않을 경우 기본 설정입니다.</p> <p>WORD</p> <p>정리 목록 항목을 입력 문자열의 각 단어 하위 문자열과 비교합니다. 정리 목록 항목이 입력 문자열에서 문자열의 시작, 문자열의 끝 또는 공백 문자 경계 옆에 있는 하위 문자열일 경우에만 일치가 성공합니다.</p> <p>ANYWHERE</p> <p>정리 목록 항목을 입력 문자열의 임의의 부분과 비교합니다. 정리 목록 항목이 입력 문자열의 하위 문자열일 경우 이 하위 문자열이 입력 문자열에서 나타나는 위치에 관계없이 일치가 성공합니다.</p> <p>참고: 문자열 비교는 대/소문자를 구분합니다.</p>
replaceAllOccurrences	<p>입력 문자열에서 일치하는 하위 문자열을 해당 정리 목록 항목으로 바꾸는 정도를 지정합니다. 다음 값 중 하나입니다.</p> <p>TRUE</p> <p>입력 문자열에서 일치하는 모든 하위 문자열을 해당 정리 목록 항목으로 바꿉니다.</p> <p>FALSE</p> <p>입력 문자열에서 일치하는 모든 하위 문자열 중 첫 번째 하위 문자열만 해당 정리 목록 항목으로 바꿉니다. replaceAllOccurrences를 지정하지 않을 경우 기본 설정입니다.</p> <p>참고: Strip 매개 변수가 TRUE일 경우 일치하는 하위 문자열이 바뀌지 않고 제거됩니다.</p>

속성	설명
stopOnHit	<p>입력 문자열에서 일치하는 항목을 하나 발견한 후 정리 목록의 나머지를 계속 처리할지 여부를 지정합니다. 다음 값 중 하나입니다.</p> <p>TRUE</p> <p>입력 문자열에서 첫 번째 정리 목록 항목이 발견된 후 즉시 정리 목록 처리를 중지합니다. 단, searchType 조건을 만족해야 합니다. stopOnHit를 지정하지 않을 경우 기본 설정입니다.</p> <p>FALSE</p> <p>일치하는 하위 문자열을 더 찾기 위해 입력 문자열에서 정리 목록의 나머지 항목을 계속 검색합니다.</p>
Strip	<p>입력 문자열에서 일치하는 텍스트를 제거할지, 아니면 바꿀지를 지정합니다. 다음 값 중 하나입니다.</p> <p>TRUE</p> <p>입력 문자열에서 일치하는 텍스트를 바꾸지 않고 제거합니다.</p> <p>FALSE</p> <p>입력 문자열에서 일치 텍스트를 바꿉니다. Strip을 지정하지 않을 경우 기본 설정입니다.</p> <p>참고: replaceAllOccurrences 매개 변수에 따라 입력 문자열에서 모든 일치 항목 또는 첫 번째 일치 항목만 바꾸거나 제거할지가 결정됩니다.</p>
defaultValue	<p>입력 문자열에서 정리 목록 항목을 발견하지 못했을 경우 출력에 사용하는 값입니다. 이 속성을 지정하지 않을 경우 일치 항목이 없으면 원래 입력 문자열이 출력으로 사용됩니다.</p>

출력 속성

다음 테이블에는 정리 목록의 출력 속성이 나열되어 있습니다.

속성	설명
output	정리 목록 함수의 출력 값입니다.
matched	정리 목록의 마지막으로 일치된 값입니다.
matchFlag	일치 항목이 목록에 있는지(true) 또는 없는지(false) 여부를 나타냅니다.

정리 목록 속성 편집

새로운 정리 목록은 비어 있으므로 정리 목록을 편집하여 일치 및 출력 문자열을 추가해야 합니다.

정리 목록을 편집하여 일치 및 출력 문자열을 추가하려면

1. 정리 함수 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 구성할 정리 목록을 선택합니다.

정리 함수 도구의 오른쪽 창에 정리 목록에 대한 정보가 표시됩니다.
4. 원하는 경우 오른쪽 창에서 변경할 값 옆에 있는 **편집** 단추를 클릭하여 표시 이름과 설명을 변경합니다.

5. 세부 정보 탭을 클릭합니다.

정리 함수 도구에 정리 목록 세부 정보가 표시됩니다.

6. 오른쪽 창에서 **추가** 단추를 클릭합니다.

정리 함수 도구에 출력 문자열 대화 상자가 표시됩니다.

7. 검색 문자열, 출력 문자열, 일치 유형을 지정하고 **확인**을 클릭합니다.

검색 문자열은 정리하려는 입력 문자열이며, 작업이 완료된 후 이에 따라 출력 문자열이 생성됩니다.

중요: Informatica MDM Hub는 문자열이 입력된 순서대로 각 문자열을 검색합니다. 따라서 항목을 지정하는 순서가 나타나는 결과에 영향을 줄 수 있습니다. 사용할 수 있는 일치 유형에 대한 자세한 내용은 [“문자열 일치 유형” 페이지 345](#)을 참조하십시오.

참고: 문자열을 정리 목록에 추가하는 즉시 정리 목록이 저장됩니다.

지정한 문자열은 정리 목록 세부 정보 섹션에 표시됩니다.

8. 문자열을 추가 및 제거할 수 있습니다. 또한 정리 목록에서 문자열을 앞뒤로 이동할 수 있으며, 이 경우 런타임 실행 시퀀스의 순서에 영향을 주고 그에 따라 나타나는 결과에도 영향을 줍니다.

9. 검색 문자열과 일치하지 않는 모든 입력 문자열에 "기본값"을 지정할 수도 있습니다.

기본값을 지정하지 않으면 검색 문자열과 일치하지 않는 모든 입력 문자열이 변경 없이 출력 문자열로 전달됩니다.


문자열 일치 유형

출력 문자열에 대해 다음 일치 유형 중 하나를 지정할 수 있습니다.

일치 유형	설명
정확히 일치	텍스트 문자열(예: "IBM"). 문자열 일치에는 대/소문자가 구분되지 않습니다. 예를 들어 문자열 test는 TEST 또는 Test와도 일치합니다.
정규식	정규식에 Java 구문을 사용하는 패턴. 예를 들어 "I.M.*"은 "IBM", "IB Corp" 및 "IXM Inc."와 일치합니다. 이름, 중간 이름 및 성으로 구성된 이름 필드를 구문 분석하려면 어떤 이름 부분을 지정하든 관계없이 성을 제공하는 정규식(\S+\$)을 사용하면 됩니다. 매개 변수로 입력한 정규식은 문자열에 대해 사용되며 일치하는 출력은 아웃렛으로 보내집니다. 정규식의 내부 그룹과 일치하는 그룹 번호를 지정할 수도 있습니다. 정규식 구성 및 그룹 작동 방식에 대한 설명은 Javadoc에서 java.util.regex.Pattern을 참조하십시오.
SQL 일치	SQL의 LIKE 연산자에 SQL 구문을 사용하는 패턴. 예를 들어 "I_M%"은 "IBM", "IBM Corp" 및 "IXM Inc."와 일치합니다. 파이프라인 기호()와 같은 메타 문자를 사용할 경우에는 메타 문자를 백슬래시(\) 등의 이스케이프 시퀀스로 구분해야 합니다.

일치 문자열 가져오기

파일 또는 데이터베이스 테이블 같은 일치 문자열을 가져오려면

1.  단추를 클릭합니다.

일치 문자열 가져오기 마법사가 열립니다.

2. 데이터 소스의 연결 속성을 지정하고 **다음**을 클릭합니다.

정리 함수 도구에 가져올 수 있는 테이블 목록이 표시됩니다.

3. 가져올 테이블을 선택하고 **다음**을 클릭합니다.

정리 함수 도구에 가져올 수 있는 열 목록이 표시됩니다.

4. 가져올 열을 클릭하고 **다음**을 클릭합니다.


정리 함수 도구에 가져올 수 있는 일치 문자열 목록이 표시됩니다.

샘플 데이터의 레코드를 구(각 레코드에 대한 한 항목) 또는 단어(각 레코드의 각 단어에 대한 한 항목)로 가져올 수 있습니다. 일치 문자열을 단어 또는 구로 가져올지를 선택하고 **마침**을 클릭합니다.


정리 목록 세부 정보 상자가 이제 지정된 소스의 데이터로 채워집니다.


참고: 가져온 일치 문자열은 일치 목록에 속하지 않습니다. 이 문자열을 일치 목록에 추가하려면 오른쪽의 검색 문자열로 이동해야 합니다.

- 검색 문자열과 출력 문자열의 일치 문자열 값과 함께 일치 문자열을 일치 목록에 추가하려면 일치 문자

열 목록에서 해당 문자열을 선택하고  단추를 클릭합니다.

- 정의하려는 출력 문자열 값과 함께 일치 문자열을 일치 목록에 추가하려면 추가한 레코드를 클릭하고 새 검색 및 출력 문자열을 지정합니다.


- 모든 일치 문자열을 일치 목록에 추가하려면  단추를 클릭합니다.

- 일치 목록에서 모든 일치 문자열을 지우려면  단추를 클릭합니다.

- 완전한 일치 목록이 구성될 때까지 이러한 단계를 반복합니다.

일치 출력 문자열 가져오기

파일 또는 데이터베이스 테이블 같은 일치 출력 문자열을 가져오려면

1. 오른쪽 창에서  단추를 클릭합니다.
일치 출력 문자열 가져오기 마법사가 열립니다.
2. 데이터 소스의 연결 속성을 지정합니다.
3. **다음**을 클릭합니다.
정리 함수 도구에 가져올 수 있는 테이블 목록이 표시됩니다.
4. 가져올 테이블을 선택합니다.
5. **다음**을 클릭합니다.
정리 함수 도구에 가져올 수 있는 열 목록이 표시됩니다.
6. 가져올 열을 선택합니다.
7. **다음**을 클릭합니다.
정리 함수 도구에 가져올 수 있는 일치 문자열 목록이 표시됩니다.
8. **마침**을 클릭합니다.
정리 목록 세부 정보 상자가 이제 지정된 소스의 데이터로 채워집니다.

Informatica 플랫폼에서 데이터 정리 설정

Informatica 플랫폼 준비를 수행하고 준비 프로세스 동안 데이터를 정리해야 하는 경우 개발자 도구에서 데이터 정리를 구성합니다. **Data Quality** 변환을 사용하여 데이터를 정리합니다.

데이터 정리를 설정하려면 맵렛에 변환을 추가합니다. 맵렛은 맵핑에서 재사용할 수 있는 구성 요소이므로 맵렛에 변환을 추가하는 경우 이러한 설정은 다시 사용할 수 있습니다.

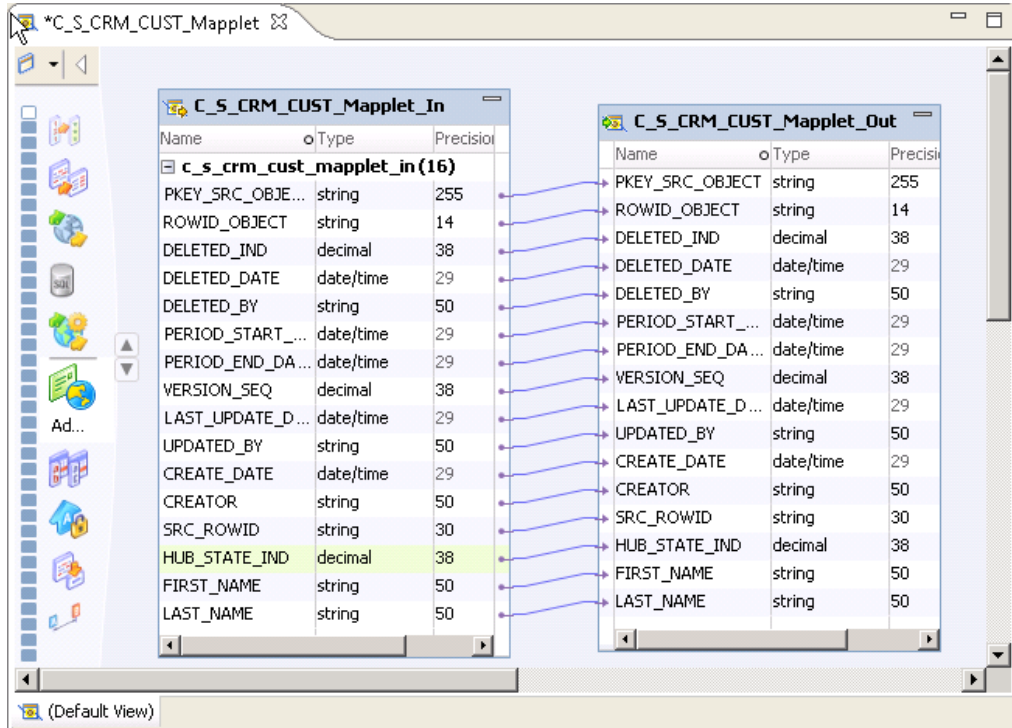
맵릿에 변환 추가

데이터 정리 작업을 수행하기 위해 데이터 품질 변환을 맵릿에 추가할 수 있습니다.

1. 개체 탐색기 보기에서 마우스 오른쪽 단추로 맵릿을 클릭하고 **열기**를 클릭합니다.

맵릿 편집기에서 맵릿이 열립니다.

다음 이미지는 C_S_CRM_CUST_Mapplet 맵릿을 보여 줍니다.

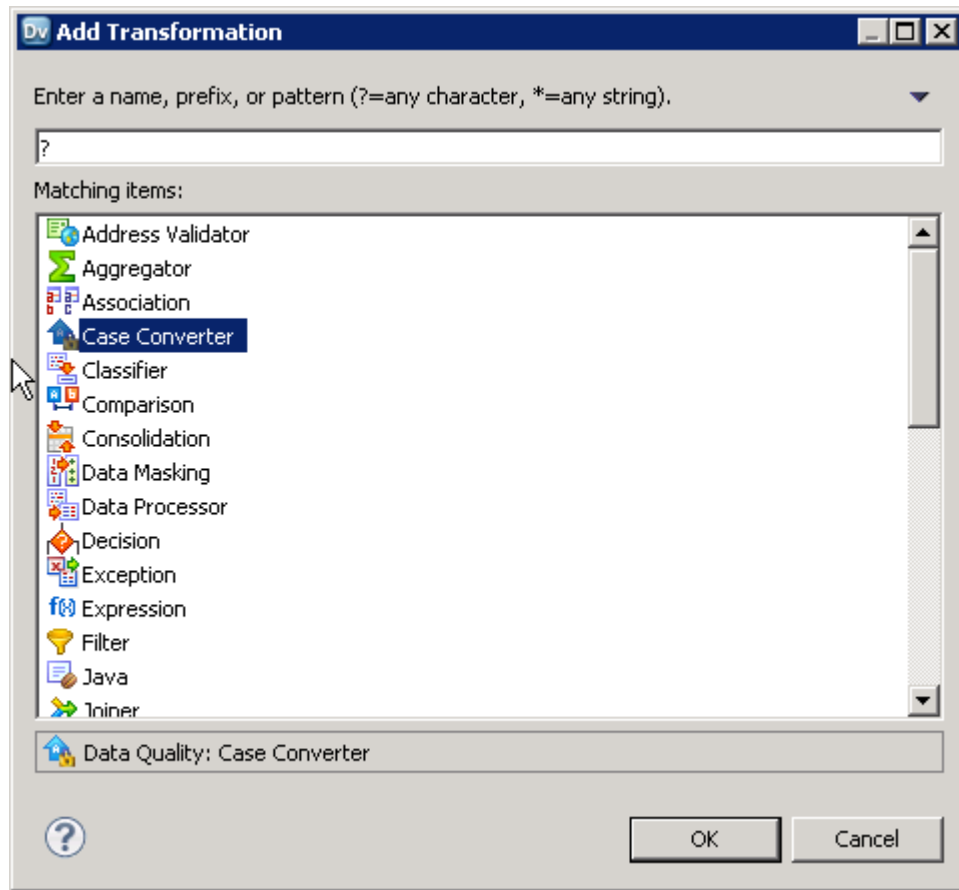


C_S_CRM_CUST_Mapplet 맵릿에는 C_S_CRM_CUST_Mapplet_In 입력 변환 및 C_S_CRM_CUST_Mapplet_Out 출력 변환이 포함되어 있습니다.

2. 맵릿 편집기를 마우스 오른쪽 단추로 클릭하고 **변환 추가**를 클릭합니다.

변환 추가 대화 상자가 나타납니다.

다음 이미지는 **변환 추가** 대화 상자를 보여 줍니다.



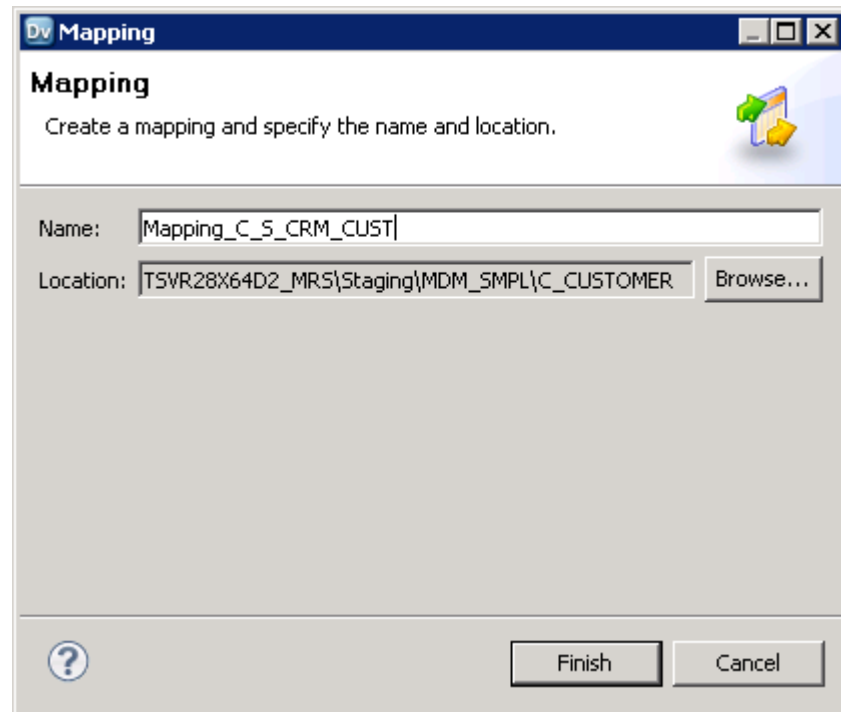
3. 원하는 변환을 선택하고 **확인**을 클릭합니다.
빈 변환이 맵렛 편집기에 표시됩니다.
4. 편집기에서 변환을 선택하고 변환을 구성합니다.

매핑 구성

소스, 대상 및 변환 개체로 매핑을 생성해야 합니다. 매핑 개체를 추가한 후에는 매핑 개체 간의 포트를 연결해야 합니다. 마지막으로 매핑의 유효성을 검사합니다.

1. 매핑을 생성하여 데이터를 변환하고 준비 테이블에 로드합니다.
 - a. **개체 탐색기** 보기에서 프로젝트 또는 폴더를 선택합니다.
 - b. **파일 > 새로 만들기 > 매핑**을 클릭합니다.

다음 이미지는 이름 및 위치 필드가 있는 매핑 대화 상자를 보여 줍니다.

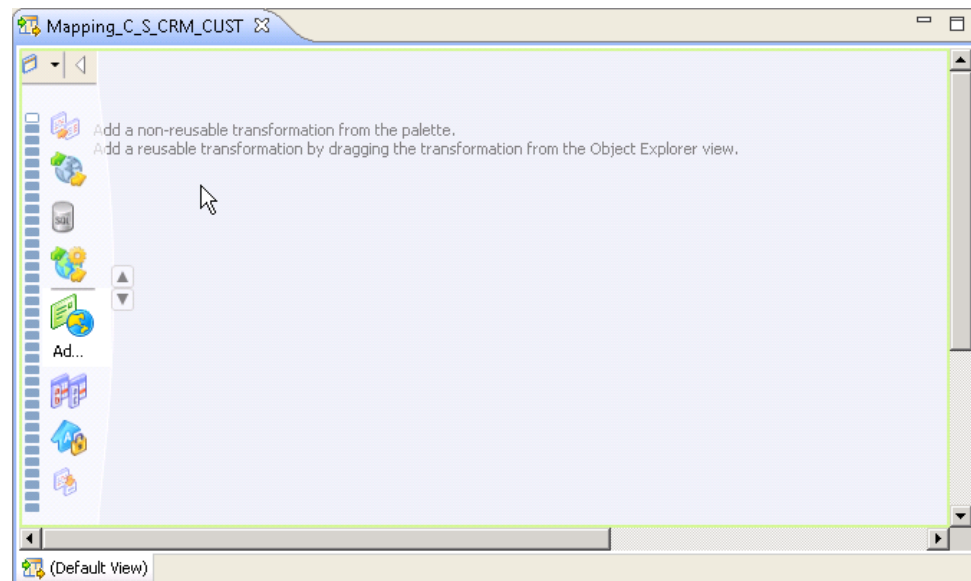


c. 매핑 이름을 입력합니다.

d. 마침을 클릭합니다.

빈 매핑이 편집기에 표시됩니다.

다음 이미지는 빈 매핑 Mapping_C_S_CRM_CUST를 보여 줍니다.

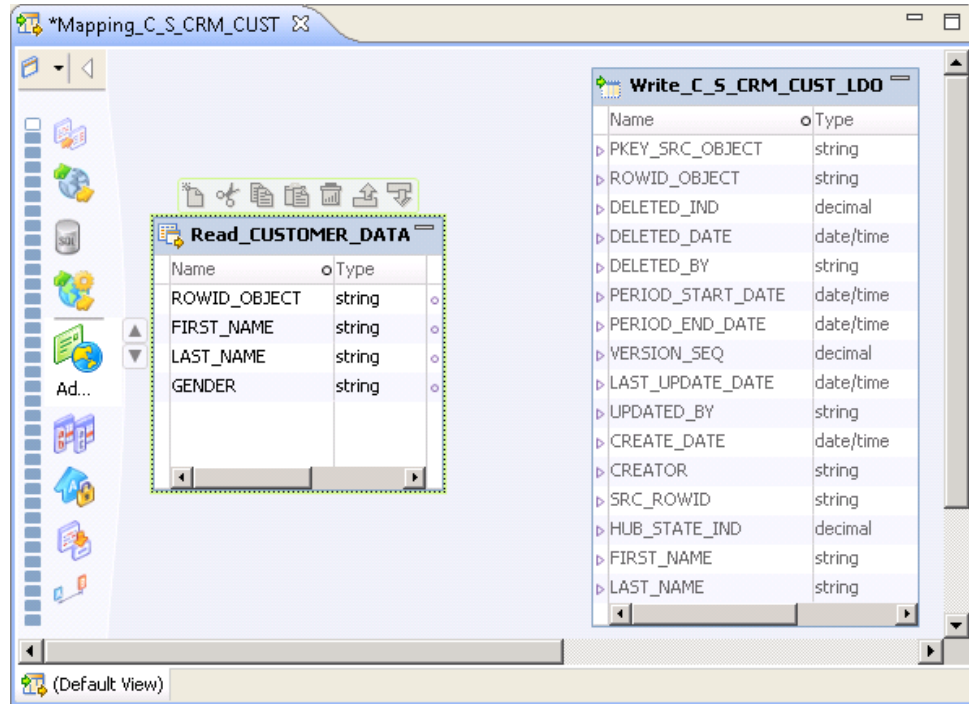


2. 개체를 매핑에 추가하여 소스와 대상 간의 데이터 흐름을 정합니다.

a. 소스에 대해 생성한 실제 데이터 개체를 편집기로 끌고 **읽기**를 선택하여 데이터 개체를 소스로 추가합니다.

- b. 준비 테이블을 나타내는 논리적 데이터 개체를 편집기로 끌고 **쓰기**를 선택하여 데이터 개체를 대상으로 추가합니다.

다음 이미지는 실제 데이터 개체 및 논리적 데이터 개체가 있는 **Mapping_C_S_CRM_CUST** 매핑을 보여 줍니다.

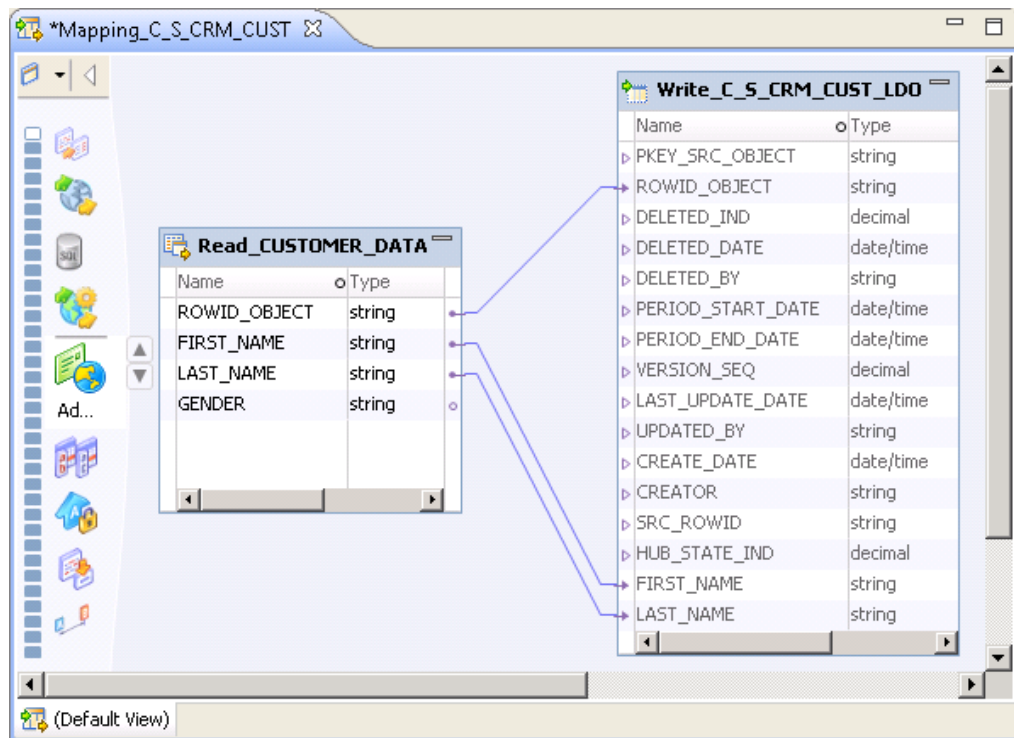


3. 매핑 개체 간의 포트를 연결합니다.

수동으로 또는 자동으로 포트를 연결할 수 있습니다.

참고: 준비 테이블 열을 편집, 추가 또는 삭제하고 모델 리포지토리와 **Hub** 저장소를 동기화한 후에는 해당하는 맷 포트의 연결이 해제됩니다. 그러므로 영향을 받은 포트를 수동으로 연결해야 합니다.

다음 이미지는 실제 데이터 개체 및 논리적 데이터 개체 간의 링크가 있는 Mapping_C_S_CRM_CUST 매핑을 보여 줍니다.



4. 매핑의 유효성을 검사하여 데이터 통합 서비스가 전체 매핑을 읽고 처리할 수 있는지 확인합니다.
 - a. **편집 > 유효성 검사**를 클릭합니다.
유효성 검사 로그 보기에 오류가 표시될 수도 있습니다.
 - b. 오류를 수정하고 매핑의 유효성을 다시 검사합니다.

제 20 장

로드 프로세스 구성

이 장에 포함된 항목:

- [개요, 352](#)
- [시작하기 전에, 352](#)
- [데이터를 로드하기 위한 태스크 구성, 353](#)
- [준비 테이블 구성, 353](#)
- [초기 데이터 로드 구성, 360](#)
- [소스 시스템에 대한 트러스트 구성, 360](#)
- [유효성 검사 규칙 구성, 365](#)

개요

이 장에서는 Informatica MDM Hub 구현에서 로드 프로세스를 구성하는 방법에 대해 설명합니다.

시작하기 전에

로드 프로세스를 구성하려면 먼저 다음 태스크를 완료해야 합니다.

- Informatica MDM Hub를 설치하고 Hub 저장소를 생성합니다.
- 스키마를 작성합니다
- 정의된 소스 시스템
- 랜딩 테이블을 생성합니다
- 준비 테이블을 생성합니다
- 로드 프로세스에 대해 알아봅니다

데이터를 로드하기 위한 태스크 구성

Informatica MDM Hub 구현에 데이터 로드 프로세스를 설정하려면 Hub 콘솔에서 다음 태스크를 완료해야 합니다.

- 준비 테이블을 구성합니다.
- 초기 데이터 로드를 구성합니다.
- 소스 시스템에 대한 트러스트를 구성합니다.
- 유효성 검사 규칙을 구성합니다.

로드 프로세스에 영향을 줄 수 있는 추가 구성 설정은 다음을 참조하십시오.

- [“행 ID별 로드” 페이지 303](#)
- [“고유 시스템” 페이지 476](#)
- [“일치 토큰 생성\(선택 사항\)” 페이지 267](#)
- [“로드 프로세스” 페이지 260](#)

준비 테이블 구성

MDM Hub에서는 랜딩 테이블에서 기본 개체로 데이터가 이동하는 흐름에서 중간의 임시 저장소 역할을 하는 준비 테이블을 사용합니다.

준비 테이블에는 Hub 저장소의 한 기본 개체 테이블에 대한 한 소스 시스템의 데이터가 포함됩니다. 일괄 준비 작업에서는 랜딩 테이블에서 준비 테이블을 채웁니다. 그러면 일괄 로드 작업에서는 준비 테이블에서 기본 개체를 채웁니다.

준비 테이블의 구조는 통합된 데이터를 포함할 대상 개체의 구조를 기반으로 합니다. 모델 작업 영역의 스키마 관리자를 사용하여 준비 테이블을 구성합니다. 준비 테이블을 정의하려면 하나 이상의 소스 시스템이 정의되어 있어야 합니다.

준비 테이블 열

준비 테이블에는 시스템 열 및 사용자 정의 열이 포함됩니다.

시스템 준비 테이블 열

스키마 관리자는 시스템 준비 테이블 열을 작성 및 유지 관리합니다.

다음 테이블에는 준비 테이블의 시스템 열이 설명되어 있습니다.

실제 이름	MDM Hub 데이터 유형(크기)	설명
PKEY_SRC_OBJECT	VARCHAR(255)	소스 시스템의 기본 키입니다. PKEY_SRC_OBJECT 값은 고유해야 합니다. 소스 레코드에 고유한 단일 열이 없는 경우 여러 열의 값을 연결하여 레코드를 고유하게 식별합니다.
ROWID_OBJECT	CHAR(14)	MDM Hub의 기본 키입니다. MDM Hub는 준비 프로세스 동안 고유한 ROWID_OBJECT 값을 할당합니다.

실제 이름	MDM Hub 데이터 유형(크기)	설명
DELETED_IND	INT	나중에 사용하기 위해 예약되어 있습니다.
DELETED_DATE	TIMESTAMP	나중에 사용하기 위해 예약되어 있습니다.
DELETED_BY	VARCHAR(50)	나중에 사용하기 위해 예약되어 있습니다.
LAST_UPDATE_DATE	TIMESTAMP	소스 시스템에서 레코드를 마지막으로 업데이트한 날짜입니다. 기본 개체의 경우 LAST_UPDATE_DATE는 교차 참조 테이블의 LAST_UPDATE_DATE 및 SRC_LUD를 채우고, 트러스트 설정을 기반으로 기본 개체 테이블의 LAST_UPDATE_DATE도 채울 수 있습니다.
UPDATED_BY	VARCHAR(50)	최근 업데이트를 수행한 사용자나 프로세스입니다.
CREATE_DATE	TIMESTAMP	레코드가 작성된 날짜입니다.
PERIOD_START_DATE	TIMESTAMP	레코드 유효 기간의 시작 날짜입니다. 시간 표시 막대가 활성화된 기본 개체의 경우 PERIOD_START_DATE 값이 필요합니다.
PERIOD_END_DATE	TIMESTAMP	레코드 유효 기간의 끝 날짜입니다. 시간 표시 막대가 활성화된 기본 개체의 경우 PERIOD_END_DATE 값이 필요합니다.
CREATOR	VARCHAR(50)	레코드의 작성을 담당하는 사용자 또는 프로세스입니다.
SRC_ROWID	VARCHAR(30)	준비 테이블에서 랜딩 테이블까지 레코드를 역추적하는 데이터베이스 내부 행 ID 열입니다.
HUB_STATE_IND	INT	상태 사용 기본 개체용입니다. 해당 레코드의 상태를 나타내는 정수 값입니다. 다음 값이 유효합니다. - 0=보류 중 - 1=활성 - -1=삭제됨 기본값은 1입니다.
VERSION_SEQ	INT	시간 표시 막대가 활성화된 기본 개체의 경우 값은 동일한 기본 소스 키의 레코드들이 준비 테이블에 로드된 순서를 보여 줍니다. 시간 표시 막대가 활성화되지 않은 기본 개체의 경우 이 값은 1이어야 합니다. VERSION_SEQ 열은 사용자 정의 랜딩 테이블 열을 대상으로 매핑됩니다. 랜딩 테이블 열에는 기본 키는 같지만 기간 시작 날짜와 기간 종료 날짜가 다른 레코드의 버전 시퀀스가 포함됩니다.

사용자 정의 준비 테이블 열

불필요한 트러스트 계산으로 인한 성능 저하를 방지하려면 준비 테이블에 추가한 열이 특정 소스 시스템에서 제공된 것이어야 합니다.

소스 시스템의 열을 추가할 때 트러스트 열을 여러 준비 테이블의 데이터가 있는 열로 제한할 수 있습니다. 준비 테이블에 특정 소스 시스템의 열이 있는 경우 각 열을 모든 소스 시스템에서 제공된 것처럼 처리하지 않아도 됩니다. 각 열에 대해 트러스트를 추가할 필요가 없습니다. 또한 열에 값을 제공하지 않은 모든 소스에 대한 null 값의 트러스트를 다운그레이드하는 유효성 검사 규칙이 필요하지 않습니다.

소스 시스템 키 유지

로드 작업 중 MDM Hub는 기본 키를 생성하거나 소스 시스템의 기본 키를 사용할 수 있습니다. MDM Hub는 데이터 로드를 위해 소스 시스템 하나의 소스 시스템 키를 유지할 수 있습니다.

기본 개체에 준비 테이블을 추가할 경우 MDM Hub가 소스 시스템의 기본 키 값을 사용해야 할지 또는 MDM Hub가 생성하는 기본 키 값을 사용해야 할지 지정합니다. 소스 시스템 키를 유지하는 옵션을 활성화하면 MDM Hub는 로드 작업을 수행하는 동안 준비 테이블의 PKEY_SOURCE_OBJECT 열에 있는 소스 시스템의 기본 키 값을 가져와서 대상 기본 개체의 ROWID_OBJECT 열에 삽입합니다. 또한 소스 시스템 키를 유지하는 옵션을 활성화하면 MDM Hub는 기본 개체에 로드할 레코드에 대해 기본 키를 생성하지 않습니다.

소스 시스템 키를 유지하는 옵션을 활성화하기 전에 다음과 같은 MDM Hub 동작을 고려하십시오.

- 준비 테이블이 작성된 후에는 소스 시스템 키를 유지하는 설정을 변경할 수 없습니다.
- 같은 소스 시스템에서 가져왔더라도 각 기본 개체의 준비 테이블 하나에 소스 시스템 키를 유지하는 옵션을 설정할 수 있습니다. 예약된 키 범위는 초기 로드 시에만 설정됩니다.
- 로드 프로세스 동안 여러 레코드에 동일한 PKEY_SRC_OBJECT 값이 포함된 경우, 존속 레코드는 LAST_UPDATE_DATE가 최근인 레코드입니다. 다른 레코드는 거부 테이블로 이동됩니다.
- 소스 시스템 키를 유지하는 옵션이 활성화된 소스 시스템에 대해 첫 번째 레코드를 로드할 때 MDM Hub는 최상위 예약된 키를 예약합니다.
- MDM Hub가 최상위 예약된 키까지의 모든 소스 시스템 키를 유지하도록 하려면 소스 시스템 키를 유지하는 옵션이 활성화된 소스 시스템을 먼저 로드한 후 다른 소스 시스템을 로드해야 합니다.
- 다른 소스 시스템에서 사용 중이고 해당 값이 최상위 예약된 키보다 순위가 낮은 소스 시스템 키가 레코드에 포함되어 있는 경우에는 로드 작업이 현재 레코드 블록을 처리하지 않고 경고를 생성합니다.
- 숫자가 아닌 ROWID_OBJECT는 MDM Hub에서 지원하지 않습니다. MDM Hub는 MDM Hub가 숫자가 아닌 소스 시스템 키에 대해 생성하는 ROWID_OBJECT를 사용합니다.

최상위 예약된 키 지정

최상위 예약된 키는 최상위 소스 시스템 키입니다. MDM Hub에서 생성하는 키가 소스 시스템 키와 충돌하지 않게 하려면 최상위 소스 시스템 키를 예약합니다. 소스 시스템 키를 유지하는 옵션을 활성화한 경우에는 레코드에 사용할 최상위 소스 시스템 키를 지정할 수 있습니다.

최상위 예약된 키를 소스 시스템 키의 상한으로 설정합니다. 소스 시스템 키의 예상 범위에 버퍼를 추가하려면 소스 시스템의 상한보다 조금 높은 값을 최상위 예약된 키의 값으로 설정합니다.

기본 개체에 대해 초기 데이터 로드를 수행할 때 MDM Hub는 다음과 같은 태스크를 수행합니다.

- 준비 테이블의 PKEY_SOURCE_OBJECT 열에 있는 값을 기본 개체의 ROWID_OBJECT 열에 삽입합니다.
- 생성해야 하는 기본 키의 시작 값을 최상위 예약된 키보다 1이 큰 값으로 설정합니다.

초기 데이터 로드 이후부터 기본 개체에 데이터를 로드할 때 MDM Hub는 다음과 같은 태스크를 수행합니다.

- 소스 시스템 키를 유지하는 소스 시스템의 경우 MDM Hub는 준비 테이블에 있는 PKEY_SOURCE_OBJECT 열의 값을 기본 개체의 ROWID_OBJECT 열에 삽입합니다.
다음과 같은 시나리오에서는 MDM Hub가 기본 키를 생성합니다.
 - 소스 시스템 키가 최상위 예약된 키보다 큰 경우
 - 소스 시스템 키가 숫자 값이 아닌 경우
- 소스 시스템 키를 유지하지 않는 소스 시스템에 대해 MDM Hub가 기본 키를 생성합니다.

최상위 예약된 키 예제

준비 테이블에 대한 소스 시스템 키를 유지하고 레코드에 사용할 최상위 소스 시스템 키를 100으로 설정합니다. 소스 시스템 키를 유지한 준비 테이블에서 추가 레코드를 로드합니다. 그런 다음 소스 시스템 키를 유지한 준비 테이블 외의 다른 준비 테이블에서 레코드를 로드합니다.

다음 로드 작업 중에 ROWID_OBJECT 열의 키 값 변경을 확인할 수 있습니다.

1. 소스 시스템 키를 유지할 준비 테이블에서 기본 소스 시스템 키 20, 21 및 22를 사용하여 초기 데이터 로드를 수행합니다.
MDM Hub가 ROWID_OBJECT 값이 20, 21 및 22인 레코드 세 개를 연결된 기본 개체에 로드합니다.
2. 초기 데이터 로드 후 소스 시스템 키를 유지할 준비 테이블에서 기본 소스 시스템 키를 가진 5개 레코드를 로드합니다. 이러한 레코드의 기본 소스 시스템 키는 23, 24와 25 그리고 AB와 YZ입니다.
MDM Hub가 ROWID_OBJECT 값이 23, 24, 25, 101 및 102인 레코드 다섯 개를 연결된 기본 개체에 로드합니다.
3. 소스 시스템 키를 유지하지 않을 준비 테이블에서 기본 키가 포함된 레코드 3개를 로드.
MDM Hub가 ROWID_OBJECT 값이 103부터 105까지인 레코드 세 개를 연결된 기본 개체에 로드합니다.

셀 업데이트 활성화

성능을 개선하고 MDM Hub가 변경된 값으로 셀을 업데이트할 수 있도록 하려면 셀 업데이트 옵션을 활성화하십시오. 기본적으로 로드 프로세스에서 더 높은 트러스트 수준을 가진 모든 인바운드 레코드의 대상 기본 개체에 있는 셀 값을 대체합니다. 대체된 값이 동일하더라도 로드 프로세스에서 셀 값을 대체합니다.

값이 변경되지 않은 경우에도 MDM Hub는 셀의 마지막 업데이트 날짜를 수신 레코드와 연결된 날짜로 업데이트하고 동일한 트러스트 수준을 셀에 새로운 값으로 할당합니다. 준비 테이블을 구성할 때 셀 업데이트 활성화 동작을 변경하려면 다음을 수행하십시오. 셀 업데이트가 활성화된 경우 로드 작업 중에 MDM Hub는 기본 개체에서 대상 레코드를 업데이트하기 전에 먼저 셀 값을 교차 참조 테이블의 최신 콘텐츠와 비교합니다. 시스템의 교차 참조 레코드에 셀과 동일한 값이 있을 경우 MDM Hub는 Hub 저장소에서 셀을 업데이트하지 않습니다.

MDM Hub에서 대상 기본 개체 레코드의 마지막 업데이트 날짜와 트러스트 값을 업데이트할 필요가 없을 경우 로드 작업 동안 성능을 향상하려면 셀 업데이트를 활성화하십시오.

준비 테이블의 열에 대한 속성

준비 테이블의 열에 대한 속성은 외래 키 조회에 대한 정보를 제공합니다. 또한 준비 테이블 열에 null 값이 포함된 경우 속성을 사용하여 일괄 로드 및 Put API 동작을 구성할 수 있습니다.

참고: MDM Hub에서 빈 문자열은 빈 문자열을 적용하는 데이터베이스 유형에 상관없이 null 값에 해당합니다.

준비 테이블 열에는 다음과 같은 속성이 포함됩니다.

열

연결된 기본 개체에서 정의된 열의 이름입니다.

조회 시스템

조회 테이블이 교차 참조 테이블인 경우 조회 시스템의 이름입니다.

조회 테이블

준비 테이블의 외래 키 열의 경우 조회 열을 포함한 테이블의 이름입니다.

조회 열

준비 테이블의 외래 키 열의 경우 조회 테이블의 조회 열 이름입니다.

Null 외래 키 허용

활성화하면 조희 열에 null 값이 포함된 경우 로드 일괄 작업 또는 Put API가 데이터를 로드할 수 있습니다. 외래 키 관계가 필요한 경우 **Null 외래 키 허용**을 활성화하지 마십시오.

비활성화하면 조희 열에 null 값이 포함된 경우 로드 일괄 작업 또는 Put API가 데이터를 로드할 수 없습니다. Hub 콘솔에서 레코드를 거부하고 레코드를 로드하지 않습니다.

Null 업데이트 허용

한 소스가 열에 대해 null 값을 적용하는 반면 다른 소스는 같은 열에 대해 null이 아닌 값을 갖는 경우를 제어합니다. BVT(최선의 진실, Best Version of the Truth) 계산을 수행하는 모든 프로세스(로드, 업데이트 로드, 넣기, 정리 넣기, 병합, 병합 해제, BVT 다시 계산 및 유효성 다시 검사)는 이 속성을 사용합니다.

- **True.** 활성화하면 이 열에 대한 가장 신뢰할 수 있는 값이 null 값인 경우 프로세스는 null 값을 기본 개체 레코드에 쓸 수 있습니다.
- **False.** 기본값입니다. 비활성화하면 다른 소스에서 열에 대해 null이 아닌 값을 적용할 경우 프로세스는 기본 개체 레코드에 null 값을 쓸 수 없습니다.

프로세스가 실행되면 null 값을 적용하는 각 소스에 대해 프로세스는 소스의 준비 테이블을 확인합니다. 열에서 **Null 업데이트 허용** 속성을 false로 설정하는 경우 프로세스는 해당 열에 트러스트를 0 미만으로 다운그레이드합니다. 그런 다음 프로세스는 조정된 트러스트 점수를 사용하여 BVT를 계산합니다. 이 방법을 사용하면 프로세스는 기본 개체 레코드에 쓸 가장 신뢰할 수 있는 null이 아닌 값을 선택할 수 있습니다.

다음의 특별한 경우에 프로세스는 준비 테이블에서 **Null 업데이트 허용** 속성을 무시하고 대신 기본 개체 테이블의 열에 **Null 값 적용** 속성을 사용합니다.

- 프로세스는 소스와 관련된 여러 준비 테이블을 검색합니다. 준비 테이블에 열의 **Null 업데이트 허용** 속성에 대한 혼합 설정이 포함됩니다.
- 프로세스는 소스에 대한 준비 테이블을 찾지만 준비 테이블에 열을 구성하지는 않습니다. 예를 들어 서비스 호출은 항상 열 값을 업데이트하므로 준비 테이블에 열을 구성하지 않습니다.
- 프로세스가 소스에 대한 준비 테이블을 찾을 수 없습니다. 예를 들어 교차 참조 레코드의 STG_ROWID_TABLE 열에 값이 없고 준비 테이블 결정을 위한 대체 방법이 명확하지 않습니다.

프로세스는 다음 시나리오에서 **Null 업데이트 허용** 속성과 **Null 값 적용** 속성을 모두 무시합니다.

- 모든 소스가 null이 아닌 값을 제공하는 경우 트러스트 수준이 가장 높은 소스의 값이 존속됩니다.
- 모든 소스가 null 값을 제공하는 경우 null 값이 존속됩니다.
- 기본 개체에 소스 시스템이 하나뿐인 경우 해당 소스의 값(null 또는 null이 아닌 값)이 기본 개체에 저장됩니다.

Null 업데이트 허용 예

영향을 주는 세 개의 소스와 함께 고객 기본 개체가 있습니다. 업데이트 로드 프로세스는 중간 이름이 삭제된(즉, 값이 null인) 소스 A에서 데이터를 로드합니다. 소스 B 및 소스 C에 고객의 중간 이름이 있습니다.

다음 테이블에서는 세 개의 소스, 준비 테이블의 중간 이름 열 설정, 트러스트 조정 및 BVT 계산 결과를 보여 줍니다.

소스	준비 테이블 중간 이름 트러스트	준비 테이블 중간 이름 Null 업데이트 허용	XREF 레코드 중간 이름 값	조정 후 트러스트	기본 개체 레코드 BVT 값
소스 A	90	false	null	< 0	-
소스 B	60	false	Edward	60	-
소스 C	80	true	Edwin	80	Edwin

업데이트 로드 프로세스가 중간 이름 열에 대한 BVT 계산을 시작합니다. 처음에 소스 A는 트러스트가 가장 높은 90이지만 값은 null입니다. 프로세스는 소스 A의 준비 테이블을 찾고 중간 이름 열의 **Null 업데이트 허용** 속성을 확인합니다. 속성은 **false**입니다. 프로세스는 소스 A의 중간 이름 열에 대한 트러스트를 0 미만으로 다운그레이드합니다. 트러스트 조정 후 소스 C의 트러스트가 80으로 가장 높습니다. 프로세스는 소스 C의 중간 이름을 선택하고 기본 개체 레코드에 Edwin을 씁니다.

준비 테이블의 속성 변경

필요한 경우 준비 테이블 속성을 변경할 수 있습니다.

- 스키마 관리자를 시작합니다.
- 쓰기 잠금을 획득합니다.
- 스키마 트리에서 **기본 개체** 노드를 확장하고 이 준비 테이블과 연결된 기본 개체의 노드를 확장합니다.
준비 테이블이 기본 개체와 연결되어 있는 경우 **준비 테이블** 노드를 확장하여 기본 개체를 표시합니다.
- 구성할 준비 테이블을 선택합니다.
스키마 관리자에 선택한 테이블의 속성이 표시됩니다.
- 준비 테이블 속성을 지정합니다.
편집할 각 속성에서 옆에 있는 **편집** 단추를 클릭하고 새 값을 지정합니다.
참고: 준비 테이블과 관련 지원 테이블(예: 원시 테이블 및 기본 랜딩 테이블)이 비어 있는 경우 소스 시스템을 변경할 수 있습니다.
준비 테이블 또는 관련 테이블에 데이터가 포함되어 있는 경우에는 소스 시스템을 변경하지 마십시오.
- 기본 개체의 열 목록에서 이 소스 시스템이 제공할 열을 변경합니다.
 - 각 열을 개별적으로 클릭할 필요 없이 모든 열을 선택하려면 **모두 선택** 단추를 클릭합니다.
 - 선택된 모든 열을 선택 취소하려면 **모두 지우기** 단추를 클릭합니다.**참고:** "Rowid 개체"와 "마지막 업데이트 날짜"는 자동으로 선택됩니다. 이러한 열은 선택 취소하거나 속성을 변경할 수 없습니다.
- 필요한 경우 열 속성을 변경합니다.
- 필요한 경우 외래 키 열에 대한 조회를 변경합니다. 열을 선택하고 **편집** 단추를 클릭하여 조회 열을 구성합니다.
- 셀 업데이트를 변경하려면 **셀 업데이트** 확인란을 클릭합니다.
- 필요한 경우 준비 테이블의 열 구성을 변경합니다.
- 필요한 경우 이 준비 테이블에 대해 감사 내역 및 델타 검색을 구성합니다.

12. 저장 단추를 클릭하여 변경 내용을 저장합니다.

외래 키 열에 대한 조회

조회를 사용하여 로드 작업 중에 상위 테이블에서 데이터를 검색할 수 있습니다. 준비 테이블의 외래 키 열이 상위 테이블의 기본 키와 관련된 경우 상위 테이블에서 데이터를 검색하도록 조회를 구성하십시오.

조회 테이블의 대상 열은 기본 키 등의 고유 열이어야 합니다.

조회를 정의한 후 기본 개체에서 로드 작업이 실행될 경우 MDM Hub이 소비자 코드 교차 참조 테이블의 소스 시스템 열에서 기본 키에 있는 소스 시스템의 소비자 코드 값을 조회합니다. 조회 후 MDM Hub이 소스 소비자 유형에 해당하는 고객 유형 ROWID_OBJECT 값을 반환합니다.

예제

조직의 MDM Hub 구현에 Consumer 상위 기본 개체 및 Address 하위 기본 개체라는 두 개의 기본 개체가 있습니다. 기본 개체의 관계는 다음과 같습니다.

```
Consumer.Rowid_object = Address.Consumer_Fkey
```

이 경우 Consumer_Fkey가 주소 준비 테이블에 있고 일부 열의 데이터를 조회합니다.

참고: Address.Consumer_Fkey는 Consumer.Rowid_object와 동일해야 합니다.

이 예제에서 다음 유형의 조회를 구성할 수 있습니다.

- 소비자 기본 개체 조회 테이블의 ROWID_OBJECT(기본 키)에 대한 조회.
- 소비자 기본 개체에 대한 교차 참조 테이블의 기본 키(PKEY_SRC_OBJECT 열)에 대한 조회.
이 경우에는 조회 시스템도 정의해야 합니다. 교차 참조 테이블의 PKEY_SRC_OBJECT 열에 대한 조회를 구성한 경우 이 준비 테이블에 연결된 소스 시스템과는 다른 소스 시스템에 연결된 상위 테이블을 가리키십시오.
- 기본 개체 또는 교차 참조 테이블의 다른 고유 열(사용 가능한 경우)에 대한 조회.

조회 구성

외래 키 관계를 통해 조회를 구성할 수 있습니다.

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 트리에서 **기본 개체** 노드를 확장하고 이 준비 테이블과 연결된 기본 개체의 노드를 확장합니다.
4. 구성할 준비 테이블을 선택합니다.
5. 구성할 외래 키 열의 행을 선택합니다.

외래 키 열에 대해서만 **조회 편집** 단추가 활성화됩니다.

6. **조회 편집** 단추를 클릭합니다.

스키마 관리자에 조회 정의 대화 상자가 표시됩니다.

조회 정의 대화 상자에는 상위 기본 개체 및 해당 교차 참조 테이블이 고유 열과 함께 표시됩니다.

7. 조회 대상 열을 선택합니다.

- 기본 개체에 대한 조회를 정의하려면 기본 개체를 확장하고 Rowid_Object(이 기본 개체의 기본 키)를 선택합니다.
- 교차 참조 테이블에 대한 조회를 정의하려면 "PKey 소스 개체"(이 교차 참조 테이블의 소스 시스템에 대한 기본 키)를 선택합니다.

- 다른 고유 열에 대한 조회를 정의하려면 해당 열을 선택합니다.
- 참고:** 관계를 삭제하면 조회가 지워집니다.
8. 조회 열이 관계 테이블의 PKey 소스 개체인 경우 조회 시스템 목록에서 조회 시스템을 선택합니다.
 9. **확인**을 클릭합니다.
 10. 필요한 경우 **Null 업데이트 허용** 확인란을 구성하여 로드 작업을 통해 null이 아닌 값이 이미 포함된 셀에 null 값을 지정할 경우의 결과를 지정합니다.
 11. 각 열에 대해 **Null 외래 키 허용** 옵션을 구성하여 외래 키 열에 null 값(사용 가능한 조회 값 없음)이 포함될 경우의 결과를 지정합니다.
 12. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

초기 데이터 로드 구성

기본 개체로의 초기 데이터 로드 중에 일괄 작업을 병렬화하여 성능을 개선할 수 있습니다.

초기 데이터 로드 성능을 개선하려면 C_REPOS_TABLE에서 다음 매개 변수를 구성합니다.

PARALLEL_BATCH_JOB_THRESHOLD

일괄 작업을 병렬화합니다. MDM Hub에서 사용 가능한 CPU 코어 수보다 작은 숫자로 값을 설정합니다. 예를 들어 시스템에 4개의 CPU가 있지만 MDM Hub에서는 2개만 사용 가능하고 각 CPU에 8개의 프로세서 코어가 있는 경우 설정할 수 있는 최대값은 15입니다. 기본값은 1,000입니다.

소스 시스템에 대한 트러스트 구성

이 섹션에서는 Informatica MDM Hub 구현에서 트러스트를 구성하는 방법에 대해 설명합니다.

트러스트 정보

여러 소스 시스템에 기본 개체 테이블의 동일한 열에 해당하는 특성이 포함될 수 있습니다.

예를 들어 여러 시스템에 고객의 주소가 저장되어 있지만 해당 데이터에 대한 소스로는 한 시스템이 다른 시스템보다 신뢰도가 높을 수 있습니다. 이와 같이 시스템 간에 차이가 있는 경우 Informatica MDM Hub에서는 사용할 최적의 값을 결정해야 합니다.

서로 다른 여러 소스 시스템의 열 데이터 간에 상대적 신뢰도를 비교하기 쉽도록 Informatica MDM Hub에서는 열의 트러스트를 구성하는 기능을 제공합니다. 트러스트는 특정 데이터 부분의 상대적 정확도에 대한 신뢰도를 나타냅니다. 각 소스의 각 열에 대해 0에서 100 사이의 수(0이 가장 낮은 트러스트이고 100이 가장 높은 트러스트)로 표현되는 트러스트 수준을 정의할 수 있습니다. 이 값은 그 자체로는 의미가 없고 다른 트러스트 값과 비교하여 더 높은 쪽을 확인할 때만 의미가 있습니다.

트러스트는 데이터 보존 기간, 시간에 따라 신뢰성이 약화되는 정도 및 데이터의 유효성을 고려합니다. 트러스트는 두 레코드가 통합될 때 존속되는 레코드를 결정하고 소스 시스템의 업데이트가 마스터 레코드를 업데이트하기에 충분히 신뢰할 수 있는지 여부를 확인하는 데 사용됩니다.

트러스트 수준

트러스트 수준은 0에서 100 사이의 수입니다. 이 값은 그 자체로는 의미가 없고 다른 트러스트 값과 비교할 때만 의미가 있습니다.

시간에 따른 데이터 신뢰도 붕괴

지정된 소스 시스템의 데이터 신뢰도가 시간이 지남에 따라 붕괴(약화)될 수 있습니다. 트러스트 계산에서 이 사실을 반영할 수 있도록 Informatica MDM Hub에서는 트러스트가 활성화된 열에 대해 붕괴 특성을 구성할 수 있습니다. **붕괴 기간**은 트러스트 수준이 최대 트러스트 수준에서 최소 트러스트 수준으로 붕괴하는 데 걸리는 기간입니다.

트러스트 계산

로드 프로세스는 기본 개체에서 트러스트가 활성화된 열의 트러스트를 계산합니다. 트러스트가 활성화된 열이 있는 레코드의 경우 로드 프로세스는 셀 데이터에 트러스트 점수를 할당합니다. 이 트러스트 점수는 초기에는 해당 열에 대해 구성된 트러스트 설정을 기반으로 합니다. 이후에 트러스트 계산 후 로드 프로세스에서 유효성 검사 규칙(트러스트가 활성화된 열에 대해 구성된 경우)이 적용되면 트러스트 점수가 다운그레이드될 수 있습니다.

모든 트러스트 계산은 시스템 날짜를 기준으로 하지만 시간 표시 막대가 사용되는 기본 개체의 기록 쿼리에 대한 트러스트 계산은 기록 날짜를 기준으로 합니다. 트러스트가 활성화된 기본 개체에 대한 트러스트 계산은 유효 날짜를 기준으로 하지 않습니다.

업데이트형 로드 작업에 대한 트러스트 계산

로드 프로세스 도중 준비 테이블의 레코드가 업데이트형 로드 작업에 사용되며 해당 레코드의 트러스트가 활성화된 열에 변경된 셀 값이 들어 있으면 로드 프로세스는 다음 항목에 대한 트러스트 점수를 계산합니다.

- 준비 테이블의 소스 레코드에 있는 셀 데이터(업데이트된 정보가 포함됨)
- 기본 개체의 대상 레코드에 있는 셀 데이터(기존 정보가 포함됨)

소스 레코드에 있는 셀 데이터의 트러스트 점수가 대상 레코드에 있는 셀 데이터의 트러스트 점수보다 높으면 Informatica MDM Hub는 기본 개체 레코드의 셀을 준비 테이블 레코드의 셀 데이터로 업데이트합니다.

두 개의 기본 개체 레코드를 통합할 경우의 트러스트 계산

기본 개체에 있는 두 개의 레코드를 통합할 경우 Informatica MDM Hub는 병합할 두 레코드의 트러스트된 각 열에 대해 트러스트 점수를 계산합니다. 통합된 최종 레코드에는 트러스트 점수가 가장 높은 셀이 존속됩니다. 트러스트 점수가 동일할 경우 Informatica MDM Hub는 레코드를 비교합니다.

트러스트가 활성화된 열에 대한 제어 테이블

기본 개체 레코드에서 트러스트가 활성화된 각 열에 대해 Informatica MDM Hub는 해당 제어 테이블에서 마지막 업데이트 날짜와 소스 시스템 식별자가 포함된 레코드를 유지 관리합니다. 이러한 설정을 기반으로 Informatica MDM Hub는 항상 열 값에 대한 최신 트러스트를 계산할 수 있습니다.

기본 개체에 대해 기록이 활성화된 경우 Informatica MDM Hub는 기본 개체의 기록 테이블 및 해당 교차 참조 테이블 외에 제어 테이블에 대한 별도의 기록 테이블도 유지 관리합니다.

기본 개체 레코드와 교차 참조 레코드의 셀 값

기본 개체의 교차 참조 테이블에는 각 소스 시스템의 가장 최신 값이 포함됩니다. 트러스트 설정이 없는 기본 상태에서 기본 개체에는 데이터를 가져오는 소스 시스템에 관계없이 가장 최신 값이 포함됩니다.

트러스트가 활성화된 열의 경우 기본 개체 레코드의 셀 값이 교차 참조 테이블의 해당 레코드와 다를 수도 있습니다. 트러스트 계산 후 로드 프로세스에서 실행되는 유효성 검사 규칙이 이전에 셀 값을 제공했던 소스가 셀을 업데이트할 수 없도록 셀에 대한 트러스트를 다운그레이드할 수 있습니다.

트러스트 점수 재정의

데이터 스튜어드는 특정 값이 정확하다고 직접적으로 알고 있는 경우 계산된 트러스트 설정을 수동으로 재정의할 수 있습니다. 또한 데이터 스튜어드는 기본 개체의 레코드에 값을 바로 입력할 수도 있습니다. 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

상태 사용 기본 개체의 트러스트

상태 사용 기본 개체의 경우 트러스트는 **ACTIVE**, **PENDING** 및 **DELETED** 상태의 레코드에 대해 계산됩니다. **DELETED** 상태의 레코드가 **ACTIVE** 및 **PENDING** 상태의 레코드보다 더 높은 트러스트 점수를 갖지 않도록 **DELETED** 상태인 레코드의 경우에는 트러스트가 추가로 다운그레이드됩니다.

일괄 작업 시 트러스트가 활성화된 열 수에 대한 제약 조건

기본 개체에 트러스트가 활성화된 열이 다수 포함된 경우 동기화 일괄 작업이 실패할 수 있습니다.

마찬가지로, 트러스트가 활성화된 열 또는 유효성 검사 사용 열이 다수인 경우에는 자동 병합 작업이 실패할 수 있습니다. 작업이 실패하게 되는 정확한 열 수는 변수이며, 열 이름의 길이와 트러스트가 활성화된 열(자동 병합 작업의 경우 유효성 검사 사용 열도 포함)의 개수에 따라 달라집니다. 허용 가능한 최대 길이인 26자에 근접한 열 이름은 긴 이름으로 간주됩니다. 이 문제를 방지하려면 트러스트가 활성화된 열의 개수를 100개 미만으로 유지하거나 열 이름의 길이를 짧게 하십시오. 해결 방법은 기본 개체를 저장하기 전에 모든 트러스트/유효성 검사 열을 활성화하여 동기화 작업이 실행되지 않도록 하는 것입니다.

트러스트 속성

이 섹션에서는 트러스트가 활성화된 열에 대해 구성할 수 있는 트러스트 속성에 대해 설명합니다.

트러스트 속성은 기본 개체의 트러스트가 활성화된 열에 레코드를 제공할 수 있는 각 소스 시스템에 대해 개별적으로 구성됩니다.

최대 트러스트

최대 트러스트(시작 트러스트)는 데이터 값이 변경된 직후에 적용되는 트러스트 수준입니다. 예를 들어 소스 시스템 X에서 전화 번호 필드가 555-1234에서 555-4321로 변경되는 경우 새 값에는 해당 시스템 X에서 전화 번호 필드에 부여하는 최대 트러스트 수준이 적용됩니다. 최대 트러스트 수준을 상대적으로 높게 설정하면 일반적으로 소스 시스템의 변경 내용을 기본 개체에 적용할 수 있습니다.

최소 트러스트

최소 트러스트는 오래된 데이터 값(붕괴 기간이 경과한 데이터 값)에 적용되는 트러스트 수준입니다. 이 값은 최대 트러스트보다 작거나 같아야 합니다.

참고: 최대 트러스트와 최소 트러스트가 같은 경우에는 붕괴 곡선이 수평선으로 나타나고 붕괴 기간과 붕괴 유형이 적용되지 않습니다.

단위

붕괴 기간을 계산하는 데 사용되는 단위(일, 주, 개월, 분기 또는 년)를 지정합니다.




붕괴

붕괴 기간을 계산하는 데 사용되는 시간 단위(일, 주, 개월, 분기 또는 년)의 수를 지정합니다.

참고: 그래프 보기를 최적화하려면 지정하는 붕괴 기간을 1에서 100 사이로 제한하십시오.

그래프 유형

붕괴 기간 중 트러스트 수준이 감소하는 방식을 나타내는 붕괴 패턴입니다. 그래프 유형은 다음과 같은 설정의 붕괴 패턴을 표시합니다.

아이콘	그래프 유형	설명
	선형	가장 단순한 붕괴 패턴입니다. 붕괴 패턴이 최대 트러스트에서 최소 트러스트를 잇는 직선 형태로 나타납니다.
	RISL(초기에 빠르다 나중에는 느려짐)	트러스트가 붕괴 기간의 초반에 집중적으로 감소합니다. 붕괴 패턴은 오목한 곡선 형태로 나타납니다. 소스 시스템의 그래프 유형이 이 유형인 경우 해당 시스템에서 가져오는 새 값은 트러스트되기는 하지만 조만간 재정의될 가능성이 매우 높습니다.
	SIRL(초기에 느리다 나중에는 빨라짐)	트러스트가 붕괴 기간의 후반에 집중적으로 감소합니다. 붕괴 패턴은 볼록한 곡선 형태로 나타납니다. 소스 시스템의 그래프 유형이 이 유형인 경우 소스 시스템에서 설정한 값은 붕괴 기간의 거의 끝에 도달하기 전까지는 다른 시스템에 의해 재정의될 가능성이 비교적 낮습니다.

테스트 오프셋 날짜

기본적으로 트러스트 붕괴 그래프에 표시되는 트러스트 붕괴 시작 날짜는 현재 시스템 날짜입니다. 지정된 소스 시스템에 대해 시작 날짜를 달리 하여 트러스트 붕괴 영향을 확인하려면 다른 테스트 오프셋 날짜를 지정합니다.

트러스트 값 설정 시 고려 사항

올바른 트러스트 값을 선택하는 과정은 복잡할 수 있습니다.

각 시스템을 개별적으로 고려하는 것으로는 충분하지 않고, 특정 열에 데이터를 제공하는 모든 소스 시스템에 대한 트러스트 설정 조합으로 원하는 동작을 얻을 수 있는지 확인해야 합니다. 소스 시스템의 트러스트 수준은 절대적이지 않습니다. 즉, 트러스트가 활성화된 열에 데이터를 제공하는 다른 소스 시스템의 트러스트 수준과 관련해서만 의미가 있습니다.

트러스트를 결정할 때는 다음 사항을 고려해야 합니다.

- 소스 시스템에서 이 데이터 값의 유효성이 검사되는지 여부와 이러한 유효성 검사의 신뢰도
- 소스 시스템의 사용자 입장에서 다른 데이터 값과 비교할 때 이 데이터 값이 갖는 중요도. 대개 사용자는 자신의 작업에 핵심적인 데이터의 유효성을 검사하는 데 가장 큰 노력을 기울입니다.
- 소스 시스템이 업데이트되는 빈도
- 특정 특성이 업데이트될 수 있는 빈도

열 트러스트

기본 개체 열에 대해 트러스트를 활성화 및 구성합니다. 연산 참조 저장소의 다른 테이블에 대해서는 트러스트를 활성화할 수 없습니다.

Informatica MDM Hub는 기본적으로 열 트러스트를 비활성화합니다. 트러스트가 비활성화되면 Informatica MDM Hub는 데이터를 가져오는 소스 시스템에 관계없이 가장 최근에 실행된 로드 프로세스의 데이터를 사용합니다. 기본 개체 열 데이터를 한 시스템에서 가져오는 경우 해당 열에 대해 트러스트가 비활성화된 상태를 유지합니다.

데이터를 여러 소스 시스템에서 가져올 수 있는 열의 경우에는 트러스트를 활성화해야 합니다. 열에 대해 트러스트를 활성화하는 경우 열에 데이터를 제공할 수 있는 특정 소스 시스템의 상대적 신뢰도를 지정하기 위한 트러스트 수준을 할당합니다.

트러스트 열 및 유효성 검사 규칙을 추가하면 일괄 로드 및 일괄 병합 성능이 저하됩니다. 또한 트러스트 열 및 유효성 검사 규칙을 추가하면 제어 테이블 업데이트 문의 길이가 늘어납니다. 40개가 넘는 트러스트 열을 활성화하면 Informatica MDM Hub가 한 번에 최대 40개의 열 청크로 제어 테이블을 업데이트합니다. 열의 가장 최근에 로드된 데이터를 BVT(최선의 진실)로 고려할 때 열에 대해 트러스트를 활성화하지 마십시오.

지나치게 많은 열에 대해 트러스트 및 유효성 검사를 활성화하지 마십시오. Microsoft SQL Server에서 페이지 크기 제한은 8KB입니다. 유니코드 문자를 지원하기 위해 Informatica MDM Hub에서는 NCHAR 및 NVARCHAR 데이터 유형을 사용합니다. 더블바이트 지원으로 인해 최대 레코드 크기는 4000자입니다. 레코드 크기가 4000자를 초과하면 일괄 처리가 예기치 않게 실패할 수 있습니다.

열의 트러스트 활성화

기본 개체 열의 트러스트를 활성화하려면 기본 개체 열 속성을 편집합니다.

1. **모델** 작업 영역의 Hub 콘솔에서 **스키마**를 선택합니다.
2. 쓰기 잠금을 획득합니다.
3. 스키마 관리자의 왼쪽 창에서 트러스트를 적용할 열이 있는 기본 개체를 확장합니다. **열**을 선택합니다.
4. 열의 **트러스트** 확인란을 활성화합니다. **저장** 단추를 클릭합니다.

트러스트가 활성화된 열의 트러스트 구성 전 작업

트러스트가 활성화된 열의 트러스트를 구성하기 전에 다음을 수행해야 합니다.

- 기본 개체 열의 트러스트 활성화
- 스키마 관리자에서 준비 테이블 구성(연결된 소스 시스템과 기본 개체 열에 해당하는 준비 테이블 열 포함)

관리 소스 시스템의 트러스트 지정

최소한 관리 소스 시스템(기본 이름: *Admin*)에서 트러스트가 활성화된 열에 대해 트러스트 설정을 지정해야 합니다. 이 소스 시스템은 Informatica MDM Hub에서 사용자가 수행하는 수동 업데이트를 나타내며, 트러스트가 활성화된 모든 열에 데이터를 제공할 수 있습니다. 이 소스 시스템에 대한 트러스트 설정을 다른 소스 시스템에 비해 상대적으로 높은 값으로 설정하면 수동 업데이트 시 다른 소스 시스템의 기존 값을 재정의할 수 있습니다.

기본 개체의 트러스트가 활성화된 열에 트러스트 수준 할당

기본 개체의 트러스트가 활성화된 열에 트러스트 수준을 할당하려면

1. 시스템 및 트러스트 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 트러스트 노드를 확장합니다.
시스템 및 트러스트 도구에 트러스트가 활성화된 열이 있는 모든 기본 개체가 표시됩니다.
4. 기본 개체를 선택합니다.
시스템 및 트러스트 도구에 선택한 기본 개체의 트러스트가 활성화된 열이 읽기 전용 보기로 표시됩니다. 해당 열에 특정 소스 시스템이 데이터를 제공하는지 여부를 나타내는 확인 표시도 함께 표시됩니다.
참고: 트러스트가 활성화된 열과 소스 시스템 간의 연결은 기본 개체에 대한 준비 테이블에서 지정됩니다.
5. 기본 개체를 확장하여 트러스트가 활성화된 열을 표시합니다.
6. 트러스트가 활성화된 열 중 구성하려는 열을 선택합니다.
트러스트가 활성화된 열의 경우 시스템 및 트러스트 도구에 선택한 열과 연결된 소스 시스템의 목록이 표시되고, 각 소스 시스템에 대해 구성할 편집 가능한 트러스트 설정과 트러스트 봉괴 그래프도 표시됩니다.

7. 각 열에 대한 트러스트 속성을 지정합니다.
8. 필요한 경우 오프셋 날짜를 변경할 수 있습니다.
9. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

이 트러스트가 활성화된 열의 각 소스 시스템에 대해 지정한 트러스트 설정에 따라 트러스트 붕괴 그래프가 새로 고쳐집니다.

X 축에는 트러스트 점수가 표시되고 Y 축에는 시간이 표시됩니다.

트러스트가 활성화된 열의 오프셋 날짜 변경

기본적으로 트러스트 붕괴 그래프에서는 현재 시스템 날짜 이후의 모든 소스 시스템에 대한 트러스트 붕괴를 보여 줍니다. 미래 날짜 등의 다른 날짜를 지정하여 현재 트러스트 설정을 테스트하고 해당 날짜 이후의 트러스트 붕괴 방식을 확인할 수 있습니다. 오프셋 날짜는 저장되지 않습니다.

트러스트가 활성화된 열의 오프셋 날짜를 변경하려면

1. 시스템 및 트러스트 도구에서 트러스트가 활성화된 열을 선택합니다.
2. 다른 오프셋 날짜를 지정할 소스 시스템 옆의 **달력** 단추를 클릭합니다.
날짜를 지정하라는 메시지가 표시됩니다.
3. 다른 날짜를 선택합니다.
4. **확인**을 선택합니다.

현재 트러스트 설정과 지정한 오프셋 날짜를 기준으로 트러스트 붕괴 그래프가 업데이트됩니다.

오프셋 날짜를 제거하려면

- 오프셋 날짜를 제거할 소스 시스템 옆의 **삭제** 단추를 클릭합니다.
현재 트러스트 설정과 현재 시스템 날짜를 기준으로 트러스트 붕괴 그래프가 업데이트됩니다.

트러스트 설정 변경 후 동기화 일괄 작업 실행

레코드를 기본 개체에 로드한 후 특정 열에 대해 트러스트를 활성화한 경우 또는 해당 기본 개체에서 트러스트가 활성화된 모든 열에 대해 트러스트 설정을 변경한 경우 통합 프로세스를 실행하기 전에 동기화 일괄 작업을 실행해야 합니다. 이 일괄 작업을 실행하지 않으면 통합 프로세스를 수행하는 동안 오류가 발생합니다.

유효성 검사 규칙 구성

이 섹션에서는 Informatica MDM Hub 구현에서 유효성 검사 규칙을 구성하는 방법에 대해 설명합니다.

유효성 검사 규칙 정보

유효성 검사 규칙은 셀 값이 특정 조건과 일치할 때 셀 값의 트러스트를 다운그레이드합니다.

참고: MDM Hub에서는 IBM DB2 또는 Microsoft SQL Server에 대해 사용자 지정 유효성 검사 규칙을 지원하지 않습니다.

각 유효성 검사 규칙은 다음 사항을 지정합니다.

- 셀 값이 유효한지 여부를 확인하는 조건
- 조건이 충족될 경우 수행할 작업(일정 백분율만큼 트러스트 다운그레이드)

예를 들어 다음과 같은 유효성 검사 규칙이 있다고 가정합니다.

Downgrade trust on First_Name by 50% if Length < 3'

이 규칙을 구성하는 요소는 다음과 같습니다.

조건

Length < 3

작업

First_Name의 트러스트를 50% 다운그레이드

열에 대해 최소 트러스트 보유 플래그가 설정되어 있는 경우에는 트러스트가 열의 최소 트러스트 설정 아래로 다운그레이드될 수 없습니다. 기본 개체에 대한 유효성 검사 규칙을 구성하려면 스키마 관리자를 사용합니다.

유효성 검사 규칙은 로드 프로세스 도중 기본 개체의 트러스트가 활성화된 열에 대해 트러스트가 계산된 이후에 실행됩니다. 유효성 검사 규칙이 정의된 경우 로드 프로세스에서는 해당 규칙을 적용하여 최종 트러스트 점수를 확인하고, 최종 트러스트 값을 사용하여 기본 개체의 레코드를 업데이트된 레코드의 셀 데이터로 업데이트할지 여부를 결정합니다.

유효성 검사 확인

유효성 검사는 기본 개체의 모든 열에 대해 수행할 수 있습니다. 유효성 검사를 수행한 결과, 동일한 열과 유효성을 검사할 수 있는 다른 모든 열에 다운그레이드가 적용될 수 있습니다. 따라서 한 열에 잘못된 데이터가 있으면 여러 열의 트러스트가 다운그레이드될 수 있습니다.

예를 들어 주소가 완전할 경우 OK 플래그가 지정되고 주소가 완전하지 않을 경우 BAD 플래그가 지정되는 주소 확인 플래그를 사용한 경우, 확인 플래그가 OK가 아니면 모든 주소 필드의 트러스트를 다운그레이드하는 유효성 검사 규칙을 구성할 수 있습니다. 이 경우 확인 플래그도 다운그레이드되어야 합니다.

필요한 열

유효성 검사 규칙은 수신 데이터 소스에 상관없이 적용됩니다. 단, 준비 테이블이나 입력(SIF(서비스 통합 프레임 워크) 요청)에 필요한 열이 모두 포함된 경우에만 적용됩니다. 필요한 열이 누락된 경우에는 유효성 검사 규칙이 적용되지 않습니다.

유효성 검사 규칙 변경 후 트러스트 점수 다시 계산

기본 개체에 기존 데이터가 포함되어 있는 경우 유효성 검사 규칙을 변경하려면 유효성 다시 검사 작업을 실행하여 새 데이터와 기존 데이터의 트러스트 점수를 다시 계산해야 합니다.

유효성 검사 규칙 및 상태 사용 기본 개체

상태 사용 기본 개체의 경우 유효성 검사 규칙은 ACTIVE, PENDING 및 DELETED 상태의 레코드에 적용됩니다. BVT를 계산할 때는 DELETED 상태의 레코드가 ACTIVE 및 PENDING 상태의 레코드보다 더 높은 트러스트 점수를 갖지 않도록 DELETE 상태인 레코드의 경우 트러스트가 추가로 다운그레이드됩니다.

자동 병합 작업 시 유효성 검사 열 수에 대한 제약 조건

유효성 검사 사용 열이 다수인 경우에는 자동 병합 작업이 실패할 수 있습니다.

작업이 실패하게 되는 정확한 열 수는 변수이며, 열 이름의 길이와 유효성 검사 사용 열의 개수에 따라 달라집니다. 허용 가능한 최대 길이인 26자에 근접한 열 이름은 긴 이름으로 간주됩니다. 이 문제를 방지하려면 유효성 검사 사용 열의 개수를 60개 미만으로 유지하거나 열 이름의 길이를 짧게 하십시오. 해결 방법은 기본 개체를 저장하기 전에 모든 트러스트/유효성 검사 열을 활성화하여 동기화 작업이 실행되지 않도록 하는 것입니다.

열의 유효성 검사 규칙 활성화

유효성 검사 규칙은 스키마 관리자에서 기본 개체의 열별로 활성화되고 구성됩니다.

유효성 검사 규칙은 연산 참조 저장소의 다른 모든 테이블의 열에는 적용되지 않습니다.

유효성 검사 규칙은 기본적으로 비활성화됩니다. 그러나 트러스트 다운그레이드를 위해 유효성 검사 규칙을 사용할 트러스트가 활성화된 열의 경우 유효성 검사 규칙을 활성화해야 합니다.

기본 개체를 업데이트할 때 트러스트가 활성화된 열의 값을 제공하지 않으면 트러스트가 활성화된 열을 사용하는 유효성 검사 규칙이 연결된 교차 참조 레코드에 적용되지 않습니다.

다운그레이드 백분율이 적용되는 방식

유효성 검사 규칙에서 다음 알고리즘에 따라 트러스트 점수를 다운그레이드합니다.

$$\text{Final trust} = \text{Trust} - (\text{Trust} * \text{Validation_Downgrade} / 100)$$

예를 들어 유효성 검사 다운그레이드 백분율이 50%이고 계산된 트러스트 수준이 60이라고 가정합니다.

$$\text{Final Trust Score} = 60 - (60 * 50 / 100)$$

최종 트러스트 점수는 다음과 같습니다.

$$\text{Final Trust Score} = 60 - 30 = 30$$

유효성 검사 규칙 시퀀스

MDM Hub는 사용자가 지정한 유효성 검사 규칙 시퀀스대로 유효성 검사 규칙을 실행합니다.

한 열에 대해 여러 유효성 검사 규칙을 구성한 경우 MDM Hub는 사용자가 구성한 시퀀스대로 유효성 검사 규칙을 실행합니다. 또한 MDM Hub는 각 유효성 검사 규칙의 다운그레이드 백분율을 고려합니다. 다운그레이드 백분율은 누적되지 않습니다. 다운그레이드 백분율이 가장 높은 유효성 검사 규칙은 다른 변경 내용을 덮어씁니다.

여러 유효성 검사 규칙의 다운그레이드 백분율이 동일하게 가장 높은 경우 MDM Hub는 최소 트러스트 보유 설정을 확인합니다. 최소 트러스트 보유 설정이 모든 유효성 검사 규칙에 대해 활성화된 경우 MDM Hub는 다운그레이드 백분율이 가장 높은 첫 번째 유효성 검사 규칙을 적용합니다.

유효성 검사 규칙 노드로 이동

유효성 검사 규칙을 구성하려면 스키마 관리자에서 기본 개체의 유효성 검사 규칙 노드로 이동합니다.

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 구성할 기본 개체 트리를 확장한 후 해당 **유효성 검사 규칙 설정** 노드를 클릭합니다.

스키마 관리자에 스키마 관리자에 유효성 검사 규칙 편집기가 표시됩니다.

유효성 검사 규칙 편집기는 다음과 같은 섹션으로 구분됩니다.

창	설명
규칙 수	선택한 기본 개체에 대해 구성된 유효성 검사 규칙 수입입니다.
유효성 검사 규칙	선택한 기본 개체에 대해 구성된 유효성 검사 규칙 목록입니다.
속성 창	선택한 유효성 검사 규칙의 속성입니다.

유효성 검사 규칙 속성

유효성 검사 규칙에는 다음과 같은 속성이 있습니다.

규칙 이름

이 유효성 검사 규칙에 대한 고유한 설명 이름입니다.

참고: 규칙 이름에 쉼표를 삽입하지 마십시오. 쉼표를 사용할 경우 유효성 검사 규칙의 순서에 영향을 미칠 수 있습니다.

규칙 유형

유효성 검사 규칙 유형입니다. 규칙 유형의 값은 다음 중 하나일 수 있습니다.

규칙 유형	설명
존재 확인	셀에 null 값이 포함된 경우(즉, 셀 값이 없음) 트러스트가 다운그레이드됩니다.
도메인 확인	셀 값이 허용된 값 범위나 목록에 포함되지 않는 경우 트러스트가 다운그레이드됩니다.
참조 무결성	셀 값이 다른 테이블에서 열의 값 집합에 없는 경우 트러스트가 다운그레이드됩니다. 이 규칙은 명시적인 외래 키가 정의되지 않은 경우 사용되며, 보다 높은 트러스트를 포함한 정확한 셀 값이 없을 경우 부정확한 셀 값이 허용될 수 있습니다.
패턴 유효성 검사	셀 값이 지정된 패턴과 일치(LIKE)하거나 일치하지 않을(NOT LIKE) 경우 트러스트가 다운그레이드됩니다.
사용자 지정	복잡한 유효성 검사 규칙 입력 시 사용자 지정 규칙을 사용합니다. SQL 함수(예: LENGTH 및 ABS)가 필요한 경우 또는 정적 테이블에 대한 복잡한 조인이 필요한 경우에만 사용자 지정 규칙을 사용합니다. 사용자 지정 SQL 코드는 데이터베이스 플랫폼의 SQL 구문을 준수해야 합니다. 입력한 SQL은 디자인 시 유효성이 검사되지 않습니다. 로드 프로세스 실행 시 올바르게 않은 SQL 구문 오류는 문제를 발생시킵니다.

규칙 열

각 열에 대해 다운그레이드 백분율과 최소 트러스트를 보유할지 여부를 지정합니다.

다운그레이드 백분율

이 유효성 검사 규칙 조건이 충족된 경우 지정된 열의 트러스트 수준이 줄어드는 백분율입니다. 백분율 값이 클수록 다운그레이드되는 정도가 많아집니다. 예를 들어 0%는 트러스트에 아무 영향을 주지 않으며 100%는 트러스트를 완전히 다운그레이드합니다(최소 트러스트 보유가 지정되지 않은 경우에는 최소 트러스트와 같도록 100%가 트러스트를 다운그레이드함).

트러스트가 100%만큼 다운그레이드되고 열에 대해 최소 트러스트 보유를 설정하지 않은 경우에는 해당 열 값이 기본 개체에 채워지지 않습니다.

최소 트러스트 보유

다운그레이드로 인해 트러스트 수준이 열의 최소 트러스트 수준 이하로 떨어질 경우 수행되는 동작을 지정합니다. 최소 트러스트를 보유할 수 있습니다(트러스트 수준이 최소 트러스트로 줄어들며, 최소 트러스트 아래로 떨어지지 않음). 이 확인란을 선택 취소하면 지정된 백분율로 인해 트러스트 수준이 최소 트러스트보다 낮아지더라도 트러스트 수준이 지정된 백분율만큼 줄어듭니다.

규칙 SQL

유효성 검사 규칙에 대한 조건을 **SQL WHERE** 절로 지정합니다. 로드 프로세스가 유효성 검사 규칙을 실행합니다. 레코드가 규칙 **SQL** 필드에 지정된 기준을 충족하면 트러스트 값이 이 유효성 검사 규칙에 대해 구성된 다운그레이드 백분율만큼 다운그레이드됩니다.

유효성 검사 규칙 편집기에 유효성 검사 규칙에 대해 선택한 규칙 유형을 기준으로 **SQL WHERE** 절을 구성하는 메시지가 표시됩니다. 로드 프로세스 동안 준비 테이블의 데이터 유효성을 확인하는 데 이 쿼리가 사용됩니다.

다음 테이블에는 규칙 유형이 나열되고 각 규칙 유형에 대한 **SQL WHERE** 절의 예제가 포함되어 있습니다.

규칙 유형	WHERE 절	예	결과
존재 확인	WHERE S.ColumnName IS NULL	WHERE S.MIDDLE_NAME IS NULL	중간 이름이 null인 레코드에 대해 영향을 받는 열이 다운그레이드됩니다. 조건을 충족하지 않는 레코드는 영향을 받지 않습니다.
도메인 확인	WHERE S.ColumnName IN ('?', '?', '?')	WHERE S.Gender NOT IN ('M', 'F', 'U')	성별이 M, F 또는 U 이외의 값인 경우 영향을 받는 열이 다운그레이드됩니다.
참조 무결성	WHERE NOT EXISTS (SELECT <blank>'a' FROM ? WHERE ?? = S.<Column_Name> WHERE NOT EXISTS (SELECT <blank>'a' FROM <Ref_Table> WHERE <Ref_Table>.<Ref_Column> = S.<Column_Name>	WHERE NOT EXISTS (SELECT DISTINCT 'a' FROM ACCOUNT_TYPE WHERE ACCOUNT_TYPE.Account_Type = S.Account_Type	계정 유형 값이 계정 유형 테이블에 없는 레코드의 경우 영향을 받는 열이 다운그레이드됩니다.
패턴 유효성 검사	WHERE S.ColumnName LIKE 'Pattern'	WHERE S.eMail_Address NOT LIKE '%@%'	전자 메일 주소에 @ 문자가 포함되지 않은 경우 다운그레이드가 적용됩니다.
사용자 지정	WHERE	WHERE LENGTH(S.ZIP_CODE) > 4	우편 번호 열 길이가 4보다 작을 경우 다운그레이드가 적용됩니다.

테이블 별칭 및 와일드카드

와일드카드 문자(*)를 사용하여 별칭으로 테이블을 참조할 수 있습니다.

- s.*는 준비 테이블의 별칭입니다.
- i.*는 임시 테이블의 별칭으로, 업데이트되는 레코드에 대한 ROWID_OBJECT, PKEY_SRC_OBJECT 및 ROWID_SYSTEM 정보를 제공합니다. i 별칭은 유효성 검사 규칙의 ROWID_OBJECT, PKEY_SRC_OBJECT 및 ROWID_SYSTEM 열에만 사용할 수 있습니다.

사용자 지정 규칙 유형 및 SQL WHERE 구문

사용자 지정 유효성 검사 규칙의 경우 올바른 형식의 잘 조정된 SQL 문을 작성합니다. SQL 문은 들어오는 데이터에 대한 WHERE 조건의 일부로 실행됩니다.

참고: 사용자 지정 유효성 검사 규칙을 사용하면 로드 프로세스의 성능이 저하됩니다. 필요한 경우에만 사용하십시오.

보유한 데이터베이스 플랫폼에 필요한 SQL 문을 사용합니다. SQL WHERE 절 구문 및 와일드카드 패턴에 대한 자세한 내용은 Informatica MDM Hub 구현 환경에서 사용되는 데이터베이스 플랫폼의 제품 설명서를 참조하십시오.

우선 순위를 지정할 때는 괄호를 사용해야 합니다. 괄호를 잘못 지정하거나 누락하면 예상치 못한 결과가 나타나고 쿼리 실행이 오래 걸릴 수 있습니다. 예를 들어 다음 문은 모호하게 작성되어 데이터베이스 서버가 우선 순위를 결정해야 하는 상황이 됩니다.

```
WHERE conditionA AND conditionB OR conditionC
```

다음 문은 괄호를 사용하여 우선 순위를 명확하게 지정합니다.

```
WHERE (conditionA AND conditionB) OR conditionC  
WHERE conditionA AND (conditionB OR conditionC)
```

이 두 가지 문은 레코드를 평가할 때 상당히 다른 결과를 생성합니다.

테이블 간 조인을 사용하는 사용자 지정 유효성 검사 규칙 예제

두 테이블의 데이터를 조인하는 사용자 지정 유효성 검사 규칙에 대한 조건을 작성할 수 있습니다. 한 테이블에는 들어오는 데이터의 상위 레코드가 포함됩니다. 다른 테이블은 값의 정적 목록이 포함된 조회 테이블입니다.

참고: 상위 레코드의 값을 조건에 사용하십시오. 하위 레코드의 값이 필요한 조건을 작성할 경우 규칙이 결과를 반환하지 않습니다. 사용자 지정 유효성 검사 규칙에서는 상위 테이블에서 하위 테이블로 조인할 수 없습니다.

들어오는 데이터는 별칭 S라는 테이블로 참조합니다. 조인된 테이블은 별칭 I로 참조합니다.

다음 코드는 Put(BO - C_AAA_BO)으로 표현된 조인에 대한 유효성 검사 규칙 SQL 문을 보여 줍니다.

```
WHERE NOT EXISTS (SELECT 1 FROM C_B_LU_ADDR_TP T WHERE T.ADDR_TP = S.COL1)
```

다음 코드는 생성된 SQL 문을 보여 줍니다.

```
SELECT S.PKEY_SRC_OBJECT ,  
       (SELECT 'a' FROM dual WHERE NOT EXISTS (SELECT 1 FROM C_B_LU_ADDR_TP T WHERE T.ADDR_TP = S.COL1 )  
        AND ROWNUM <= 1 ) RULE1  
FROM  
       (SELECT NULL AS ROWID_OBJECT ,  
              'SVR1.3GDX' AS PKEY_SRC_OBJECT ,  
              'SYS0' AS ROWID_SYSTEM  
        FROM dual) I  
CROSS JOIN  
       (SELECT '1' AS COL2 ,  
              '1' AS COL1 ,  
              'SVR1.3GDX' AS PKEY_SRC_OBJECT ,  
              'SYS0' AS ROWID_SYSTEM  
        FROM dual) S
```

유효성 검사 규칙 추가

유효성 검사 규칙을 추가하려면

1. 유효성 검사 규칙 편집기로 이동합니다.
2. 유효성 검사 규칙 추가 단추를 클릭합니다.

스키마 관리자에 유효성 검사 규칙 추가 대화 상자가 표시됩니다.

- 유효성 검사 규칙의 속성을 지정합니다.
- 필요한 경우 **편집** 단추를 클릭하여 유효성 검사 규칙에 사용할 규칙 열을 선택합니다.
유효성 검사 규칙 편집기에 규칙 열 선택 대화 상자가 표시됩니다.
사용 가능한 열은 유효성 검사 플래그가 활성화된 열입니다.
이 유효성 검사 규칙의 **WHERE** 절에 지정된 조건이 충족될 경우 트러스트 수준을 다운그레이드할 열을 선택한 다음 **확인**을 클릭합니다.
참고: 유효성 검사 규칙에 날짜를 사용해야 하는 경우에는 **to_date** 함수를 사용하고, 실제 날짜 형식을 지정하거나 날짜가 데이터베이스에 필요한 형식으로 지정되도록 합니다.
- 확인**을 클릭합니다.
유효성 검사 규칙 목록에 새 규칙이 추가됩니다.
참고: 기본 개체에 기존 데이터가 포함되어 있는 경우 유효성 검사 규칙을 변경하려면 유효성 다시 검사 작업을 실행하여 새 데이터와 기존 데이터의 트러스트 점수를 다시 계산해야 합니다.

유효성 검사 규칙 속성 편집



유효성 검사 규칙을 편집하려면

- 스키마 관리자에서 유효성 검사 규칙 편집기로 이동합니다.
- 탐색 규칙 목록에서 구성할 탐색 규칙을 선택합니다.
유효성 검사 규칙 편집기에 선택한 유효성 검사 규칙에 대한 속성이 표시됩니다.
- 이 유효성 검사 규칙의 편집 가능한 속성을 지정합니다. 규칙 유형은 변경할 수 없습니다.
- 필요한 경우 **편집** 단추를 클릭하여 이 유효성 검사 규칙에 대한 규칙 열을 선택합니다.
유효성 검사 규칙 편집기에 규칙 열 선택 대화 상자가 표시됩니다.
사용 가능한 열은 유효성 검사 플래그가 활성화된 열입니다.
이 유효성 검사 규칙의 **WHERE** 절에 지정된 조건이 충족될 경우 트러스트 수준을 다운그레이드할 열을 선택한 다음 **확인**을 클릭합니다.
- 저장** 단추를 클릭하여 변경 내용을 저장합니다.
참고: 기본 개체에 기존 데이터가 포함되어 있는 경우 유효성 검사 규칙을 변경하려면 유효성 다시 검사 작업을 실행하여 새 데이터와 기존 데이터의 트러스트 점수를 다시 계산해야 합니다.

유효성 검사 규칙 순서 변경

유효성 검사 규칙의 실행 순서는 매우 중요합니다.

다음 단추를 사용하여 목록에 표시된 유효성 검사 규칙의 순서를 변경할 수 있습니다.

클릭	작업
	선택한 유효성 검사 규칙을 더 높은 순서로 이동합니다.
	선택한 유효성 검사 규칙을 더 낮은 순서로 이동합니다.

유효성 검사 규칙 제거

유효성 검사 규칙을 제거하려면

1. 스키마 관리자에서 유효성 검사 규칙 편집기로 이동합니다.
2. 유효성 검사 규칙 목록에서 제거할 유효성 검사 규칙을 선택합니다.
3. **삭제** 단추를 클릭합니다.

스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.

4. **예**를 클릭합니다.

참고: 기본 개체에 기존 데이터가 포함되어 있는 경우 유효성 검사 규칙을 변경하려면 유효성 다시 검사 작업을 실행하여 새 데이터와 기존 데이터의 트러스트 점수를 다시 계산해야 합니다.

제 21 장

일치 프로세스 구성

이 장에 포함된 항목:

- [시작하기 전에, 373](#)
- [일치 프로세스에 대한 태스크 구성, 373](#)
- [일치/병합 설정 세부 정보 대화 상자로 이동, 375](#)
- [기본 개체에 대한 일치 속성 구성, 376](#)
- [관련 레코드의 일치 경로 구성, 380](#)
- [경로 구성 요소 구성, 387](#)
- [일치 열 구성, 389](#)
- [일치 규칙 집합에 대한 일치 열 규칙 구성, 402](#)
- [기본 키 일치 규칙 구성, 423](#)
- [일치 키의 분포 조사, 425](#)
- [일치 프로세스에서 레코드 제외, 427](#)
- [근접 검색, 427](#)
- [경량 일치, 429](#)

시작하기 전에

시작하기 전에 Informatica MDM Hub를 설치하고 *Multidomain MDM 설치 가이드*의 지침을 사용하여 Hub 저장소를 생성하고, 스키마를 작성해야 합니다.

일치 프로세스에 대한 태스크 구성

이 섹션에서는 일치 프로세스와 관련된 구성 태스크에 대해 간략하게 설명합니다.

데이터 이해

일치 규칙을 정의하려면 먼저 데이터에 대해 잘 알고 있고 다음을 이해하고 있어야 합니다.

- 중복 레코드를 확인하는 데 사용하려는 열의 값 분포
- 중복된 총 레코드 수의 전체 비율

일치 프로세스와 연결된 기본 개체 속성

다음 기본 개체 속성은 일치 프로세스의 동작에 영향을 줍니다.

속성	설명
중복 일치 임계값	초기 데이터 로드 중 "중복 데이터에 대한 일치" 작업에만 사용됩니다.
최대 일치 경과 시간(분)	일치 규칙을 실행할 때 적용되는 제한 시간(분)입니다. 이 시간이 초과되면 일치 프로세스가 종료됩니다.
일치 플래그 감사 테이블	이 속성을 활성화하면 감사 테이블(<i>BusinessObjectName_FMHA</i>)이 생성되고 병합 관리자에서 자동 병합을 위해 수동 일치 레코드를 대기열에 넣은 사용자의 사용자 ID로 채워집니다.

일치 규칙을 정의하기 위한 구성 단계

일치 규칙을 정의하려면

1. 기본 개체의 일치 속성을 구성합니다.
2. 일치 열을 정의합니다.
3. 일치 규칙의 일치 규칙 집합을 정의합니다.
4. 규칙 집합의 일치 규칙을 정의합니다.
5. 일치 규칙을 모두 생성할 때까지 3~4단계를 반복합니다.
6. 데이터에 대한 정보를 기반으로 기본 키 기준의 일치가 필요한지 여부를 결정합니다.
7. 데이터가 기본 키 일치에 적절한 경우 기본 키 일치 규칙을 생성합니다.
8. 규칙을 조정합니다. 이 프로세스는 반복적 프로세스로, 이를 통해 대표 데이터 집합에 일치 규칙을 적용하고, 결과를 분석하고, 일치 성능을 최적화하도록 설정을 조정할 수 있습니다.

다국어 데이터가 포함된 기본 개체 구성

Informatica MDM Hub에서는 미국 외 인구집단의 데이터가 포함된 기본 개체에 대한 일치뿐만 아니라 미국 및 중국과 같이 서로 다른 인구집단의 데이터가 포함된 기본 개체에 대한 일치도 지원합니다.

분산 일치 구성

연산 참조 저장소에 대해 여러 처리 서버를 구성하는 경우 분산 일치를 구성할 수 있습니다. 여러 처리 서버를 병렬로 실행하여 일치 프로세스의 처리량을 늘릴 수 있습니다.

이를 위해서는 `cmxcleanse.properties` 파일에서 분산 일치를 구성해야 합니다. 분산 일치를 활성화하려면 `cmx.server.match.distributed_match` 속성을 1로 설정해야 합니다. 기본적으로 이 속성은 비활성화되어 있습니다.

데이터 로드 구성

토큰화 및 일치 프로세스에 대해 데이터를 데이터베이스로 직접 로드하거나 중간 파일을 사용하도록 데이터 로드 프로세스를 구성할 수 있습니다. `cmxcleanse.properties` 파일에서 속성을 구성하여 데이터 로드 방법 및 일괄 처리 크기를 지정할 수 있습니다. 기본값은 직접 로드입니다.

기본 동작을 변경하려면 `cmxcleanse.properties` 파일에 데이터 로드 속성을 추가합니다. `cmxcleanse.properties` 파일은 다음 디렉터리에 있습니다.

Windows의 경우. <MDM Hub 설치 디렉터리>\hub\cleanse\resources

UNIX의 경우, <MDM Hub 설치 디렉터리>/hub/cleanse/resources

다음 테이블에는 토큰화 및 일치에 대한 데이터 로드 속성이 설명되어 있습니다.

속성	설명
cmx.server.tokenize.file_load	토큰화에 대해 데이터를 데이터베이스로 로드하는 데 중간 파일을 사용할지 여부를 지정합니다. 중간 파일을 사용하여 데이터를 로드하려면 true로 설정합니다. 데이터를 직접 로드하려면 false로 설정합니다. Oracle 및 IBM DB2 환경의 경우 기본값은 true입니다. Microsoft SQL Server 환경의 경우 기본값은 false입니다. 참고: 데이터 로드를 위한 중간 파일 사용은 Microsoft SQL Server에 적용되지 않습니다.
cmx.server.tokenize.loader_batch_size	직접 로드 중에 데이터베이스로 보낼 수 있는 최대 삽입 문 수입니다. 기본값은 1000입니다.
cmx.server.match.file_load	일치에 대해 데이터를 데이터베이스로 로드하는 데 중간 파일을 사용할지 여부를 지정합니다. 중간 파일을 사용하여 데이터를 로드하려면 true로 설정합니다. 데이터를 직접 로드하려면 false로 설정합니다. Oracle 및 IBM DB2 환경의 경우 기본값은 true입니다. 외부 일치에 대해 구성된 Microsoft SQL Server 환경 및 IBM DB2 환경의 경우 기본값은 false입니다. 참고: 데이터 로드를 위한 중간 파일 사용은 Microsoft SQL Server에 적용되지 않습니다.
cmx.server.match.loader_batch_size	직접 로드 중에 데이터베이스로 보낼 수 있는 최대 삽입 문 수입니다. 기본값은 1000입니다.

일치/병합 설정 세부 정보 대화 상자로 이동

기본 개체에 대해 일치 및 병합 프로세스를 설정하려면 다음 단계를 수행하여 시작합니다.

1. 스키마 관리자를 시작합니다.
2. 스키마 탐색 트리에서 일치 속성을 정의할 기본 개체를 확장합니다.
3. 스키마 탐색 트리에서 **일치/병합 설정**을 선택합니다.

스키마 관리자에 일치/병합 설정 세부 정보 대화 상자가 표시됩니다.

설정을 변경하려면 쓰기 잠금을 획득해야 합니다.

일치/병합 설정 세부 정보 대화 상자에는 다음과 같은 탭이 포함됩니다.

탭 이름	설명
속성	일치/병합 설정을 요약하고 구성 가능한 여러 일치/병합 설정을 제공합니다.
경로	다른 기본 개체나 동일한 기본 개체에서 레코드에 대한 상위/하위 관계에 사용하도록 일치 경로를 구성할 수 있습니다.
일치 열	일치 열 규칙에 대해 일치 열을 구성할 수 있습니다.

탭 이름	설명
일치 규칙 집합	일치 규칙 집합을 사용하여 검색 전략 및 규칙을 정의할 수 있습니다.
기본 키 일치 규칙	기본 키 일치 규칙을 정의할 수 있습니다.
일치 키 분산	일치 키의 분산을 표시합니다.
병합 설정	설정을 병합 및 연결할 수 있습니다.

기본 개체에 대한 일치 속성 구성

일치 열 및 일치 규칙 등의 다른 일치 기능을 구성하려면 먼저 기본 개체에 대한 일치 속성을 설정해야 합니다. 이러한 일치 속성은 기본 개체에 대한 모든 규칙에 적용됩니다.

일치 속성 설정

각 기본 개체의 일치 속성을 구성합니다. 이 설정은 해당하는 모든 일치 규칙 및 규칙 집합에 적용됩니다.

기본 개체에 대해 일치 속성을 구성하려면

1. 스키마 관리자에는 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 창이 표시됩니다.
2. 일치/병합 설정 세부 정보 창에서 **속성** 탭을 클릭합니다.
스키마 관리자에 속성 탭이 표시됩니다.
3. 쓰기 잠금을 획득합니다.
4. 해당되는 경우 필드 옆의 **편집** 단추를 클릭하여 변경할 속성 설정을 편집합니다.
5. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

일치 속성

이 섹션에서는 [속성] 탭의 구성 설정에 대해 설명합니다.

계산된 읽기 전용 필드

속성 탭에는 다음과 같은 읽기 전용 필드가 표시됩니다.

테이블 1. 읽기 전용 일치 속성

속성	설명
일치 열	해당 기본 개체에 대해 구성된 일치 열의 수입입니다. 읽기 전용.
일치 규칙 집합	해당 기본 개체에 대해 구성된 일치 규칙 집합의 수입입니다. 읽기 전용.

속성	설명
활성 집합의 일치 규칙	현재 활성 집합으로 선택된 규칙 집합에서 이 기본 개체에 대해 구성된 일치 규칙의 수입입니다. 읽기 전용.
기본 키 일치 규칙	이 기본 개체에 대해 구성된 기본 키 일치 규칙의 수입입니다. 읽기 전용.

수동 통합의 최대 일치 항목 수

이 설정을 통해 데이터 스튜어드에게 표시되는, 수동 통합을 위한 일치 항목 수를 제한할 수 있습니다.

이 항목은 데이터 스튜어드가 결정해야 하는 일치 항목 목록에 대해 제한을 설정합니다(기본값: 1000). 이 제한에 도달하면 Informatica MDM Hub가 수동 통합을 위해 준비된 레코드 수가 줄어들 때까지 일치 프로세스를 중지합니다.

이 값은 `consolidation_ind`가 2로 설정된 레코드 수를 확인하여 계산됩니다. 이 수는 각 자동 일치 및 병합 주기 마지막에 확인되며, 이 수가 수동 통합에 대한 최대 일치 항목 수를 초과할 경우 자동 일치 및 병합 프로세스가 종료됩니다.

일치 작업 일괄 처리 주기당 행 수

이 설정은 일치 프로세스를 실행하는 동안(일치 또는 자동 일치 및 병합 작업) Informatica MDM Hub가 일치를 위해 처리하는 레코드 수에 대한 상한을 지정합니다. 일치 프로세스가 실행되면 *일치 작업 일괄 처리*에 레코드가 포함되도록 플래그를 지정하여 시작됩니다. 일치를 위해 준비된 새 레코드/통합되지 않은 레코드 풀에서 (`CONSOLIDATION_IND=4`) 일치 프로세스가 `CONSOLIDATION_IND`를 3으로 바꿉니다. 플래그가 지정된 레코드 수는 [일치 작업 일괄 처리 주기당 행 수]에 따라 결정됩니다. 그러면 일치 프로세스가 일치 작업 일괄 처리의 레코드를 기본 개체의 모든 레코드와 일치시킵니다.

일치 작업 일괄 처리의 레코드 수는 일치 프로세스 실행에 소요되는 시간에 영향을 줍니다. 지정하는 값은 데이터 집합 크기, 일치 규칙 복잡성, 일치 프로세스를 실행하는 데 사용 가능한 시간 범위 등에 따라 달라집니다. 기본 일치 일괄 처리 크기는 낮음(10)입니다. 기본 개체의 레코드 수 및 해당 일치 규칙을 기준으로 하는 레코드에 대해 생성된 일치 항목 수를 바탕으로 이 크기를 늘릴 수 있습니다.

- 일치 일괄 처리 크기가 작을수록 일치 및 통합 프로세스를 실행하는 데 필요한 시간이 길어집니다.
- 반면 일치 일괄 처리 크기가 클수록 각 일치 및 통합 프로세스에서 수행할 작업이 많아집니다.

기본 개체마다 최적의 일치 일괄 처리 크기를 얻을 수 있는 중간 영역이 있습니다. 환경에서 조정 중인 성능의 일부로 최적의 일괄 처리 크기를 식별해야 합니다. 일치 일괄 처리 크기를 일치 및 병합할 레코드 볼륨의 10%로 시작하고 일치 작업만 실행한 후 일치 규칙에 따라 생성된 일치 항목 수를 확인한 후 적절하게 이 수를 상향 조정하거나 하향 조정합니다.

일치되지 않은 모든 행을 고유 행으로 허용

Informatica MDM Hub에서 일치 프로세스를 거쳤지만 일치 항목이 식별되지 않은 레코드를 고유 레코드로 표시(`CONSOLIDATION_IND=1`)하도록 하려면 이 기능을 활성화(예로 설정)합니다.

이 기능이 활성화된 경우 이러한 레코드의 상태는 Informatica MDM Hub에서 자동으로 *통합됨*으로 변경됩니다. 즉, 통합 표시기가 2에서 1로 변경됩니다. 통합된 레코드는 자동 병합 일괄 작업을 통해 데이터 스튜어드의 대기열에서 제거됩니다.

기본적으로 이 옵션은 비활성화되어 있습니다. 개발 환경에서 예를 들어 특정 일치 규칙 집합에 대해 고유한 것으로 식별되는 레코드를 확인하기 위해 일치 규칙을 반복적으로 테스트하고 조정하는 동안에는 대개 이 옵션을 비활성화합니다.

프로덕션 환경에서는 항상 이 옵션이 활성화되어 있어야 합니다. 그렇지 않으면 통합 표시기가 2인 레코드가 너무 많아질 수 있습니다. 이 레코드 백로그가 수동 통합의 최대 일치 항목 수 설정을 초과할 경우에는 레코드 일치 및 통합을 계속하기 전에 이러한 레코드를 처리해야 합니다.

일치/검색 전략

일치/검색 전략을 선택하여 필요한 성능에 대한 일치 신뢰도를 지정할 수 있습니다.

다음 옵션 중 하나를 선택합니다.

전략 옵션	설명
유사 항목	<p>철자 차이, 잘못된 철자의 가능성 및 일치 레코드를 다르게 만들 수 있는 기타 차이를 고려하는 확률 일치입니다. 기본 개체의 일치하는 데이터에 대한 기본 방법입니다. 이 설명서에서는 <i>유사 항목 일치 기본 개체</i>라고 합니다.</p> <p>참고: 유사 항목 일치/검색 전략을 지정한 경우 유사 항목 일치 키를 지정해야 합니다.</p>
정확히	<p>일치 열에서 같은 값을 가진 레코드만 일치시킵니다. 정확히 일치를 지정한 경우 이 기본 개체에 대해 정확히 일치 열만 정의할 수 있습니다. 정확히 일치 기본 개체에는 유사 항목 일치 열이 포함될 수 없습니다. 이 설명서에서는 <i>정확히 일치하는 기본 개체</i>라고 합니다.</p>

정확히 일치 전략은 수행 속도는 빠르지만 데이터가 불완전한 경우 정확히 일치 실행 시 일부 일치 항목이 누락되는 경우가 있습니다. 최적의 선택 옵션은 데이터의 특성, 데이터에 대한 지식 및 특정 일치 및 통합 요구 사항에 따라 달라집니다.

유사 항목 인구집단

일치/검색 전략이 유사 항목일 경우 일치시키려는 레코드에 대해 특정한 특성을 정의하는 *인구집단*을 선택해야 합니다.

데이터 특성은 국가별로 다를 수 있습니다. 기본적으로 Informatica MDM Hub는 데모 인구집단과 함께 제공되지만 Informatica는 각 국가에 대한 표준 인구집단을 제공합니다. 다른 인구집단이 필요한 경우 Informatica 지원부에 문의하십시오. 정확히 일치/검색 전략을 선택한 경우 이 값은 무시됩니다.

인구집단에서는 일치를 위해 다음 기능을 수행합니다.

- 이름, 주소 및 기타 식별 데이터에 있을 수 있는 불가피한 변형과 오류를 처리합니다.
예를 들어 미국의 인구집단에는 주민등록번호와 같이 미국 데이터에 사용되는 일반적인 식별 번호에 대한 인텔리전스가 포함되어 있습니다. 인구집단에는 일반적인 이름의 분포에 관한 인텔리전스도 포함되어 있습니다. 예를 들어 미국의 인구집단에서는 Smith라는 성의 백분율이 상대적으로 높습니다. 그러나 비영어권 국가의 인구집단에서는 Smith라는 이름이 일반적인 이름에 속하지 않습니다.
- Informatica MDM Hub가 일치 토큰을 구성하는 방식을 지정합니다.
- 검색 전략 및 일치 프로세스가 일치시킬 데이터의 인구집단에 대해 작동하는 방식을 지정합니다.

이전 Rowid 개체만 일치

현재 레코드와 더 작은 ROWID_OBJECT 값을 가진 레코드를 일치시키려면 이 속성을 활성화합니다.

예를 들어 현재 레코드에 100의 ROWID_OBJECT 값이 있으면 해당 레코드가 100보다 작은 ROWID_OBJECT 값을 포함한 기본 개체의 다른 레코드와만 일치됩니다. 100보다 큰 ROWID_OBJECT 값이 있는 레코드는 일치 프로세스 동안 무시됩니다.

필요한 일치 항목 수를 줄이고 성능을 높이려면 이전 **Rowid** 개체만 일치 속성을 사용합니다. 단, **PUT API** 호출을 실행한 경우 또는 레코드가 **rowid** 순서를 벗어나 삽입된 경우에는 레코드가 완전히 일치하지는 않을 수 있습니다. 이 경우 데이터의 특성과 관련 일치 요구 사항을 기준으로 성능과 일치 수량 간의 균형을 유지해야 합니다.

초기 데이터 로드의 성능을 향상시키기 위해 이전 **Rowid** 개체만 일치 속성을 활성화할 수 있습니다. 증분 데이터 로드에서 이 속성을 활성화할 경우 잠재적 일치 항목을 잃을 수 있습니다. 기본적으로 이 옵션은 비활성화되어 있습니다.

한 번만 일치

[“이전 Rowid 개체만 일치” 페이지 378](#)가 선택된 경우에만 유사 키 일치에서 사용할 수 있습니다.

한 번만 일치가 활성화(선택)된 경우 일치하는 레코드를 찾으면 **Informatica MDM Hub**가 이 검색 범위(유사 일치 키 값의 집합) 내에서 더 이상 일치하는 레코드를 찾지 않습니다. 이 기능을 사용하면 중복을 줄이고 성능을 향상시킬 수 있습니다. **Informatica MDM Hub**가 검색 범위에서 레코드의 모든 일치 항목을 찾는 것이 아니라 각 레코드에 대해 단일 일치 항목을 찾을 수 있습니다. 후속 일치 주기에서 병합 프로세스는 이러한 일치 항목을 기본 개체와 연결된 큰 **XREF** 레코드 그룹에 배치합니다.

기본적으로 이 옵션은 선택되어 있지 않습니다(비활성화됨). 이 기능을 활성화하면 일치 항목이 누락될 수 있습니다. 예를 들어 레코드 **A**가 레코드 **B**와 일치하는 동시에 레코드 **C**와 일치하지만 레코드 **B**와 레코드 **C**는 일치하지 않는다고 가정해 보십시오. 이 경우 데이터의 특성과 관련 일치 요구 사항을 기준으로 성능과 일치 수량 간의 균형을 유지해야 합니다.

동적 일치 분석 임계값

일치 프로세스 중에 동적 일치 분석에서 일치 프로세스에 지나치게 오랜 시간이 걸릴지 여부를 확인합니다.

동적 일치 분석 임계값은 허용 가능한 최대 비교 횟수를 지정합니다.

동적 일치 임계값을 활성화하려면 **0**이 아닌 값을 지정하십시오. 매우 유사한(일치의 과도한 집중) 데이터가 있는 경우 데이터의 핫스팟에 소비되는 작업량을 줄이려면 이 기능을 활성화하십시오. 핫스팟은 과도 일치된 데이터(일치 항목의 큰 교집합)를 나타내는 레코드 그룹입니다. 동적 일치 분석 임계값이 활성화되어 있으면 일치 프로세스 중에 지정된 수보다 많은 잠재적 일치 후보를 생성하는 레코드를 건너뛰게 됩니다. 기본적으로 이 옵션은 **0**(비활성화)입니다.

Informatica MDM Hub에서는, 지정된 검색 범위에 대한 일치를 실행하기 전에 검색 레코드(일치 항목을 검색할 레코드)의 수를 계산하여 이 값에 파일 레코드의 수(비교해야 할 일치 키 테이블에서 반환된 레코드의 수)를 곱합니다. 이 결과가 지정된 동적 일치 분석 임계값보다 크면 해당 데이터 범위에서는 비교가 수행되지 않고 구체적으로 조사할 수 있도록 응용 프로그램 서버 로그에 범위가 기록됩니다.

보류 중인 레코드에서 일치 활성화

기본적으로 일치 프로세스에는 **ACTIVE** 레코드만 포함되고 **PENDING** 레코드는 무시됩니다. 상태 관리 사용 개체의 경우 일치 프로세스에 **PENDING** 상태인 레코드를 포함하려면 이 확인란을 선택합니다. 이 설정에 관계없이 **DELETED** 레코드는 일치 프로세스에서 무시됩니다.

긴 ROWID_OBJECT 값 지원

기본 개체의 레코드 수가 너무 많아 **ROWID_OBJECT** 값이 12자리를 초과할 경우에는 처리 서버에서 보다 긴 값을 지원하도록 명시적으로 설정해야 합니다.

처리 서버에서 긴 **Rowid** 개체 값을 사용할 수 있도록 하려면 **cmxcleanse.properties** 파일을 편집하여 **cmx.server.bmg.use longs** 설정을 구성합니다.

```
cmx.server.bmg.use longs=1
```

기본적으로 이 옵션은 비활성화되어 있습니다.

관련 레코드의 일치 경로 구성

이 섹션에서는 Informatica MDM Hub 구현에서 일치에 사용되는 관련 레코드의 일치 경로를 구성하는 방법에 대해 설명합니다.

일치 경로

일치 경로를 통해 계층이 기본 개체 사이에 있던지(테이블 간 경로) 단일 기본 개체에 있던지(테이블 내 경로) 상관없이 레코드 간 계층을 이동할 수 있습니다. 일치 경로는 별도의 테이블이나 동일한 테이블에 있는 관련 레코드와 연관된 일치 열 규칙을 구성하는 데 사용됩니다.

외래 키 관계 및 필터

다른 레코드를 가리키는 일치 경로를 구성하는 작업에는 두 가지 기본 구성 요소가 포함됩니다.

구성 요소	설명
외래 키 관계	관계를 다른 레코드로 이동하는 데 사용됩니다. 상위-하위 관계와 하위-상위 관계를 지정할 수 있습니다.
필터(선택 사항)	ADDRESS_TYPE 또는 PARTY_TYPE 같은 지정된 열의 값을 기준으로 레코드를 선택적으로 포함 또는 제외할 수 있습니다.

관계 기본 개체

이러한 유형의 관계, 특히 다대다 관계에 대해 일치 규칙을 구성하려면 레코드 간 관계를 Informatica MDM Hub에 설명하는 *관계 기본 개체*로 사용할 별도의 기본 개체를 생성해야 합니다. Informatica MDM Hub 프로세스(랜드, 준비 및 로드)가 아닌 Informatica MDM Hub 외부에서 실행되는 데이터 관리 도구를 사용하여 이 관계 기본 개체를 관계에 대한 정보로 채웁니다.

각 관계 유형에 대해 별도의 관계 기본 개체를 구성합니다. 관계 유형의 추가 특성(예: 시작 날짜, 종료 날짜 및 기타 관계 세부 정보)을 포함할 수 있습니다. 관계 기본 개체는 일치 열 규칙을 구성할 수 있도록 해주는 일치 경로를 정의합니다.

중요: 테이블 내 또는 테이블 간 일치 경로의 레코드 간 관계를 정의하는 데 사용되는 기본 개체에 대해서는 일치 및 통합 프로세스를 실행하지 마십시오. 그러면 관계 데이터가 변경되어 레코드 간의 연관이 손실될 수 있습니다.

테이블 간 경로

테이블 간 경로는 서로 다른 두 기본 개체의 레코드 간 관계를 정의합니다. 간단히 외래 키 관계를 구성하여 이 관계를 정의할 수 있는 경우가 많습니다. 즉, 하위 기본 개체의 키 열이 상위 기본 개체의 기본 키를 가리키도록 구성합니다.

하지만 일부 경우에는 레코드 간의 관계가 훨씬 복잡하기 때문에 두 테이블의 레코드 간 관계를 정의하는 중간 기본 개체를 사용해야 합니다.

테이블 간 경로에 대한 기본 개체 예

Informatica MDM Hub 구현에 기본 개체 두 개가 있는 다음과 같은 예를 살펴보겠습니다.

기본 개체	설명
Person	조직의 직원, 다른 조직의 직원(잠재 고객, 고객, 공급업체 또는 파트너), 하청업자 등과 같은 모든 유형의 개인을 포함합니다.
Address	모든 주소 유형(우편 주소, 배송지, 집, 회사 등)을 포함합니다.

이 예에서는 다대다 관계가 발생할 가능성이 있습니다.

- 한 개인이 여러 개의 주소(예: 집 주소와 회사 주소)를 가질 수 있습니다.
- 한 주소(예: 회사 또는 집)에 여러 개인이 포함될 수 있습니다.

서로 다른 기본 개체의 레코드 간에 이런 종류의 관계에 대한 일치 규칙을 구성하기 위해, 두 기본 개체의 레코드 간에 어떤 관계가 있는지 Informatica MDM Hub에 설명하는 별도의 기본 개체(예: PersAddrRel)를 생성할 수 있습니다.

예제 기본 개체의 열

Person 기본 개체에 다음과 같은 열이 있다고 가정합니다.

열	유형	설명
ROWID_OBJECT	CHAR(14)	기본 키입니다. 기본 개체에서 개인을 고유하게 식별합니다.
TYPE	CHAR(14)	개인의 유형(직원 또는 고객 연락처)입니다.
NAME	VARCHAR(50)	개인의 이름입니다(이 예제에서는 단순화됨).
EMPLOYER	VARCHAR(50)	개인의 고용주입니다.
...

Address 기본 개체에 다음과 같은 열이 있다고 가정합니다.

열	유형	설명
ROWID_OBJECT	CHAR(14)	기본 키입니다. 직원을 고유하게 식별합니다.
TYPE	CHAR(14)	주소의 유형(예: 집, 직장, 우편 또는 배송 주소)입니다.
NAME	VARCHAR(50)	주소에 거주 또는 상주하는 개인이나 조직의 이름입니다.
ADDRESS_1	VARCHAR(50)	첫 번째 주소 행입니다.
ADDRESS_2	VARCHAR(50)	두 번째 주소 행입니다.
CITY	VARCHAR(50)	구/군/시입니다.
STATE_PROV	VARCHAR(50)	시/도입니다.

열	유형	설명
POSTAL_CODE	VARCHAR(50)	우편 번호
...

두 기본 개체에 있는 레코드 간의 관계를 정의하기 위해 **PersonAddrRel** 기본 개체에 다음과 같은 열을 포함할 수 있습니다.

열	유형	설명
ROWID_OBJECT	CHAR(14)	기본 키입니다. 기본 개체에서 개인을 고유하게 식별합니다.
PERS_FK	CHAR(14)	Person 기본 개체의 ROWID_OBJECT 열에 대한 외래 키입니다.
ADDR_FK	CHAR(14)	Address 기본 개체의 ROWID_OBJECT 열에 대한 외래 키입니다.

외래 키 열의 열 유형 "CHAR(14)"는 해당 열이 가리키는 기본 키와 일치합니다.

예제 구성 단계

관계 기본 개체(**PersonAddrRel**)를 구성한 후 다음 작업을 수행하게 됩니다.

- 이 기본 개체를 사용하여 **Person** 및 **Address** 기본 개체의 ROWID_OBJECT에 대한 외래 키를 구성합니다.
- 레코드 간의 관계를 설명하는 데이터를 사용하여 **PersAddrRel** 기본 개체를 로드합니다.

ROWID_OBJECT	PERS_FKEY	ADDR_FKEY
1	380	132
2	480	920
3	786	432
4	786	980
5	12	1028
6	922	1028
7	1302	110
...

이 예에서는 **Person 786**의 주소가 2개이고 **Address 1028**의 개인이 2명임에 유의하십시오.

- 관련 레코드에 대한 일치 열 규칙을 구성할 때 **PersonAddrRel** 기본 개체를 사용합니다.

테이블 내 경로

기본 개체 내에서 개별 레코드 간에 상위/하위 관계가 존재할 수 있습니다. **Informatica MDM Hub**에서는 동일한 기본 개체 내 레코드 간의 관계를 명확히 할 수 있으며 열 일치 규칙을 구성할 때 이러한 관계를 사용할 수 있습니다.

테이블 내 경로에 대한 예제 기본 개체

다음 그림에는 직원들 간에 보고 관계가 존재하는 Employee 기본 개체가 나열되어 있습니다.

직원 이름	직함
Maria Alvarez	CEO
Todd Garvey	사장
Lisa Cho	부사장
Anil Shahani	이사
Jane Rickets	부장
Mark Sullivan	분석가
Rajan Vir	분석가
Roberta Zimmer	분석가
...	...

직원들 간의 관계는 계층적입니다. CEO는 계층의 맨 위에 있으며, 이러한 레코드를 **글로벌 최종 상위** 레코드라고 합니다.

예제 기본 개체의 열

Employee 기본 개체에 다음과 같은 열이 있다고 가정합니다.

열	유형	설명
ROWID_OBJECT	CHAR(14)	기본 키입니다. 기본 개체에서 직원을 고유하게 식별합니다.
NAME	VARCHAR(50)	직원 이름입니다.
TITLE	VARCHAR(50)	직원의 직위입니다.
...

관계 기본 개체 생성

이 유형의 개체에 대한 일치 규칙을 구성하려면 Informatica MDM Hub에 레코드 간의 관계를 설명하는 별도의 기본 개체를 생성합니다.

예를 들어 다음 열을 사용하여 EmplRepRel 기본 개체를 생성하고 구성할 수 있습니다.

열	유형	설명
ROWID_OBJECT	CHAR(14)	기본 키입니다. 이 관계 레코드를 고유하게 식별합니다.
EMPLOYEE_FK	CHAR(14)	직원 레코드의 ROWID_OBJECT에 대한 외래 키입니다.
REPORTS_TO_FK	CHAR(14)	관리자 레코드의 ROWID_OBJECT에 대한 외래 키입니다.

참고: 외래 키 열의 열 유형 CHAR(14)는 해당 키가 가리키는 기본 키와 일치합니다.

예제 구성 단계

이 기본 개체를 구성한 후 다음 태스크를 완료해야 합니다.

1. 이 기본 개체에서 **Employee** 기본 개체의 **ROWID_OBJECT**에 대한 외래 키를 구성합니다.
2. 레코드 간의 관계를 설명하는 데이터를 사용하여 이 기본 개체를 로드합니다.

ROWID_OBJECT	EMPLOYEE	REPORTS_TO
1	7	93
2	19	71
3	24	82
4	29	82
5	31	82
6	31	71
7	48	16
8	53	12

레코드 간의 다대다 관계를 정의할 수 있습니다. 예를 들어 **ROWID_OBJECT**가 31인 직원은 서로 다른 두 관리자(**ROWID_OBJECT**=82 및 **ROWID_OBJECT**=71)에게 보고하고 **ROWID_OBJECT**가 82인 관리자에게는 보고서 세 개(**ROWID_OBJECT**=24, 29 및 31)가 있습니다.

3. 관련 레코드의 일치 열 규칙을 구성하는 경우 **EmplRepRel** 기본 개체를 사용합니다.
예를 들어 보다 정확히 일치를 생성하기 위해 직원의 관리자를 고려한 일치 규칙을 생성할 수 있습니다.

참고: 이 예에서는 **REPORTS_TO** 필드를 사용하여 관계를 정의하지만, **RELATIONSHIP_TYPE**과 같이 보다 일반적이고 유연한 정보를 사용하여 레코드를 연결할 수도 있습니다.

경로 탭으로 이동

기본 개체의 [경로] 탭으로 이동하려면

1. 스키마 관리자에서 구성할 기본 개체의 [일치/병합 설정 세부 정보] 대화 상자로 이동합니다.
2. **경로** 탭을 클릭합니다.
스키마 관리자에 [경로 구성 요소] 창이 표시됩니다.

경로 탭의 섹션

[경로] 탭에는 다음과 같은 두 개의 섹션이 포함됩니다.

섹션	설명
경로 구성 요소	관계를 이동하는 데 사용되는 외래 키를 구성합니다.
필터	일치 작업 시 레코드를 포함하거나 제외하는 데 사용되는 필터를 구성합니다.

루트 기본 개체

루트 기본 개체는 [경로 구성 요소] 섹션에 자동으로 표시되며 항상 사용할 수 있습니다.

루트 기본 개체는 하위 또는 상위 관계가 없는 항목을 나타냅니다. 상위 또는 하위 레코드와 관련된 일치 규칙을 구성하려면 루트 기본 개체에 경로 구성 요소를 명시적으로 추가해야 하며, 이 관계가 사전에 구성되어 있어야 합니다.

일치 경로에 대한 필터 구성

사용자가 지정하는 열 값을 기반으로 일치 경로의 필터에 일치하는 레코드를 포함하거나 제외할 수 있습니다. 열에 대한 필터를 정의할 때는 일치 처리를 적용할 수 있는 레코드를 결정하는 하나 이상의 값으로 필터 조건을 지정합니다. 예를 들어 배송 주소와 청구 주소가 모두 포함된 **Address** 기본 개체의 경우 청구 주소를 포함하고 배송 주소는 제외하는 필터를 구성할 수 있습니다. 일치 프로세스가 실행되면 **MDM Hub**가 일치 일괄 처리의 레코드를 청구 주소 레코드와 일치시킵니다.

참고: 날짜 열에 필터 조건을 지정할 경우 데이터베이스 로컬과 호환되는 날짜 값의 올바른 문자열 표현을 사용합니다.

Informatica MDM Hub에서 필터의 속성은 다음과 같습니다.

설정	설명
열	현재 선택된 기본 개체에서 구성할 열입니다.
연산자	이 필터에 사용할 연산자입니다. 다음 값 중 하나입니다. <ul style="list-style-type: none">- IN - 지정된 값이 들어 있는 열을 포함합니다.- NOT IN - 지정된 값이 들어 있는 열을 제외합니다.
값	이 필터에 사용할 하나 이상의 값입니다.

예를 들어 **Address** 기본 개체의 메일 주소만을 기준으로 일치시키려면 다음을 지정하십시오.

설정	예제 값
열	ADDR_TYPE
연산자	IN
값	MAILING

이 예에서는 메일 주소만 일치 조건으로 사용되어 열 필드에서 "MAILING"이 포함된 레코드만 필터링됩니다. 다른 모든 레코드는 무시됩니다.

필터 추가

여러 필터를 추가할 경우 Informatica MDM Hub에서는 논리적 AND 연산자를 사용하여 전체 식을 평가합니다. 예를 들면 다음과 같습니다.

xExpr AND yExpr AND zExpr

필터를 추가하려면

1. 스키마 관리자에서 **경로** 탭으로 이동합니다.
2. 쓰기 잠금을 획득합니다.

3. 필터 섹션에서 **추가** 단추를 클릭합니다.
스키마 관리자에 필터 추가 대화 상자가 표시됩니다.
4. 이 경로 구성 요소의 속성을 지정합니다.
5. 이 필터의 값을 지정합니다.
6. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

필터 값 편집

필터의 값을 편집하려면

1. 다음 작업 중 하나를 수행합니다.
 - 필터를 추가합니다.
 - 필터 속성을 편집합니다.
2. 필터 추가 또는 필터 편집 대화 상자에서 값 필드 옆에 있는 **편집** 단추를 클릭합니다.
스키마 관리자에 값 편집 대화 상자가 표시됩니다.
3. 이 필터에 대한 값을 구성합니다.
 - 값을 추가하려면 **추가** 단추를 클릭합니다. 메시지가 표시되면 값을 지정하고 **확인**을 클릭합니다.
 - 값을 삭제하려면 값 편집 대화 상자에서 값을 선택하고 **삭제** 단추를 클릭한 다음 값을 삭제할지 묻는 메시지가 나타나면 **예**를 클릭합니다.
4. **확인**을 클릭합니다.
5. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

필터 속성 편집

필터 속성을 편집하려면

1. 스키마 관리자에서 **경로** 탭으로 이동합니다.
2. 쓰기 잠금을 획득합니다.
3. 필터 섹션에서 **편집** 단추를 클릭합니다.
스키마 관리자에 필터 추가 대화 상자가 표시됩니다.
4. 이 경로 구성 요소의 속성을 지정합니다.
5. 이 필터의 값을 지정합니다.
6. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

필터 삭제

필터를 삭제하려면

1. 스키마 관리자에서 **경로** 탭으로 이동합니다.
2. 쓰기 잠금을 획득합니다.
3. 필터 섹션에서 삭제할 필터를 선택한 다음 **삭제** 단추를 클릭합니다.
스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.
4. **예**를 클릭합니다.

경로 구성 요소 구성

이 섹션에서는 스키마 관리자에서 경로 구성 요소를 구성하는 방법에 대해 설명합니다. 경로 구성 요소는 일치 열에서 상위 또는 하위 테이블의 열을 사용할 목적으로 외래 키를 사용하여 상위 테이블과 하위 테이블 간의 관계를 정의하는 데 사용됩니다.

표시 이름

Hub 콘솔에 표시되는 이 경로 구성 요소의 이름입니다.

실제 이름

데이터베이스의 경로 구성 요소에 대한 실제 이름입니다. Informatica MDM Hub에서는 사용자가 입력한 표시 이름에 기반하여 경로 구성 요소의 실제 이름을 제안합니다.

누락된 하위 레코드 허용

누락된 하위 레코드 허용 옵션은 일치 경로의 하위 기본 개체에 레코드가 있는지 확인하는 검사를 기반으로 상위 레코드를 일치 항목으로 간주해야 하는지 여부를 나타냅니다.

일치 경로 구성 요소 수준에서 누락된 하위 레코드를 허용하는 옵션을 활성화할 수 있습니다. 일치 경로 구성 요소 수준은 주 상위 기본 개체의 일치 경로에 있는 여러 하위 기본 개체 수준으로 구성될 수 있습니다.

일치 경로 구성 요소에서 누락된 하위 레코드를 허용하는 옵션을 활성화하면 상위 기본 개체 레코드와 이 레코드에 연결된 하위 기본 개체 레코드 간에 일치가 발생합니다. 이 옵션이 활성화된 경우 하위 기본 개체의 하위 레코드가 상위 기본 개체 레코드에 없는 경우에도 일치가 발생합니다. 기본적으로 누락된 하위 레코드를 허용하는 옵션은 상위 기본 개체의 일치 경로에 있는 모든 하위 기본 개체에 대해 활성화됩니다. 이 옵션은 활성화 시 포괄적으로 적용되므로 상위 기본 테이블과 이 옵션이 활성화된 하위 기본 테이블 간에 데이터베이스 외부 조인이 구현됩니다. Informatica Data Director 기본 검색에서는 하위 레코드가 없는 레코드에 대해 결과가 반환됩니다.

모든 일치 경로 구성 요소에 대해 누락된 하위 레코드를 허용하는 옵션을 비활성화할 경우 모든 하위 기본 개체 레코드가 있는 상위 기본 개체 레코드는 일치 프로세스를 거칩니다. 이 옵션이 비활성화된 하위 기본 개체의 하위 레코드가 상위 레코드에 없는 경우에도 상위 레코드는 일치 프로세스를 거치지 않습니다. 이 옵션은 비활성화 시 배타적으로 적용되므로 일반적인 데이터베이스 동등 조인과 유사합니다. 즉, 상위 레코드에 하위 레코드가 없으면 상위 또는 하위 레코드가 반환되지 않습니다. 이 옵션을 비활성화하면 외부 조인 시 발생하는 성능 저하가 방지됩니다. Informatica Data Director 기본 검색에서는 하위 레코드가 없는 레코드에 대해 결과가 반환되지 않습니다.

기본 개체에 대한 유사 항목 일치를 수행해야 하는 경우에는 상위 기본 개체 레코드의 토큰화가 발생해야 합니다. 누락된 하위 레코드를 허용하는 옵션이 비활성화된 모든 하위 기본 개체에 관련 하위 기본 개체 레코드가 있는 경우, 상위 기본 개체 레코드의 토큰화가 발생합니다. 누락된 하위 레코드를 허용하는 옵션이 비활성화되어 있고 아직 레코드가 포함되지 않은 하위 기본 개체가 상위 기본 개체 레코드에 있으면 상위 레코드가 토큰화되지 않습니다.

데이터가 완전하면 상위 레코드 간의 일치가 예상대로 발생하며 MDM Hub에서는 일치 조건에 따른 일치에 모든 상위 레코드를 포함합니다. 상위 기본 개체의 일치 경로에 있는 각 하위 기본 개체의 하위 레코드가 상위 레코드에 있고 상위 기본 개체에 대한 일치 규칙에 하위 열을 포함한 경우에는 데이터가 완전합니다. 그러나 데이터가 불완전하면 레코드가 없는 하위 기본 개체의 경로 구성 요소에 대해 누락된 하위 항목을 확인하는 옵션을 활성화해야 상위 레코드가 일치될 수 있습니다. 한 하위 기본 개체의 레코드가 누락된 상위 레코드와 다른 하위 기본 개체의 레코드가 누락된 다른 상위 레코드가 데이터에 함께 포함되어 있는 경우 이 데이터는 불완전합니다. 또한 상위 일치 규칙에 하위 열을 포함하지 않은 경우에도 데이터가 불완전합니다. 이 경우 상위 기본 개체에 하위 기본 개체의 레코드가 없으면 일치 경로 열의 기반이 되는 열을 포함하는 상위 레코드 및 연결된 하위 기본 개체 레코드는 일치에서 제외되지 않습니다.

참고: 누락된 하위 레코드를 허용하는 옵션이 활성화된 경우 Informatica MDM Hub는 상위 테이블과 하위 테이블 간의 외부 조인을 수행합니다. 이 경우 이 옵션이 활성화된 각 일치 경로 구성 요소에서 성능에 영향이 있습니다. 따라서 필요하지 않은 경우에는 이 옵션을 비활성화하는 것이 더 효율적입니다.

제약 조건

속성	설명
테이블	스키마에 포함된 테이블의 목록입니다.
방향	외래 키의 방향입니다. 상위 대 하위 하위 대 상위 N/A
외래 키 대상	외래 키가 가리키는 열입니다. 이 열은 다른 기본 개체나 동일한 기본 개체에 있을 수 있습니다.

경로 구성 요소 추가

경로 구성 요소를 추가하려면

1. 스키마 관리자에서 경로 탭으로 이동합니다.
2. 쓰기 잠금을 획득합니다.
3. 경로 구성 요소 섹션에서 **추가** 단추를 클릭합니다.
스키마 관리자에 경로 구성 요소 추가 대화 상자가 표시됩니다.
4. 이 경로 구성 요소의 속성을 지정합니다.
5. **확인**을 클릭합니다.
6. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

경로 구성 요소 편집

경로 구성 요소를 편집하려면

1. 스키마 관리자에서 **경로** 탭으로 이동합니다.
2. 쓰기 잠금을 획득합니다.
3. 경로 구성 요소 트리에서 삭제할 경로 구성 요소를 선택합니다.
4. 경로 구성 요소 섹션에서 **편집** 단추를 클릭합니다.
스키마 관리자에 경로 구성 요소 편집 대화 상자가 표시됩니다.
5. 이 경로 구성 요소의 속성을 지정합니다. 다음 값을 변경할 수 있습니다.
 - 표시 이름
 - 누락된 하위 레코드 허용
6. **확인**을 클릭합니다.
7. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

경로 구성 요소 삭제

경로 구성 요소는 삭제할 수 있지만 루트 기본 개체는 삭제할 수 *없습니다*. 경로 구성 요소를 삭제하려면

1. 스키마 관리자에서 **경로** 탭으로 이동합니다.
2. 쓰기 잠금을 획득합니다.
3. 경로 구성 요소 트리에서 삭제할 경로 구성 요소를 선택합니다.
4. 경로 구성 요소 섹션에서 **삭제** 단추를 클릭합니다.
스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.
5. **예**를 클릭합니다.
6. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

일치 열 구성

이 섹션에서는 일치 열을 일치 열 규칙에 사용할 수 있도록 구성하는 방법에 대해 설명합니다. 이 방법 대신 기본 키 일치 규칙을 구성하려면 [“기본 키 일치 규칙 구성” 페이지 423](#)의 지침을 참조하십시오.

일치 열 정보

일치 열은 이름 또는 주소 열과 같이 일치 규칙에 사용할 열입니다. 규칙 정의에 열을 사용하려면 먼저 해당 열을 일치 규칙에 사용할 수 있는 열로 지정하고 해당 열에 포함된 데이터에 대한 정보를 제공해야 합니다.

참고: 일치 규칙을 정의하기 위한 열을 선택할 때 숫자 또는 10진수 데이터 유형의 열은 일치 열로 사용할 수 없습니다.

일치 열 유형

일치 규칙에 대한 유사 항목 또는 정확히 일치 열 유형을 구성할 수 있습니다.

다음 테이블에서는 일치 규칙에 대한 열 유형에 대해 설명합니다.

열 유형	설명
유사 항목	확률 일치. 철자, 약어, 단어 순서, 완성도, 신뢰도 및 기타 불일치가 서로 다른 데이터가 포함된 열에 사용됩니다. 예를 들어 유사 항목 일치 열에는 주소, 지리적 좌표(예: 위도, 경도, 고도) 및 사람 또는 조직의 이름이 포함됩니다.
정확히	확정적 일치. 일관되고 예측 가능한 패턴이 포함된 열에 사용됩니다. 정확히 일치 열에서는 동일한 데이터만 일치됩니다. ID, 우편 번호, 산업 코드 또는 기타 잘 정의된 정보가 이에 해당됩니다.

일치 열은 검색 전략에 따라 달라짐

구성할 수 있는 일치 열의 유형은 구성하는 기본 개체의 유형에 따라 달라집니다.

기본 개체의 유형은 선택한 일치/검색 전략에 의해 정의됩니다.

일치 전략	설명
유사 항목 일치 기본 개체	정확히 일치 열과 유사 항목 일치 열을 구성할 수 있습니다.
정확히 일치하는 기본 개체	정확히 일치 열은 구성할 수 있지만 유사 항목 일치 열은 구성할 수 없습니다.

경로 구성 요소

이 경로 구성 요소는 일치 열 정의 시 사용할 소스 테이블이거나 레코드 계층 탐색 시 사용되는 일치 경로입니다. 일치 경로는 별도의 테이블이나 동일한 테이블에 있는 관련 레코드와 연관된 일치 열 규칙을 구성하는 데 사용됩니다. 경로 구성 요소를 지정하려면 일치 경로를 구성해야 합니다.

일치 열의 경로 구성 요소를 지정하려면

1. [경로 구성 요소] 필드 옆의 **편집** 단추를 클릭합니다.
스키마 관리자에 [일치 경로 구성 요소 선택] 대화 상자가 표시됩니다.
2. 일치 경로 구성 요소를 선택합니다.
3. **확인**을 클릭합니다.

필드 이름

유사 항목 일치 열을 일치 규칙에 추가할 때 목록에서 필드 이름을 선택할 수 있습니다.

다음 테이블에서는 일치 규칙에 대한 유사 항목 일치 열을 추가할 때 선택할 수 있는 필드 이름에 대해 설명합니다.

필드 이름	설명
Address_Part1	로컬리티 마지막 행은 포함하지 않고 이 행까지 주소의 부분을 포함합니다. 주소 구성 요소의 위치는 데이터 인구집단에서와 같이 일반 단어 순서와 동일해야 합니다. 이 데이터를 한 필드로 전달합니다. 기본 개체를 기준으로 일치 전에 이러한 특성을 한 필드로 연결할 수도 있습니다. 예를 들어 미국의 경우 Address_Part1 문자열에 Care-of + Building Name + Street Number + Street Name + Street Type + Apartment Details 필드가 포함됩니다. Address_Part1에는 주소용으로 특별히 설계된 메서드와 옵션이 사용됩니다. 과도 일치를 방지하기 위해 일치 프로세스에서는 필드에 10개 이상의 문자가 포함되어 있는 경우에만 전위 오류를 고려합니다. 예를 들어 'PO Box 38'은 'PO Box 83'과 일치하지 않지만, '13 Capital Street'는 '31 Capital Street'와 일치합니다.
Address_Part2	주소의 로컬리티 행입니다. 예를 들어 미국의 경우 일반적인 Address_Part2에 City + State + Zip (+ Country) 필드가 포함됩니다. Address_Part2의 일치에는 주소용으로 특별히 설계된 메서드와 옵션이 사용됩니다.
Attribute1, Attribute2	두 개의 범용 필드입니다. MDM Hub는 전위 및 누락 문자 또는 숫자를 보충하는 범용 문자열 일치 알고리즘을 사용하여 특정 필드를 일치시킵니다.
Date	생년월일, 만료 날짜, 계약 날짜, 변경 날짜, 생성 날짜 등 모든 날짜 유형을 일치시킵니다. 일+월+년 형식으로 날짜를 전달합니다. SSA_Date 필드 이름에는 날짜 구성 요소 간에 구분자를 사용하거나 사용하지 않을 수 있습니다. 날짜의 일치에는 날짜용으로 특별히 설계된 메서드와 옵션이 사용됩니다. 날짜의 일치는 해당 데이터 유형에서 발견되는 일반적인 오류와 변형을 처리합니다.

필드 이름	설명
Geocode	지리적 좌표, 위도, 경도 및 고도를 일치시킵니다. 다음과 같은 순서로 지리적 좌표를 지정합니다. 1. 위도 2. 경도 3. 고도 하나의 필드 또는 여러 필드에서 이 데이터를 전달할 수 있습니다. 하나의 필드에서 좌표 부여 데이터를 연결하는 경우 쉼표 또는 공백으로 값을 구분합니다. 좌표 부여는 전위 및 누락 문자 또는 숫자를 보충하는 문자열 일치 알고리즘을 사용합니다.
ID	모든 유형의 ID(예: 계정 번호, 고객 번호, 신용 카드 번호, 운전면허번호, 여권 번호, 정책 번호, SSN 또는 기타 ID 코드 및 VIN)를 일치시킵니다. ID 필드는 전위 및 누락 문자 또는 숫자를 보충하는 문자열 일치 알고리즘을 사용합니다.
Organization_Name	조직의 이름(예: 조직 이름, 비즈니스 이름, 기관 이름, 부서 이름, 대리점 이름, 거래 이름)을 일치시킵니다. 이 필드는 단일 이름 또는 복합 이름(예: 법적 이름과 해당 거래 스타일)에 대한 일치를 지원합니다. 일치에 대한 단일 Organization_Name 열에서 여러 개의 이름(예: 법적 이름과 거래 스타일)을 사용할 수도 있습니다.
Person_Name	개인의 이름을 일치시킵니다. 개인의 전체 이름을 사용합니다. 이름, 중간 이름 및 성의 위치는 인구집단에서 사용되는 일반적인 단어 순서와 동일해야 합니다. 예를 들어 영어권 국가에서 일반적인 순서는 이름 + 중간 이름 + 성입니다. 기본 개체 디자인을 기준으로 일치 전에 이러한 필드를 한 필드로 연결할 수 있습니다. 이 필드는 단일 이름 또는 계정 이름(예: JOHN & MARY SMITH)에 대한 일치를 지원합니다. 여러 개의 이름(예: 결혼 전 이름과 결혼 후 이름)을 사용할 수도 있습니다.
Postal_Area	Address_Part2 필드를 사용하지 않고 우편 번호를 더욱 강조하는 데 사용합니다. ZIP 코드를 비롯한 모든 유형의 우편 번호에 사용합니다. Postal_Area 필드 이름은 전위 및 누락 문자 또는 숫자를 보충하는 문자열 일치 알고리즘을 사용합니다.
Telephone_Number	전화 번호를 일치시키는 데 사용합니다. Telephone_Number 필드 이름은 전위 및 누락 숫자 또는 지역 코드를 보충하는 문자열 일치 알고리즘을 사용합니다.

일치를 위해 여러 열 선택

일치시킬 둘 이상의 열을 지정한 경우

- 값이 일치 목적에서 사용된 필드로 연결되며, 각 값 간 공백이 삽입됩니다. 예를 들어 기본 개체에서 **first**, **middle**, **last** 및 **suffix** 열을 선택할 수 있습니다. 연결된 필드는 다음과 같이 표시됩니다(문자열 마지막 단어 뒤에 공백이 포함됨).

first middle last suffix

(예:

Anna Maria Gonzales MD

- 공백이나 null 데이터를 포함한 데이터의 경우
 - 데이터에 공백이 있으면 공백은 그대로 남아 있으며 필드는 NULL이 아닙니다.
 - 모든 필드가 null일 경우 결합된 값도 null입니다.
 - 결합된 필드의 특정 구성 요소가 null이면 null을 대체하기 위해 공백이 추가되지 않습니다.

참고: 정확히 일치 열의 경우에는 열을 연결하지 않는 것이 좋습니다.

유사 항목 일치 기본 개체에 대한 일치 열 구성

유사 항목 일치 기본 개체에는 유사 항목 일치 열과 정확히 일치 열이 모두 포함될 수 있습니다. 정확히 일치 기본 개체의 경우에는 [“정확히 일치하는 기본 개체에 대한 일치 열 구성” 페이지 395](#)을 참조하십시오.

유사 항목 일치 기본 개체에 대한 일치 열 탭으로 이동

유사 항목 일치 기본 개체에 대해 일치 열을 정의하려면

1. 스키마 관리자에서 구성할 유사 항목 일치 기본 개체를 선택합니다.
2. **일치/병합 설정** 노드를 클릭합니다.
3. **일치 열** 탭을 클릭합니다.

스키마 관리자에 유사 항목 일치 기본 개체에 대한 [일치 열] 탭이 표시됩니다.

유사 항목 일치 기본 개체의 [일치 열] 탭에는 다음과 같은 섹션이 포함됩니다.

속성	설명
유사 항목 일치 키	유사 항목 일치 키의 속성입니다.
일치 열	일치 열 및 해당 속성: <ul style="list-style-type: none">- 필드 이름- 열 유형- 경로 구성 요소- 소스 테이블 - 기본 개체 또는 경로 구성 요소에서 참조된 테이블(경로 구성 요소가 루트인 경우)
일치 열 내용	일치를 위해 선택된 열과 기본 개체에서 사용 가능한 열의 목록입니다.

유사 항목 일치 키 속성 구성

이 섹션에서는 유사 항목 일치 기본 개체의 일치 열 속성을 구성하는 방법에 대해 설명합니다. *유사 항목 일치 키*는 일치 열에 유사 항목 일치/검색 전략이 사용되는 경우 스키마 관리자가 추가하는 기본 개체의 특수 열입니다. 이 열은 검색 및 일치 도중에 해당 기본 개체에 대한 일치 후보를 생성하기 위해 사용되는 기본 필드입니다. 모든 유사 항목 일치 기본 개체에는 유사 항목 일치 키가 하나만 있습니다.

키 유형

*일치 키 유형*은 열과 관련된 중요 특성을 Informatica MDM Hub에 알려 줍니다. Informatica MDM Hub에는 이름 및 주소 정보를 인식하는 몇 가지 기능이 있으므로, 이러한 정보를 사용하면 Informatica MDM Hub가 키를 올바르게 생성하고 향상된 검색을 수행할 수 있습니다. 이것이 잠재적 일치 후보의 초기 목록을 작성하는 검색의 기본 조건입니다. 이 키 유형은 유사 항목 일치 키를 구성하는 실제 열에 있는 기본 데이터 유형을 기반으로 해야 합니다.

유사 항목 일치 기본 개체의 경우 다음 키 유형 중 하나를 선택할 수 있습니다.

키 유형	설명
Person_Name	유사 항목 일치 키가 개인에 대한 데이터만 포함하는 경우에 사용됩니다.
Organization_Name	유사 항목 일치 키가 조직에 대한 데이터만 포함하거나 조직과 개인 모두에 대한 데이터를 포함하는 경우에 사용됩니다.
Address_Part1	유사 항목 일치 키가 통합할 주소 데이터를 포함하는 경우에 사용됩니다.

참고: 키 유형은 선택된 인구집단을 기반으로 합니다. 위의 키 유형 목록이 기본 인구집단(US)에 적용됩니다. 그 외 인구집단의 키 유형은 각기 다를 수 있습니다. 다른 인구집단이 필요한 경우 Informatica 지원부에 문의하십시오.

키 너비

*일치 키 너비*는 유사 항목 일치 키의 분석 완전성, 반환되는 가능한 일치 후보의 수 및 키가 사용하는 디스크 공간의 양을 결정합니다. 키 너비는 유사 항목 일치 개체에만 적용됩니다.

키 너비	설명
표준	신뢰도와 공간 사용량 간에 균형을 유지하며, 대부분의 유사 항목 일치 키에 적합합니다.
확장됨	더 많은 일치 후보를 얻을 수 있지만 키 생성에 더 긴 처리 시간이 걸립니다. 이 옵션은 열 연결과 관련된 몇 가지 추가적인 일치 기능을 제공합니다. 다음과 같은 경우에 이 키 너비를 사용하는 것이 좋습니다. <ul style="list-style-type: none"> - 데이터 집합이 아주 크기 않은 경우 - 데이터 집합이 완성되지 않은 경우 - 처리 시간과 디스크 공간 요구 사항을 해결할 수 있는 충분한 리소스가 있는 경우
제한됨	일치의 신뢰도를 다소 낮춰 디스크 공간을 절약합니다. 이 옵션을 사용하면 일치 후보 수가 줄어들지만 검색 속도는 빨라집니다. 키에 대해 더 적은 디스크 공간을 사용하는 보다 빠른 검색을 위해 부족 일치를 허용하려는 경우 이 옵션을 사용할 수 있습니다. 제한적 키를 사용하면 표준 키에 비해 더 적은 수의 단어 순서 변형을 포함하는 레코드가 일치됩니다. 이 옵션은 표준 키 집합의 하위 집합을 제공하지만 디스크 공간이 제한적이거나 데이터 양이 엄청나게 큰 경우에 최선의 선택일 수 있습니다.
기본 설정	기본 개체 레코드 하나당 키 하나를 생성합니다. 이 옵션은 일치의 신뢰도를 낮춰 수행해야 하는 일치의 수를 줄임으로써 성능을 높이고, 일치 키 테이블의 크기를 줄여 디스크 공간을 절약합니다. 데이터의 특성에 따라 기본 설정 키 너비에서 일치 후보 수가 더 적을 수 있습니다.

유사 항목 일치 키 속성의 구성 단계

유사 항목 일치 기본 개체에 대해 유사 항목 일치 키 속성을 구성하려면

1. 스키마 관리자에서 일치 열 탭으로 이동합니다.
2. 쓰기 잠금을 획득합니다.

- 이 유사 항목 일치 기본 개체에 대해 다음과 같은 설정을 구성합니다.

속성	설명
키 유형	일치에 주로 사용되는 필드 유형입니다. 이것이 잠재적 일치 후보의 초기 목록을 작성하는 검색의 기본 조건입니다. 이 키 유형은 기본 개체에 저장된 기본 데이터 유형에 기준해야 합니다.
키 너비	키가 생성된 검색 범위 크기입니다.
경로 구성 요소	이 유사 항목 일치 키의 경로 구성 요소입니다. 키 유형: 기본 개체, 하위 기본 개체 테이블 또는 교차 참조 테이블로 지정할 열을 포함한 테이블입니다.

- 저장** 단추를 클릭하여 변경 내용을 저장합니다.

유사 항목 일치 기본 개체에 대한 유사 항목 일치 열 추가

유사 항목 일치 기본 개체에 대한 유사 항목 일치 열을 정의하려면

- 스키마 관리자에서 일치 열 탭으로 이동합니다.
- 쓰기 잠금을 획득합니다.
- 유사 항목 일치 열을 추가하려면 **추가** 단추를 클릭합니다.
스키마 관리자에서 "유사 항목 일치 열 추가" 대화 상자가 표시됩니다.
- 다음 설정을 지정합니다.

속성	설명
일치 경로 구성 요소	이 유사 항목 일치 열에 대한 일치 경로 구성 요소입니다. 유사 항목 일치 열의 경우 소스 테이블은 상위 테이블, 상위 상호 참조 테이블 또는 하위 기본 개체 테이블일 수 있습니다.
필드 이름	필드의 이름입니다. 일치 열에 대한 데이터 유형을 선택합니다.

- 유사 항목 일치에 대한 기본 개체 열을 지정합니다.
선택한 열 목록에 열을 추가하려면 열 이름을 선택하고 오른쪽 화살표 단추를 클릭합니다.
참고: 여러 열을 추가할 경우 값이 연결되고 이렇게 연결된 값 사이에 구분 기호 공백이 추가됩니다.
- 확인**을 클릭합니다.
스키마 관리자의 일치 열 목록에 일치 열이 추가됩니다.
- 저장** 단추를 클릭하여 변경 내용을 저장합니다.

유사 항목 일치 기본 개체에 대한 정확히 일치 열 추가

유사 항목 일치 기본 개체에 대한 정확히 일치 열을 정의하려면

- 스키마 관리자에서 **일치 열** 탭으로 이동합니다.
- 쓰기 잠금을 획득합니다.
- 정확히 일치 열을 추가하려면 **추가** 단추를 클릭합니다.
스키마 관리자에서 "정확히 일치 열 추가" 대화 상자가 표시됩니다.

4. 다음 설정을 지정합니다.

속성	설명
일치 경로 구성 요소	이 정확히 일치 열에 대한 일치 경로 구성 요소입니다. 정확히 일치 열의 경우 소스 테이블은 상위 테이블 및/또는 하위 실제 열일 수 있습니다.
필드 이름	Hub 콘솔에서 표시하려는 필드에 대한 이름입니다.

5. 정확히 일치에 대한 기본 개체 열을 지정합니다.
6. **선택한 열** 목록에 열을 추가하려면 열 이름을 선택하고 오른쪽 화살표 단추를 클릭합니다.
- 참고:** 여러 열을 추가할 경우 값이 연결되고 이렇게 연결된 값 사이에 구분 기호 공백이 추가됩니다. 정확히 일치 열에 대해서는 열을 연결하지 마십시오.
7. **확인**을 클릭합니다.
- 스키마 관리자의 일치 열 목록에 일치 열이 추가됩니다.
8. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

유사 항목 일치 기본 개체에 대한 일치 열 속성 편집

일치 열 속성을 편집하는 대신 다음을 수행해야 합니다.

- 일치 열 삭제
- 새 일치 열 추가

유사 항목 일치 기본 개체에 대한 일치 열 삭제

유사 항목 일치 기본 개체에 대한 일치 열을 삭제하려면

1. 스키마 관리자에서 **일치 열** 탭으로 이동합니다.
 2. 쓰기 잠금을 획득합니다.
 3. 일치 열 목록에서 삭제하려는 일치 열을 선택합니다.
 4. **삭제** 단추를 클릭합니다.
- 스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.
5. **예**를 클릭합니다.
 6. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

정확히 일치하는 기본 개체에 대한 일치 열 구성

일치 열 규칙을 정의하려면 먼저 일치 열 규칙의 기준이 되는 일치 열을 정의해야 합니다. 정확히 일치 기본 개체에는 정확히 일치 열만 포함될 수 있습니다. 유사 항목 기본 개체에 대한 일치 열을 추가하는 방법에 대한 자세한 내용은 [“유사 항목 일치 기본 개체에 대한 일치 열 구성” 페이지 392](#)을 참조하십시오.

정확히 일치하는 기본 개체에 대한 일치 열 탭으로 이동

정확히 일치하는 기본 개체에 대해 일치 열을 정의하려면

1. 스키마 관리자에서 구성하려는 정확히 일치 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.

2. **일치 열** 탭을 클릭합니다.

스키마 관리자에 정확히 일치하는 기본 개체에 대한 [일치 열] 탭이 표시됩니다.

정확히 일치하는 기본 개체의 [일치 열] 탭에는 다음과 같은 섹션이 포함됩니다.

속성	설명
일치 열	일치 열 및 해당 속성: <ul style="list-style-type: none">- 필드 이름- 열 유형- 경로 구성 요소- 소스 테이블 - 기본 개체 또는 경로 구성 요소에서 참조된 테이블(경로 구성 요소가 루트인 경우)
일치 열 내용	일치를 위해 선택된 열 및 사용 가능한 열 목록

정확히 일치하는 기본 개체에 대한 일치 열 추가

정확히 일치 기본 개체의 경우 정확히 일치 열만 추가할 수 있습니다. 유사 항목 일치 열은 허용되지 않습니다.

정확히 일치 기본 개체에 대한 정확히 일치 열을 추가하려면

1. 스키마 관리자에서 **일치 열** 탭으로 이동합니다.
2. 쓰기 잠금을 획득합니다.
3. 정확히 일치 열을 추가하려면 **추가** 단추를 클릭합니다.
스키마 관리자에 "정확히 일치 열 추가" 대화 상자가 표시됩니다.
4. 다음 설정을 지정합니다.

속성	설명
일치 경로 구성 요소	해당 정확히 일치 열에 대한 일치 경로 구성 요소입니다. 정확히 일치 열의 경우 소스 테이블은 상위 테이블 및/또는 하위 실제 열일 수 있습니다.
필드 이름	Hub 콘솔에 표시되는 필드 이름입니다.

5. 정확히 일치에 사용할 기본 개체 열을 지정합니다.
6. 선택한 열 목록에 열을 추가하려면 열 이름을 선택하고 오른쪽 화살표를 클릭합니다.

참고:

- 여러 열을 추가할 경우 값이 연결되고 값 사이에 구분 기호로 공백이 추가됩니다.
- 정확히 일치 열의 경우에는 열을 연결하지 않는 것이 좋습니다.

7. **확인**을 클릭합니다.
스키마 관리자의 일치 열 목록에 선택한 일치 열이 추가됩니다.
8. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

정확히 일치하는 기본 개체에 대한 일치 열 속성 편집

정확히 일치 열에 루트 경로가 아닌 경로가 있는 경우 일치 열을 **기본 일치 하위 유형**으로 설정할 수 있습니다. 일치 규칙 집합에서 유사 항목 일치 규칙을 이 정확히 일치 열과 함께 추가하면 **일치 하위 유형** 옵션이 선택됩니다.

참고: 다른 모든 속성을 편집하려면 일치 열을 삭제하고 저장한 다음 업데이트된 속성을 사용하여 일치 열을 추가합니다.

1. **일치 열** 탭에서 다음 특성을 가진 일치 열을 선택합니다.
 - **열 유형** = 정확히
 - **경로 구성 요소** = <루트를 제외한 모든 경로>
2. **편집**을 클릭합니다.
3. **기본 일치 하위 유형**을 선택하고 **확인**을 클릭합니다.
4. **저장**을 클릭합니다.

관련 항목:

- [“일치 하위 유형” 페이지 411](#)

정확히 일치하는 기본 개체에 대한 일치 열 삭제

정확히 일치하는 기본 개체의 일치 열을 삭제하려면

1. 스키마 관리자에서 **일치 열** 탭으로 이동합니다.
2. 쓰기 잠금을 획득합니다.
3. 일치 열 목록에서 삭제하려는 일치 열을 선택합니다.
4. **삭제** 단추를 클릭합니다.
스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.
5. **예**를 클릭합니다.
6. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

일치 규칙 집합

일치 규칙 집합은 몇 가지 공통적인 속성이 있는 일치 열 규칙의 논리적 컬렉션입니다.

일치 규칙 집합은 일치 열 규칙에만 연결되고 기본 키 일치 규칙에는 연결되지 않습니다.

일치 규칙 집합을 사용하면 시점에 따라 서로 다른 일치 열 규칙 집합을 실행할 수 있습니다. 일치 프로세스가 실행될 때마다 하나의 일치 규칙 집합만 사용됩니다. 서로 다른 일치 규칙 집합을 사용하여 일치 항목을 찾으려면 일치 규칙 집합을 선택하고 일치 프로세스를 다시 실행합니다.

구성한 일치 규칙 집합은 **Data Director**에서 확장된 쿼리를 생성하는 데 사용됩니다. 일치 규칙 집합을 수정하거나 삭제하면 해당 일치 규칙 집합을 사용하는 쿼리가 삭제됩니다.

참고: 레코드 간의 일치를 선언하려면 일치 규칙 집합의 일치 열 규칙이 하나만 성공해야 합니다.

일치 규칙 집합이 지정하는 사항

일치 규칙 집합에는 다음이 포함됩니다.

- 검색 전략을 지정하는 검색 수준
- 자동 및 수동 일치 열 규칙의 수

- 일치 프로세스 중에 일치 일괄 처리에서 레코드를 선택적으로 포함하거나 제외하는 데 사용할 수 있는 필터 (선택 사항)

여러 일치 규칙 집합 및 지정된 기본값

규칙 집합은 원하는 만큼 구성할 수 있습니다.

사용자는 일치 일괄 작업을 실행할 때 기본 개체에 대해 정의된 규칙 집합 목록에서 하나의 규칙 집합을 선택합니다.

스키마 관리자에서 하나의 일치 규칙 집합을 기본값으로 지정합니다.

일치 규칙 집합을 사용하는 경우

일치 규칙 집합을 사용하면 시점에 따라 서로 다른 일치 열 규칙 요구 사항을 충족할 수 있습니다.

예를 들어 초기 데이터 로드를 위한 일치 규칙 집합과 후속 증분 로드를 위한 일치 규칙 집합을 각각 따로 만들어 사용할 수 있습니다. 마찬가지로 모든 레코드를 처리하기 위한 일치 규칙 집합과 레코드의 하위 집합만 처리하도록 필터를 포함하는 일치 규칙 집합을 각각 따로 만들어 사용할 수 있습니다.

규칙 집합 평가

일치 규칙 집합에 일치 규칙 집합의 일치 규칙에 대한 변경 내용을 비롯하여 모든 변경 내용을 저장하기 전에 스키마 관리자는 일치 규칙 집합을 분석하고 일치 규칙 집합에 문제가 있는 경우 사용자에게 경고 메시지를 표시합니다.

참고: 이 메시지는 단순한 경고 메시지입니다. 메시지를 무시하고 변경 내용을 저장하도록 선택할 수 있습니다.

문제 예로 다음과 같은 일치 규칙 집합을 들 수 있습니다.

- 기존의 일치 규칙 집합과 같은 일치 규칙 집합
- 일치 열 규칙이 추가되지 않은 빈 일치 규칙 집합
- 유사 항목 일치 기본 개체에 대해 유사 항목 일치 열을 포함하지 않은 일치 규칙 집합
- 하나 이상의 유사 항목 일치 열을 포함하지만 정확히 일치 열이 없는(일치 성능에 영향을 줄 수 있음) 일치 규칙 집합
- 같은 소스 열을 가진 유사 항목 일치 열 및 정확히 일치 열을 포함한 일치 규칙 집합

일치 규칙 집합 속성

이 섹션에서는 일치 규칙 집합의 속성에 대해 설명합니다.

이름

규칙 집합의 이름입니다. 고유한 설명 이름을 지정합니다.

검색 수준

유사 항목 일치 기본 개체에만 사용됩니다. 일치 규칙 집합을 구성할 경우 후보 일치 항목을 검색하는 데 있어 얼마나 엄격하게 그리고 철저하게 검색할지 Informatica MDM Hub에 지시하는 검색 수준을 정의하게 됩니다.

일치 프로세스의 목적은 데이터에 대해 최적의 일치 항목 수를 찾는 것입니다.

- 너무 적으면(이름: 부족 일치) 안 됨 - 관련 일치 항목 누락

- 너무 많으면(이름: *과도 일치*) 안 됨 - 관련되지 않은 일치 항목을 비롯한 너무 많은 일치 항목 생성

유사 항목 일치 키의 이름이나 주소의 경우 **Informatica MDM Hub**는 가능한 일치 후보 레코드 및 일치 열 규칙이 적용되는 레코드를 결정할 목적으로 정의된 검색 수준을 사용하여 다양한 키 범위를 생성합니다.

다음과 같은 검색 수준 중 하나를 선택할 수 있습니다.

검색 수준	설명
한정됨	가능한 일치 후보를 검색하는 데 있어 가장 엄격한 수준입니다. 이 검색 수준은 속도는 빠르지만 다른 검색 수준이 생성할 수 있는 일치 항목 수보다 적은 수의 일치 항목을 반환하며 경우에 따라 부족 일치가 야기될 수 있습니다. [한정됨]은 데이터가 상대적으로 정확하고 완전한 경우 또는 고도로 일치되는 데이터를 포함한 대량의 데이터 집합의 경우 적합합니다.
표준	대부분의 규칙 집합에 적합합니다.
포괄적	표준 수준보다 더 많은 일치 후보 집합을 생성합니다. 이 수준은 다른 검색 수준이 생성할 수 있는 일치 항목 수보다 많은 수의 일치 항목을 반환하며 경우에 따라 과도 일치가 야기되어 시간이 오래 걸릴 수 있습니다. 이 수준은 덜 완전한 소량의 데이터 집합에 적합합니다.
최대	다른 수준에 비해 훨씬 많은 일치 후보 집합을 생성하며, 이로 인해 과도 일치가 야기되어 시간이 매우 오래 걸릴 수 있습니다. 이 수준은 덜 완전한 소량의 데이터 집합이나 매우 많은 일치 레코드를 식별할 때 적합합니다.

선택하는 검색 수준은 데이터 집합 크기, 시간 제약 및 일치 항목의 중요도 등에 따라 결정되어야 합니다. 사용자 상황 및 요구 사항에 따라 부족 일치가 더 적합한 경우도 있고 과도 일치가 더 적합한 경우도 있습니다. 상대적으로 신뢰도가 높고 완전한 데이터를 처리하는 구현에서는 [한정됨] 수준을 사용할 수 있으며, 신뢰도가 떨어지거나 보다 중요한 문제를 처리하는 구현에서는 [포괄적] 또는 [최대] 수준을 사용해야 합니다.

검색 수준은 프로젝트 단계에 따라 다를 수도 있습니다. 초기 일치의 경우 완화된 수준([포괄적] 또는 [최대])을 사용하고 데이터의 중복이 제거되면 엄격한 수준을 사용해야 할 수 있습니다.

규칙별 검색 활성화

규칙별 검색 활성화에 대해 일치 규칙 집합을 사용하도록 설정하면 **SearchMatch** API에만 배타적으로 사용하도록 이 특정 일치 규칙 집합을 예약할 수 있습니다. 규칙별 검색 활성화 일치 규칙 집합을 사용하려면

SearchMatch MatchType이 **NONE**이어야 하고 **SearchMatch** 요청에 일치 규칙 집합이 지정되어야 합니다. 규칙별 검색 활성화를 사용하면 **MDM Hub**에서 유사 항목 일치만 사용합니다.

일부 필드가 비어 있는 상태로 검색을 수행하려는 경우가 있습니다. **SearchMatch**에서 규칙별 검색 활성화 일치 규칙 집합으로 검색을 수행하는 경우 검색 요청에서 제공한 비어 있는 필드가 무시됩니다. 이렇게 하면 검색 결과에 관련된 레코드가 제외되는 상황을 방지할 수 있습니다.

SearchMatch에 대한 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

참고: 유사 항목 일치 기본 개체에 대해 정확히 일치를 수행하기 위해 선택적으로 **Hub** 서버 속성을 활성화하면 일치 규칙에 영향을 미칩니다. 정확히 일치 및 유사 항목 일치에 대해 동일한 소스 열을 사용하면 **MDM Hub**에서 일치 결과를 반환하지 않습니다. 이 문제를 피하기 위해 **cmxserver.properties** 파일에 있는 **cmx.server.match.exact_match_fuzzy_bo_api** 속성을 비활성화해야 합니다. **Hub** 서버 속성에 대한 자세한 내용은 *Multidomain MDM 구성 가이드*를 참조하십시오.

필터링 활성화

이 일치 규칙 집합에 대해 필터링을 활성화할지 여부를 지정합니다.

- 선택된 경우 이 일치 규칙 집합에 대해 필터를 정의할 수 있습니다. 일치 작업을 실행할 때 필터가 정의되어 있는 일치 규칙 집합을 선택하여 일치 작업에서 필터 조건을 충족하는 레코드 하위 집합만 처리하게 만들 수 있습니다.
- 선택되지 않은 경우 일치 일괄 작업을 실행하면 일치 규칙 집합에서 모든 레코드가 처리됩니다.

예를 들어 **Organization** 기본 개체에 여러 조직 유형(고객, 공급업체, 잠재 고객, 파트너 등)이 포함된 경우 일치에 사용할 레코드 유형만 선택적으로 처리하는 다양한 일치 규칙 집합을 정의할 수 있습니다. 예를 들어 **MatchAll**(필터 없음), **MatchCustomersOnly**, **MatchVendorsOnly** 등을 정의할 수 있습니다.

참고: 일치 규칙 집합에는 필터를 사용할 수 있지만 하위 기본 개체 및 그룹 함수에는 필터를 사용할 수 없습니다.

필터링 SQL

기본적으로 일치 일괄 작업이 실행되면 일치 규칙 집합이 모든 레코드를 처리합니다.

필터링 활성화와 확인란이 선택되어 있으면 필터 조건을 지정하여 해당 조건을 만족하는 규칙만 처리하도록 제한할 수 있습니다. *필터*는 SQL 문의 **WHERE** 절과 비슷합니다. 필터 식은 해당 데이터베이스 플랫폼에서 사용되는 **WHERE** 절 구문에 대해 유효한 식이면 어느 식이나 가능합니다.

참고: 일치 규칙 집합 필터는 일치 풀의 레코드(일치시킬 대상 레코드)가 아닌 *일치 일괄 처리*(일치시킬 소스 레코드)에 대해서만 선택된 기본 개체 레코드에 적용됩니다.

예를 들어 구현 환경에 여러 가지 유형의 조직(고객, 공급업체, 잠재 고객, 파트너 등)이 포함된 조직 기본 개체가 있다고 가정해 봅니다. 필터를 사용하여 고객 데이터만 처리하는 일치 규칙 집합(**MatchCustomersOnly**)을 정의할 수 있습니다.

```
org.type='C'
```

다른 비고객 레코드는 일치 작업에서 무시되고 처리되지 않습니다.

참고: 일치 작업 중에 레코드를 올바르게 필터링하는 적절한 SQL 문을 지정하는 것은 관리자의 책임입니다. 스키마 관리자는 해당 데이터베이스 플랫폼에 따라 SQL 구문의 유효성을 검사하지만 필터 조건의 논리나 적합성을 검사하지는 않습니다.

일치 규칙

이 창 영역에는 선택한 일치 규칙 집합에 대해 구성된 일치 열 규칙 목록이 표시됩니다.

일치 규칙 집합 탭으로 이동

일치 규칙 집합 탭으로 이동하려면

1. 스키마 관리자에서 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. **일치 규칙 집합** 탭을 클릭합니다.

스키마 관리자에 선택한 기본 개체에 대한 일치 규칙 집합 탭이 표시됩니다.

3. 일치 규칙 집합 탭은 다음과 같은 섹션으로 구성됩니다.

섹션	설명
일치 규칙 집합	구성된 일치 규칙 집합 목록입니다.
속성	선택한 일치 규칙 집합의 속성입니다.

일치 규칙 집합 추가

새 일치 규칙 집합을 추가하려면

- 스키마 관리자에서 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자의 **일치 규칙 집합** 탭을 표시합니다.
- 쓰기 잠금을 획득합니다.
- 추가** 단추를 클릭합니다.
스키마 관리자에서 일치 규칙 집합 추가 대화 상자가 표시됩니다.
- 이 새 일치 규칙 집합의 고유한 설명 이름을 입력합니다.
- 확인**을 클릭합니다.
스키마 관리자의 목록에 새 일치 규칙 집합이 추가됩니다.
- 일치 규칙 집합을 구성합니다.

일치 규칙 집합 속성 편집

일치 규칙 집합의 속성을 편집하려면

- 스키마 관리자에서 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자의 **일치 규칙 집합** 탭을 표시합니다.
- 쓰기 잠금을 획득합니다.
- 구성할 일치 규칙 집합을 선택합니다.
스키마 관리자의 속성 패널에 해당 속성이 표시됩니다.
- 이 일치 규칙 집합의 속성을 구성합니다.
- 이 일치 규칙 집합의 일치 열을 구성합니다.
- 저장** 단추를 클릭하여 변경 내용을 저장합니다.
스키마 관리자는 변경 내용을 저장하기 전에 일치 규칙 집합을 분석하고, 일치 규칙 집합에 포함된 항목이 적합하지 않을 경우 메시지를 표시합니다.
- 변경 내용을 저장할지 묻는 메시지가 표시되면 **확인** 단추를 클릭하여 변경 내용을 저장합니다.

일치 규칙 집합 이름 바꾸기

일치 규칙 집합 이름을 바꾸려면

- 스키마 관리자에서 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자의 **일치 규칙 집합** 탭을 표시합니다.
- 쓰기 잠금을 획득합니다.
- 이름을 바꿀 일치 규칙 집합을 선택합니다.

4. **편집** 단추를 클릭합니다.
스키마 관리자에 [규칙 집합 이름 편집] 대화 상자가 표시됩니다.
5. 이 일치 규칙 집합에 대해 고유한 설명 이름을 지정합니다.
6. **확인**을 클릭합니다.
스키마 관리자가 목록에서 일치 규칙 집합의 이름을 업데이트합니다.

일치 규칙 집합 삭제

일치 규칙 집합을 삭제하려면

1. 스키마 관리자에서 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자의 **일치 규칙 집합** 탭을 표시합니다.
2. 쓰기 잠금을 획득합니다.
3. 삭제할 일치 규칙 집합의 이름을 선택합니다.
4. **삭제** 단추를 클릭합니다.
스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.
5. **예**를 클릭합니다.
스키마 관리자가 삭제된 일치 규칙 집합과 함께 해당 일치 규칙 집합에 포함된 모든 일치 열 규칙을 목록에서 제거합니다.

일치 규칙 집합에 대한 일치 열 규칙 구성

*일치 열 규칙*은 일치 프로세스 중 일치를 구성하는 항목을 결정합니다.

일치 열 규칙은 두 레코드가 통합될 수 있을 만큼 유사한지 여부를 확인합니다. 각 일치 규칙은 유사점을 검사하는 데 필요한 하나 이상의 일치 열을 포함하는 집합으로 정의됩니다. 일치 규칙을 구성하려면 소스 시스템 내에서 서와 소스 시스템 간에 일치하는 레코드를 식별하기 위한 조건을 설정합니다.

일치 열 규칙을 구성하기 위한 선행 조건

다음 선행 조건을 수행한 후에만 일치 열 규칙을 구성할 수 있습니다.

- 일치 규칙에 사용할 열 구성
- 하나 이상의 일치 규칙 집합 생성

정확하게 일치하는 기본 개체와 유사 항목 일치 기본 개체 간의 일치 열 규칙 차이점

정확히 일치하는 기본 개체와 유사 항목 일치 기본 개체 간의 일치 열 규칙의 속성은 다릅니다.

- 정확히 일치하는 기본 개체의 경우 정확히 일치 열 유형만 구성할 수 있습니다.
- 유사 항목 일치 기본 개체의 경우에는 유사 항목 일치 또는 정확히 일치 열 유형을 구성할 수 있습니다.

각 일치 열 규칙에 대해 일치된 레코드를 자동으로 통합할지 수동으로 통합할지 여부를 결정합니다.

일치된 레코드에 대한 통합 옵션 지정

각 일치 열 규칙에 대해 일치된 레코드를 자동으로 통합할지 수동으로 통합할지 여부를 결정합니다.

유사 항목 일치 기본 개체만을 위한 일치 규칙 속성

이 섹션에서는 유사 항목 일치 기본 개체의 일치 규칙 속성에 대해 설명합니다. 이 속성은 정확히 일치 기본 개체에는 적용되지 않습니다.

일치/검색 전략

유사 항목 일치 기본 개체의 경우 일치/검색 전략은 일치 프로세스에서 레코드 검색 및 일치에 사용되는 전략을 정의합니다. 일치/검색 전략은 유사 항목 일치 또는 정확히 일치 옵션을 사용하여 후보 A가 후보 B와 일치되는 방식을 결정합니다.

참고: 유사 항목 일치 기본 개체에 대한 자세한 내용은 [“일치/검색 전략” 페이지 378](#) 항목을 참조하십시오.

일치/검색 전략은 일치 후보의 양과 질에 영향을 줄 수 있습니다. 정확히 일치 전략을 사용하려면 정리된 완벽한 데이터가 필요합니다. 데이터가 정리되지 않았거나 데이터가 완전하지 않은 경우 일치 프로세스에서 일부 일치 항목이 누락될 수 있습니다. 유사 항목 전략에서는 더 많은 일치 항목을 찾지만 많은 항목이 중복 항목이 아닐 수 있습니다. 일치 규칙 속성을 정의할 때는 가능한 모든 후보를 찾는 것과 관련이 없는 후보를 피하는 것 사이에서 최적의 균형을 찾아야 합니다.

다음 전략 옵션 중 하나를 선택합니다.

일치/검색 전략 옵션	설명
유사 항목 전략	철자 변형, 가능한 철자 오류 및 기타 차이점을 고려하는 확률 일치입니다.
정확한 항목 전략	동일한 레코드를 일치시키는 정확히 일치하는 항목입니다.

모든 유사 항목 일치 기본 개체에는 일치 경로 구성 요소에 지정한 열을 기준으로 생성되는 유사 항목 일치 키가 있습니다. 유사 항목 전략을 사용하는 경우 일치 프로세스는 일치 키 인덱스를 검색하여 일치 후보를 식별합니다. 일치 프로세스에서 후보를 식별하려면 일치 규칙에 유사 항목 일치 키의 열을 하나 이상 지정해야 합니다. 예를 들어 일치 키가 이름 열을 기준으로 생성되는 경우 일치 규칙에 이름 열이 포함되어야 합니다.

다음 테이블에는 일치 규칙의 유형, 일치/검색 전략 및 각 조합에 대한 설명이 나열되어 있습니다.

일치 규칙 유형	사용되는 일치/검색 전략	설명
유사 항목 일치 규칙	유사 항목 전략	유사 항목 일치 열이 포함되어 있으며 정확히 일치 열이 포함되어 있을 수 있습니다. 유사 항목 일치 규칙은 철자가 잘못된 이름과 같이 데이터 변형이 포함된 열에 권장됩니다.
정확히 일치 규칙	정확한 항목 전략	정확히 일치 열만 포함합니다. 일치 프로세스는 데이터베이스 행에서 SQL 문을 실행하여 일치 항목을 식별합니다. 정확히 일치 규칙은 유사 항목 일치 기능이 필요하지 않은 ID, 생년월일과 같은 열에 가장 적합합니다.
필터링된 일치 규칙	유사 항목 전략	정확히 일치 열만 포함합니다. 규칙은 연결된 일치 키를 기반으로 필터링된 일치 후보에 대해 실행됩니다. 일치 프로세스는 일치 키 인덱스를 검색하여 일치 후보를 식별합니다. 정확히 일치 열 중 하나에 유사 항목 일치 키와 동일한 소스 열이 있는 경우 필터링된 일치 규칙을 사용하는 것이 좋습니다. 필터링된 일치 규칙은 데이터베이스 대신 처리 서버에서 정확히 일치 규칙을 실행하여 성능을 향상시킵니다.

팁: 데이터베이스 서버와 관련된 성능 문제로 제약이 있을 경우 정확히 일치 규칙 대신 필터링된 항목 일치 규칙을 사용해 보십시오. 필터링된 항목 일치 규칙을 사용하면 정확히 일치 규칙에서보다 더 큰 일괄 처리를 실행할 수 있습니다. 또한 필터링된 일치에서 일괄 처리 크기가 크게 증가할 경우 일치 작업 시간이 상대적으로 근소하게 늘어납니다.

일치 목적

유사 항목 일치 기본 개체의 경우 *일치 목적*은 일치 규칙에 부속된 주목적을 정의합니다. 예를 들어 두 개의 레코드가 동일인에 대한 주소인지 여부를 결정하는 데 있어 주소가 결정적 요소인 사람에 대한 일치 항목을 식별할 경우 "거주자"라고 하는 일치 목적을 선택합니다.

정의하는 모든 일치 규칙에 대해 Informatica에서 제공하는 미리 정의된 일치 목적 목록에서 규칙 목적을 선택해야 합니다. 각 일치 목적에는 일치 목적을 달성하기 위해 두 개의 레코드를 최적으로 비교하는 방법에 대한 정보가 포함됩니다. Informatica MDM Hub에서는 일치 규칙을 적용하는 기준으로 선택한 일치 목적을 사용하여 일치하는 레코드를 결정합니다. 규칙의 동작은 선택한 목적에 따라 다릅니다. 사용 가능한 일치 목적 목록은 사용된 인구집단에 따라 다릅니다.

일치 목적이 결정하는 사항

일치 목적은 다음 사항을 결정합니다.

- 일치 규칙의 동작 방식
- 필요한 열
- Informatica MDM Hub에서 일치 프로세스에 사용되는 각 열에 기울여야 하는 주의 수준

목적은 제외한 모든 특성이 동일한 두 규칙이 있을 경우 각 규칙은 목적이 서로 다르기 때문에 서로 다른 일치 집합을 반환합니다.

필수 필드 및 선택적 필드

각 일치 목적은 필수 필드와 선택적 필드의 조합을 지원합니다. 각 필드에는 일치 결정에 미치는 필드의 영향력에 따라 가중치가 적용됩니다. 몇 가지 목적에 따라 일부 필드를 그룹화할 수 있습니다. 이러한 그룹화에는 두 가지 유형이 있습니다.

- 필수 - 필드 멤버 중 적어도 하나가 null이 아니어야 합니다.
- 최적 - 그룹에 속한 필드의 최고 점수만 전체 일치 점수에 제공됩니다.

예를 들어 개별 일치 목적의 경우에는 다음과 같습니다.

- Person_Name은 필수 필드입니다.
- ID 번호 또는 생년월일 중 하나가 필요합니다.
- 기타 특성은 선택적입니다.

각 목적에서 반환되는 전체 점수는 참여하는 필드 점수에 해당 가중치를 곱하고 모든 필드 가중치의 합계로 나눈 값을 더하여 계산합니다. 필드가 선택적이고 제공되지 않으면 가중치 계산에 포함되지 않습니다.

이름 형식

Informatica MDM Hub 일치에는 성이 필요한 위치를 알려주는 기본 이름 형식이 포함됩니다. 옵션은 다음과 같습니다.

- 왼쪽 - 성이 전체 이름의 시작 부분에 있습니다(예: Smith Jim).
- 오른쪽 - 성이 전체 이름의 끝에 있습니다(예: Jim Smith).

Informatica MDM Hub가 사용하는 이름 형식은 사용하는 목적에 따라 다릅니다. 조직을 사용하는 경우 기본값은 성, 이름, 중간 이름입니다. 개인 또는 거주자를 사용하는 경우 기본값은 이름, 중간 이름, 성입니다.

참고: 일치시킬 데이터를 포맷할 때 Informatica MDM Hub가 사용하는 이름 형식이 사용하는 목적에 따라 다르다는 점을 기억해야 합니다. 특별한 경우, 특히 선택된 인구집단에 없는 이름의 경우 선택하는 이름 형식과 일치 목적이 차이를 만듭니다.

일치 목적 목록

다음 테이블에서는 Informatica가 제공하는 일치 목적에 대해 설명합니다.

일치 목적	설명
Person_Name	<p>이름별로 개인을 식별합니다. 이 목적은 이름만 사용하는 조회가 필요하고 사람이 결정을 내릴 수 있는 온라인 검색에 사용합니다. 일치에는 일치 항목을 결정하는 데 이름과 함께 다른 특성이 필요합니다. 이 목적을 사용할 때 규칙에 주소 필드를 포함하면 안 됩니다. 이 목적은 주소가 있는 사람과 주소가 없는 사람을 일치시킵니다. 규칙에 주소 필드가 포함된 경우에는 거주자 목적을 대신 사용합니다.</p> <p>이 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Person_Name(필수) - Address_Part1 - Address_Part2 - Postal_Area - Telephone_Number - ID - 날짜 - Attribute1 - Attribute2 - 좌표 부여 <p>Address_Part2와 Postal_Area 간에 최적의 점수를 얻으려면 Address_Part2 필드에서 Postal_Area를 반복 값으로 사용하십시오.</p>
개인	<p>이름, ID 번호 및 생년월일 특성별로 개인을 식별합니다.</p> <p>개인 일치 목적에는 Person_Name 일치 목적에서 제공하는 추가 정보가 필요하므로 Person_Name별 검색 후 이 일치 목적을 사용합니다.</p> <p>이 일치 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Person_Name(필수) - ID(필수) - 날짜(필수) - Attribute1 - Attribute2
거주자	<p>주소로 개인을 식별합니다. 이 일치 목적은 Person_Name 또는 Address_Part1별 검색 후 사용합니다. 더 많은 정보를 사용할 수 있는 경우 선택적 입력 필드를 사용하여 일치를 한정하거나 순위를 지정할 수 있습니다.</p> <p>Address_Part2와 Postal_Area 간에 최적의 점수를 얻으려면 Address_Part2 필드에서 Postal_Area를 반복 값으로 전달하십시오.</p> <p>이 일치 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Person_Name(필수) - Address_Part1(필수) - Address_Part2 - Postal_Area - Telephone_Number - ID - 날짜 - Attribute1 - Attribute2 - 좌표 부여

일치 목적	설명
가정	<p>성이 동일하거나 유사한 여러 개인이 동일한 주소를 공유하는 경우 일치 항목을 식별합니다.</p> <p>이 일치 목적은 Address_Part1별 검색 후 사용합니다.</p> <p>참고: Person_Name별 검색은 결과적으로 Person_Name의 한 단어와 일치해야 하고 한 단어 검색은 대부분의 상황에서 효과적이지 않기 때문에 실용적이지 않습니다.</p> <p>Person_Name 필드의 주요 단어인 성은 일치에서 레코드를 표현하는 방식 중에 단어 순서가 중요시되는 흔치 않은 사례 중 하나이기 때문에 특히 강조됩니다.</p> <p>하지만 한 이름의 주요 단어와 다른 이름의 또 다른 단어 사이에서 일치가 발생하는 경우에만 합당한 점수가 생성됩니다.</p> <p>이 일치 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Person_Name(필수) - Address_Part1(필수) - Address_Part2 - Postal_Area - Telephone_Number - Attribute1 - Attribute2 - 좌표 부여 <p>Address_Part2와 Postal_Area 간에 최적의 점수를 얻으려면 Address_Part2 필드에서 Postal_Area를 반복 값으로 전달하십시오.</p>
가족	<p>성이 동일하거나 유사한 여러 개인이 동일한 주소 또는 동일한 전화 번호를 공유하는 경우 일치 항목을 식별합니다.</p> <p>이 일치 목적은 Address_Part1 및 Telephone_Number별 계층화된 검색(다중 검색) 후에 사용합니다.</p> <p>참고: Person_Name별 검색은 결과적으로 Person_Name의 한 단어와 일치해야 하고 한 단어 검색은 대부분의 상황에서 효과적이지 않기 때문에 실용적이지 않습니다.</p> <p>Person_Name 필드의 주요 단어인 성은 일치에서 레코드를 표현하는 방식 중에 단어 순서가 중요시되는 흔치 않은 사례 중 하나이기 때문에 특히 강조됩니다.</p> <p>하지만 한 이름의 주요 단어와 다른 이름의 또 다른 단어 사이에서 일치가 발생하는 경우에만 합당한 점수가 생성됩니다.</p> <p>이 일치 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Person_Name(필수) - Address_Part1(필수) - Telephone_Number(필수) (점수는 최적의 Address_Part_1 및 Telephone_Number에 기 반함) - Address_Part2 - Postal_Area - Attribute1 - Attribute2 - 좌표 부여 <p>Address_Part2와 Postal_Area 간에 최적의 점수를 얻으려면 Address_Part2 필드에서 Postal_Area를 반복 값으로 전달하십시오.</p>

일치 목적	설명
Wide_Household	<p>성이 동일하거나 전화 번호가 동일한 여러 개인이 동일한 주소를 공유하는 경우 일치 항목을 식별합니다.</p> <p>이 일치 목적은 Address_Part1별 검색 후 사용됩니다.</p> <p>참고: Person_Name별 검색은 결과적으로 Person_Name의 한 단어와 일치해야 하고 한 단어 검색은 대부분의 상황에서 효과적이지 않기 때문에 실용적이지 않습니다.</p> <p>Person_Name 필드의 주요 단어인 성은 일치에서 레코드를 표현하는 방식 중에 단어 순서가 중요시되는 흔치 않은 사례 중 하나이기 때문에 특히 강조됩니다.</p> <p>하지만 한 이름의 주요 단어와 다른 이름의 또 다른 단어 사이에서 일치가 발생하는 경우에만 합당한 점수가 생성됩니다.</p> <p>이 일치 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Address_Part1(필수) - Person_Name(필수) - Telephone_Number(필수) (점수는 최적의 Person_Name 및 Telephone_Number에 기 반함) - Address_Part2 - Postal_Area - Attribute1 - Attribute2 - 좌표 부여 <p>Address_Part2와 Postal_Area 간에 최적의 점수를 얻으려면 Address_Part2 필드에서 Postal_Area를 반복 값으로 전달하십시오.</p>
Address	<p>주소 일치를 식별합니다. 주소는 우편 번호, 거주지, 배송지, 설명 형식, 공식 또는 비공식 주소 일 수 있습니다.</p> <p>필수 필드는 Address_Part1입니다. Address_Part2, Postal_Area, Telephone_Number, ID, Date, Attribute1 및 Attribute2 필드는 주소를 세부적으로 구분하기 위한 선택적 입력 필드입니다. 예를 들어 구/군/시 또는 시/도 이름이 Address_Part2로 제공되는 경우 이 필드를 사용하여 서로 다른 위치의 일반 주소인 [100번지]를 구분할 수 있습니다.</p> <p>이 일치 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Address_Part1(필수) - Address_Part2 - Postal_Area - Telephone_Number - ID - 날짜 - Attribute1 - Attribute2 - 좌표 부여 <p>Address_Part2와 Postal_Area 간에 최적의 점수를 얻으려면 Address_Part2에서 Postal_Area를 반복 값으로 전달하십시오. 이 경우 사용되는 Address_Part2 점수는 점수가 지정된 두 필드 중에서 더 높은 점수입니다.</p>

일치 목적	설명
Organization	<p>기본적으로 이름으로 조직을 일치시킵니다. 이 목적은 이름만 사용하는 조회가 필요하고 사람이 결정을 내릴 수 있는 온라인 검색을 위한 것입니다. 일괄 처리의 일치에서는 일치 항목을 결정하는 데 이름과 함께 다른 특성이 필요합니다. 이 일치 목적은 규칙에 주소 필드가 포함되지 않은 경우에 사용합니다. 이 일치 목적에서는 주소가 있는 조직과 주소가 없는 조직 간의 일치가 허용됩니다. 규칙에 주소 필드가 포함된 경우에는 부서 일치 목적을 사용합니다.</p> <p>이 일치 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Organization_Name(필수) - Address_Part1 - Address_Part2 - Postal_Area - Telephone_Number - ID - 날짜 - Attribute1 - Attribute2 - 좌표 부여 <p>Address_Part2와 Postal_Area 간에 최적의 점수를 얻으려면 Address_Part2 필드에서 Postal_Area를 반복 값으로 전달하십시오.</p>
부서	<p>주소로 조직을 식별합니다. 이 일치 목적은 Organization_Name이나 Address_Part1 또는 둘 모두로 검색한 후에 사용합니다.</p> <p>이 일치 목적은 Address_Part1이 필수 필드라는 것을 제외하면 조직의 일치 목적과 근본적으로 동일합니다. 예를 들어 이 일치 목적은 여러 개의 주소가 제공된 경우 주소가 Y인 회사 X 또는 주소가 W인 회사 Z의 일치 항목을 찾도록 설계되었습니다.</p> <p>이 일치 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Organization_Name(필수) - Address_Part1(필수) - Address_Part2 - Postal_Area - Telephone_Number - ID - Attribute1 - Attribute2 - 좌표 부여 <p>Address_Part2와 Postal_Area 간에 최적의 점수를 얻으려면 Address_Part2 필드에서 Postal_Area를 반복 값으로 전달하십시오.</p>
연락처	<p>조직 내에서 특정 위치의 연락처를 식별합니다.</p> <p>이 일치 목적은 Person_Name별 검색 후 사용합니다. 하지만 Organization_Name이나 Address_Part1을 검색 조건으로 사용할 수도 있습니다.</p> <p>이 일치 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Person_Name(필수) - Organization_Name(필수) - Address_Part1(필수) - Address_Part2 - Postal_Area - Telephone_Number - ID - 날짜 - Attribute1 - Attribute2 - 좌표 부여 <p>Address_Part2와 Postal_Area 간에 최적의 점수를 얻으려면 Address_Part2 필드에서 Postal_Area를 반복 값으로 전달하십시오.</p>

일치 목적	설명
Corporate_Entity	<p>INC, LTD 등과 같은 법인 접미사를 비롯한 법인 이름으로 조직을 식별합니다. 이 일치 목적은 ABC TRADING INC와 ABC TRADING LTD 같은 이름을 구분해야 하는 상황을 위해 만들어졌습니다.</p> <p>이 일치 목적은 Organization_Name별 검색 후 사용합니다. 보다 엄격한 일치를 수행하고 법인 접미사를 노이즈로 간주하지 않는다는 것을 제외하면 이 일치 목적은 조직의 일치 목적과 근본적으로 동일합니다.</p> <p>이 일치 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Organization_Name(필수) - Address_Part1 - Address_Part2 - Postal_Area - Telephone_Number - ID - Attribute1 - Attribute2 - 좌표 부여 <p>Address_Part2와 Postal_Area 간에 최적의 점수를 얻으려면 Address_Part2 필드에서 Postal_Area를 반복 값으로 전달하십시오.</p>
Wide_Contact	<p>위치를 고려하지 않고 조직 내에서 연락처를 대략적으로 식별합니다.</p> <p>이 일치 목적은 Person_Name별 검색 후 사용합니다.</p> <p>필수 필드와 함께 선택적으로 ID, Attribute1 및 Attribute2를 제공하여 일치에서 연락처를 추가로 한정할 수 있습니다.</p> <p>이 일치 목적에는 다음과 같은 필드가 사용됩니다.</p> <ul style="list-style-type: none"> - Person_Name(필수) - Organization_name(필수) - ID - Attribute1 - Attribute2
필드	<p>특정 용도가 아닌 범용 목적으로 제공됩니다. 이 일치 목적에는 필수 필드가 없습니다. 모든 필드 유형을 선택적 입력 필드로 사용할 수 있습니다.</p>

일치 수준

유사 항목 일치 기본 개체의 경우 일치 수준에 따라 일치의 정확성이 결정됩니다. 유사 항목 일치 기본 개체에 대해 다음 일치 수준 중 하나를 지정할 수 있습니다.

테이블 2. 일치 수준

수준	설명
표준	대부분의 일치에 적절합니다.
보수적	표준 수준보다 더 적은 항목이 일치 항목으로 검색됩니다. 일부 데이터의 경우 실제로 일치해도 일치 항목으로 플래그가 지정되지 않고 일치 프로세스를 통과할 수 있습니다. 이러한 상황을 <i>부족 일치</i> 라고 합니다.
느슨함	표준 수준보다 더 많은 항목이 일치 항목으로 검색됩니다. 느슨함 일치에서는 실제로 일치 항목이 아닌 상당한 수의 일치 후보가 일치하는 것으로 검색될 수 있습니다. 이러한 상황을 <i>과도 일치</i> 라고 합니다. 수동 병합에 대한 일치 규칙에서 이 수준을 사용하여 더욱 엄격한 다른 일치 규칙에서 잠재적인 일치 항목을 누락하지 않도록 할 수도 있습니다.

일치할 데이터에 대한 지식을 바탕으로 표준, 보수적(더 적은 일치 항목) 또는 느슨함(더 많은 일치 항목) 중에 수준을 선택합니다. 확실하지 않으면 표준을 사용합니다.

좌표 부여 반지름

지정된 반지름 내에 있는 레코드를 식별하려고 할 때 좌표 부여 반지름을 유사 항목 일치 기본 개체에 대한 일치 규칙 속성으로 사용합니다. 일치 열에 대한 좌표 부여 필드 이름을 선택하면 좌표 부여 반지름 필드가 활성화됩니다.

미터 단위의 좌표 부여 반지름을 지정합니다. 좌표 부여 반지름은 **C_REPOS_MATCH_RULE** 테이블의 **GEOCODE_RADIUS** 열에 저장되어 있습니다. 레코드가 지정한 좌표 부여 반지름에 근접하면 일치 점수가 높습니다. 레코드가 지정한 좌표 부여 반지름으로부터 멀리 있으면 일치 점수가 감소합니다.

허용 한도 조정

유사 항목 일치 기본 개체의 경우 **허용 한도**는 일치 항목의 허용 가능성을 결정하는 수입입니다. 이 설정은 일치 수준과 동일하지만 보다 세부적인 제한이 가능합니다. 허용 한도는 Informatica에 의해 인구집단 내에 일치 목적에 따라 정의됩니다. 허용 한도 조정을 통해 이 일치 규칙에 대해 일치 항목으로 간주할 항목을 대략적으로 조정할 수 있습니다.

- 양수로 조정하면 좀 더 보수적인 일치가 수행됩니다.
- 음수로 조정하면 좀 더 느슨한 일치가 수행됩니다.

예를 들어 특정 필드에서 특정 인구집단을 사용할 때 표준 일치 수준, 느슨한 일치 수준 및 보수적 일치 수준의 허용 한도가 각각 80, 70, 90이라고 가정합니다. 조정 값으로 양수(예: 3)를 지정하면 허용 수준이 약간 더 보수적이 됩니다. 음수(예: -2)를 지정하면 허용 수준이 좀 더 느슨해집니다.

이 설정을 구성하면 일치 설정을 선택적으로 세밀하게 조정할 수 있으므로 일부 경우에 유용합니다. 허용 한도를 조금이라도 조정하면 일치에 큰 영향을 주게 되어 과도 일치 또는 부족 일치를 초래할 수 있습니다. 따라서 조정 값을 조금씩 늘리면서 다양한 설정을 반복적으로 테스트해 보고 데이터에 최적의 설정을 결정하는 것이 좋습니다.

일치 규칙에 대한 일치 열 속성

이 섹션에서는 일치 규칙에 대해 구성할 수 있는 일치 열 속성을 설명합니다.

일치 하위 유형

서로 다른 유형의 데이터를 포함한 기본 개체의 경우 **일치 하위 유형** 옵션을 통해 같은 기본 개체 내 특정 데이터 유형에 일치 규칙을 적용할 수 있습니다. 상위/하위 경로 구성 요소를 포함한 **정확히 일치 열**에 대해 일치 하위 유형을 활성화하거나 비활성화할 수 있습니다. 일치 하위 유형은 다음의 경우에만 사용할 수 있습니다.

- 루트 이외의 경로 구성 요소를 기준으로 하는 정확히 일치 열 유형
- 유사 항목 일치/검색 전략을 포함한 일치 규칙

일치 하위 유형을 사용하려면 각 일치 규칙에 대해 사용할 "하위 유형" 열 역할을 하는 하나 이상의 **정확히 일치 열**을 지정합니다. 정확히 일치 열이 세그먼트 일치에 사용되는지 여부에 상관없이 이 열에 대해 하위 유형 표시기를 설정할 수 있습니다. 일치 프로세스 동안 하위 유형 열 평가는 다른 일치 열의 평가보다 먼저 수행됩니다. 일치 하위 유형은 신중하게 사용해야 합니다. 이 유형을 사용할 경우 일치 프로세스의 성능에 영향을 줄 수 있기 때문입니다.

일치 하위 유형은 일치 하위 유형으로 표시된 일치 열이 일치 중인 모든 레코드에 대해 같아야 하는 추가 요구 사항을 포함한 표준 상위/하위 일치 시나리오처럼 작동합니다. 다음 예에서 일치 하위 유형 열은 주소 유형이고 일치 규칙은 주소 행 1, 구/군/시, 시/도로 구성됩니다.

상위 ID	주소 행 1	구/군/시입니다.	상태	주소 유형
3	123 Main	NYC	ON	청구
3	50 John St	토론토	NY	배송
5	123 Main	토론토	BC	청구
5	20 Adelaide St	마크햄	AB	배송
5	50 John St	오타와	ON	청구
7	50 John St	배리	BC	청구
7	20 Adelaide St	토론토	NB	배송
7	90 Yonge St	토론토	ON	청구

일치 하위 유형이 없으면 상위 ID 3은 5 및 7과 일치합니다. 단, 일치 하위 유형이 있으면 상위 ID 3은 5 및 7과 일치하지 않습니다. 일치 행이 서로 다른 주소 유형에 걸쳐 분산되어 있기 때문입니다. 상위 ID 5와 7은 서로 일치합니다. 일치 행이 모두 '청구' 주소 유형에 포함되기 때문입니다.

관련 항목:

- [“정확히 일치하는 기본 개체에 대한 일치 열 속성 편집” 페이지 397](#)

NULL 일치

NULL 일치를 사용하여 null 값이 다른 null 값과 일치할 때 일치 프로세스에서 처리하는 방법을 지정합니다.

참고: Null 일치와 세그먼트 일치는 함께 사용할 수 없습니다. NULL 일치는 정확히 일치 열에만 적용됩니다.

기본적으로 Null 일치는 비활성화되어 있고 MDM Hub은 일치를 검색할 때 null을 같지 않은 값으로 처리합니다. Null 일치가 비활성화된 경우 Null 값이 다른 값과 일치하지 않습니다.

Null 일치를 활성화하려면 일치 열에 대해 다음 Null 일치 옵션 중 하나를 선택합니다.

- NULL이 NULL과 일치
- NULL이 NULL 아닌 값과 일치

NULL이 NULL과 일치

NULL이 NULL과 일치 옵션을 활성화하면 일치 프로세스에서 두 Null 값이 일치하는 것으로 간주됩니다.

Null 일치 시나리오에 따라 다음 값 쌍 중 하나가 일치로 간주됩니다.

- 두 셀 값이 모두 Null입니다.
- 두 셀의 값이 동일합니다.

NULL이 NULL 아닌 값과 일치

NULL이 NULL 아닌 값과 일치 옵션을 활성화하면 일치 프로세스에서 Null 값이 Null이 아닌 값과 일치하는 것으로 간주됩니다.

Null 일치 시나리오에 따라 다음 값 쌍이 일치로 간주됩니다.

- 한 셀의 값이 null 이고 다른 셀의 값이 null이 아닙니다.
- 두 셀의 값이 동일합니다.

중요: NULL이 NULL 아닌 값과 일치 옵션을 활성화하면 일치 그룹 빌드에서 Null이 아닌 일치에 대한 단일 Null 만 그룹에 허용하므로 불필요한 전이적 일치 가능성이 줄어듭니다. 동일한 값이 삭제되지 않도록 하려면 동일한 정확히 일치 규칙 2개를 생성합니다. 하나의 정확히 일치 규칙에서 NULL이 NULL 아닌 값과 일치 옵션을 활성화합니다. 다른 정확히 일치 규칙에서 NULL이 NULL 아닌 값과 일치 옵션을 비활성화합니다. 일치 규칙 실행 시퀀스에서 NULL이 NULL 아닌 값과 일치 옵션이 비활성화된 정확히 일치 규칙이 NULL이 NULL 아닌 값과 일치 옵션이 활성화된 정확히 일치 규칙보다 앞에 있는지 확인하십시오.

하나의 셀 값이 Null이 아닐 경우 Null이 Null과 일치 예제

조직에 병합 작업을 수행하려는 CUSTOMER 기본 개체가 있습니다. 정확히 일치 규칙에 대해 Null 값을 다른 Null 값과 일치하는 옵션을 활성화합니다. CUSTOMER 기본 개체에는 John Smith에 대한 레코드 3개가 포함되어 있는데 레코드 2개에는 내선 전화 번호에 Null 값이 있습니다.

John Smith의 기본 개체 레코드를 보면 레코드 3개 중 2개의 내선 전화 번호가 Null인 것을 알 수 있습니다.

Rowid_Object	개인 이름	내선	전화 번호
1	John Smith	6053128215	-
2	John Smith	6053128215	-
3	John Smith	6053128215	236

일치 및 병합 작업 후에 기본 개체를 보면 Null 값이 포함된 레코드가 병합된 것을 알 수 있습니다.

Rowid_Object	개인 이름	내선	전화 번호
1	John Smith	6053128215	-
3	John Smith	6053128215	236

일치 및 병합 작업 후에 교차 참조 테이블을 보면 Null 값이 있는 레코드 중 하나의 Rowid_Object와 Null이 아닌 값이 있는 레코드가 존속되는 것을 알 수 있습니다.

Rowid_XREF	Rowid_Object	개인 이름	내선	전화 번호
1	1	John Smith	6053128215	-
2	1	John Smith	6053128215	-
3	3	John Smith	6053128215	236

참고: 이 시나리오에서는 Null 값이 다른 Null 값과 일치하므로 두 레코드가 일치인지 여부를 확인할 수 없습니다. Null 값이 있는 레코드가 일치인지 여부를 확인하려면 이러한 레코드에 수동 병합 플래그가 지정되도록 MDM Hub를 구성합니다.

하나의 셀 값이 Null일 경우 Null이 Null 아닌 값과 일치 예제

조직에 병합 작업을 수행하려는 CUSTOMER 기본 개체가 있습니다. 정확히 일치 규칙에 대해 Null 값을 Null 아닌 값과 일치하는 옵션을 활성화합니다. CUSTOMER 기본 개체에는 John Smith에 대한 레코드가 포함되어 있고, 이 레코드 중 하나에는 내선 전화 번호에 Null 값이 있습니다.

John Smith의 기본 개체 레코드를 보면 그의 내선 전화 번호가 236 또는 Null임을 알 수 있습니다.

Rowid_Object	개인 이름	내선	전화 번호
1	John Smith	6053128215	236
2	John Smith	6053128215	-

일치 및 병합 작업 후 기본 개체에서 Null 값을 가진 레코드가 Null 아닌 값을 가진 레코드로 병합됨을 표시합니다.

Rowid_Object	개인 이름	내선	전화 번호
1	John Smith	6053128215	236

일치 및 병합 작업 후 교차 참조 테이블에서 Null 값을 가진 레코드가 병합되는 레코드의 Rowid_Object가 존속됨을 보여 줍니다.

Rowid_XREF	Rowid_Object	개인 이름	내선	전화 번호
1	1	John Smith	6053128215	236
2	1	John Smith	6053128215	-

참고: 이 시나리오에서는 Null 값이 다른 Null 값과 일치하므로 두 레코드가 일치인지 여부를 확인할 수 없습니다. Null 값과 Null이 아닌 값이 있는 레코드가 일치 항목인지 확인하려면 이러한 레코드에 수동 병합 플래그가 지정되도록 MDM Hub를 구성합니다.

두 셀 값이 동일한 경우 Null이 Null 아닌 값과 일치 예제

조직에 일치 및 병합 작업을 수행하려는 CUSTOMER 기본 개체가 있습니다. 정확히 일치 규칙에 대해 Null 값을 Null 아닌 값과 일치하는 옵션을 활성화합니다. CUSTOMER 기본 개체에 John Smith에 대한 레코드가 포함되어 있습니다. 두 레코드에서 내선 전화 번호는 값이 동일합니다.

John Smith의 기본 개체 레코드를 보면 그의 내선 전화 번호가 236임을 알 수 있습니다.

Rowid_Object	개인 이름	내선	전화 번호
1	John Smith	6053128215	236
2	John Smith	6053128215	236

일치 및 병합 작업 후 기본 개체에는 하나의 레코드가 표시됩니다.

Rowid_Object	개인 이름	내선	전화 번호
1	John Smith	6053128215	236

일치 및 병합 작업 후 Rowid_Object 1이 있는 레코드와 동일한 값이 포함된 Rowid_Object 2가 있는 레코드가 Rowid_Object 1 레코드로 병합됩니다.

일치 및 병합 작업 후 교차 참조 테이블을 보면 동일한 값을 가진 레코드가 병합되는 대상 레코드의 Rowid_Object가 존속되는 것을 알 수 있습니다.

Rowid_XREF	Rowid_Object	개인 이름	내선	전화 번호
1	1	John Smith	6053128215	236
2	1	John Smith	6053128215	236

동일한 값이 있는 다른 레코드로 병합되는 동일한 값을 가진 레코드에는 병합 대상 레코드의 Rowid_Object가 존속됩니다.

하나의 셀 값은 Null이고 다른 셀 값은 동일한 경우 Null이 Null 아닌 값과 일치 예제

조직에 일치 및 병합 작업을 수행하려는 CUSTOMER 기본 개체가 있습니다. 정확히 일치 규칙에 대해 Null 값을 Null 아닌 값과 일치하는 옵션을 활성화합니다. CUSTOMER 기본 개체에 John Smith에 대한 레코드가 포함되어 있습니다. 레코드 중 하나에 내선 전화 번호에 대한 Null 값이 있고 다른 레코드에는 동일한 값이 있습니다.

John Smith의 기본 개체 레코드를 보면 그의 내선 전화 번호가 236 또는 Null임을 알 수 있습니다.

Rowid_Object	개인 이름	내선	전화 번호
1	John Smith	6053128215	236
2	John Smith	6053128215	-
3	John Smith	6053128215	236

일치 및 병합 작업 후 기본 개체에서 두 개의 레코드를 표시합니다.

Rowid_Object	개인 이름	내선	전화 번호
1	John Smith	6053128215	236
3	John Smith	6053128215	236

일치 및 병합 작업 후 Rowid_Object 2가 있는 레코드는 Rowid_Object 1이 있는 레코드로 병합됩니다. 일치 그룹 빌드에서는 Null이 아닌 일치에 대한 단일 Null만 그룹에 허용하기 때문입니다. Rowid_Object 1이 포함된 레코드의 값과 동일한 값이 포함된 Rowid_Object 3이 있는 레코드는 병합되지 않습니다.

일치 및 병합 작업 후 교차 참조 테이블에서 Null 값을 가진 레코드가 병합되는 레코드의 Rowid_Object가 존속됨을 보여 줍니다.

Rowid_XREF	Rowid_Object	개인 이름	내선	전화 번호
1	1	John Smith	6053128215	236
2	1	John Smith	6053128215	-
3	3	John Smith	6053128215	236

Null 아닌 값을 가진 레코드에 병합되는 Null 값을 가진 레코드에는 자신이 병합되는 레코드의 Rowid_Object가 있습니다. 병합되지 않는 레코드의 Rowid_Object 값은 동일하게 유지됩니다.

참고: 이 시나리오에서는 Null 값이 Null이 아닌 값과 일치하므로 두 레코드가 일치인지 여부를 확인할 수 없습니다. Null 값과 Null이 아닌 값이 있는 레코드가 일치 항목인지 확인하려면 이러한 레코드에 수동 병합 플러그가 지정되도록 MDM Hub를 구성합니다. 또한 Rowid_Object 1 및 Rowid_Object 3이 있는 동일한 레코드가 일치인지 확인하려면 동일한 정확히 일치 규칙 2개를 생성합니다. 하나의 정확히 일치 규칙에서 NULL이 NULL 아닌 값과 일치 옵션을 활성화합니다. 다른 정확히 일치 규칙에서 NULL이 NULL 아닌 값과 일치 옵션을 비활성화합니다. 일치 규칙 실행 시퀀스에서 NULL이 NULL 아닌 값과 일치 옵션이 비활성화된 정확히 일치 규칙이 NULL이 NULL 아닌 값과 일치 옵션이 활성화된 정확히 일치 규칙보다 앞에 있는지 확인하십시오.

같지 않은 일치

일치 규칙에서 같지 않은 일치 옵션을 사용하면 열에서 일치하는 값이 서로 일치하지 않도록 할 수 있습니다. 같지 않은 일치는 정확히 일치 열에만 적용됩니다.

참고: 같지 않은 일치와 세그먼트 일치는 함께 사용할 수 없습니다. 두 항목 중 하나가 선택된 경우 나머지 항목은 선택할 수 없습니다.

같지 않은 일치 옵션은 일치 방지 옵션으로 생각하면 됩니다. 옵션이 활성화된 경우의 일치 결과는 옵션이 비활성화된 경우의 일치 결과와 반대입니다.

같지 않은 일치를 사용하여 레코드를 유사 조직 이름 열 및 정확히 같지 않은 조직 유형 열과 일치시킬 수 있습니다. 조직 이름이 같더라도 조직 유형이 같지 않은 경우에만 레코드가 일치합니다.

NULL 일치가 없는 같지 않은 일치

먼저 NULL 일치가 비활성화된 경우 같지 않은 일치 옵션의 결과를 고려하십시오. 다음 테이블에서 볼 수 있듯이 NULL 값은 절대 일치하지 않으며 같은 값들이 일치합니다. 같지 않은 일치 옵션이 활성화되면 반대로 적용됩니다. 모든 NULL 값은 일치하고 같은 값들은 일치하지 않습니다.

다음 테이블은 같지 않은 일치이 활성화되기 전후의 단순 일치 결과를 보여 줍니다.

값	같지 않은 일치=False	같지 않은 일치= True
- 레코드 1 = NULL - 레코드 2 = "Fred"	일치하지 않음	일치
- 레코드 1 = NULL - 레코드 2 = NULL	일치하지 않음	일치
- 레코드 1 = "Bill" - 레코드 2 = "Fred"	일치하지 않음	일치
- 레코드 1 = "Fred" - 레코드 2 = "Fred"	일치	일치하지 않음

NULL이 NULL과 일치하는 같지 않은 일치

마찬가지로 NULL이 NULL과 일치하는 같지 않은 일치 옵션을 활성화하면 일치 결과가 반대 결과로 전환됩니다.

다음 테이블은 NULL이 NULL과 일치하는 같지 않은 일치이 활성화되기 전후의 일치 결과를 보여 줍니다.

값	같지 않은 일치=False NULL이 NULL과 일치=True	같지 않은 일치= True NULL이 NULL과 일치=True
- 레코드 1 = NULL - 레코드 2 = "Fred"	일치하지 않음	일치
- 레코드 1 = NULL - 레코드 2 = NULL	일치	일치하지 않음
- 레코드 1 = "Bill" - 레코드 2 = "Fred"	일치하지 않음	일치
- 레코드 1 = "Fred" - 레코드 2 = "Fred"	일치	일치하지 않음

NULL이 NULL 아닌 값과 일치하는 같지 않은 일치

NULL이 NULL 아닌 값과 일치하는 같지 않은 일치 옵션을 활성화하면 일치 결과가 반대 결과로 전환됩니다. 한 가지 상황에서 NULL 값을 가진 레코드가 일치하지 않습니다. 설명은 아래 테이블을 참조하십시오.

다음 테이블은 NULL이 NULL 아닌 값과 일치하는 같지 않은 일치이 활성화되기 전후의 일치 결과를 보여 줍니다.

값	같지 않은 일치=False NULL이 NULL 아닌 값과 일치=True	같지 않은 일치= True NULL이 NULL 아닌 값과 일치=True
- 레코드 1 = NULL - 레코드 2 = "Fred"	일치	일치하지 않음
- 레코드 1 = NULL - 레코드 2 = NULL	일치하지 않음	일치*

값	같지 않은 일치=False NULL이 NULL 아닌 값과 일치=True	같지 않은 일치= True NULL이 NULL 아닌 값과 일치=True
- 레코드 1 = "Bill" - 레코드 2 = "Fred"	일치하지 않음	일치
- 레코드 1 = "Fred" - 레코드 2 = "Fred"	일치	일치하지 않음

* 일치 규칙으로 정확히 필터링됨을 사용하면 결과는 일치입니다. 하지만 일치 규칙으로 정확히를 사용하면 결과는 일치하지 않습니다.

세그먼트 일치

참고: 세그먼트 일치와 같지 않은 일치는 함께 사용할 수 없습니다. 두 항목 중 하나가 선택된 경우 나머지 항목은 선택할 수 없습니다. 세그먼트 일치와 NULL 일치도 함께 사용할 수 없습니다. 두 항목 중 하나가 선택된 경우 나머지 항목은 선택할 수 없습니다.

정확히 일치 열에 한해 *세그먼트 일치*를 사용하여 일치 규칙을 특정 데이터 하위 집합으로 제한할 수 있습니다. 예를 들어 서로 다른 국가의 고객에 대해 서로 다른 일치 규칙을 정의하여 세그먼트 일치를 통해 특정 규칙을 특정 국가 코드로 제한할 수 있습니다. 세그먼트 일치는 정확히 일치 기본 개체 및 유사 항목 일치 기본 개체 둘 다에 적용됩니다.

[세그먼트 일치] 확인란이 선택된 경우 또 다른 두 개의 옵션 즉 [세그먼트가 모든 데이터와 일치] 및 [세그먼트 일치 값]을 구성할 수 있습니다.

세그먼트가 모든 데이터와 일치

선택 취소(기본값)되면 Informatica MDM Hub가 세그먼트 일치 값에 정의된 값 집합 내 레코드와만 일치 작업을 수행합니다.

예를 들어 기본 개체에 잠재 고객, 파트너, 고객 및 공급자가 포함된다고 가정하면 세그먼트 일치 값에 잠재 고객 및 파트너 값이 포함되고 [세그먼트가 모든 데이터와 일치]가 선택 취소된 경우 Informatica MDM Hub는 잠재 고객이나 파트너를 포함하는 레코드 내에서만 일치 작업을 수행합니다. 고객이나 공급자 레코드는 모두 무시됩니다.

[세그먼트가 모든 데이터와 일치]가 선택된 경우에는 잠재 고객 및 파트너가 고객 및 공급자와 일치됩니다. 단, 고객 및 공급자 서로가 일치하지는 않습니다.

여러 열의 값 연결

연결된 열에 대해 세그먼트 일치가 활성화된 상태에서 정확히 일치하는 항목의 경우 연결된 필드에 있는 각 데이터 부분에 공백 문자를 추가해야 합니다.

참고: 정확히 일치 열의 경우에는 열을 연결하지 않는 것이 좋습니다.

세그먼트 일치 값

세그먼트 일치의 경우 세그먼트 일치 작업 시 사용할 세그먼트 값 목록을 지정합니다.

일치 열에 대해 세그먼트 일치를 정의하는 하나 이상의 값을 지정해야 합니다. 예를 들어 지정된 일치 규칙에 대해 성별에 따라 세그먼트 일치를 정의한다고 가정해 봅시다. 세그먼트 일치 값으로 M(남성의 경우)을 지정한 경우 해당 일치 규칙에 대해 Informatica MDM Hub가 다른 일치 열을 바탕으로 남성 레코드에 대해서만 일치 항목

을 검색하며 [세그먼트가 모든 데이터와 일치]를 활성화한 경우가 아니라면 또 다른 남성 레코드와만 일치시킬 수 있습니다.

참고: 세그먼트 일치 값은 대소문자를 구분합니다. 유사 항목 및 정확한 기본 개체에 대해 세그먼트 일치를 사용할 경우 일치 일괄 작업을 실행하면 설정한 값이 대소문자를 구분합니다.

일치 열 규칙의 정확히 일치 열에 대한 요구 사항

정확히 일치 열에는 다음과 같은 규칙이 적용됩니다.

- 정확히 일치 열 이름은 26자를 초과할 수 없습니다.
- 정확히 일치 열 유형은 VARCHAR 또는 CHAR여야 합니다.
- 일치 열은 기본 개체의 텍스트 열 조합이나 모든 텍스트 열에서 일치 작업을 수행하는 데 사용될 수 있습니다.
- 숫자나 날짜를 사용하려면 숫자나 날짜를 기본 개체에 로드하기 전에 정리 함수를 사용하여 VARCHAR로 변환해야 합니다.
- 일치 열은 하위 기본 개체의 열에서 일치 작업을 수행하는 데에도 사용할 수 있으며, 하위 기본 개체의 텍스트 열 조합이나 모든 텍스트 열에 기준할 수 있습니다. 하위 기본 개체의 일치 열에 대한 일치치는 테이블 간 일치라고 합니다.
- 테이블 간 일치를 사용하고 외래 키를 통해 하위 테이블에 대한 일치 규칙을 생성할 경우 하위의 각 일치 규칙에 상위 테이블의 외래 키를 포함해야 합니다. 그렇지 않을 경우 하위가 병합되면 상위 레코드에서 이전에 이 레코드에 속한 하위 레코드가 손실됩니다.

열 일치 규칙을 구성하기 위한 명령 단추

일치 규칙 집합 탭의 목록에서 일치 규칙 집합을 선택하면 스키마 관리자에 다음과 같은 명령 단추가 표시됩니다.

단추	설명
	일치 규칙을 추가합니다.
	선택한 일치 규칙의 속성을 편집합니다.
	선택한 일치 규칙을 삭제합니다.
	선택한 일치 규칙을 순서에서 위로 이동합니다.
	선택한 일치 규칙을 순서에서 아래로 이동합니다.
	수동 통합 규칙을 자동 통합 규칙으로 변경합니다. 수동 통합 레코드를 선택하고 이 단추를 클릭합니다.
	자동 통합 규칙을 수동 통합 규칙으로 변경합니다. 자동 통합 레코드를 선택하고 이 단추를 클릭합니다.

참고: 일치 후에 일치 규칙을 변경하면 일치 항목을 재설정하라는 메시지가 표시됩니다. 일치 항목을 재설정하면 일치 테이블과 통합 표시기가 2인 레코드에서 모든 항목이 삭제되고 통합 표시기는 4로 재설정됩니다.

일치 열 규칙 추가

유사 항목 일치 규칙, 정확히 일치 규칙 또는 필터링된 일치 규칙을 추가할 수 있습니다.

일치 열을 사용하여 정확히 일치 규칙 또는 유사 항목 일치 규칙을 새로 추가하려면

1. 스키마 관리자에서 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. 쓰기 잠금을 획득합니다.
3. **일치 규칙 집합** 탭을 클릭합니다.
4. 목록에서 일치 규칙 집합을 선택합니다.

스키마 관리자에 선택한 일치 규칙 집합의 속성이 표시됩니다.

5. 화면의 일치 규칙 섹션에서 **추가** 단추를 클릭합니다.

스키마 관리자에 일치 규칙 편집 대화 상자가 표시됩니다. 이 대화 상자는 정확히 일치 기본 개체의 경우와 유사 항목 일치 기본 개체의 경우에 약간 다르게 나타납니다.

6. 유사 항목 일치 기본 개체의 경우 대화 상자 위쪽에서 일치 규칙 속성을 구성합니다.
7. 일치 규칙에 대한 일치 열을 구성합니다.

- a. 일치 열 목록 옆의 **편집** 단추를 클릭합니다.

일치 열 추가/제거 대화 상자가 나타나며 정의한 일치 열만 표시됩니다. 정확히 일치/검색 전략을 사용하는 정확히 일치 기본 개체나 일치 규칙의 경우 정확히 일치 열 유형만 사용할 수 있습니다. 유사 항목 일치 기본 개체의 경우 유사 항목 일치 또는 정확히 일치 열 유형을 선택할 수 있습니다.

- b. 일치 열 규칙에 추가할 필드를 선택합니다.

- c. **확인**을 클릭합니다.

스키마 관리자의 일치 열 목록에 선택한 필드가 표시됩니다.

8. 일치 열 목록의 각 일치 열에 대한 일치 속성을 구성합니다.
9. **확인**을 클릭합니다.
10. 정확히 일치의 경우 이 일치 규칙에 대한 일치 속성을 지정합니다. **확인**을 클릭합니다.
11. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

스키마 관리자는 변경 내용을 저장하기 전에 일치 규칙 집합을 분석하고, 일치 규칙 집합에 포함된 항목이 적합하지 않을 경우 메시지를 표시합니다.

12. 변경 내용을 저장할지 묻는 메시지가 표시되면 **확인** 단추를 클릭하여 변경 내용을 저장합니다.

필터링된 일치 규칙 구성

정확히 일치 열이 포함된 유사 항목 일치 기본 개체에 대해 필터링된 일치 규칙을 구성할 수 있습니다.

필터링된 일치 규칙을 구성해야 하는 유사 항목 일치 기본 개체에는 정확히 일치 규칙이 이미 추가되어 있어야 합니다.

정확히 일치 열을 사용하여 필터링된 일치 규칙을 추가하려면

1. 스키마 관리자에서 구성하려는 유사 항목 일치 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. 쓰기 잠금을 획득합니다.
3. **일치 규칙 집합** 탭을 클릭합니다.
4. 정확히 일치 규칙 유형의 일치 규칙을 선택하고 **편집** 단추를 클릭합니다.

일치 규칙 편집 대화 상자가 나타납니다.

참고: 유사 항목 일치 규칙을 필터링된 일치 규칙으로 직접 변경할 수는 없습니다.

5. 일치/검색 전략 드롭다운 목록에서 **유사 항목**을 선택하고 **확인**을 클릭합니다.
일치 규칙 유형이 "필터링됨"으로 변경됩니다.
6. **저장** 단추를 클릭하여 변경 내용을 저장합니다.
스키마 관리자는 변경 내용을 저장하기 전에 일치 규칙 집합을 분석하고, 일치 규칙 집합에 포함된 항목이 적합하지 않을 경우 메시지를 표시합니다.
7. 변경 내용을 저장할지 묻는 메시지가 표시되면 **확인** 단추를 클릭하여 변경 내용을 저장합니다.

일치 열 규칙 편집

기존 일치 규칙에 대한 속성을 편집하려면

1. 스키마 관리자에서 구성하려는 정확히 일치 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. 쓰기 잠금을 획득합니다.
3. **일치 규칙 집합** 탭을 클릭합니다.
4. 목록에서 일치 규칙 집합을 선택합니다.
스키마 관리자에 선택한 일치 규칙 집합의 속성이 표시됩니다.
5. 화면의 일치 규칙 섹션에서 **편집** 단추를 클릭합니다.
스키마 관리자에 일치 규칙 편집 대화 상자가 표시됩니다. 이 대화 상자는 정확히 일치 기본 개체의 경우와 유사 항목 일치 기본 개체의 경우에 약간 다르게 나타납니다.
6. 유사 항목 일치 기본 개체에 대해 원하는 경우 대화 상자 위쪽에서 일치 규칙 속성을 변경합니다.
7. 원하는 경우 이 일치 규칙에 대한 일치 열을 구성합니다.
일치 열로 정의되어 있는 열만 표시됩니다.
 - 정확히 일치/검색 전략을 사용하는 정확히 일치 기본 개체나 일치 규칙의 경우 정확히 일치 열 유형만 사용할 수 있습니다.
 - 유사 항목 일치 기본 개체에 대해 유사 항목 또는 정확히 일치 열 유형을 선택할 수 있습니다.
 - a. 일치 열 목록 옆의 **편집** 단추를 클릭합니다.
스키마 관리자에 일치 열 추가/제거 대화 상자가 표시됩니다.
 - b. 포함할 열 옆에 있는 확인란을 선택합니다.
 - c. 생략할 열 옆의 확인란을 선택 취소합니다.
 - d. **확인**을 클릭합니다.
스키마 관리자의 일치 열 목록에 선택한 열이 표시됩니다.
8. 편집하려는 모든 일치 열에 대해 일치 속성을 변경합니다.
9. **확인**을 클릭합니다.
10. 정확히 일치/검색 전략의 경우 이 일치 규칙에 대한 일치 속성을 지정합니다. **확인**을 클릭합니다.
11. **저장** 단추를 클릭하여 변경 내용을 저장합니다.
스키마 관리자는 변경 내용을 저장하기 전에 일치 규칙 집합을 분석하고, 일치 규칙 집합에 포함된 항목이 적합하지 않을 경우 메시지를 표시합니다.
12. 변경 내용을 저장할지 묻는 메시지가 표시되면 **확인** 단추를 클릭하여 변경 내용을 저장합니다.

일치 열 규칙 삭제

일치 열 규칙을 삭제하려면

1. 스키마 관리자에서 구성하려는 정확히 일치 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. 쓰기 잠금을 획득합니다.
3. **일치 규칙 집합** 탭을 클릭합니다.
4. 목록에서 일치 규칙 집합을 선택합니다.
5. 일치 규칙 섹션에서 삭제할 일치 규칙을 선택합니다.
6. **삭제** 단추를 클릭합니다.
스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.
7. **예**를 클릭합니다.

일치 열 규칙의 실행 순서 변경

일치 열 규칙의 실행 순서를 변경하려면

1. 스키마 관리자에서 구성하려는 정확히 일치 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. 쓰기 잠금을 획득합니다.
3. **일치 규칙 집합** 탭을 클릭합니다.
4. 목록에서 일치 규칙 집합을 선택합니다.
5. 일치 규칙 섹션에서 위 또는 아래로 이동할 일치 규칙을 선택합니다.
6. 다음 작업 중 하나를 수행합니다.
 - 선택한 일치 규칙을 실행 순서에서 위로 이동하려면 **위로** 단추를 클릭합니다.
 - 선택한 일치 규칙을 실행 순서에서 아래로 이동하려면 **아래로** 단추를 클릭합니다.
7. **저장** 단추를 클릭하여 변경 내용을 저장합니다.
스키마 관리자는 변경 내용을 저장하기 전에 일치 규칙 집합을 분석하고, 일치 규칙 집합에 포함된 항목이 적합하지 않을 경우 메시지를 표시합니다.
8. 변경 내용을 저장할지 묻는 메시지가 표시되면 **확인** 단추를 클릭하여 변경 내용을 저장합니다.

일치 열 규칙에 대한 통합 옵션 지정

일치 프로세스 동안 일치 열 규칙은 일치된 레코드가 수동 통합을 위해 대기될지 자동 통합을 위해 대기될지 여부를 결정해야 합니다.

참고: 자동 통합에 대해 구성된 일치 규칙이 있는 경우에는 기본 개체에 200개 이상의 사용자 정의 열이 포함될 수 없습니다.

일치 규칙에 대한 수동 통합 및 자동 통합 간 전환하려면

1. 스키마 관리자에서 구성하려는 정확히 일치 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. 쓰기 잠금을 획득합니다.
3. **일치 규칙 집합** 탭을 클릭합니다.
4. 목록에서 일치 규칙 집합을 선택합니다.
5. 일치 규칙 섹션에서 구성할 일치 규칙을 선택합니다.


6. 다음 작업 중 하나를 수행합니다.
 - 위로 단추를 클릭하여 수동 통합 규칙을 자동 통합 규칙으로 변경합니다.
 - 아래로 단추를 클릭하여 자동 통합 규칙을 수동 통합 규칙으로 변경합니다.
7. 저장 단추를 클릭하여 변경 내용을 저장합니다.
스키마 관리자는 변경 내용을 저장하기 전에 일치 규칙 집합을 분석하고, 일치 규칙 집합에 포함된 항목이 적합하지 않을 경우 메시지를 표시합니다.
8. 변경 내용을 저장할지 묻는 메시지가 표시되면 **확인** 단추를 클릭하여 변경 내용을 저장합니다.

열의 일치 가중치 구성

유사 항목 일치 열의 경우 일치 규칙 편집 대화 상자에서 일치 가중치를 변경할 수 있습니다. Informatica MDM Hub에서는 각 열에 대해 내부 *일치 가중치*를 할당합니다. 일치 가중치는 일치 대상 열의 중요도를 테이블의 다른 열에 상대적인 값으로 나타낸 수입니다. 일치 가중치는 선택한 일치 목적 및 인구집단에 따라 달라집니다. 예를 들어 일치 목적이 **Person_Name**인 경우 Informatica MDM Hub에서는 일치 항목을 평가할 때 주소 등의 다른 열에 있는 데이터 일치 항목보다 중요도가 높은 이름 열의 데이터 일치 항목을 확인합니다.

열의 일치 가중치를 조정하여 해당 열에 더 높은 가중치를 할당하고, Informatica MDM Hub에서 일치 항목의 값이 분석될 때 다른 열과 비교한 해당 열의 상대적 중요성을 높일 수 있습니다.

열의 일치 가중치를 구성하려면

1. 일치 규칙 편집 대화 상자의 목록에서 열을 선택합니다.
2.  **일치 가중치 조정** 단추를 클릭합니다.
조정된 경우 선택한 열의 이름이 굵은 글꼴로 표시됩니다.
3. 저장 단추를 클릭하여 변경 내용을 저장합니다.
스키마 관리자는 변경 내용을 저장하기 전에 일치 규칙 집합을 분석하고, 일치 규칙 집합에 포함된 항목이 적합하지 않을 경우 메시지를 표시합니다.
4. 변경 내용을 저장할지 묻는 메시지가 표시되면 **확인** 단추를 클릭하여 변경 내용을 저장합니다.

열에 대한 세그먼트 일치 구성

세그먼트 일치는 정확히 일치 열에 사용되어 일치 규칙을 특정 데이터 하위 집합으로 제한합니다.

정확히 일치 열에 대한 세그먼트 일치를 구성하려면

1. 일치 규칙 편집 대화 상자의 일치 열 목록에서 정확히 일치 열을 선택합니다.
2. **세그먼트 일치** 확인란을 선택하여 이 기능을 활성화합니다.
3. 필요한 경우 **세그먼트가 모든 데이터와 일치** 확인란을 선택합니다.
4. 세그먼트 일치에 대한 세그먼트 일치 값을 지정합니다.
 - a. **편집** 단추를 클릭합니다.
스키마 관리자에 값 편집 대화 상자가 표시됩니다.
 - b. 다음 작업 중 하나를 수행합니다.
 - 값을 추가하려면 **추가** 단추를 클릭하고 추가할 값을 입력한 다음 **확인**을 클릭합니다.
 - 값을 삭제하려면 목록에서 값을 선택하고 **삭제** 단추를 클릭한 다음 삭제 확인 메시지가 표시될 때 **예**를 선택합니다.
5. **확인**을 클릭합니다.
6. 저장 단추를 클릭하여 변경 내용을 저장합니다.

스키마 관리자는 변경 내용을 저장하기 전에 일치 규칙 집합을 분석하고, 일치 규칙 집합에 포함된 항목이 적합하지 않을 경우 메시지를 표시합니다.

7. 변경 내용을 저장할지 묻는 메시지가 표시되면 **확인** 단추를 클릭하여 변경 내용을 저장합니다.

기본 키 일치 규칙 구성

이 섹션에서는 Informatica MDM Hub 구현의 기본 키 일치 규칙을 구성하는 방법에 대해 설명합니다.

이 방법 대신 일치 열 일치 규칙을 구성하려면 [“일치 열 구성” 페이지 389](#)의 지침을 참조하십시오.

기본 키 일치 규칙 정보

기본 키에 대한 일치는 기본 개체에 대한 두 개 이상의 서로 다른 소스 시스템에 동일한 기본 키 값이 있는 경우에 사용할 수 있습니다.

소스 시스템에서 이러한 경우는 드물게 발생하기는 하지만, 이러한 경우가 발생할 경우 Informatica MDM Hub의 기본 키 일치 옵션을 사용하여 일치하는 기본 키가 있는 소스 시스템의 레코드를 빠르게 일치시키고 자동으로 통합할 수 있습니다.

예를 들어 두 시스템에서 동일한 고객 ID 집합을 사용할 수 있습니다. 두 시스템 모두에서 동일한 기본 키 값을 사용하여 고객 XYZ123에 대한 정보를 제공할 경우 두 시스템은 분명히 동일한 고객을 참조하고 있으며 레코드는 자동으로 통합되어야 합니다.

기본 키 일치를 지정하는 경우에는 동일한 기본 키 값이 있는 소스 시스템을 지정하기만 하면 됩니다. 또한 Informatica MDM Hub에서 병합 또는 링크 일괄 작업을 실행할 때 일치하는 레코드를 자동으로 통합하도록 하려면 **일치하는 레코드 자동 병합** 확인란을 선택합니다.

기본 키 일치 규칙 추가

새 기본 키 일치 규칙을 추가하려면

1. 스키마 관리자에서 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. 쓰기 잠금을 획득합니다.
3. **기본 키 일치 규칙** 탭을 클릭합니다.

기본 키 일치 규칙 탭이 표시됩니다.

기본 키 일치 규칙 탭에는 다음 열이 있습니다.

열	설명
키 조합	일치를 위해 이 기본 일치 키 규칙이 사용되는 두 개의 소스 시스템입니다. 이러한 소스 시스템은 Informatica MDM Hub에 이미 구성되어 있어야 하며, 이 기본 개체에 대한 준비 테이블이 이러한 소스 시스템과 연결되어 있어야 합니다.
자동 병합	이 기본 키 일치 규칙의 결과로 자동 통합을 수행할지 수동 통합을 수행할지를 지정합니다.

4. **추가** 단추를 클릭하여 기본 일치 키 규칙을 추가합니다.
기본 키 일치 규칙 추가 대화 상자가 표시됩니다.
5. 기본 키를 기준으로 레코드를 일치시킬 두 개의 소스 시스템 옆에 있는 확인란을 선택합니다.

6. 기본 키가 동일한 레코드는 서로 일치하는 것이 확실한 경우 **일치하는 레코드 자동 병합** 확인란을 선택합니다.
필요한 경우 나중에 **일치하는 레코드 자동 병합**의 선택을 변경할 수 있습니다.
7. **확인**을 클릭합니다.
스키마 관리자의 **기본 키 규칙** 탭에 새 규칙이 표시됩니다.
8. **저장** 단추를 클릭하여 변경 내용을 저장합니다.
기존 일치 항목을 재설정할지 묻는 메시지가 표시됩니다.
9. 필요한 경우 **예**를 선택하여 일치 테이블에 현재 저장되어 있는 모든 일치 항목을 삭제합니다.

기본 키 일치 규칙 편집

기본 키 일치 규칙을 정의한 후 **일치하는 레코드 자동 병합** 확인란 값을 변경할 수 있습니다.

기존 기본 키 일치 규칙을 편집하려면

1. 스키마 관리자에서 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. 쓰기 잠금을 획득합니다.
3. **기본 키 일치 규칙** 탭을 클릭합니다.
기본 키 일치 규칙 탭이 표시됩니다.
4. 편집할 기본 키 일치 규칙으로 스크롤합니다.
5. **일치하는 레코드 자동 병합** 확인란을 선택하거나 선택 취소하여 자동 병합을 활성화하거나 비활성화합니다.
6. **저장** 단추를 클릭하여 변경 내용을 저장합니다.
기존 일치 항목을 재설정할지 묻는 메시지가 표시됩니다.
7. 필요한 경우 **예**를 선택하여 일치 테이블에 현재 저장되어 있는 모든 일치 항목을 삭제합니다.

기본 키 일치 규칙 삭제

기존 기본 키 일치 규칙을 삭제하려면

1. 스키마 관리자에서 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. 쓰기 잠금을 획득합니다.
3. **기본 키 일치 규칙** 탭을 클릭합니다.
기본 키 일치 규칙 탭이 표시됩니다.
4. 삭제할 기본 키 일치 규칙을 선택합니다.
5. **삭제** 단추를 클릭합니다.
스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.
6. **예**를 선택합니다.
삭제하는 규칙이 기본 키 일치 규칙 탭에서 제거됩니다.
7. **저장** 단추를 클릭하여 변경 내용을 저장합니다.
기존 일치 항목을 재설정할지 묻는 메시지가 표시됩니다.
8. 원하는 경우 **예**를 선택하여 일치 테이블에 현재 저장된 모든 일치 항목을 삭제합니다.

일치 키의 분포 조사

이 섹션에서는 일치 키 테이블에서 일치 키 분포를 확인하는 방법을 설명합니다.

일치 키 분포 정보

일치 키는 일치 후보를 식별하는 데 사용되는 유사 항목 일치 키 열의 데이터를 인코딩하는 문자열입니다.

토큰화 프로세스에서는 기본 개체의 모든 레코드에 대해 일치 키를 생성하여 해당 일치 키 테이블에 저장합니다. 기본 개체 레코드의 특성에 따라 토큰화 프로세스에서는 각 기본 개체 레코드에 대해 하나 이상의 일치 키(대개 여러 개의 일치 키)를 생성합니다. 일치 키는 이후에 일치 프로세스에서 기본 개체 레코드 간의 가능한 일치 항목을 확인하는 데 사용됩니다.

스키마 관리자의 일치/병합 설정 세부 정보 창에 있는 일치 키 분포 탭에서는 일치 키 테이블의 일치 키 분포를 조사할 수 있습니다. 이 도구는 데이터에서 잠재적 핫스팟(과도 일치치를 초래할 수 있는 일치 키의 과도한 집중)을 식별하는 데 도움을 줍니다. 핫스팟이 있는 경우 일치 프로세스에서 관련 없는 일치를 포함하여 지나치게 많은 일치를 생성하게 됩니다. 데이터에서 핫스팟이 발생하는 위치를 알고 있으면 데이터 정리 및 일치 규칙을 세부적으로 조정하여 핫스팟을 줄이고 일치 프로세스에 사용할 수 있는 최적의 분포로 일치 키를 생성할 수 있습니다. 모든 키에 비교적 고른 분포가 가장 좋습니다.

일치 키 분포 탭으로 이동

일치 키 배포 탭으로 이동하려면

1. 스키마 관리자에서 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. **일치 키 배포** 탭을 클릭합니다.

스키마 관리자에 [일치 키 배포] 탭이 표시됩니다.

일치 키 분포 탭의 구성 요소

일치 키 분포 탭에는 히스토그램, 일치 키 및 일치 열이 표시됩니다.

히스토그램

히스토그램은 일치 키 테이블에 있는 일치 키의 통계적 분포를 보여 줍니다.

축	설명
키(X축)	일치 키의 시작 문자입니다. 필터가 적용되지 않은 경우(기본값) 일치 키의 시작 문자입니다. 필터가 적용된 경우에는 일치 키의 시작 문자 시퀀스이며 가장 왼쪽의 문자로 시작됩니다.
개수(Y축)	일치 키 테이블에서 시작 문자로 시작되는 일치 키의 수입니다. 일치 키 테이블의 핫스팟은 히스토그램의 다른 문자와 비교하여 불균형적으로 높은 스파이크(높은 수의 일치 키)를 보여 줍니다.

일치 키 목록

일치 키 분포 탭의 일치 키 목록에는 일치 키 테이블의 레코드가 표시됩니다.




각 레코드에 대해 다음과 같은 열의 셀 데이터가 표시됩니다.

열 이름	설명
ROWID	기본 개체에서 이 일치 키와 연결된 레코드를 고유하게 식별하는 ROWID_OBJECT입니다.
KEY	생성된 일치 키입니다. 일치 키 테이블의 SSA_KEY 열입니다.

구성된 일치 규칙과 레코드의 데이터 특성에 따라, 기본 개체 테이블의 단일 레코드에 생성된 일치 키 여러 개가 있을 수 있습니다.

일치 키 테이블의 레코드를 통한 페이징

다음 명령 단추를 사용하여 일치 키 테이블의 레코드를 탐색할 수 있습니다.

단추	설명
	일치 키 테이블의 레코드에 대한 첫 번째 페이지를 표시합니다.
	일치 키 테이블의 레코드에 대한 이전 페이지를 표시합니다.
	일치 키 테이블의 레코드에 대한 다음 페이지를 표시합니다.
<input type="text" value="1"/>	입력한 페이지 번호로 이동합니다.

일치 열

일치 키 분포 탭의 일치 열 영역에는 일치 키 목록에서 선택된 레코드에 대한 일치 열 데이터가 표시됩니다.

일치 키 테이블의 SSA_DATA 열입니다. 이 기본 개체에 대해 구성된 각각의 일치 열의 열 이름과 셀 데이터가 표시됩니다.

일치 키 필터링

일치 키 필터를 사용하여 핫스팟 또는 다른 일치 키 분포 패턴에 조사를 집중할 수 있습니다.

일치 키 필터는 필터 조건을 만족하는 일치 키의 하위 집합으로 히스토그램 및 일치 키 목록의 데이터를 제한합니다. 기본적으로 필터는 정의되어 있지 않습니다. 따라서 일치 키 테이블의 모든 레코드가 표시됩니다.

필터 조건은 해당되는 일치 키에 대해 왼쪽에서 오른쪽으로 평가되는 시작 문자열 시퀀스를 지정합니다. 예를 들어 글자 M으로 시작되는 일치 키만 표시하려면 필터에서 M을 선택합니다. 일치 키를 한층 더 제한하여 글자 MD로 시작되는 일치 키에 대한 데이터만 표시하려면 필터에 글자 D를 추가합니다. 필터 식이 길어질수록 표시가 더욱 제한됩니다.



필터 설정

필터를 설정하려면

1. 필터에 추가할 문자와 연결된, 히스토그램의 세로 막대를 클릭합니다.
2. X축에서 문자 위의 세로 막대를 클릭합니다. 그러면 히스토그램이 새로 고쳐지고 해당 문자로 시작하는 모든 일치 키에 대한 분포가 표시됩니다.
일치 키 목록에는 필터 조건과 일치하는 일치 키만 표시됩니다.

필터 탐색

다음 명령 단추를 사용하여 필터를 탐색할 수 있습니다.

단추	설명
	필터를 지웁니다. 기본 보기를 표시합니다(필터 없음).
	이전에 선택한 필터를 표시합니다(필터의 가장 오른쪽 문자가 제거됨).

일치 프로세스에서 레코드 제외

Informatica MDM Hub에는 일치 프로세스에서 선택적으로 레코드를 제외하는 메커니즘이 있습니다. 예를 들어 데이터에 일치 프로세스에서 무시해야 할 레코드가 포함된 경우 이 기능을 사용할 수 있습니다.

이 기능을 구성하려면 스키마 관리자에서 기본 개체에 **EXCLUDE_FROM_MATCH**라는 열을 추가합니다. 이 열은 기본값이 0인 integer 유형이어야 합니다.

일치 작업에서 레코드를 제외하려면 테이블을 채운 다음 일치 작업을 실행하기 전에 데이터 관리자에서 **EXCLUDE_FROM_MATCH** 열의 레코드 값을 1로 변경합니다. 일치 작업을 실행하면 **EXCLUDE_FROM_MATCH** 값이 0인 레코드만 토근화되고 처리되며 다른 모든 레코드는 무시됩니다.

다음에 대한 일치 프로세스에서 레코드 제외를 사용할 수 있습니다.

- 정확히 일치하는 기본 개체 및 유사 항목 일치 기본 개체 모두
- 중복 항목에 대해 일치하지 않는 일치 열 규칙의 경우에만(기본 키 일치 규칙 아님)

근접 검색

지정한 좌표 부여 반지름에 근접한 레코드를 검색할 수 있습니다. 열이 위도, 경도 및 고도(선택 사항)로 채워진 기본 개체에 대한 근접 검색을 수행할 수 있습니다.

선택한 기본 개체에 위도, 경도, 고도와 같은 지리적 좌표에 대한 단일 열이나 여러 열이 있는지 확인합니다. 지리적 좌표는 서명된 도 형식이어야 하며 서로 다른 열이나 점표나 공백으로 구분된 단일 열에 포함될 수 있습니다. 예를 들어 사용자는 모든 지리적 좌표 -23.399437, -52.090904, 203이 포함된 좌표 부여라고 하는 단일 열을 가질 수 있습니다. 또는 값이 -23.399437인 위도, 값이 -52.090904인 경도, 값이 203인 고도의 3개 열을 가질 수도 있습니다.

1000미터와 같은 좌표 부여 반지름을 지정하면 1000미터 이내에 있는 레코드를 검색할 수 있습니다. 근접 검색을 사용하려면 MDM Hub가 일치 키를 생성하는 데 사용하는 **Person_Name**과 같은 다른 유사 항목 일치 열이 필요합니다. MDM Hub는 이러한 일치 키를 기준으로 레코드를 그룹화합니다. 그런 다음 MDM Hub는 좌표 부여 반지름이 포함된 일치 규칙을 사용하여 각 일치된 그룹에서 1000미터 이내에 있는 레코드를 서로 일치시킵니다.

근접 검색 구성

근접 검색을 구성하려면 좌표 부여 열과 하나 이상의 기타 유사 항목 일치 열을 구성해야 합니다. MDM Hub는 사용자가 구성하는 좌표 부여 열을 제외한 유사 항목 일치 열을 기준으로 근접 검색에 대한 일치 키를 생성합니다.

1. 스키마 도구를 선택하고 쓰기 잠금을 획득합니다.
스키마 탐색기가 나타납니다.
2. 근접 검색을 구성할 기본 개체를 확장한 후 일치/병합 설정을 선택합니다.
일치/병합 설정 세부 정보 창이 나타납니다.
3. **속성** 탭을 클릭하고 기본 개체에 대한 일치 속성을 구성합니다.
일치/검색 전략 속성의 값을 유사 항목으로 설정했는지 확인합니다.
4. **경로** 탭을 클릭하고 관련 레코드가 있는 경우 일치 경로를 구성합니다.
5. **일치 열** 탭을 클릭하고 일치 규칙에 대한 일치 열을 구성합니다.
 - a. 유사 항목 일치 키 섹션에서 유사 항목 일치 키 속성을 구성합니다.
일치 키를 생성하는 기준으로 **키 유형** 목록에서 키 유형을 선택해야 합니다.
유사 항목 일치 열이 일치 열 섹션에 추가됩니다.
 - b. 유사 항목 일치 열에 대한 소스 열을 선택한 후 선택한 열 목록으로 이동합니다.
 - c. **유사 항목 일치 열 추가**를 클릭합니다.
유사 항목 일치 열 추가 대화 상자가 나타납니다.
 - d. **필드 이름** 목록에서 좌표 부여를 선택합니다.
 - e. 지리적 좌표 데이터가 포함된 기본 개체 열을 선택한 열 목록으로 이동하여 근접 검색에 대한 일치 열을 생성합니다.
모든 지리적 좌표 데이터가 단일 열에 있는 경우 해당 열을 선택한 열 목록으로 이동합니다.
 - f. **확인**을 클릭합니다.
6. **일치 규칙 집합** 탭을 클릭하고 일치 규칙 집합을 구성합니다.
 - a. 일치 규칙 집합 섹션에서 **추가**를 클릭합니다.
일치 규칙 집합 추가 대화 상자가 나타납니다.
 - b. 일치 규칙 집합의 이름을 입력하고 **확인**을 클릭합니다.
일치 규칙 집합 섹션에 일치 규칙 집합이 표시됩니다.
 - c. 일치 규칙 섹션에서 **추가**를 클릭합니다.
일치 규칙 편집 대화 상자가 나타납니다.
 - d. **편집**을 클릭합니다.
일치 열 추가/제거 대화 상자가 나타납니다.
 - e. 일치 키를 생성하는 기준이 되는 **좌표 부여** 및 유사 항목 일치 열을 선택합니다.
 - f. **확인**을 클릭합니다.
일치 규칙 편집 대화 상자가 나타납니다.

g. **좌표 부여 반지름**을 포함한 일치 규칙 속성을 설정한 후 **확인**을 클릭합니다.

일치 규칙 섹션에 일치 규칙이 표시됩니다.

이제 일괄 작업 또는 서비스 통합 프레임워크 API를 사용하여 근접 검색을 수행할 수 있습니다.

경량 일치

일치 성능을 향상시키려면 경량 일치를 구성합니다. 경량 일치는 점수 예측값을 매우 빠르게 생성합니다. 이 알고리즘은 명백한 불일치가 포함된 후보를 전체 평가로 전달하는 대신 거부합니다. 일반 데이터 집합에서 이 알고리즘은 후보의 99% 이상을 거부하므로 성능이 향상됩니다.

팁: ID 번호 또는 생년월일과 같은 값을 포함하는 열에서 엄격한 정확히 일치 규칙을 이미 사용 중인 경우라면 경량 일치를 사용해도 현저한 성능 향상은 제공되지 않을 수 있습니다.

SSA-NAME3은 경량 일치 점수를 기반으로 후보를 거부합니다. SSA-NAME3은 SSA-NAME3 평가에 높은 일치를 보이는 후보를 거부할 수 있습니다. 경량 일치를 적용할 필드를 신중하게 선택하고 임계값 조정을 사용하여 이러한 위험을 완화할 수 있습니다.

필드에 경량 일치를 적용하려면 `LWM_FIELDS` 컨트롤을 사용합니다. 경량 일치 점수에 대한 거부 및 허용 한도를 설정하려면 `LWM_LIMIT` 컨트롤을 사용합니다.

경량 일치 구성

`cmxcleanse.properties` 파일에서 속성을 설정하여 경량 일치를 구성합니다.

예를 들어 다음 예는 지정된 필드에 대해 경량 일치를 활성화합니다.

```
cmx.server.match.lwm=true
cmx.server.match.lwm_param=LWM_FIELDS=Organization_Name,50,Address_Part1,50
LWM_LIMIT=75,85
cmx.server.match.stats=false
```

다음 속성 정의는 경량 일치 속성을 함께 사용하는 방법에 대해 설명합니다.

`cmx.server.match.lwm`

선택 사항입니다. 수동으로 추가해야 합니다. 경량 일치 기능을 제어합니다. 일치하는 레코드에 대한 전체 평가와 함께 경량 일치 기능을 활성화하려면 `Y`로 설정합니다. 일치하는 레코드에 대한 전체 평가 없이 경량 일치 기능을 활성화하려면 `ONLY`로 설정합니다. 기본값은 `N`입니다.

`cmx.server.match.lwm_param` 및 `cmx.server.match.stats` 속성과 함께 이 속성을 사용합니다.

`cmx.server.match.lwm_param`

선택 사항입니다. 수동으로 추가해야 합니다. `cmx.server.match.lwm` 속성을 `Y` 또는 `ONLY`로 설정해야 합니다. SSA-NAME3 컨트롤에 대한 속성 값을 다음 형식으로 설정합니다.

```
LWM=Y LWM_FIELDS=<field1>,<weight1>[,...,<fieldn>,<weightn>] LWM_LIMIT=<Reject>[,<Accept>]
```

`cmx.server.match.stats`

선택 사항입니다. 수동으로 추가해야 합니다. `cmx.server.match.lwm` 속성을 `Y` 또는 `ONLY`로 설정해야 합니다.

SSA-NAME3 컨트롤

`cmx.server.match.lwm_param` 속성 내에 다음 SSA-NAME3 컨트롤을 추가할 수 있습니다. 컨트롤은 공백으로 구분합니다.

LWM=Y/N/ONLY

경량 일치를 활성화 또는 비활성화합니다. 경량 일치를 활성화하려면 Y 값을 사용합니다. 경량 일치는 빠른 점수 예측을 사용하여 명백한 불일치를 거부합니다. 경량 일치에서 전달하는 레코드는 강력한 평가 및 순위 지정을 위해 전체 평가로 이동합니다. SSA-NAME3은 전체 점수 및 결정을 호출자에게 반환합니다.

참고: SDF 마법사를 사용하여 시스템 정의 파일을 생성하는 경우 경량 일치는 기본적으로 활성화됩니다.

경량 일치를 비활성화하려면 N 값을 사용합니다. SSA-NAME3 일치는 모든 일치 레코드에 대해 전체 평가를 수행합니다.

경량 일치를 활성화하고 전체 평가를 비활성화하려면 ONLY 값을 사용합니다. 경량 일치는 예측값을 호출자에게 최종 점수로 반환합니다.

LWM_FIELDS

경량 일치와 해당 가중치를 적용할 필드를 지정합니다. 이러한 값은 런타임 동안 일치 목적에서 정의한 값을 재정의합니다. SSA-NAME3은 경량 일치 점수를 기반으로 명백한 불일치를 거부합니다. 값을 설정하지 않는 경우 SSA-NAME3은 일치 목적에서 필드를 검색하고 여기에 동일한 가중치를 할당합니다.

LWM_FIELDS 컨트롤의 구문은 다음과 같습니다.

LWM_FIELDS=<field1>,<weight1>[,...,<fieldn>,<weightn>]

여기서 field는 목적 컨트롤에서 정의한 유효한 필드 이름이며, weight는 다른 필드와 비교했을 때 지정된 필드(0-100)의 상대적 중요성입니다.

예: LWM_FIELDS=Person_Name,5,Address_Part1,1

경량 일치는 주소와 같이 변형이 적은 필드에 적용할 때 유용합니다. 경량 일치는 변형이 큰 필드에는 효과적이지 않습니다. 필드의 변형이 큰 경우 SSA-NAME3은 편집 목록을 통해 변형을 처리하므로 경량 일치가 레코드를 잘못 거부할 수 있습니다.

LWM_LIMIT

경량 일치 점수에 대한 허용 한도와 거부 한도를 지정합니다. SSA-NAME3은 이러한 한도에 따라 검색 결과를 허용 또는 거부합니다.

LWM_LIMIT 컨트롤의 구문은 다음과 같습니다.

LWM_LIMIT=<Reject>[,<Accept>]

여기서 Reject 및 Accept는 0~100 범위의 정수 값입니다.

예: LWM_LIMIT=50,90

LWM=N인 경우 LWM_LIMIT 컨트롤은 아무런 영향을 미치지 않습니다.

LWM=Y인 경우 SSA-NAME3은 거부 한도보다 작은 경량 일치 점수를 거부합니다. 허용 한도는 영향을 미치지 않으므로 생략할 수 있습니다.

LWM=ONLY인 경우 SSA-NAME3은 거부 한도보다 작은 경량 일치 점수를 거부하고 허용 한도보다 큰 점수를 허용합니다. 거부 한도보다 크거나 같고 허용 한도보다 작은 레코드의 점수는 미결정으로 표시합니다.

기본 거부 한도는 65이고 기본 허용 한도는 90입니다. 허용 한도를 설정하지 않고 거부 한도가 90보다 큰 경우 허용 한도는 거부 한도와 같습니다.

제 22 장

일치 규칙 구성 예제

이 장에 포함된 항목:

- [일치 규칙 구성 예제 개요, 431](#)
- [일치 규칙 구성 시나리오, 432](#)
- [일치 규칙 구성, 433](#)
- [1단계. 데이터 검토, 433](#)
- [2단계. 일치 작업에 대한 기본 개체 식별, 434](#)
- [3단계. 일치 속성 구성, 434](#)
- [4단계. 일치 경로 정의, 435](#)
- [5단계. 일치 열 정의, 440](#)
- [6단계. 일치 규칙 집합 정의, 445](#)
- [7단계. 일치 규칙 추가, 447](#)
- [8단계. 일치 규칙에 대해 병합 옵션 설정, 449](#)
- [9단계. 일치 속성 검토, 450](#)
- [10단계. 일치 규칙 테스트, 452](#)

일치 규칙 구성 예제 개요

일치 규칙 구성 예제에는 MDM Hub의 중복 레코드를 일치/병합하도록 일치 규칙을 구성하는 방법이 소개됩니다. 일치 규칙 구성 예제는 MDM Hub 리소스 키트에 포함된 MDM Hub 샘플 ORS(연산 참조 저장소)에서 사용 가능한 데이터를 기반으로 합니다.

중복 레코드(서로 일치하는 레코드)를 결정하기 위해 MDM Hub는 일치 규칙을 사용합니다. 기본 개체 레코드에 대한 일치 규칙을 구성합니다. 정확한 또는 유사 항목 일치 작업에 대해 일치 규칙을 정의할 수 있습니다. 정확히 일치 규칙은 일치 열에 동일한 값이 있는 레코드를 일치시킵니다. 유사 항목 일치 규칙은 데이터 패턴에 발생할 수 있는 맞춤법 오류, 전치, 생략, 음운 변이 등의 변이를 고려한 확률 일치 항목을 기반으로 유사한 레코드를 일치시킵니다. 구성할 일치 규칙은 데이터 특성, 특정 일치 및 병합 요구 사항에 따라 달라집니다.

일치 규칙 구성 예제에는 구성 프로세스가 단계별로 소개됩니다. 이 예제는 MDM Hub 샘플 ORS에서 사용할 수 있는 당사자 기본 개체 데이터를 기반으로 합니다. 이 예제는 이름 및 주소를 기반으로 데이터를 일치시키고 병합할 당사자 기본 개체의 개인 데이터에 대해 일치 규칙을 구성하는 방법을 보여 줍니다. 예제에 설명된 일치 규칙과 유사한 미리 구성된 일치 규칙을 보려면 MDM Hub 샘플 ORS를 설정하십시오. 샘플 연산 참조 저장소에 대한 자세한 내용은 *Multidomain MDM 샘플 ORS 가이드*를 참조하십시오.

일치 규칙 구성 시나리오

조직에는 당사자 기본 개체 및 다른 관련 기본 개체에 저장된 고객 정보가 있습니다. 당사자 기본 개체에는 여러 고객에 대한 중복 레코드가 포함됩니다. 병합할 중복 레코드를 식별하고 대기열에 추가할 일치 규칙을 구성하려고 합니다.

당사자 기본 개체에는 성이나 이름이 누락되거나, 올바르게 알려지지 않거나 일관되지 않은 항목을 가진 중복 고객 레코드가 포함됩니다. 표시 이름 열에는 각 레코드에 대한 값이 있습니다. 레코드는 개인 또는 조직 당사자 유형으로 범주화됩니다. 고객 주소는 관련 주소 기본 개체에 저장됩니다. 당사자 주소 관계 관계 기본 개체는 당사자 및 주소 기본 개체의 레코드 간의 관계를 정의합니다.

당사자 기본 개체의 개인 당사자 유형이 있는 중복 고객 레코드를 일치시키는 일치 규칙을 만들려고 합니다. 개인 당사자 유형이 있는 레코드는 개인에게 속합니다. 당사자 및 주소 기본 개체에 저장된 이름과 주소를 기반으로 일치 작업을 수행할 수 있습니다.

중복 레코드를 일치시키려면 충분히 유사하고 자동 병합 대기열에 추가할 수 있는 자동 병합 일치 규칙을 작성해야 합니다. 중복 가능성이 있지만 병합 프로세스 이전에 데이터 스튜어드에서 검토해야 하는 레코드에 대해 수동 병합 일치 규칙을 작성해야 합니다.

당사자 기본 개체에는 중요한 열과 샘플 레코드가 표시됩니다.

Rowid	개체	이름	성	조직 이름	표시 이름	당사자 유형
1019		NULL	NULL	NULL	WILL R DE HAAN	개인
1072		NULL	NULL	NULL	AHMED RAUF	개인
1106		RACHEL	ARSEN	NULL	RACHEL ARSEN	개인
1154		RACHEL	ARSEN	NULL	RACHEL ARSEN	개인
1191		WILLIAM	DE HAAN	NULL	WILLIAM DE HAAN	개인
1419		BILL	DE HAAN	NULL	BILL ROGER DE HAAN	개인
1475		NULL	NULL	RYERSON AREA	RYERSON AREA MED CTR	조직
1642		AHMED	RAUF	NULL	AHMED RAUF	개인
1800		NULL	NULL	NULL	RYERSON AREA MED CTR	조직

주소 기본 개체에는 중요한 열과 샘플 레코드가 표시됩니다.

Rowid	개체	주소 행 1	주소 행 2	구/군/시 이름	시 코드	우편 번호
920		69 BUTLER ST	NULL	ATLANTA	NULL	30303
991		7610 ROSENWALD LN	NULL	NOKESVILLE	NULL	20181
1221		RR 1 BOX 4	NULL	SUGAR RUN	NULL	18846-9701
1279		RR 1 BOX 3	NULL	SUGAR RUN	NULL	18846-9701
1711		69 JESSE HILL JR DR SE	NULL	ATLANTA	NULL	30303-3033
1860		5493 S QUEENS RD	NULL	ROCHELLE	NULL	61068
1909		5193 S QUEENS RD	NULL	ROCHELLE	NULL	61068
1960		669 BUTLER ST SE	NULL	ATLANTA	NULL	30303-3033
2005		7601 ROSENWALD LN	NULL	NOKESVILLE	NULL	20181

LU 주소 유형 조회 기본 개체에는 중요한 열과 샘플 주소 유형 조회가 표시됩니다.

Rowid	개체	주소 유형	주요 유형 표시	주요 유형 설명
1		BILL	BILLING	BILLING
4		LGL	LEGAL	LEGAL
10		RSID	RESIDENCE	RESIDENCE
11		SHIP	SHIPPING	SHIPPING

당사자 주소 관계 관계 기본 개체에는 중요한 열과 샘플 레코드가 표시됩니다.

Rowid	개체	당사자 ID	주소 ID	주소 유형	상태 코드
976		1019	920	BILL	NULL
1051		1072	991	BILL	NULL
1080		1191	1960	LGL	NULL

Rowid	개체	당사자 ID	주소 ID	주소 유형	상태 코드
1100		1419	1711	BILL	NULL
2001		1154	1909	LGL	NULL
2002		1106	1860	LGL	NULL
2004		1642	2005	LGL	NULL
2049		1475	1279	LGL	NULL
2050		1800	1221	LGL	NULL

일치 규칙 구성

일치 규칙을 구성하려면 다음 태스크를 수행하십시오.

1. 데이터를 검토합니다.
2. 일치 작업에 대한 기본 개체를 식별합니다.
3. 일치 속성을 구성합니다.
4. 일치 경로를 정의합니다.
5. 일치 열을 정의합니다.
6. 일치 규칙 집합을 정의합니다.
7. 일치 규칙을 추가합니다.
8. 일치 규칙에 대해 병합 옵션을 설정합니다.
9. 일치 속성을 검토합니다.
10. 일치 규칙을 테스트합니다.

1단계. 데이터 검토

일치 규칙을 작성하기 전에 데이터를 검토하고 파악하십시오. 기본 개체에는 올바르게 않거나, 일치하지 않거나, 누락된 값이 있을 수 있습니다.

데이터 값, 일관성, 고유성 및 논리 품질에 대해 고객 데이터를 검토하십시오. 개인을 일치시킬 일치 규칙을 작성하려면 개인과 관련된 특성을 이해해야 합니다.

샘플 데이터 집합에는 당사자 및 주소 기본 개체가 포함됩니다. 당사자 기본 개체에 누락된 값이 포함된 이름 및 성 열이 있습니다. 누락된 값 때문에 성 및 이름 열을 일치 규칙에 배치하는 것은 좋지 않습니다. 표시 이름 열에는 모든 레코드에 대한 값이 있으므로 일치 규칙에서 일치 열로 사용하는 것이 좋습니다.

당사자 기본 개체의 당사자 유형 열에서 고객 레코드가 개인용인지 조직용인지 식별할 수 있습니다. 조직에 속한 고객 레코드에 대한 일치 항목을 찾지 않으려면 고객 레코드를 필터링하도록 당사자 유형 열을 사용할 수 있습니다. 일치 프로세스에 적합하지 않은 데이터를 필터링하면 일치 성능을 높일 수 있습니다.

주소 기본 개체에는 일치 규칙의 일치 열로 사용할 수 있는 주소 행 1, 구/군/시 이름 및 우편 번호 등의 열이 있습니다. 주소 기본 개체의 열은 당사자 기본 개체에 있는 중복 레코드를 식별하는 데 도움이 됩니다. 이름 및 주소 특성을 기반으로 개인을 일치시킬 일치 규칙을 작성할 수 있습니다.

2단계. 일치 작업에 대한 기본 개체 식별

이름 및 주소 특성에 기반하여 개인 데이터를 일치시키려고 합니다. 개인 데이터에 대해 일치 규칙을 작성하기 전에 먼저 일치 규칙을 배치하려는 특정 데이터가 포함된 기본 개체를 식별해야 합니다.

이 예제에서는 개인 데이터가 당사자 및 주소 기본 개체에 저장됩니다. 개인 이름은 당사자 기본 개체에 있으며 개인 주소는 주소 기본 개체에 있습니다. 일치 항목을 결정하고 당사자 및 주소 기본 개체의 이름 및 주소를 기반으로 중복된 개인 레코드를 제거할 수 있습니다. 중복 항목을 찾기 위해 개인 이름이 포함된 당사자 기본 개체에서 일치 작업을 실행할 수 있습니다.

3단계. 일치 속성 구성

일치 규칙에 대해 일치 열을 구성하기 전에 일치 및 검색 전략과 같은 당사자 기본 개체에 대해 일치 속성을 구성하십시오. 개인에 대한 중복 레코드가 있기 때문에 당사자 기본 개체에 대해 일치 속성을 구성해야 합니다.

데이터 특성, 데이터 파악 정도 및 일치/병합 요구 사항을 기반으로 일치 속성을 결정해야 합니다. 당사자 기본 개체의 데이터에 철자 변형 및 전위와 같은 오류가 있습니다. 확률 일치를 기반으로 하는 유사 항목 일치 및 검색 전략이 당사자 기본 개체 데이터에 적합합니다. 유사 항목 일치 및 검색 전략을 설정할 때 일치 규칙 인구집단을 설정해야 합니다. 인구집단은 일치에 사용할 데이터 특성을 정의합니다. 일치 작업에 선택할 데이터가 미국 중심인 경우 일치 속성으로 미국 인구집단을 설정해야 합니다.

일치 속성 구성

일치 속성을 구성하려면 Hub 콘솔의 스키마 도구를 사용하십시오.

1. Hub 콘솔에서 스키마 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일치 속성을 구성할 당사자 기본 개체를 확장합니다.
4. **일치/병합 설정**을 클릭합니다.
일치/병합 설정 세부 정보 페이지가 나타납니다.
5. **속성** 탭을 클릭합니다.
당사자 일치/병합 설정 세부 정보 섹션이 나타납니다.
6. 일치 규칙을 구성하려면 다음 속성을 설정해야 합니다.

속성	값 및 설명
일치/병합 전략	유사 항목. 개인 데이터에 발생하는 철자 변형, 가능한 철자 오류 및 전위 오류를 고려하는 확률 일치 전략입니다.
유사 항목 인구집단	KR. 일치하는 레코드에 대한 특정 특성을 정의하는 인구집단입니다.

다음 이미지는 **일치/병합 설정 세부 정보** 페이지의 **속성** 탭을 보여 줍니다.

Match/Merge Setup Details	
Match Rule Sets	Primary key match rules
Properties	Paths
Match Columns	Match Columns
Match Columns	0
Match Rule Sets	0
Match Rules in Active Set	0
Primary key match rules	0
Maximum matches for manual consolidation	1000
Number of rows per match job batch cycle	10
Accept All Unmatched Rows as Unique	No
Match/Search Strategy	Fuzzy
Fuzzy Population	US
Match Only Previous Rowid Objects	<input type="checkbox"/>
Match Only Once	<input type="checkbox"/>
Dynamic Match Analysis Threshold (0=disa...	0

4단계. 일치 경로 정의

관련 레코드에 대해 일치 규칙을 구성하려면 일치 경로 구성 요소를 구성하십시오. 관련 레코드는 하나의 기본 개체 또는 여러 기본 개체에 상주할 수도 있습니다.

하위 레코드를 포함하는 일치 규칙을 구성하려면 루트 기본 개체에 경로 구성 요소를 추가해야 합니다. 구성 요소를 일치 경로에 추가할 때 관련 상위 및 하위 기본 개체 간에 연결을 정의합니다.

다음 구성 요소를 일치 경로에 추가하십시오.

- 당사자. 개인 및 조직 데이터를 포함한 기본 개체입니다.
- 주소. 개인 또는 조직에 대한 고객 주소를 포함한 기본 개체입니다.
- LU 주소 유형. 주소 유형 조회를 포함한 기본 개체를 조회합니다.
- 당사자 주소 관계. 당사자 및 주소 기본 개체의 레코드 간에 관계를 정의하는 관계 기본 개체입니다.

일치 경로 구성 요소 추가

일치 경로는 상위 항목과 하위 기본 개체 간의 연결을 정의합니다. 상위 및 하위 레코드가 포함될 일치 규칙을 구성하려면 일치 경로 구성 요소를 루트 기본 개체에 추가하십시오. 또한, 일치 프로세스 동안 데이터를 포함하거나 제외할 필터도 추가하십시오.

1. 당사자 기본 개체에 대한 **일치/병합 설정 세부 정보** 페이지에서 **경로** 탭을 클릭합니다.

2. 당사자 주소 관계 관계 기본 개체를 일치 경로 구성 요소로 추가합니다.
 - a. 경로 구성 요소 섹션에서 **C_Party**의 루트 일치 경로 구성 요소를 선택합니다.
다음 이미지는 일치/병합 설정 세부 정보 페이지의 경로 탭을 보여 줍니다.

Match/Merge Setup Details				
Primary key match rules		Match Key Distribution	Merge Settings	
Properties	Paths	Match Columns	Match Rule Sets	
Path Components				
Display name	Component Name	Table Name	Direction	Check Mis...
Root for C_Party	N/A	Party	N/A	N/A

Filters		
Column	Operator	Values

- b. 경로 구성 요소 섹션에서 **추가**를 클릭합니다.
다음 이미지는 경로 구성 요소 추가 대화 상자를 보여 줍니다.
경로 구성 요소 추가 대화 상자가 표시됩니다.

Add Path Component		
Identity		
Display name		
Physical name	C_MT_	
Check For Missing Children	<input checked="" type="checkbox"/>	
Constraints:		
Table	Direction	Foreign Key On
Party Cross-Reference	Parent-to-Child	Rowid Object

OK Cancel

- c. **표시 이름** 필드에 당사자 주소 관계를 Hub 콘솔에 표시할 일치 경로 구성 요소의 이름으로 입력합니다.

입력한 일치 경로 구성 요소의 표시 이름을 기반으로 **실제 이름** 필드가 채워집니다. 실제 이름은 데이터 베이스의 경로 구성 요소 이름입니다.

다음 이미지는 **경로 구성 요소 추가** 대화 상자를 보여 줍니다.

경로 구성 요소 추가 대화 상자에 이름 필드가 채워져서 표시됩니다.

Identity	
Display name	Party Address Rel
Physical name	C_MT_PARTY_ADDRESS_REL
Check For Missing Children	<input checked="" type="checkbox"/>

Constraints:		
Table	Direction	Foreign Key On
Party Address Rel	Parent-to-Child	Rowid Object

d. **누락된 하위 항목 확인** 옵션이 활성화되었는지 확인합니다.

이 옵션은 기본적으로 활성화됩니다. 활성화된 경우 상위 기본 개체 레코드와 관련 하위 기본 개체 레코드 간에 일치가 발생합니다. 이 옵션이 활성화된 경우 하위 기본 개체의 하위 레코드가 상위 기본 개체 레코드에 없는 경우에도 일치가 발생합니다.

참고: 일치 키 생성은 **누락된 하위 항목 확인** 옵션의 구성에 따라 다릅니다. **누락된 하위 항목 확인** 옵션이 비활성화되고 상위 레코드에 하위 레코드가 없으면 해당 상위 레코드에 대해 일치 키가 생성되지 않습니다.

- e. 제약 조건 섹션에서 제약 조건을 선택한 후 **확인**을 클릭합니다.

당사자 주소 관계 경로 구성 요소가 경로 구성 요소 섹션에 표시됩니다.

다음 이미지는 **일치/병합 설정 세부 정보** 페이지의 **경로** 탭을 보여 줍니다.

Display name	Component Name	Table Name	Direction	Check Mis...
[-] Root for C_PARTY	N/A	Party	N/A	N/A
[-] Party Address Rel	C_MT_PARTY_ADDRESS_REL	Party Address Rel	Parent-to-Child	Yes

3. 주소 기본 개체를 당사자 주소 관계 일치 경로 구성 요소의 하위 항목으로 일치 경로에 추가합니다.
 - a. **경로** 탭의 경로 구성 요소 섹션에서 **당사자 주소 관계** 일치 경로 구성 요소를 선택합니다.
 - b. **추가**를 클릭합니다.
경로 구성 요소 추가 대화 상자가 표시됩니다.
 - c. Hub 콘솔에 표시하려는 주소 기본 개체 이름을 **표시 이름** 필드에 입력합니다.

- d. 제약 조건 섹션에서 제약 조건을 선택한 후 **확인**을 클릭합니다.

주소 경로 구성 요소가 당사자 주소 관계 일치 경로 구성 요소의 하위 항목으로 경로 구성 요소 섹션에 표시됩니다. 관계 방향이 하위 대 상위로 표시됩니다.

다음 이미지는 경로 구성 요소가 있는 **일치/병합 설정 세부 정보**의 **경로** 탭을 보여 줍니다.

Display name	Component Name	Table Name	Direction	Check Mis...
Root for C_PARTY	N/A	Party	N/A	N/A
Party Address Rel	C_MT_PARTY_ADDRESS_REL	Party Address Rel	Parent-to-Child	Yes
Address	C_MT_ADDRESS	Address	Child-to-Parent	Yes

4. 일치 작업에 대해 개인은 포함하고 조직을 제외하려면 당사자 유형 열이 포함된 당사자 기본 개체에 대해 필터를 구성합니다.

- a. 경로 구성 요소 섹션에서 **C_Party**의 루트 일치 경로 구성 요소를 선택합니다.

- b. 필터 섹션에서 **추가**를 클릭합니다.

필터 추가 대화 상자가 표시됩니다.

- c. **열** 목록에서 **당사자 유형**을 선택합니다.

- d. **연산자** 목록에서 **IN**을 선택합니다.

- e. 값 필드 옆에 있는 **편집**을 클릭합니다.

값 편집 대화 상자가 표시됩니다.

- f. **추가**를 클릭합니다.

값 추가 대화 상자가 나타납니다.

- g. 값 필드에 **개인**을 입력하고 **확인**을 클릭합니다.

- h. **확인**을 클릭합니다.

당사자 유형 열에 대한 필터가 필터 섹션에 표시됩니다. 당사자 유형 열에 대한 필터를 사용하면 일치 키를 작성할 때 조직 당사자 유형에 속한 **Ryerson Area Med Ctr**과 같은 레코드가 제외됩니다. 일치 키를 작성할 때 **Rachel Arsen** 및 **Ahmed Rauf**와 같이 개인 당사자 유형에 속한 레코드만 포함됩니다.

5단계. 일치 열 정의

일치 경로 구성 요소를 구성한 후 일치 규칙에 사용할 일치 열을 정의하십시오. 당사자 기본 개체에 대한 유사 항목 일치 및 검색 전략을 선택했기 때문에 유사 항목 일치 및 정확히 일치 열을 구성할 수 있습니다.

개인에 대한 중복 레코드를 일치시키도록 일치 규칙을 작성하려고 합니다. 일치 규칙에는 중복 레코드를 일치시킬 개인 이름 및 주소 데이터가 있는 열이 포함되어야 합니다.

개인에 대한 중복 레코드를 일치시키려면 다음 열을 일치 열로 정의하십시오.

- 표시 이름
- 주소 행 1
- 주소 행 2
- 구/군/시 이름
- 시 코드
- 우편 번호

일치 열 정의

당사자 기본 개체에 대한 일치 열을 정의하십시오.

1. 당사자 기본 개체에 대한 **일치/병합 설정 세부 정보** 페이지에서 **일치 열** 탭을 클릭합니다.
2. 당사자 기본 개체에 대해 유사 항목 일치 키 설정을 구성합니다.

- a. **키 유형** 목록에서 **개인 이름**을 선택합니다.

경로 구성 요소가 루트(당사자)로 설정되어 있으며, 이는 일치 열 섹션에 추가될 **Person_Name** 유사 항목 일치 키에 대한 경로 구성 요소입니다.

- b. 키 너비 목록에서 **표준**을 선택합니다.

MDM Hub에서 일치 키를 작성할 검색 범위가 표준 크기로 설정되어 있습니다.

다음 이미지는 개인 이름 유사 항목 일치 키가 구성된 **일치 열** 탭을 보여 줍니다.

Match/Merge Setup Details

Primary key match rules | Match Key Distribution | Merge Settings

Properties | Paths | Match Columns | Match Rule Sets

Fuzzy Match Key

Key Type	Person Name
Key Width	Standard
Path Component	Root (Customer)

Match Columns

Field Name	Column Type	Path Component	Source Table
Person_Name	Fuzzy Match Key	Root	Party

Match Column Contents - Source Table: Customer

Available columns:

- Display Name
- First Name
- Last Name
- Middle Name
- Organization Name
- Party Type

Selected columns:

MDM Hub에서 구성한 개인 이름 유사 항목 일치 키를 기준으로 일치 키를 작성합니다.

3. 표시 이름 열을 **Person_Name** 유사 항목 일치 키 열에 대한 소스 열로 정의하십시오.

- a. 일치 열 콘텐츠 섹션의 **사용 가능한 열** 목록에서 **표시 이름**을 선택합니다.

- b. 오른쪽 화살표를 클릭합니다.

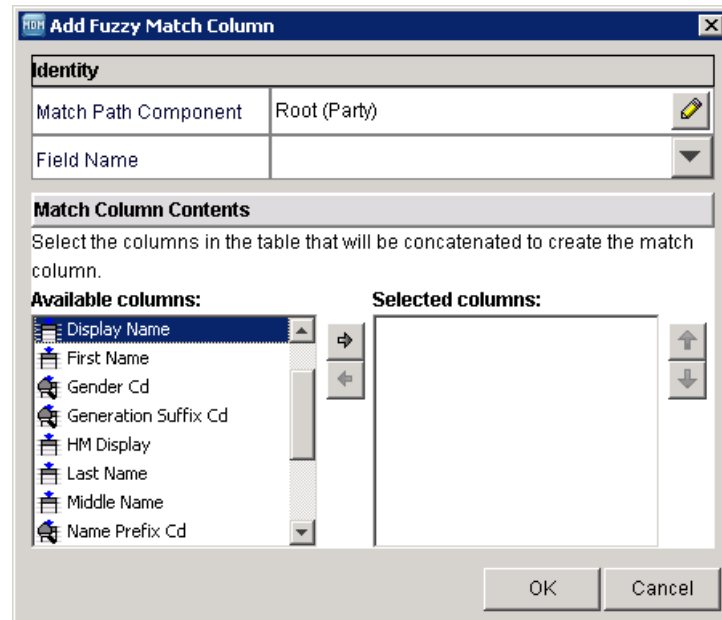
표시 이름 열이 **선택한 열** 목록으로 이동되고 **Person_Name** 일치 열에 대해 소스 열이 정의됩니다.

4. 주소가 포함되도록 유사 항목 일치 열인 Address_Part1을 추가하십시오.

a. 유사 항목 일치 열 추가를 클릭합니다.

유사 항목 일치 열 추가 대화 상자가 나타납니다.

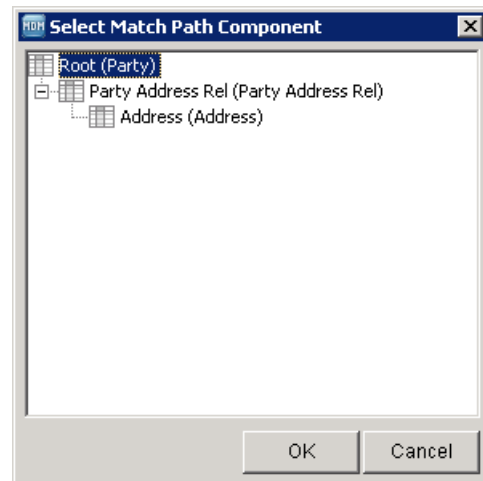
다음 이미지는 유사 항목 일치 열 추가 대화 상자를 보여 줍니다.



b. 일치 경로 구성 요소 필드 옆에 있는 편집을 클릭합니다.

일치 경로 구성 요소 대화 상자가 표시됩니다.

다음 이미지는 일치 경로 구성 요소 선택 대화 상자를 보여 줍니다.



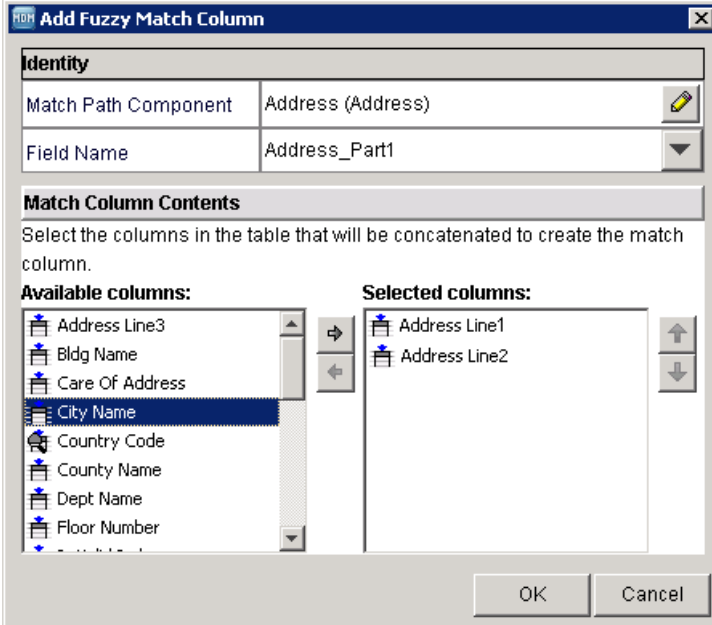
c. 고객 주소 데이터가 포함된 주소 기본 개체를 선택한 후 **확인**을 클릭합니다.

d. 필드 이름 목록에서 **Address_Part1**을 선택합니다.

e. 사용 가능한 열 목록에서 Address_Part1 일치 열을 작성하기 위해 연결하려는 주소 행 1 및 주소 행 2 열을 선택합니다.

f. 해당 열을 선택한 열 목록으로 이동하려면 오른쪽 화살표를 클릭합니다.

주소 행 1 및 주소 행 2 열이 선택한 열 목록으로 이동합니다. Address_Part1 일치 열을 작성하기 위해 열이 연결됩니다. 유사 항목 일치 열 추가 대화 상자가 나타납니다.
다음 이미지는 유사 항목 일치 열 추가 대화 상자를 보여 줍니다.



The dialog box is titled "Add Fuzzy Match Column". It contains two main sections: "Identity" and "Match Column Contents".

Identity Section:

Match Path Component	Address (Address)
Field Name	Address_Part1

Match Column Contents Section:

Select the columns in the table that will be concatenated to create the match column.

Available columns:

- Address Line3
- Bldg Name
- Care Of Address
- City Name
- Country Code
- County Name
- Dept Name
- Floor Number

Selected columns:

- Address Line1
- Address Line2

At the bottom, there are "OK" and "Cancel" buttons.

g. **확인**을 클릭합니다.

Address_Part1 일치 열이 일치 열 섹션에 추가됩니다.

다음 이미지는 개인 이름 및 주소 데이터에 대해 구성된 일치 열이 포함된 **일치 열** 탭을 보여 줍니다.

Match/Merge Setup Details

Primary key match rules | Match Key Distribution | Merge Settings

Properties | Paths | Match Columns | Match Rule Sets

Fuzzy Match Key

Key Type	Person Name
Key Width	Standard
Path Component	Root (Party)

Match Columns

Field Name	Column Type	Path Component	Source Table
Address_Part1	Fuzzy	Address	Address
Person_Name	Fuzzy Match Key	Root	Party

Match Column Contents - Source Table: Address

Available columns:

- City Name
- Country Code
- County Name
- Dept Name
- Floor Number
- Is Valid Ind
- Latitude

Selected columns:

- Address Line1
- Address Line2

5. 구/군/시 이름 및 시 코드가 포함되도록 고객 주소 데이터에 대해 유사 항목 일치 열인 Address_Part2를 추가하십시오.

a. **유사 항목 일치 열 추가**를 클릭합니다.

유사 항목 일치 열 추가 대화 상자가 나타납니다.

b. **일치 경로 구성 요소** 필드 옆에 있는 **편집**을 클릭합니다.

일치 경로 구성 요소 대화 상자가 표시됩니다.

c. 고객 주소 데이터가 포함된 주소 기본 개체를 선택한 후 **확인**을 클릭합니다.

d. **필드 이름** 목록에서 **Address_Part2**를 선택합니다.

e. **사용 가능한 열** 목록에서 Address_Part2 일치 열을 작성하기 위해 연결할 구/군/시 이름 및 시 코드 열을 선택합니다.

f. 해당 열을 **선택한 열** 목록으로 이동하려면 오른쪽 화살표를 클릭합니다.

구/군/시 이름 및 시 코드 열이 **선택한 열** 목록으로 이동합니다. Address_Part2 일치 열을 작성하기 위해 열이 연결됩니다.

g. **확인**을 클릭합니다.

Address_Part2 일치 열이 일치 열 섹션에 추가됩니다.

6. 정확히 일치 열인 **Postal_Area**를 추가하여 고객 주소에 대한 우편 번호가 포함되도록 합니다.
 - a. **정확히 일치 열 추가**를 클릭합니다.
정확히 일치 열 추가 대화 상자가 나타납니다.
 - b. **일치 경로 구성 요소** 필드 옆에 있는 **편집**을 클릭합니다.
일치 경로 구성 요소 선택 대화 상자가 표시됩니다.
 - c. 고객 주소 데이터가 포함된 주소 기본 개체를 선택한 후 **확인**을 클릭합니다.
 - d. **필드 이름** 목록에서 **Postal_Area**를 선택합니다.
 - e. **사용 가능한 열** 필드 이름에서 **Postal_Area** 일치 열을 작성하는 데 사용할 **우편 번호** 열을 선택합니다.
 - f. 해당 열을 **선택한 열** 목록으로 이동하려면 오른쪽 화살표를 클릭합니다.
우편 번호 열이 **선택한 열** 목록으로 이동합니다.
 - g. **확인**을 클릭합니다.

Postal_Area 일치 열이 일치 열 섹션에 추가됩니다.

다음 이미지는 개인 이름 및 주소 데이터에 대해 구성된 일치 열이 포함된 **일치 열** 탭을 보여 줍니다.

Match/Merge Setup Details

Primary key match rules | Match Key Distribution | Merge Settings
 Properties | Paths | Match Columns | Match Rule Sets

Fuzzy Match Key

Key Type	Person Name
Key Width	Standard
Path Component	Root (Customer)

Match Columns

	Field Name	Column Type	Path Compon...	Source Table
	Address_Part1	Fuzzy	Address	Address
	Address_Part2	Fuzzy	Address	Address
	Person_Name	Fuzzy Match Key	Root	Party
	Postal_Area	Exact	Address	Address

Match Column Contents - Source Table: Address

Available columns:

- Address Line3
- Bldg Name
- Care Of Address
- City Name
- Country Code

Selected columns:

- Address Line1
- Address Line2

6단계. 일치 규칙 집합 정의

검색 수준이 표준으로 설정된 일치 규칙 집합을 정의하려고 합니다. 정의한 일치 규칙 집합에 일치 규칙을 작성할 수 있습니다. 하나 이상의 일치 규칙 집합을 작성해야 합니다. 일치 규칙 집합에는 일부 일반 속성이 있는 일치 규칙의 논리적 집합이 포함될 수 있습니다.

일치 규칙 집합 정의

일치 규칙을 추가할 규칙 집합을 정의하십시오.

1. 당사자 기본 개체에 대한 **일치/병합 설정 세부 정보** 페이지에서 **일치 규칙 집합** 탭을 클릭합니다.
2. 일치 규칙 집합을 추가하려면 **추가**를 클릭합니다.
일치 규칙 집합 추가 대화 상자가 나타납니다.
3. WS와 같이 고유한 이름을 규칙 집합에 대해 입력하고 **확인**을 클릭합니다.
4. **검색 수준** 목록에서 **표준**을 선택합니다.
일치 후보를 검색할 검색 수준을 표준으로 설정합니다.
5. SearchMatch API에서만 일치 규칙 집합을 베타적으로 사용하도록 예약하지 않으려면 **규칙별 검색 활성화** 옵션을 비활성화해야 합니다.
6. **필터링 활성화** 옵션을 비활성화했는지 확인합니다.

참고: 필터링 활성화 옵션을 비활성화하면 일치 일괄 작업이 실행될 때 일치 규칙 집합에서 모든 레코드를 처리합니다. 필터링 활성화 옵션이 활성화되어 있으면 이 일치 규칙 집합에 대해 필터를 정의할 수 있고 일치 규칙 집합에서 필터링된 레코드만 처리합니다.

다음 이미지는 WS 일치 규칙 집합이 구성된 일치/병합 설정 세부 정보 페이지를 보여 줍니다.

Match/Merge Setup Details

Primary key match rules | Match Key Distribution | Merge Settings

Properties | Paths | Match Columns | Match Rule Sets

Match Rule Set

WS (*)

Match Rule Set

Name	WS
Search Level	Typical
Enable Search by Rules	<input type="checkbox"/>
Enable Filtering	<input type="checkbox"/>
Filtering SQL	

Match Rules

Rule #	Auto	Type	Accept Limit

7단계. 일치 규칙 추가

일치 규칙을 추가하기 전에 자동 병합 또는 수동 병합에 대한 일치 규칙을 설정할 것인지 결정해야 합니다. 주소로 개인을 일치시키려면 이름 및 주소 구성 요소가 있는 일치 목적이 필요합니다.

자동 병합하도록 일치 규칙을 설정하면 데이터 스튜어드의 개입 없이 **MDM Hub**에서 일치하는 레코드를 병합합니다. 수동 병합하도록 일치 규칙을 구성하면 **MDM Hub**에서 일치하는 레코드를 자동으로 병합하지 않습니다. 데이터 스튜어드에서 일치하는 레코드를 검토한 후 병합 작업을 시작할 수 있습니다.

또한, 유사 항목 일치, 정확한 일치 및 필터링된 일치 규칙을 설정할지 결정해야 합니다. 불일치 항목이 있을 수 있는 데이터에 대해 유사 항목 일치 규칙이 필요합니다. 데이터 품질이 좋으면 정확히 일치 규칙을 구성할 수 있습니다. 대량의 일괄 작업을 실행하는 동시에 최상의 성능을 보장하려면 필터링된 일치 규칙을 구성할 수 있습니다. 필터링된 일치 규칙은 유사 항목 일치 키를 정확히 일치 결과 함께 사용하는 정확히 일치 규칙입니다.

주소로 개인을 일치시키려면 이름 및 주소 구성 요소가 있는 거주자 일치 목적이 필요합니다.

일치 규칙 추가

이름 및 주소를 기반으로 중복된 개인에 대해 일치 작업을 하려면 거주자 일치 목적이 포함된 유사 항목 일치 규칙을 **WS** 일치 규칙 집합에 추가하십시오. 이전 단계에서 **WS** 일치 규칙 집합을 작성했습니다.

1. 당사자 기본 개체에 대한 **일치/병합 설정 세부 정보** 페이지에서 **일치 규칙 집합** 탭을 클릭합니다.
2. 표준 일치 수준과 함께 수동 병합 일치 규칙을 추가합니다.

- a. 일치 규칙 섹션에서 **추가**를 클릭합니다.

일치 규칙 편집 대화 상자가 나타납니다.

- b. **일치/검색 전략** 목록에서 **유사 항목 일치**를 선택합니다.

유사 항목 일치/검색 전략은 철자 변형, 철자 오류, 전위 오류가 발생한 레코드를 포함하는 확률 일치 전략입니다.

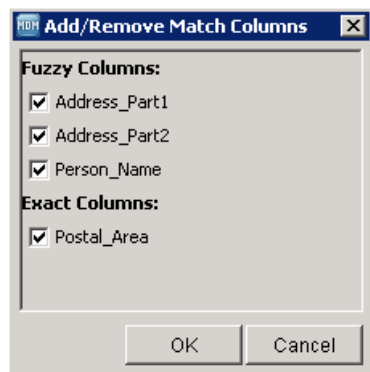
- c. **일치 열** 필드 옆에 있는 **편집**을 클릭합니다.

일치 열 추가/제거 대화 상자가 나타납니다.

- d. 다음 열을 선택합니다.

- Address_Part1
- Address_Part2
- Person_Name
- Postal_Area

다음 이미지는 선택한 일치 열이 포함된 **일치 열 추가/제거** 대화 상자를 보여 줍니다.



- e. **확인**을 클릭합니다.

일치 열 이름이 **일치 열** 필드에 표시됩니다.

- f. 주소를 기반으로 개인 이름에 대한 일치 항목을 식별하려면 **일치 목적** 목록에서 **거주자**를 선택하십시오.
- g. 레코드 간의 일치 정밀도를 정의하려면 **일치 수준** 필드에서 **표준**을 선택하십시오.

개인을 일치시키는 거의 모든 일치 작업에 대해 표준 일치 수준을 사용하는 것이 좋습니다. 보수적 일치 수준을 사용하면 표준 일치 수준보다 더 적은 수의 일치 항목을 반환하고 일부 잠재적 일치 항목이 일치 플래그가 표시되지 않은 채 일치 프로세스를 통과할 수 있습니다. 느슨한 일치 수준은 표준 일치 수준보다 훨씬 더 많은 수의 일치 항목을 반환할 수 있으며 실제로는 일치하지 않은 항목도 포함될 수 있습니다.

다음 이미지는 유사 항목 일치 규칙 설정이 포함된 **일치 규칙 편집** 대화 상자를 보여 줍니다.

- h. **확인**을 클릭합니다.

거주자 일치 목적이 있는 수동 병합 일치 규칙이 일치 규칙 섹션에 표시됩니다.

3. 느슨한 일치 수준과 함께 수동 병합 일치 규칙을 추가하십시오.

- a. 일치 규칙 섹션에서 **추가**를 클릭합니다.

일치 규칙 편집 대화 상자가 나타납니다.

- b. **일치/검색 전략** 목록에서 **유사 항목 일치**를 선택합니다.

유사 항목 일치/검색 전략은 철자 변형, 철자 오류, 전위 오류가 발생한 레코드를 포함하는 확률 일치 전략입니다.

- c. **일치 열** 필드 옆에 있는 **편집**을 클릭합니다.

일치 열 추가/제거 대화 상자가 나타납니다.

- d. 유사 항목 일치 열 옵션에서 다음 유사 항목 일치 열을 선택합니다.
 - Address_Part1
 - Person_Name
- e. 확인을 클릭합니다.
유사 항목 일치 열이 일치 열 필드에 표시됩니다.
- f. 주소를 기반으로 개인 이름에 대한 일치 항목을 식별하려면 일치 목적 목록에서 거주자를 선택하십시오.
- g. 레코드 간의 일치 정밀도를 정의하려면 일치 수준 필드에서 느슨함을 선택하십시오.
- h. 확인을 클릭합니다.

거주자 일치 목적이 있는 수동 병합 일치 규칙이 일치 규칙 섹션에 표시됩니다.

다음 이미지는 WS 일치 규칙 집합에 대해 구성된 두 개의 유사 항목 일치 규칙과 수동 병합 일치 규칙이 포함된 일치 규칙 집합 탭을 보여 줍니다.

The screenshot shows the 'Match/Merge Setup Details' dialog box. The 'Match Rule Set' tab is active, showing a list of rule sets on the left with 'WS (*)' selected. The main area displays the configuration for the 'WS' rule set. Below this, the 'Match Rules' section contains a table with two rules.

Rule #	Auto	Type	Accept Limit	Purpose(Level)	Columns
1	No	Fuzzy	0	Resident(Typical)	Address_Part1 (Fuzzy) Address_Part2 (Fuzzy) Person_Name (Fuzzy) Postal_Area (Fuzzy)
2	No	Fuzzy	0	Resident(Loose)	Address_Part1 (Fuzzy) Person_Name (Fuzzy)

8단계. 일치 규칙에 대해 병합 옵션 설정

일치 프로세스 동안 일치 규칙에서 자동 병합이나 수동 병합 작업을 위해 일치하는 레코드를 대기열에 추가할 것인지 결정해야 합니다. 일치 규칙에 대해 수동 병합과 자동 병합 간에 전환할 수 있습니다. 일치 프로세스 후 레코드를 자동 병합하도록 일치 규칙에 대해 표준 일치 수준을 설정할 수 있습니다.

거주자 일치 목적 및 표준 일치 수준으로 일치 규칙의 정밀도가 지정된 경우 자체적으로 자동 병합이 수행됩니다.

자동 병합 일치 규칙으로 일치 규칙 설정

일치 프로세스 동안 자동 병합에 대해 일치하는 레코드를 대기열에 추가하려면 일치 규칙을 자동 병합 일치 규칙으로 설정하십시오.

1. 자동 병합 일치 규칙으로 변경하려는 표준 일치 수준이 있는 수동 병합 일치 규칙을 선택합니다.
2. **위로 이동**을 클릭합니다.

수동 병합 일치 규칙이 자동 병합 일치 규칙으로 변경됩니다.

다음 이미지는 **WS** 일치 규칙 집합에 대해 구성된 수동 병합 일치 규칙 및 자동 병합 일치 규칙이 포함된 **일치 규칙 집합**을 보여 줍니다.

Match/Merge Setup Details

Properties | Paths | Match Columns | Match Rule Sets | Primary key match rules | Match Key Distribution | Merge Settings

Match Rule Set

WS (*)

Match Rule Set

Name	WS
Search Level	Typical
Enable Search by Rules	<input type="checkbox"/>
Enable Filtering	<input type="checkbox"/>
Filtering SQL	

Match Rules

Rule #	Auto	Type	Accept Limit	Purpose(Level)	Columns
1	Yes	Fuzzy	0	Resident(Typical)	Address_Part1 (Fuzzy) Address_Part2 (Fuzzy) Person_Name (Fuzzy) Postal_Area (Fuzzy)
2	No	Fuzzy	0	Resident(Loose)	Address_Part1 (Fuzzy) Person_Name (Fuzzy)

3. **저장**을 클릭합니다.
규칙 집합 평가 대화 상자가 표시됩니다.
4. **확인**을 클릭합니다.
작성한 일치 규칙이 저장되었습니다.

9단계. 일치 속성 검토

일치 규칙 구성을 저장한 후 일치 속성을 검토하십시오. 일치 속성에는 일치 및 병합 설정이 요약되어 있습니다.

일치 및 병합 속성에는 다음 세부 정보가 표시되어야 합니다.

- 구성된 일치 열 수

- 구성된 일치 규칙 집합 수
- 활성 일치 규칙 집합의 일치 규칙 수
- 수동 통합의 최대 일치 항목 수
- 각 일치 일괄 작업 주기에 대한 행 수
- 일치 및 검색 전략 유형
- 유사 항목 인구집단 이름

일치 속성 검토

일치 속성 설정에 대한 요약은 일치 속성을 검토하십시오.

- ▶ 당사자 기본 개체에 대한 **일치/병합 설정 세부 정보** 페이지에서 **속성** 탭을 클릭합니다.

당사자 **일치/병합 설정 세부 정보** 섹션이 표시됩니다.

다음 테이블에는 일치 및 병합 설정이 요약되어 있습니다.

속성	값
일치 열	4
일치 규칙 집합	1
활성 집합의 일치 규칙	2
기본 키 일치 규칙	0
수동 통합의 최대 일치 항목 수	1000
일치 작업 일괄 처리 주기당 행 수	10
일치되지 않은 모든 행을 고유 행으로 허용	아니요
일치/검색 전략	유사 항목 일치
유사 항목 인구집단	KR
이전 Rowid 개체만 일치	비활성화
한 번만 일치	비활성화
동적 일치 분석 임계값(0=비활성화됨)	0

다음 이미지는 일치/병합 설정 세부 정보 페이지의 속성 탭을 보여 줍니다.

Match/Merge Setup Details	
Match Rule Sets	Primary key match rules
Properties	Paths
Match Columns	Match Columns
Match Columns	4
Match Rule Sets	1
Match Rules in Active Set	2
Primary key match rules	0
Maximum matches for manual consolidation	1000
Number of rows per match job batch cycle	10
Accept All Unmatched Rows as Unique	No
Match/Search Strategy	Fuzzy
Fuzzy Population	US
Match Only Previous Rowid Objects	<input type="checkbox"/>
Match Only Once	<input type="checkbox"/>
Dynamic Match Analysis Threshold (0=disa...	0

10단계. 일치 규칙 테스트

일치 규칙을 테스트하려면 일치 작업을 실행한 후 결과를 검토하십시오.

대량의 데이터 집합을 대표하는 샘플 데이터 집합에서 일치 규칙을 테스트할 수 있습니다. 일치 규칙을 테스트하려면 대표적인 샘플 데이터 집합에서 일치 작업을 실행하십시오. 일치 작업이 완료되면 일치 결과를 검토합니다.

당사자 기본 개체와 연결된 C_PARTY_MTCH 일치 테이블에서 일치 결과를 검토할 수 있습니다. 또한 Hub 콘솔의 병합 관리자 및 Informatica Data Director의 잠재적 일치 옵션을 통해서도 일치 결과를 볼 수 있습니다. 일치하는 레코드를 검토하여 정확성을 확인합니다.

C_PARTY_MTCH 일치 테이블에는 샘플 일치 결과가 포함된 중요한 열이 표시됩니다.

ROWID_OBJECT	ROWID_OBJECT_MATCHED	ROWID_MATCH_RULE	AUTOMERGE_IND
1191	1019	SVR1.JJ4J	0
1191	1419	SVR1.JJ4J	0
1154	1106	SVR1.JJ4E	1
1642	1072	SVR1.JJ4E	1

ROWID_OBJECT 열에는 ROWID_OBJECT_MATCHED 열의 행 ID가 있는 레코드와 일치하는 레코드의 행 ID가 포함됩니다.

ROWID_MATCH_RULE 열에는 작성하려는 일치 규칙의 행 ID가 포함됩니다. 일치 규칙은 리포지토리의 C_REPOS_MATCH_RULE 테이블에 있습니다. SVR1.JJ4J는 거주자 일치 목적 및 느슨한 일치 수준을 가진 일치 규칙의 행 ID입니다. SVR1.JJ4E는 거주자 일치 목적 및 표준 일치 수준을 가진 일치 규칙의 행 ID입니다.

이 예제에서는 행 ID 1191인 William De Haan 레코드에 대해 두 개의 일치 항목이 있습니다. 해당 레코드는 행 ID가 1019이고 Will R De Haan인 개인 및 행 ID가 1419이고 Bill Roger De Haan인 개인과 일치합니다. 일치 항

목은 이름과 주소 열에 기반하고 있으며 유사하지만 병합하기 전에 데이터 스튜어드가 검토해야 합니다. William De Haan에 대한 두 개의 일치 항목이 수동 병합을 위해 대기열에 추가되었습니다.

일치 테이블에서는 행 ID 1154인 Rachel Arsen 레코드가 행 ID 1106인 Rachel Arsen 레코드와 일치합니다. 두 개의 Rachel Arsen 레코드에는 유사하고 자동 병합을 위해 대기열에 추가되어야 할 주소들이 주소 기본 개체에 있습니다.

다음 테이블에는 자신의 레코드가 다른 유사 레코드와 일치하는 일부 개인의 이름이 표시됩니다.

표시 이름	일치하는 표시 이름
WILLIAM DE HAAN	WILL R DE HAAN
WILLIAM DE HAAN	BILL ROGER DE HAAN
RACHEL ARSEN	RACHEL ARSEN
AHMED RAUF	AHMED RAUF

일치 결과가 예제에 올바르게 표시됩니다. 더 큰 일치 작업에 대해 구성된 일치 규칙을 사용할 수 있습니다.

참고: 일치 작업 중에 발생할 수 있는 문제를 보려면 다음 디렉터리에 있는 cmxserver.log 파일을 검토하십시오.

UNIX의 경우. <infamdm_install_directory>/hub/server/logs

Windows의 경우. <infamdm_install_directory>\hub\server\logs

각 일치 작업 일괄 주기에 대한 행 수와 같이 일치 속성으로 인해 시간이 초과될 수 있습니다. **일치/병합 설정** 페이지의 **속성** 탭에서 속성을 변경할 수 있습니다.

일치 작업 진행률을 확인하고 요약 통계를 얻으려면 프로세스 서버 로그를 검토하십시오.

cmxserver.log 프로세스 서버 로그는 다음 디렉터리에 있습니다.

UNIX의 경우. <프로세스 서버 설치 디렉터리>/hub/cleanse/logs

Windows의 경우. <프로세스 서버 설치 디렉터리>\hub\cleanse\logs

제 23 장

Elasticsearch를 사용한 검색

이 장에 포함된 항목:

- [Elasticsearch를 사용한 검색 개요, 454](#)
- [Elasticsearch 아키텍처를 사용한 검색, 454](#)
- [검색 설치 및 구성, 455](#)
- [1단계. Elasticsearch 설치 및 설정, 456](#)
- [2단계. 검색에 대한 MDM Hub 속성 구성, 460](#)
- [3단계. 프로비저닝 도구를 사용하여 검색 구성, 464](#)
- [4단계. 연산 참조 저장소 유효성 검사, 473](#)
- [5단계. 검색 데이터 인덱싱, 473](#)
- [키 저장소, 트러스트 저장소 및 인증서 생성\(선택 사항\), 473](#)

Elasticsearch를 사용한 검색 개요

Data Director 응용 프로그램 또는 사용자 지정 응용 프로그램을 사용하여 특정 비즈니스 항목 내의 데이터를 검색할 수 있습니다. MDM에서는 오픈 소스의 전체 텍스트 검색 엔진인 Elasticsearch를 사용합니다. MDM Hub 설치 관리자 패키지로 제공되는 Elasticsearch 검색을 구성해야 합니다.

참고: Elasticsearch 검색은 더 이상 지원되지 않는 Solr 검색을 대체합니다.

Elasticsearch를 단일 노드 클러스터로 설정하거나 다중 노드 클러스터로 설정하여 분산 인덱싱 및 검색을 제공할 수 있습니다.

비즈니스 사용자 및 데이터 스튜어드는 데이터 검색을 수행하기 전에 Hub 서버와 처리 서버를 구성하고 데이터를 인덱싱해야 합니다. 데이터를 인덱싱하면 MDM Hub가 레코드를 처리하고 인덱싱된 레코드를 Elasticsearch 서버에 추가합니다. 데이터가 하나의 노드에 비해 클 경우 여러 노드를 사용하고 여러 shard로 데이터를 분할할 수 있습니다.

Elasticsearch 아키텍처를 사용한 검색

MDM Hub는 검색을 위해 Hub 서버와 처리 서버를 사용합니다. 실시간 인덱싱을 수행하기 위해 Hub 서버에 Elasticsearch 클라이언트가 포함되어 있습니다. 처리 서버는 초기 데이터 로드 후에 초기 인덱싱만 수행합니다.

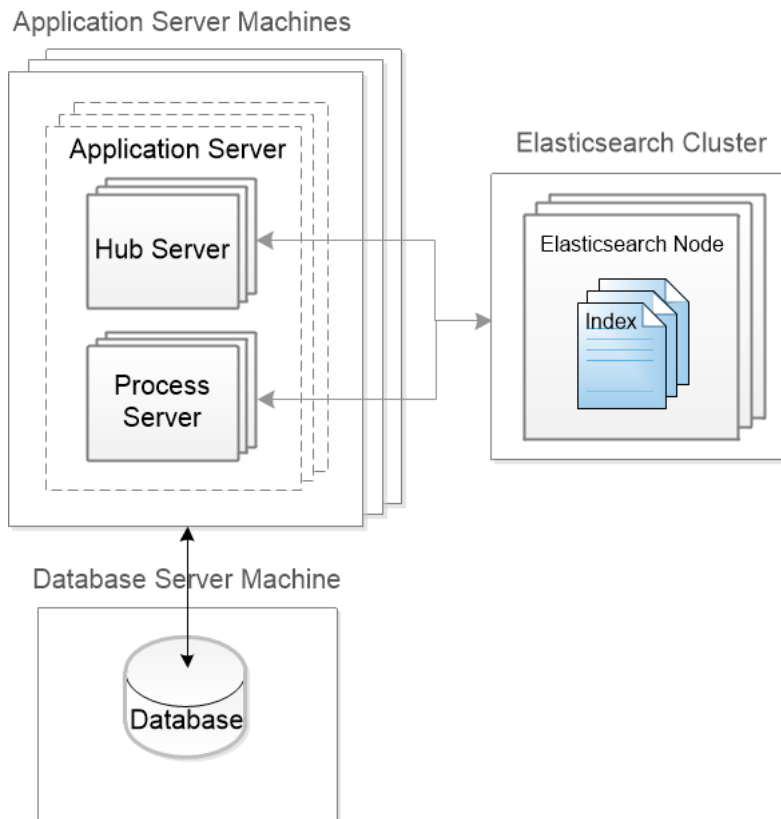
모든 MDM Hub 구성 요소를 단일 시스템에 설치하든, 여러 시스템에 설치하든 모든 Hub 서버 및 처리 서버 인스턴스를 검색에 대해 구성해야 합니다.

Elasticsearch는 MDM Hub 구성 요소가 설치되어 있는 시스템이나 별도의 시스템에 설치할 수 있습니다.

Elasticsearch에는 MDM Hub 구성 요소가 공유하지 않는, 지원되는 버전의 전용 JVM(Java Virtual Machine)이 필요합니다.

Elasticsearch를 사용하여 검색하도록 MDM Hub 구성 요소를 구성하려면 Elasticsearch 클러스터를 설정합니다. 클러스터는 하나 이상의 노드로 구성된 컬렉션입니다. 각 노드는 데이터를 저장하고 인덱싱 및 검색 작업에 관여하는 단일 서버입니다. MDM Hub 토폴로지 및 인덱싱할 데이터 양에 따라 클러스터에 대해 하나 이상의 노드를 구성할 수 있습니다. 각 노드에는 여러 인덱스가 포함될 수 있습니다. 데이터 손실을 방지하고 클러스터 안정성을 유지하기 위해 다중 노드 클러스터에 Zen Discovery를 구성할 수 있습니다.

다음 이미지에서는 검색용으로 구성된 샘플 설치 토폴로지를 보여 줍니다.



검색 설치 및 구성

검색을 사용하려면 Elasticsearch로 MDM Hub를 구성합니다.

1. Elasticsearch를 설치 및 설정합니다.
2. 검색에 대한 MDM Hub 속성을 구성합니다.
3. 프로비저닝 도구를 사용하여 검색을 구성합니다.
4. ORS(연산 참조 저장소) 유효성을 검사합니다.
5. 검색 데이터를 인덱싱합니다.

1단계. Elasticsearch 설치 및 설정

검색을 구성하려면 Elasticsearch를 설치 및 설정해야 합니다.

Elasticsearch를 설정하려면 다음 태스크를 수행합니다.

1. 설치 전 태스크를 완료합니다.
2. Elasticsearch를 설치합니다.
3. Elasticsearch JVM(Java Virtual Machine)을 구성합니다.
4. Elasticsearch 속성 파일을 구성합니다.
5. Elasticsearch를 보호합니다.
6. 분석 플러그 인을 설치합니다.
7. 검색에서 무시할 단어 목록을 사용자 지정합니다.
8. 검색에 포함할 동의어 목록을 사용자 지정합니다.
9. Elasticsearch를 시작합니다.

설치 전 태스크 완료

Elasticsearch 클러스터를 설치 및 설정하기 전에 환경을 준비하고 고가용성을 구성할지 결정합니다.

모든 환경의 태스크

다음 태스크를 수행하여 설치 환경을 준비합니다.

- 각 시스템이 지원되는 Elasticsearch 버전에 대한 하드웨어 요구 사항을 충족하는지 확인합니다. 하드웨어에 대한 자세한 내용은 Elasticsearch 설명서를 참조하십시오.
- 각 시스템이 지원 운영 체제 및 Java 버전과 같은 지원되는 Elasticsearch 버전에 대한 소프트웨어 요구 사항을 충족하는지 확인합니다. 소프트웨어 요구 사항에 대한 자세한 내용은 *Elasticsearch Support Matrix*를 참조하십시오.
- 스왑, 파일 설명자, 가상 메모리와 같은 중요 시스템 구성을 완료합니다. 중요 시스템 구성에 대한 자세한 내용은 Elasticsearch 설명서를 참조하십시오.

UNIX 환경의 태스크

UNIX 환경에서 다음 태스크를 수행합니다.

- 파일 설명자 수 부족으로 인한 데이터 손실을 방지하도록 파일 설명자의 수를 65536 이상으로 설정합니다.
- 메모리 스왑을 방지하려면 스왑을 방지하도록 시스템을 구성합니다. `mlockall`을 통해 메모리의 힙을 잠그도록 JVM(Java Virtual Machine)을 구성할 수 있습니다.

고가용성 요구 사항

인덱싱 및 검색할 대량의 데이터가 있는 경우에는 고가용성 Elasticsearch 클러스터를 구현하는 것이 좋습니다. 고가용성 클러스터에는 여러 개의 노드가 있으며 클러스터가 노드 간에 워크로드를 분배할 수 있습니다. 프로덕션 환경에서 하나의 노드가 실패하는 경우 클러스터가 워크로드를 다른 노드에 분배합니다.

설치 전 태스크의 하나로, 고가용성 Elasticsearch 클러스터를 구현할지 결정합니다. 구현할 경우 Elasticsearch 클러스터를 평소대로 구성하되 다음과 같은 추가 요구 사항을 충족해야 합니다.

- Elasticsearch 클러스터에 3개 이상의 노드가 있습니다.
팁: 소형 클러스터를 설정하여 클러스터를 시작하고 필요에 따라 확장할 수 있습니다. 워크로드를 분석하여 노드 실패를 처리할 수 있는 충분한 용량이 있는지 확인해야 합니다.
- 각 노드는 별도의 전용 시스템에 구성되어 있습니다.

- 안정성과 성능을 보장하기 위해 마스터 노드가 3개 이상입니다. Elasticsearch에서는 마스터 노드를 홀수로 구성하는 것을 권장합니다.
 - 클러스터에 노드가 3개만 있다면 모든 노드를 마스터 노드로 구성합니다.
 - 클러스터에 4개 이상의 노드가 있는 경우에는 3개의 노드를 마스터 노드로 구성하고 나머지 노드를 데이터 노드로 구성합니다.
- Elasticsearch 클러스터 크기에 따라 복제본의 수를 결정합니다. 프로비저닝 도구를 사용하여 Elasticsearch 인덱스를 구성할 때 사용할 복제본의 수를 지정할 수 있습니다.
- 각 노드의 elasticsearch.yml 구성 파일에서 다음과 같은 추가 속성을 설정합니다.
 - discovery.zen.minimum_master_nodes
 - discovery.zen.ping.unicast.hosts

하드웨어 요구 사항, 시스템 구성, 속성 값을 포함한 고가용성 클러스터에 대한 자세한 내용은 Elasticsearch 설명서를 참조하십시오.

Elasticsearch 설치

Hub 서버 및 처리 서버를 설치한 후 검색을 구성하려면 Elasticsearch 클러스터를 설치 및 설정합니다.

Elasticsearch 설치에 지원되는 운영 체제 및 Java 버전을 사용하는지 확인하십시오. 자세한 내용은 Elasticsearch Support Matrix를 참조하십시오.

Elasticsearch를 설치하고 클러스터를 설정하는 방법에 대한 자세한 내용은 Elasticsearch 설명서를 참조하십시오.

1. Elastic 웹 사이트에서 지원되는 버전의 Elasticsearch 보관 파일을 다운로드합니다.
지원되는 버전에 대한 자세한 내용은 PAM(Product Availability Matrix)을 참조하십시오. PAM은 <https://network.informatica.com/community/informatica-network/product-availability-matrices>에서 액세스할 수 있습니다.
2. Elasticsearch 보관 파일을 추출합니다.

Elasticsearch JVM(Java Virtual Machine) 구성

시스템에서 사용 가능한 RAM 양에 따라 힙 크기를 사용하도록 Elasticsearch JVM(Java Virtual Machine)을 구성합니다. JVM을 구성하려면 jvm.options 파일을 편집합니다.

1. 다음 디렉터리에서 jvm.options 파일을 찾습니다.
`<elasticsearch 설치 디렉터리>/config`
2. 텍스트 편집기를 사용하여 파일을 열고 다음 속성을 편집합니다.

속성	설명
-Xms	최소 힙 크기입니다. 기본값은 1GB입니다.
-Xmx	최대 힙 크기입니다. 기본값은 1GB입니다.
-XX:HeapDumpPath	힙 덤프 경로입니다. 기본값은 /var/lib/elasticsearch입니다. 다중 클러스터 환경에서는 이 속성을 대체 경로로 설정해야 합니다.

참고: 최소 힙 크기(Xms) 및 최대 힙 크기(Xmx)를 동일한 값으로 설정합니다. 다른 속성의 경우 기본 설정을 사용합니다.

Elasticsearch 속성 파일 구성

Informatica는 샘플 Elasticsearch 속성 파일을 제공합니다. Elasticsearch를 구성하려면 속성 파일을 편집합니다.

1. 다음 디렉터리에서 `elasticsearch.yml` 파일을 찾습니다.

<elasticsearch 설치 디렉터리>/config

2. 텍스트 편집기를 사용하여 파일을 열고 다음 속성을 편집합니다.

속성	설명
<code>bootstrap.memory_lock</code>	메모리 잠금을 설정합니다. Elasticsearch 메모리 스왑을 방지하려면 <code>true</code> 로 설정합니다. 기본값은 <code>true</code> 입니다.
<code>cluster.name</code>	Elasticsearch 클러스터의 고유한 이름을 지정합니다. 클러스터가 여러 개인 경우 각 클러스터의 이름은 고유해야 합니다. 클러스터에 다수의 노드가 있는 경우 클러스터의 각 노드에 동일한 클러스터 이름을 지정해야 합니다.
<code>discovery.zen.minimum_master_nodes</code>	다중 노드 클러스터에서 데이터 손실을 방지하고 클러스터 안정성을 유지하는 데 필요합니다. 다음 값으로 설정합니다. (마스터 적격 노드의 수/2) + 1 예를 들어 클러스터에 3개의 노드가 있고 이들 모두 마스터 노드로 사용하고 데이터를 포함할 수 있는 경우 속성을 (3/2) + 1(2로 반올림됨)로 설정합니다.
<code>discovery.zen.ping.unicast.hosts</code>	다중 노드 클러스터에 필요합니다. 이 속성은 클러스터 노드의 IP 주소 및 전송 포트 목록인 검색 설정을 지정하는 데 사용됩니다. 다음 형식을 사용하여 속성을 지정합니다. ["host1:port1", "host2:port2", "host3:port3"]
<code>http.port</code>	HTTP 요청의 포트입니다. 기본값은 9200입니다.
<code>network.host</code>	바인딩 주소로 사용할 호스트의 IP 주소입니다.
<code>node.data</code>	CRUD 및 검색 등 데이터 관련 작업을 수행하는 데이터 노드로 노드를 활성화합니다. 기본값은 <code>true</code> 입니다.
<code>node.ingest</code>	인덱싱 전에 데이터를 변환하고 보강하는 수집 노드로 노드를 활성화합니다. 기본값은 <code>true</code> 입니다.
<code>node.master</code>	클러스터를 제어하는 마스터 노드로 노드를 활성화합니다. 클러스터에 여러 개의 노드가 있는 경우 하나 이상의 노드를 마스터 노드로 활성화합니다.고가용성의 경우 여러 노드를 마스터 노드로 설정합니다. 기본값은 <code>true</code> 입니다.
<code>node.name</code>	노드의 고유한 이름을 지정합니다.
<code>path.data</code>	데이터를 저장하려는 디렉터리의 경로입니다. 여러 데이터 디렉터리를 구성할 수 있습니다. 여러 데이터 디렉터리 구성에 대한 자세한 내용은 Elasticsearch 설명서를 참조하십시오.
<code>path.logs</code>	로그 파일의 경로입니다.
<code>transport.tcp.port</code>	TCP 바인딩 포트입니다. 기본값은 9300입니다.

3. 속성 파일을 동일한 이름인 `elasticsearch.yml`로 저장합니다.

Elasticsearch 보호

Elasticsearch를 설치한 후 MDM Hub와 Elasticsearch 간 통신을 보호합니다. 또한 Elasticsearch 클러스터를 보호합니다.

Elasticsearch 보호에 대한 자세한 내용은 Elasticsearch 보안 설명서를 참조하십시오.

분석 플러그 인 설치

새로운 분석기, 토큰나이저, 토큰 필터 및 문자 필터를 추가하여 Elasticsearch를 확장하는 음성 및 일본어(kuromoji) 분석 플러그 인을 설치합니다. 음성 분석 플러그인은 토큰을 분석하여 해당하는 음성 값으로 변환합니다. 일본어(kuromoji) 분석 플러그인은 Kuromoji 분석기를 사용하여 일본어를 분석합니다.

1. 음성 및 일본어(kuromoji) 분석 플러그인을 Elastic 웹 사이트에서 다운로드합니다.
2. 다음 명령을 실행하여 각 클러스터 노드에서 음성 분석 플러그인을 설치합니다.
`sudo bin/elasticsearch-plugin install analysis-phonetic`
3. 다음 명령을 실행하여 각 클러스터 노드에서 일본어(kuromoji) 분석 플러그인을 설치합니다.
`sudo bin/elasticsearch-plugin install analysis-kuromoji`
4. 각 클러스터 노드를 다시 시작합니다.

검색에서 무시할 단어 목록 사용자 지정

검색을 수행할 때 MDM Hub는 "and", "an" 및 "is" 같은 일반적인 단어를 무시합니다. 검색에서 무시할 일반적인 단어의 기본 목록을 사용하거나 목록을 사용자 지정할 수 있습니다. 검색에서 무시할 단어 목록을 사용자 지정하려면 stopwords.txt 파일을 편집합니다.

1. 텍스트 편집기를 사용하여 다음 위치에서 stopwords.txt 파일을 엽니다.
`<elasticsearch 설치 디렉터리>/config/analysis`
2. stopwords.txt 파일을 편집하고 저장합니다.
3. stopwords.txt 파일을 편집하기 전에 데이터를 인덱싱한 경우 인덱스를 수동으로 삭제하고, Elasticsearch를 다시 시작한 다음 데이터를 다시 인덱싱합니다.

stopwords.txt 파일 업데이트에 대한 자세한 내용은 Elasticsearch 설명서를 참조하십시오.

검색에 포함할 동의어 목록 사용자 지정

검색을 수행할 때 MDM Hub는 지정된 검색 문자열의 동의어를 검색할 수 있습니다. 예를 들어 "William"을 검색하는 경우 동의어인 "Will"과 "Willy"가 검색 결과에 포함됩니다. synonyms.txt 파일에서 동의어를 정의할 수 있습니다.

검색에서 사용할 동의어를 사용자 지정하려면 stopwords.txt 파일을 편집합니다.

1. 텍스트 편집기를 사용하여 다음 위치에서 synonyms.txt 파일을 엽니다.
`<elasticsearch 설치 디렉터리>/config/analysis`
2. synonyms.txt 파일을 편집하고 저장합니다.
3. synonyms.txt 파일을 편집하기 전에 데이터를 인덱싱한 경우 인덱스를 수동으로 삭제하고, Elasticsearch를 다시 시작한 다음 데이터를 다시 인덱싱합니다.

synonyms.txt 파일 업데이트에 대한 자세한 내용은 Elasticsearch 설명서를 참조하십시오.

Elasticsearch 시작

Elasticsearch를 설정한 후 변경 내용을 적용하려면 Elasticsearch 클러스터의 각 노드를 시작합니다.

팁: Elasticsearch를 시작할 때 메모리 잠금 문제가 발생하면 `soft memlock unlimited` 및 `hard memlock unlimited`를 설정해야 할 수 있습니다.

1. 명령 프롬프트를 열고 다음 디렉터리로 변경합니다.

<elasticsearch 설치 디렉터리>/bin

2. 다음 명령을 실행합니다.

UNIX의 경우. `elasticsearch.sh`

Windows의 경우. `elasticsearch.bat`

2단계. 검색에 대한 MDM Hub 속성 구성

MDM Hub 속성을 구성하려면 Hub 콘솔, 처리 서버 속성 파일 및 Hub 서버 속성 파일을 사용합니다.

1. 처리 서버 속성을 구성합니다.
2. Hub 서버 속성을 구성합니다.

Hub 서버 속성 구성

모든 Hub 서버 인스턴스에서 검색을 활성화하도록 구성해야 합니다. Hub 콘솔의 Hub 서버 도구와 `cmxserver.properties` 파일을 사용하여 검색에 대한 Hub 서버 속성을 구성합니다.

1. 텍스트 편집기를 사용하여 다음 위치의 `cmxserver.properties` 파일을 엽니다. <MDM Hub 설치 디렉터리>\hub\server\resources\cmxserver.properties

2. 검색에 대한 다음 속성을 구성합니다.

cmx.ss.enabled

검색을 활성화할지 여부를 나타냅니다. 새 설치의 경우 기본값은 **true**입니다. 업그레이드의 경우 이 속성이 설정되어 있었다면 업그레이드 전 값이 유지됩니다. 이 속성을 설정하지 않으면 기본값이 **false**입니다.

ex.max.conn.per.host

호스트에 연결하려는 Elasticsearch 노드의 최대 수를 설정합니다. 호스트의 Elasticsearch 클러스터 노드 수로 설정합니다.

ex.max.threads

Elasticsearch 클러스터의 각 노드에 사용할 Apache 비동기식 비차단 수신기의 최대 스레드 수를 설정합니다. 기본값은 1입니다.

이 값은 Informatica 글로벌 고객 지원 센터에서 제안한 경우에만 변경합니다.

es.index.refresh.interval

처음에 스마트 검색 데이터 인덱싱 일괄 작업이 실행된 후 Elasticsearch에서 데이터 변경 내용을 커밋하는 간격(초)을 설정합니다. 이 시간 간격 후에 데이터를 검색에 사용할 수 있습니다. 기본값은 30입니다.

이 속성은 초기 인덱싱 중에 발생하는 높은 인덱싱 볼륨에 영향을 미칩니다. 이 값은 Informatica 글로벌 고객 지원 센터에서 제안한 경우에만 변경합니다.

cmx.e360.view.enabled

MDM 관리자가 Entity 360 프레임워크를 구현하는 경우 IDD 사용자는 **검색** 상자를 사용하여 레코드를 찾고 항목 탭을 사용하여 레코드를 편집 및 관리합니다. 새 설치의 경우 기본값은 **true**입니다. 업그레이드의 경우 이 속성이 설정되어 있었다면 업그레이드 전 값이 유지됩니다. 이 속성을 설정하지 않으면 기본값이 **false**입니다.

search.provisioning.numshards

선택 사항입니다. Elasticsearch 환경에 생성할 shard 수입니다. 최대 shard 수 및 총 노드 수에 따라 값이 달라집니다. 예를 들어, 최대 shard 수가 1이고 노드 수가 3인 경우 shard를 3개 작성할 수 있습니다. 기본값은 Hub 서버의 총 수입니다.

search.provisioning.numreplicas

선택 사항입니다. 서로 다른 노드에 생성할 Elasticsearch 엔진 문서의 사본 수입니다. 서로 다른 노드의 shard에 여러 개의 문서 사본을 생성하려면 복제 계수를 사용합니다. 하나 이상의 노드가 예기치 않게 종료된 경우 고가용성을 달성하려면 여러 개의 문서 사본이 필요합니다. 예를 들어 복제 계수가 2라면 두 개의 노드에서 두 개의 문서 사본을 얻을 수 있습니다. Elasticsearch의 경우 기본값은 0입니다.

cmx.task.search.records.return

사용자가 비즈니스 항목이 있는 Data Director의 태스크 관리자에서 태스크를 검색할 때 Elasticsearch 페이지 매기기를 제어합니다. 기본값은 1000입니다.

이 값은 Informatica 글로벌 고객 지원 센터에서 제안한 경우에만 변경합니다.

cmx.server.batch.smartsearch.initial.block_size

스마트 검색 데이터 초기 인덱싱 일괄 작업이 각 블록에서 처리할 수 있는 최대 레코드 수입니다. 기본값은 250입니다. 대규모 데이터 집합을 인덱싱할 때는 레코드 수를 늘리십시오. 권장되는 최대값은 1000입니다.

cmx.server.enrichcopager.thread_pool

속성을 수동으로 추가합니다. **EnrichCoPager** 속성이 스레드 풀에서 병렬 **ReadCO** 작업을 수행하기 위해 사용하는 스레드 수를 설정합니다. 기본값은 30입니다. 스레드 수를 1로 설정하면 속성이 비활성화됩니다.

cmx.server.enrichcopager.min_rec_for_multithreading

EnrichCoPager 속성에서 다중 스레드를 사용하기 전에 반환할 최소 레코드 수를 설정합니다. 기본값은 2입니다.

ssl.keyStore

응용 프로그램 서버의 **HTTPS** 포트를 사용하여 **Hub** 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 키 저장소 파일의 절대 경로 및 파일 이름입니다.

ssl.keyStore.password

응용 프로그램 서버의 **HTTPS** 포트를 사용하여 **Hub** 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 키 저장소 파일에 대한 일반 텍스트 암호입니다.

ssl.trustStore

응용 프로그램 서버의 **HTTPS** 포트를 사용하여 **Hub** 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 트러스트 저장소 파일의 절대 경로 및 파일 이름입니다.

ssl.trustStore.password

응용 프로그램 서버의 **HTTPS** 포트를 사용하여 **Hub** 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 트러스트 저장소 파일에 대한 일반 텍스트 암호입니다.

Hub 서버 속성을 업데이트한 후 **ORS**(연산 참조 저장소) 유효성을 검사하고 **Hub** 콘솔을 다시 시작해야 합니다.

처리 서버 속성 구성

모든 처리 서버 인스턴스에서 검색을 활성화하도록 구성합니다. **Hub** 콘솔의 처리 서버 도구와 **cmxcleanse.properties** 파일을 사용하여 검색에 대한 처리 서버 속성을 구성합니다.

1. 노드의 **Hub** 콘솔에서 처리 서버 도구를 시작합니다.
2. 쓰기 잠금 > 잠금 획득을 클릭합니다.
3. 처리 서버 도구의 오른쪽 창에서 **처리 서버 추가** 단추를 클릭합니다.
처리 서버 추가/편집 대화 상자가 표시됩니다.
4. 검색을 위해 다음 처리 서버 속성을 설정합니다.

속성	설명
서버	이 처리 서버가 배포된 응용 프로그램 서버의 IP 주소 또는 정규화된 호스트 이름입니다. 참고: localhost를 호스트 이름으로 사용하지 마십시오.
포트	이 처리 서버를 배포한 응용 프로그램 서버의 HTTP 또는 HTTPS 포트입니다.

속성	설명
Elasticsearch 처리	이 처리 서버에서 처음에 스마트 검색 데이터 인덱싱 일괄 작업을 처리할지 나타냅니다. 이 일괄 작업은 비즈니스 항목에서 검색 가능한 필드의 모든 값에 대한 인덱스를 생성합니다.
보안 연결(HTTPS)	이 처리 서버에서 HTTPS 프로토콜을 사용할지 나타냅니다. 선택하는 경우 포트 옵션이 HTTPS 포트 번호로 설정되어 있는지 확인합니다.

5. **확인**을 클릭한 다음 **저장**을 클릭합니다.
6. 텍스트 편집기를 사용하여 다음 위치에서 `cmxcleanse.properties` 파일을 엽니다.

<MDM Hub 설치 디렉터리>\hub\cleanse\resources\

7. 검색에 대한 다음 속성을 구성합니다.

`cmx.ss.enabled`

검색을 수행하는 데 처리 서버를 사용할지 여부를 나타냅니다. 기본값은 **False**입니다. 검색을 수행하는 데 처리 서버를 사용하려면 **true**로 설정합니다.

`ex.max.conn.per.host`

호스트에 연결하려는 Elasticsearch 노드의 최대 수를 설정합니다. 호스트의 Elasticsearch 클러스터 노드 수로 설정합니다.

`ex.max.threads`

Elasticsearch 클러스터의 각 노드에 사용할 Apache 비동기식 비차단 수신기의 최대 스레드 수를 설정합니다. 기본값은 1입니다.
이 값은 Informatica 글로벌 고객 지원 센터에서 제안한 경우에만 변경합니다.

`search.provisioning.numreplicas`

선택 사항입니다. 서로 다른 노드에 생성할 Elasticsearch 엔진 문서의 사본 수입니다. 서로 다른 노드의 **shard**에 여러 개의 문서 사본을 생성하려면 복제 계수를 사용합니다. 하나 이상의 노드가 예기치 않게 종료된 경우 고가용성을 달성하려면 여러 개의 문서 사본이 필요합니다. 예를 들어 복제 계수가 2라면 두 개의 노드에서 두 개의 문서 사본을 얻을 수 있습니다. Elasticsearch의 경우 기본값은 0입니다.

`MAX_INITIAL_RESULT_SIZE_TO_CONSIDER`

선택 사항입니다. 속성을 수동으로 추가합니다. Data Director 응용 프로그램에 표시할 검색 결과의 총 개수입니다. 권장되는 최대값은 250입니다. 기본값은 130입니다. 130보다 높은 값은 Data Director 응용 프로그램의 성능에 영향을 줍니다.

`ssl.keyStore`

응용 프로그램 서버의 HTTPS 포트를 사용하여 처리 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 키 저장소 파일의 절대 경로 및 파일 이름입니다.

`ssl.keyStore.password`

응용 프로그램 서버의 HTTPS 포트를 사용하여 처리 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 키 저장소 파일에 대한 일반 텍스트 암호입니다.

`ssl.trustStore`

응용 프로그램 서버의 HTTPS 포트를 사용하여 처리 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 트러스트 저장소 파일의 절대 경로 및 파일 이름입니다.

ssl.trustStore.password

응용 프로그램 서버의 HTTPS 포트를 사용하여 처리 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 트러스트 저장소 파일에 대한 일반 텍스트 암호입니다.

cmx.websphere.security.ssl.config.url

응용 프로그램 서버의 HTTPS 포트를 사용하여 처리 서버를 구성하는 경우 필수입니다. WebSphere에만 사용됩니다. 속성을 수동으로 추가합니다. 파일 이름을 포함한 ssl.client.props 파일의 절대 경로입니다.

8. cmxcleanse.properties 파일을 저장합니다.
9. 응용 프로그램 서버를 다시 시작합니다.

3단계. 프로비저닝 도구를 사용하여 검색 구성

Elasticsearch를 설정하고 MDM Hub 속성을 구성한 후 프로비저닝 도구를 사용하여 검색 환경을 구성합니다.

1. Elasticsearch 클러스터를 구성합니다.
2. 필요한 경우 사용자 지정 Elasticsearch 인덱스 설정을 생성합니다.
3. 검색 가능한 필드를 구성합니다.
4. 검색 및 쿼리 결과 표시를 구성합니다.
5. 필요한 경우 유사한 레코드를 표시하도록 레이아웃을 구성합니다.

Elasticsearch 클러스터 구성

프로비저닝 도구를 사용하여 MDM 응용 프로그램의 Elasticsearch 클러스터를 구성하십시오. 이 구성은 검색 API에 사용됩니다. 검색 API는 Data Director 응용 프로그램 및 모든 사용자 지정 응용 프로그램에서 사용됩니다.

참고: Elasticsearch 클러스터를 구성할 때는 클러스터의 마스터 노드만 지정해야 합니다.

1. 지원되는 브라우저를 열고 다음 URL을 입력합니다.
`https://<MDM Hub Server host name>:<MDM Hub Server port number>/provisioning/`
로그인 페이지가 나타납니다.
2. 사용자 이름 및 암호를 입력하고 **로그인**을 클릭합니다.
3. **데이터베이스** 목록에서 Elasticsearch 클러스터를 구성하려는 데이터베이스를 선택합니다.
4. **구성 > 인프라 설정**을 클릭합니다.
인프라 설정 페이지가 표시됩니다.
5. 목록에서 **Elasticsearch 클러스터**를 선택한 다음 **ESCluster**를 클릭합니다.
ESCluster가 트리 보기 패널에 나타납니다.
6. Elasticsearch 클러스터 노드를 구성하려면 트리 보기 패널에서 **esNode**를 선택하고 **생성**을 클릭합니다.

7. 구성된 Elasticsearch 클러스터의 다음 속성을 지정합니다.

속성	설명
이름	Elasticsearch 클러스터의 마스터 노드 이름입니다.
URL	Elasticsearch 클러스터의 마스터 노드 URL입니다. URL 형식은 <code>https://<호스트 이름>:<포트></code> 입니다.

8. **적용**을 클릭합니다.
9. 추가 마스터 노드를 생성하려면 **6~8** 단계를 반복합니다.
10. 변경 내용을 MDM Hub에 게시합니다.
 - a. **게시**를 클릭합니다.
변경 내용을 게시 또는 검토하라는 메시지를 표시하는 확인 대화 상자가 나타납니다.
 - b. 변경 내용을 검토하거나 검토 없이 게시합니다.
 - 검토 없이 게시하려면 **게시**를 클릭합니다.
 - 검토 후 게시하려면 **변경 내용 검토**를 클릭하고 화면에 나타나는 지시 사항을 따릅니다.

사용자 지정 Elasticsearch 인덱스 설정 생성(선택 사항)

Informatica에서 제공하는 Elasticsearch 인덱스 설정이 요구 사항을 만족하지 못하는 경우 사용자 지정 인덱스 설정을 생성할 수 있습니다. 사용자 지정 인덱스 설정에는 텍스트를 토큰 또는 조건으로 변환하는 분석기가 포함되어야 합니다. 이러한 토큰 또는 조건은 검색을 위해 반전된 인덱스에 추가됩니다.

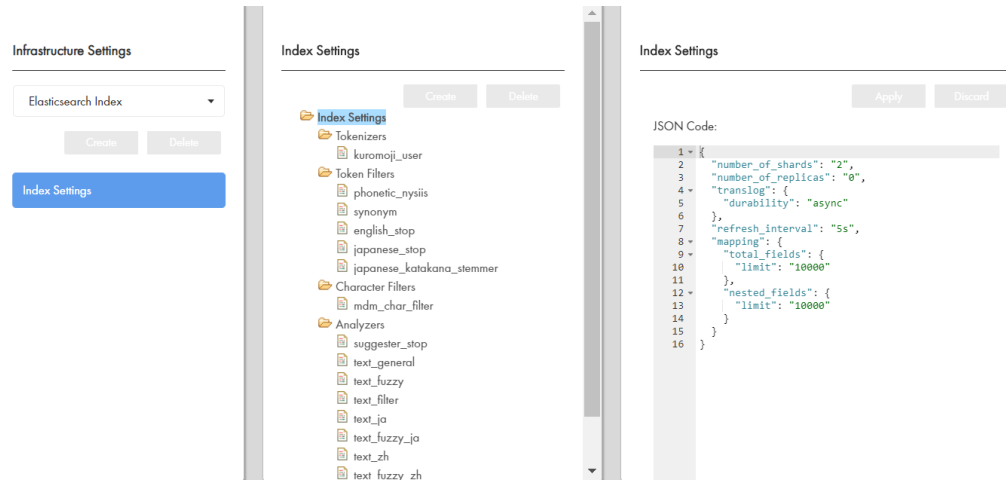
분석기에는 토큰나이저가 하나만 있어야 하며 0개 이상의 문자 필터 및 토큰 필터가 있을 수 있습니다. 토큰나이저는 토큰으로 변환되는 문자의 스트림을 수신합니다. 토큰 필터는 토큰나이저에서 생성되는 토큰의 스트림을 수신하며, 토큰을 추가, 제거 또는 변경할 수 있습니다. 문자 필터는 문자의 스트림을 수신하며, 스트림의 문자를 추가, 제거 또는 변경할 수 있습니다.

사용자 지정 분석기에서 사용하는 토큰나이저, 토큰 필터 및 문자 필터는 Informatica 기본, 사용자 지정 또는 Elasticsearch 기본 제공 구성 요소일 수 있습니다. 기본 설정은 편집할 수 없습니다. 분석기를 구성할 때 Elasticsearch 기본 제공 토큰나이저 및 토큰 필터를 선택할 수 있습니다.

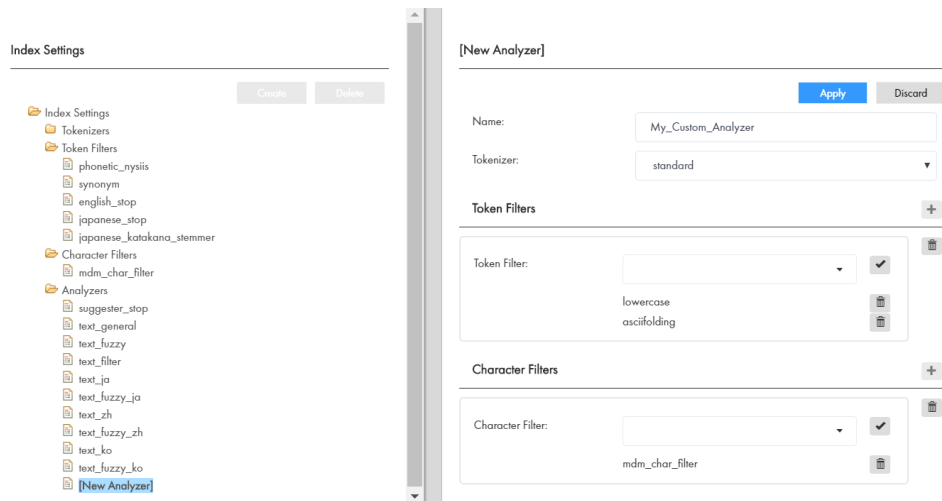
Elasticsearch 인덱스 설정에 대한 자세한 내용은 Elasticsearch 설명서를 참조하십시오.

1. 프로비저닝 도구에 로그인합니다.
2. **데이터베이스** 목록에서 Elasticsearch 인덱스 설정을 구성하려는 데이터베이스를 선택합니다.
3. **구성 > 인프라 설정**을 클릭합니다.
인프라 설정 페이지가 표시됩니다.
4. 인프라 설정 목록에서 **Elasticsearch 인덱스**를 선택하고 **인덱스 설정**을 클릭합니다.

트리 보기 패널에 **인덱스 설정**이 나타나고, 속성 패널에 해당 인덱스 설정에 대한 **JSON 코드** 상자가 나타납니다. 인덱스 설정이 변경되지 않은 경우 페이지는 기본 설정을 표시합니다.



5. **JSON 코드** 상자에 분석 모듈 이외의 모듈에 대한 인덱스 설정을 입력합니다. 또한 shard 수, 복제본 수, 새로 고침 간격과 같이 특정 인덱스 모듈과 연결되지 않은 인덱스 설정을 입력합니다.
6. 토크나이저, 토큰 필터 및 문자 필터와 같은 분석기 구성 요소를 구성합니다.
 - a. 트리 보기 패널에서 구성하려는 구성 요소를 선택하고 **생성**을 클릭합니다.
 - b. 속성 패널에서 구성 요소의 이름과 JSON 코드를 입력합니다.
 - c. **적용**을 클릭합니다.
7. 분석기를 구성합니다.
 - a. 트리 보기 패널에서 **분석기**를 선택한 다음 **생성**을 클릭합니다.
 - b. 속성 패널에서 분석기의 이름, 토크나이저, 토큰 필터 및 문자 필터를 지정합니다.
 분석기에서 사용하려는 순서대로 토큰 필터를 지정했는지 확인합니다.
 다음 이미지는 사용자 지정 분석기 구성의 예를 보여 줍니다.



- c. **적용**을 클릭합니다.

8. 변경 내용을 MDM Hub에 게시합니다.
 - a. **게시**를 클릭합니다.
변경 내용을 게시 또는 검토하라는 메시지를 표시하는 확인 대화 상자가 나타납니다.
 - b. 변경 내용을 검토하거나 검토 없이 게시합니다.
 - 검토 없이 게시하려면 **게시**를 클릭합니다.
 - 검토 후 게시하려면 **변경 내용 검토**를 클릭하고 화면에 나타나는 지시 사항을 따릅니다.
9. 응용 프로그램 서버 로그에서 인덱스 설정 관련 유효성 검사 오류를 검토하고 내용을 변경합니다.

Elasticsearch 기본 제공 토크나이저 및 토큰 필터

사용자 지정 분석기에 사용 가능한 Elasticsearch 기본 제공 토크나이저 및 토큰 필터를 선택할 수 있습니다.

다음과 같은 Elasticsearch 기본 제공 토크나이저를 사용자 지정 분석기에 사용할 수 있습니다.

- standard
- letter
- lowercase
- whitespace
- uax_url_email
- classic
- thai
- keyword

다음과 같은 Elasticsearch 기본 제공 토큰 필터를 사용자 지정 분석기에 사용할 수 있습니다.

- asciifolding
- standard
- lowercase
- uppercase
- porter_stem
- trim
- cjk_width
- cjk_bigram
- classic
- apostrophe
- kuromoji_baseform

사용자 지정 및 기본 제공 Elasticsearch 분석기 구성 요소에 대한 자세한 내용은 Elasticsearch 설명서를 참조하십시오.

검색 가능한 필드 구성

프로비저닝 도구를 사용하여 필드를 검색 가능한 필드로 구성하고 필드 속성을 설정할 수 있습니다. 검색 요청에서는 검색 가능한 필드로 구성된 필드만 검색합니다.

여러 검색 가능한 필드가 검색 요청의 성능에 영향을 미칠 수 있으므로 중요한 필드만 검색 가능한 필드로 구성하십시오. 예를 들어 국가 코드, 성별 코드 또는 주소 유형을 포함하는 필드보다는 전체 이름, 조직 이름 또는 전자 메일 주소를 포함하는 필드를 검색 가능한 필드로 구성하십시오.

1. 프로비저닝 도구에 로그인합니다.
2. **데이터베이스** 목록에서 필드를 구성하려는 데이터베이스를 선택합니다.
3. **비즈니스 항목 > 모델링**을 클릭합니다.
모델링 페이지가 나타납니다.
4. 목록에서 **비즈니스 항목**을 선택하고 검색 가능한 필드를 구성할 비즈니스 항목을 선택합니다.
5. 트리 보기의 비즈니스 항목 아래에서 **필드**를 선택하고 **생성**을 클릭합니다.
6. 요구 사항에 따라 다음 속성을 구성합니다.

이름

필드에 대한 프로비저닝 도구 트리 보기에 표시되는 이름입니다.

레이블

Data Director의 보기 내에 표시하려는 필드에 대한 레이블입니다.

읽기 전용

선택 사항입니다. 항목 보기의 필드에 대한 편집 가능 여부를 나타냅니다. 필드를 편집할 수 없는 필드로 구성하려면 속성을 선택합니다.

필수

선택 사항입니다. 필드가 필수 필드인지 여부를 나타냅니다. 필드를 필수 필드로 구성하려면 **필수**를 선택합니다. 기본적으로 필드는 필수 필드가 아닙니다.

URI

선택 사항입니다. 사용자 지정 데이터 유형이 정의된 네임스페이스입니다. 기본값은 **commonj.sdo**입니다.

유형

선택 사항입니다. 필드의 데이터 유형입니다. 기본적으로 필드의 데이터 유형은 필드와 연결하는 기본 개체 열의 데이터 유형과 동일합니다.

열

필드와 연결하려는 기본 개체 열의 이름입니다.

7. **검색 가능**을 선택합니다.
추가적인 필드 속성이 나타납니다.
8. 요구 사항을 기반으로 다음 속성 중 하나 이상을 선택합니다.
 - 검색 분석기
 - 제안기
 - 정렬 가능
 - 필터링 가능
 - 패킷 범위

- 패킷
 - 표시 가능
9. 필요한 경우 **패킷**을 선택하고 **패킷 범위** 필드에서 패킷으로 구성하는 숫자 또는 날짜 필드의 범위를 다음 형식으로 지정합니다.
- <Start Value>,<End Value>,<Frequency>
- 예: 1000,2000,50
- 참고:** 패킷 범위는 **Data Director** 응용 프로그램에 표시되지 않습니다. **REST** 웹 서비스를 사용하여 검색을 수행하는 경우 응답에서 패킷 범위가 반환될 수 있습니다.
10. **적용**을 클릭합니다.
11. 변경 내용을 **MDM Hub**에 게시합니다.
- 게시**를 클릭합니다.
변경 내용을 게시 또는 검토하라는 메시지를 표시하는 확인 대화 상자가 나타납니다.
 - 변경 내용을 검토하거나 검토 없이 게시합니다.
 - 검토 없이 게시하려면 **게시**를 클릭합니다.
 - 검토 후 게시하려면 **변경 내용 검토**를 클릭하고 화면에 나타나는 지시 사항을 따릅니다.

검색 가능 필드 속성

검색 가능 필드 속성을 구성하려면 프로비저닝 도구를 사용하거나 리포지토리에 변경 목록을 적용할 수 있습니다.

필터링 가능 패킷으로 검색 가능 참조 항목 필드를 활성화하면 **Data Director** 레코드 보기에 다음 형식으로 필터 필드 레이블이 표시됩니다.

<비즈니스 항목 조회 필드 레이블> - <참조 항목 조회 필드 레이블>

참조 항목에 대해 필터링 가능 속성을 구성하는 경우 필터링이 작동하려면 모든 종속 참조 항목에 대해 필터링 가능 속성을 구성해야 합니다.

다음 테이블에는 검색 가능 필드 속성이 설명되어 있습니다.

속성	설명
검색 가능	<p>검색 요청이 필드에서 검색 문자열을 검색할 수 있는지 나타냅니다. 검색 요청에 필드를 포함하려면 이 속성을 활성화합니다. 검색 요청에 필드를 포함하지 않으려면 이 속성을 비활성화합니다.</p> <p>검색 가능 속성이 활성화되어 있으면 검색에 대한 추가 속성을 구성할 수 있습니다.</p> <p>다음과 같은 추가 속성을 구성에 사용할 수 있습니다.</p> <ul style="list-style-type: none"> - 검색 분석기 - 제안기 - 정렬 가능 - 필터링 가능 - 패킷 범위 - 패킷 - 표시 가능
검색 분석기	<p>필드에 대해 사용할 사용자 지정 검색 분석기를 지정합니다. 필드에 포함할 데이터의 유형을 기반으로 적절한 검색 분석기를 결정합니다.</p>

속성	설명
제안기	필드의 값을 Data Director 응용 프로그램에서 검색 문자열로 제안할지 여부를 나타냅니다. 필드의 값을 검색 문자열로 제안하려면 이 속성을 활성화합니다. 필드의 값을 검색 문자열로 제안하지 않는 경우 이 속성을 비활성화합니다. 중요: 데이터 보안을 활성화하려면 중요한 데이터를 포함하는 필드에 대해 제안기 속성을 활성화하지 마십시오.
정렬 가능	이 속성을 사용하지 마십시오.
필터링 가능	필드에서 필터링을 활성화할지 나타냅니다. Data Director 응용 프로그램이 검색 작업 공간에서 필터링 가능한 필드를 필터로 표시합니다. 필드를 필터로 구성하려면 이 속성을 활성화합니다. 필드를 필터로 구성하지 않으려면 이 속성을 비활성화합니다.
패킷 범위	패킷으로 구성하는 숫자 또는 날짜 필드의 범위를 나타냅니다. 범위를 지정할 때는 다음 형식을 사용합니다. <Start Value>,<End Value>,<Frequency> 범위에서 시작 값은 포함되며 종료 값은 제외됩니다. 예를 들어 정수 필드에 대해 패킷 범위를 1000,2000,500으로 설정할 경우 검색 요청에서 다음 범위가 반환됩니다. [1000 to 1500] [1500 to 2000] 1000 ~ 1500 범위에는 1000부터 1499까지의 값이 포함되고 1500 ~ 2000 범위에는 1500부터 1999까지의 값이 포함됩니다. 범위에 대해 유효한 최소값 및 최대값을 설정하고 범위의 수를 10으로 제한하는 오프셋을 설정했는지 확인합니다. 음수에 대한 패킷을 구성할 수 없지만, 검색 요청에서는 음수 값을 표시합니다. 날짜 필드의 경우 빈도에 Y M D 접미사를 추가합니다. 여기서 Y는 년, M은 월, D는 일을 나타냅니다. 예를 들어 2M은 2개월을 나타냅니다.
패킷	필드를 패킷으로 설정할지 나타냅니다. 패킷 필드는 검색 결과 값을 그룹화하고 각 그룹의 수를 표시합니다. Data Director 응용 프로그램이 패킷 필드, 검색 결과를 기반으로 그룹화된 필드 값 및 각 그룹의 수를 검색 작업 공간에 표시합니다. 하위 레코드 필드가 패킷 필드로 설정된 경우 Data Director 응용 프로그램은 패킷 필드에 대한 도구 설명을 표시합니다. 도구 설명 텍스트의 형식은 <하위 레코드 이름>/<하위 레코드 필드 이름>입니다. 패킷 속성은 필터링 가능 속성과 함께 작동하므로 필드를 패킷으로 구성하려는 경우 필터링 가능 속성을 활성화합니다. 필드를 패킷으로 구성하지 않으려면 패킷 속성을 비활성화합니다.
표시 가능	이 속성을 사용하지 마십시오.

검색 또는 쿼리 결과 표시 구성

프로비저닝 도구를 사용하여 검색에 사용할 비즈니스 항목 보기를 구성할 수 있습니다. 검색 결과에는 검색 결과에 구성된 비즈니스 항목 보기의 일부인 필드만 포함됩니다. 검색 필터가 표시되는 순서를 구성할 수도 있습니다.

검색 가능한 보기를 구성하기 전에 검색 결과에 사용하려는 비즈니스 항목 보기를 생성합니다.

참고: 검색 결과에 비즈니스 항목의 하위 레코드 필드를 표시하려면 비즈니스 항목에서 변환된 비즈니스 항목 보기를 사용합니다. 보기가 루트 레코드 수준의 하위 레코드 필드를 포함하는지 확인합니다.

1. 프로비저닝 도구에 로그인합니다.
2. **데이터베이스** 목록에서 응용 프로그램에 연결된 데이터베이스를 선택합니다.

3. **구성 > 응용 프로그램 편집기**를 클릭합니다.
응용 프로그램 페이지가 표시됩니다.
4. **응용 프로그램** 목록에서 검색을 구성하려는 응용 프로그램을 선택합니다.
응용 프로그램이 없는 경우 검색을 구성하기 전에 생성합니다.
5. 트리 보기 패널에서 **검색 구성**을 선택한 다음, **생성**을 클릭합니다.
6. 속성 패널에서 비즈니스 항목을 선택하고 검색 또는 쿼리 결과를 표시하는 데 사용할 비즈니스 항목 보기를 선택합니다.
비즈니스 항목 보기를 선택하지 않은 경우 검색 및 쿼리 결과에 모든 비즈니스 항목 필드가 포함됩니다.
7. 필요한 경우, 검색을 구성했다면 필터를 선택하고 검색 필터의 표시 순서를 구성합니다.
 - a. **필터 표시 순서** 옆에 있는 **편집** 아이콘을 클릭합니다.
필터 표시 순서 편집 대화 상자가 나타납니다. 대화 상자에는 필터가 포함되어 있으며, 이러한 필터는 비즈니스 항목 모델에서 필터링 가능으로 구성된 필드입니다.
 - b. **사용 가능한 필터** 섹션에서 **선택한 필터** 섹션으로 필터를 끌어다 놓습니다.
 - c. 순서를 구성하려면 필터를 위 또는 아래로 끌어 이동합니다.
 - d. **확인**을 클릭합니다.
8. **적용**을 클릭합니다.
검색 구성이 임시 작업 공간에 저장됩니다.
9. 변경 내용을 **MDM Hub**에 게시합니다.
 - a. **게시**를 클릭합니다.
변경 내용을 게시 또는 검토하라는 메시지를 표시하는 확인 대화 상자가 나타납니다.
 - b. 변경 내용을 검토하거나 검토 없이 게시합니다.
 - 검토 없이 게시하려면 **게시**를 클릭합니다.
 - 검토 후 게시하려면 **변경 내용 검토**를 클릭하고 화면에 나타나는 지시 사항을 따릅니다.

유사한 레코드를 표시하도록 레이아웃 구성(선택 사항)

Data Director 응용 프로그램에 데이터를 입력하여 레코드를 생성할 때 입력한 데이터를 기반으로 검색되는 유사한 레코드를 볼 수 있습니다. 유사한 레코드를 보려면 레이아웃을 구성하여 유사한 레코드를 검색할 기준 필드를 정의해야 합니다.

1. 프로비저닝 도구에 로그인합니다.
2. **데이터베이스** 목록에서 응용 프로그램을 구성하려는 데이터베이스를 선택합니다.
3. **구성 > 구성 요소 편집기**를 클릭합니다.
구성 요소 편집기가 표시됩니다.
4. 구성 요소 유형 목록에서 **유사한 레코드**를 선택한 다음, **생성**을 클릭합니다.
5. 속성 패널에서 유사한 레코드 구성 요소의 이름을 입력합니다.
6. **XML** 필드에 유사한 레코드를 검색할 필드 목록이 포함된 XML 구성을 입력합니다.

다음 테이블에는 유사한 레코드 구성 요소를 구성하는 데 사용할 수 있는 XML 요소가 설명되어 있습니다.

요소	설명
searchableFields	검색 기반으로 사용할 하나 이상의 필드를 지정합니다. searchableFields 요소는 필드 이름 요소의 상위입니다.
필드 이름	유사한 레코드 검색의 기반이 되는 필드의 이름을 지정합니다. 필드 이름 요소는 searchableFields 요소의 하위입니다. 여러 필드 이름 요소를 구성할 수 있습니다.
searchType	수행할 검색 유형을 지정합니다. searchType 요소는 다음과 같은 하위 요소를 포함할 수 있습니다. - smartSearch - searchMatch
smartSearch	유사한 레코드를 찾기 위해 검색을 사용하고자 함을 지정합니다.
searchMatch	유사한 레코드를 찾기 위해 쿼리를 사용하고자 함을 지정합니다. searchMatch 요소는 다음과 같은 하위 요소를 포함할 수 있습니다. - fuzzy - matchRuleSet
fuzzy	유사 항목 검색을 수행할지 여부를 지정합니다. 유사 항목 검색을 수행하려면 true로 설정합니다. fuzzy 요소는 searchMatch 요소의 하위입니다. 이 요소를 추가하지 않는 경우 정확한 검색이 수행됩니다.
matchRuleSet	유사한 레코드 찾기에 사용할 일치 규칙 집합의 이름을 지정합니다. matchRuleSet 요소는 searchMatch 요소의 하위입니다.
label	검색 필드 값의 레이블 형식을 지정합니다. 레이블 형식을 구성하려면 existsFormat 특성을 사용합니다.
column	레이블 형식에 사용할 단일 열을 지정합니다. 레이블에 대한 열을 구성하려면 열의 고유 식별자인 columnId 특성을 사용합니다. column 요소는 label 요소의 하위입니다. 한 레이블에 두 개 이상의 열을 지정할 수 있습니다.

샘플 구성에 대해서는 *Multidomain MDM 프로비저닝 도구 가이드*를 참조하십시오.

7. **적용**을 클릭합니다.

생성한 유사한 레코드 구성 요소가 **구성 요소 편집기** 패널과 트리 보기 패널에 표시됩니다.

8. 변경 내용을 **MDM Hub**에 게시합니다.

a. **게시**를 클릭합니다.

변경 내용을 게시 또는 검토하라는 메시지를 표시하는 확인 대화 상자가 나타납니다.

b. 변경 내용을 검토하거나 검토 없이 게시합니다.

- 검토 없이 게시하려면 **게시**를 클릭합니다.
- 검토 후 게시하려면 **변경 내용 검토**를 클릭하고 화면에 나타나는 지시 사항을 따릅니다.

4단계. 연산 참조 저장소 유효성 검사

Elasticsearch 구성의 영향을 받는 ORS(연산 참조 저장소)의 메타데이터 유효성을 검사하려면 Hub 콘솔에서 리포지토리 관리자 도구를 사용합니다.

1. Hub 콘솔을 시작하고 MDM Hub 마스터 데이터베이스에 연결합니다.
2. 구성 작업 영역을 확장하고 리포지토리 관리자를 클릭합니다.
리포지토리 관리자가 나타납니다.
3. 유효성 검사 탭을 클릭하고 유효성을 검사할 리포지토리를 선택합니다.
4. 유효성 검사를 클릭합니다.
유효성 검사 선택 대화 상자가 나타납니다.
5. 수행할 유효성 검사를 선택합니다.
6. 확인을 클릭합니다.
리포지토리 관리자가 리포지토리의 유효성을 검사하고 문제가 있으면 발견된 문제 창에 모두 표시합니다.
7. 문제를 복구하려면 복구를 클릭합니다.

5단계. 검색 데이터 인덱싱

환경에 데이터가 포함되는 경우 처음에 스마트 검색 데이터 인덱싱 일괄 작업을 수동으로 실행하여 데이터를 인덱싱합니다. 환경에 데이터가 포함되지 않는 경우 처음에 스마트 검색 데이터 인덱싱 작업을 실행하지 않아도 됩니다. 로드 일괄 작업을 시행하여 데이터를 로드하면 로드 일괄 작업이 처음에 스마트 검색 데이터 인덱싱 일괄 작업을 자동으로 실행하고 데이터를 인덱싱합니다. 검색 요청은 인덱스를 사용하여 레코드를 검색합니다.

비즈니스 항목에 기여하는 모든 기본 개체에 대해 처음에 스마트 검색 데이터 인덱싱 일괄 작업을 실행합니다. 처음에 스마트 검색 데이터 인덱싱 일괄 작업을 기본 개체에 실행하면 Elasticsearch 서버가 검색 가능한 필드의 데이터를 인덱싱합니다. 인덱싱된 데이터는 검색 가능한 필드가 속하는 비즈니스 항목을 나타내는 모든 컬렉션에 추가됩니다. 컬렉션이 너무 큰 경우 컬렉션을 하나 이상의 shard로 분할할 수 있습니다. shard는 여러 노드에서 분할된 컬렉션의 논리적 단위입니다. 검색을 수행하면 Elasticsearch 서버가 컬렉션을 읽고 일치 필드를 반환합니다.

처음에 스마트 검색 데이터 인덱싱 일괄 작업은 작업에서 모든 레코드에 대한 인덱싱 요청을 대기열에 추가한 후에 레코드를 비동기적으로 인덱싱하고 작업의 성공적인 완료를 보고합니다. 검색 요청에서 인덱싱된 레코드는 인덱스 요청이 완료된 후에만 표시되며 몇 분이 소요될 수 있습니다.

중요: 데이터를 인덱싱한 후 필드의 검색 가능한 속성을 업데이트하는 경우 인덱스가 삭제됩니다. 처음에 스마트 검색 데이터 인덱싱 일괄 작업을 실행하여 데이터를 인덱싱해야 합니다. 또한 인덱싱 프로세스는 리소스 소모가 많은 프로세스이므로 처음에 스마트 검색 데이터 인덱싱 일괄 작업 여러 개를 동시에 실행하지 마십시오.

키 저장소, 트러스트 저장소 및 인증서 생성(선택 사항)

Elasticsearch를 설치한 후 MDM Hub와 Elasticsearch 간의 통신 보안에 필요한 키 저장소, 트러스트 저장소 및 보안 인증서를 생성할 수 있습니다. 키 저장소, 트러스트 저장소 및 인증서를 생성하려면 Hub 서버가 설치된 시

스텝 중 하나에서만 sip_ant 스크립트를 실행합니다. 그런 다음 키 저장소, 트러스트 저장소 및 인증서를 Hub 서버가 설치된 다른 모든 시스템에 복사합니다.

참고: sip_ant 스크립트를 사용하지 않고 키 저장소, 트러스트 저장소 및 인증서를 생성할 수 있습니다.

다음 테이블에는 필요한 키 저장소 및 트러스트 저장소가 설명되어 있습니다.

키 저장소/트러스트 저장소 이름	설명
MDM_ESCLIENT_FILE_JKS.keystore	클라이언트 인증서 및 키를 포함하는 Elasticsearch 키 저장소입니다.
MDM_ESKEYSTORE_FILE_JKS.keystore	클라이언트 및 노드 인증서를 포함하는 Elasticsearch 키 저장소입니다. Elasticsearch 클러스터에 여러 개의 노드가 있는 경우 모든 노드에 인증서가 사용됩니다.
MDM_ESTRUSTSTORE_FILE_JKS.keystore	클라이언트 및 Elasticsearch 노드에 대한 서명된 인증서를 포함하는 Elasticsearch 트러스트 저장소입니다.

1. 명령 프롬프트를 열고 Hub 서버가 설치된 시스템 중 하나의 다음 디렉터리로 이동합니다.
<MDM Hub 설치 디렉터리>/hub/server/bin
2. 키 저장소, 트러스트 저장소 및 인증서를 생성하려면 다음 명령을 실행합니다.
UNIX의 경우. sip_ant.sh generate_mdm_es_store
Windows의 경우. sip_ant.bat generate_mdm_es_store
3. 키 저장소 및 트러스트 저장소의 암호를 묻는 메시지가 표시되면 암호를 지정합니다.
키 저장소, 트러스트 저장소 및 인증서가 다음 디렉터리에 생성됩니다.
<MDM Hub 설치 디렉터리>/hub/server/resources/certificates
4. 다음 키 저장소 및 트러스트 저장소를 각 Elasticsearch 설치의 <Elasticsearch 설치 디렉터리>/config 디렉터리에 복사합니다.
 - MDM_ESCLIENT_FILE_JKS.keystore
 - MDM_ESKEYSTORE_FILE_JKS.keystore
 - MDM_ESTRUSTSTORE_FILE_JKS.keystore
5. 다음 키 저장소 및 트러스트 저장소를 Elasticsearch 클러스터에 포함되는 각 Hub 서버 노드의 <MDM Hub 설치 디렉터리>/hub/server/resources/certificates 디렉터리에 복사합니다.
 - MDM_ESCLIENT_FILE_JKS.keystore
 - MDM_ESTRUSTSTORE_FILE_JKS.keystore

제 24 장

통합 프로세스 구성

이 장에 포함된 항목:

- [통합 프로세스 구성 개요, 475](#)
- [통합 설정, 475](#)
- [통합 설정 변경, 479](#)

통합 프로세스 구성 개요

통합 프로세스는 일치 쌍을 단일 마스터 레코드로 병합합니다. 일치 프로세스를 구성한 후 MDM Hub 구현을 위해 통합 프로세스를 구성합니다.

통합 프로세스를 구성하려면 일치/병합 설정 세부 정보 페이지에서 병합 설정 탭을 사용합니다. 소스 시스템 특성을 지정하고 레코드 병합 해제 방법을 지정할 수 있습니다.

비즈니스 항목 서비스 또는 Data Director를 사용하여 레코드를 병합하는 경우 사용자는 병합할 레코드의 값을 재정의할 수 있습니다. 사용자가 병합할 레코드의 값을 재정의하고 워크플로우가 구성된 경우 태스크 작업을 대기 중인 보류 중인 레코드의 세부 정보는 보류 중인 제어 테이블에 저장됩니다.

통합 설정

통합 설정은 Informatica MDM Hub의 통합 프로세스 동작에 영향을 줍니다. 이 섹션에서는 일치/병합 설정 세부 정보 페이지의 병합 설정 탭에서 구성할 수 있는 설정에 대해 설명합니다.

변경할 수 없는 Rowid 개체

지정된 기본 개체에 대해 소스 시스템을 변경할 수 없는 소스로 지정할 수 있습니다. 즉, 해당 소스 시스템의 레코드가 고유 레코드(CONSOLIDATION_IND = 1)로 허용되며, 이는 병합이 발생하는 경우에도 마찬가지입니다. 해당 소스의 레코드가 완전히 통합한 후에는 더 이상 변경되지 않으며 다른 레코드가 해당 레코드와 일치될 수 있는 경우에도 다른 레코드와 일치되지 않습니다. 소스 시스템 하나만 변경할 수 없는 소스로 구성할 수 있습니다.

참고: 하위 기본 개체에 대한 "상위 항목 병합 시 대기열에 다시 넣기" 설정이 "통합되지 않은 항목만"으로 설정된 경우 상위 항목 병합이 발생하면 하위 레코드에 대한 통합 표시기가 1로 설정되어 있는 레코드를 제외한 레코드의 통합 표시기가 4로 설정됩니다. 통합 표시기가 1로 설정된 하위 레코드를 대기열에 다시 넣으려면 수동으로 "상위 항목 병합 시 대기열에 다시 넣기" 설정을 2로 설정해야 합니다.

변경할 수 없는 소스는 고유 시스템이기도 합니다. 모든 레코드가 Informatica MDM Hub에서 마스터 레코드로 저장됩니다. 변경할 수 없는 소스 시스템의 모든 소스 레코드에 대해 로드 및 PUT의 통합 표시기는 항상 1(통합된 레코드)입니다.

기본 개체에 대한 변경할 수 없는 소스를 지정하려면 변경할 수 없는 Rowid 개체 옆에 있는 드롭다운 목록을 클릭하고 소스 시스템을 선택합니다.

이 목록에 이 기본 개체와 연결된 소스 시스템이 표시됩니다. 소스 시스템 하나만 변경할 수 없는 소스 시스템으로 지정될 수 있습니다.

예를 들어 Informatica MDM Hub가 소스 데이터에 대한 유일한 지속형 저장소인 경우 변경할 수 없는 소스 시스템을 적용할 수 있습니다. 변경할 수 없는 소스 시스템을 지정하면 소스 내부의 일치가 방지되고 변경할 수 없는 소스의 레코드를 자동으로 고유 레코드로 허용할 수 있으므로 로드, 일치 및 병합 프로세스가 간소화됩니다. 변경할 수 없는 레코드 두 개를 병합해야 하는 경우에는 데이터 스튜어드가 수동 확인을 수행하여 변경을 허용해야 합니다. 이 작업을 수행할 때 데이터 스튜어드가 Informatica MDM Hub를 사용하여 유지할 키를 선택할 수 있습니다.

고유 시스템

고유 시스템은 통합되지 않고 기본 개체에 삽입되는 데이터를 제공합니다. 고유 시스템의 레코드는 동일한 시스템의 다른 레코드와 일치하지 않지만 다른 시스템의 다른 레코드(로드 시 CONSOLIDATION_IND가 4로 설정됨)와는 서로 일치시킬 수 있습니다. 고유 소스 시스템을 지정하고 각 소스 시스템에 대해 레코드를 자동 또는 수동으로 통합할지를 구성할 수 있습니다.

고유 소스 시스템

소스 시스템을 끝은 소스라고도 하는 고유 소스로 지정할 수 있습니다. 그러면 해당 소스의 레코드가 병합되지 않습니다. 예를 들어 ABC 소스가 고유 소스로 지정된 경우 일치 규칙이 이 소스에서 가져오는 두 레코드를 일치시키거나 병합하지 않습니다. 고유 소스의 레코드는 자동 일치 및 병합 프로세스에서 임시 일치를 통해 일치되지 않습니다. 이러한 레코드는 일치 대상 플래그를 지정해서 수동으로만 병합할 수 있습니다.

고유 소스 시스템을 지정하려면

1. 병합 설정 탭의 소스 시스템 목록에서 레코드의 병합을 방지하도록 시스템 내 병합을 허용하지 않을 소스 시스템을 선택합니다.
2. 각 고유 소스 시스템에 대해 자동 규칙만을 사용하도록 할지 여부를 지정합니다.

자동 규칙만

고유 시스템에만 사용할 수 있으며, 이 옵션을 활성화하면 연결된 고유 소스 시스템에 대해 실행되는 규칙 유형을 구성할 수 있습니다. Informatica MDM Hub가 이 고유 시스템에 대해 수동 통합 규칙이 아니라 자동 통합 규칙만 적용하도록 하려면 이 확인란을 선택합니다. 기본적으로 이 옵션은 비활성화(선택 취소)되어 있습니다.

상위 항목이 병합 해제된 경우 하위 항목 병합 해제(계단식 병합 해제)

중요: 이 기능은 일치 규칙 및 외래 키가 구성된 하위 기본 개체에만 적용됩니다.

하위 기본 개체의 경우 Informatica MDM Hub에서는 상위 기본 개체의 레코드가 병합 해제된 경우 수행할 작업을 지정할 수 있는 계단식 병합 해제 기능을 제공합니다. 기본적으로 이 기능은 비활성화되어 있으므로 상위 레코드를 병합 해제해도 연결된 하위 레코드는 병합 해제되지 않습니다. 병합 설정 탭의 아래쪽에 있는 "상위 항목이 병합 해제된 경우 하위 항목 병합 해제"에서 하위 기본 개체에 대해 "계단식 병합 해제" 확인란을 선택한 경우 상위 개체의 레코드가 병합 해제되면 Informatica MDM Hub에서는 하위 기본 개체의 레코드 중 영향을 받는 레코드도 병합 해제합니다.

계단식 병합 해제를 위한 선행 조건

계단식 병합 해제를 활성화하려면

- 하위 기본 개체에서 상위-하위 관계가 이미 구성되어 있어야 합니다.
- 하위 기본 개체의 외래 키 열이 일치 활성화된 열이어야 합니다.

[병합 설정] 탭 하단의 [상위 항목이 병합 해제된 경우 하위 항목 병합 해제]에서 스키마 관리자에는 외래 키로 구성된, 하위 기본 개체의 일치 활성화된 열만 표시됩니다.

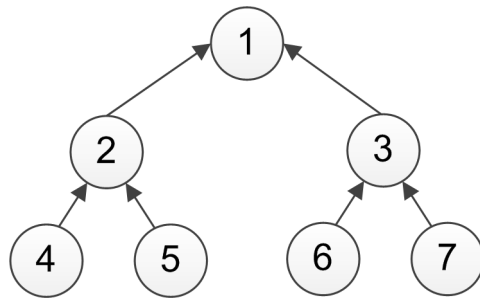
하위 기본 개체 계단식 병합 해제 동작

계단식 병합 해제 프로세스가 하위 기본 개체에 미치는 영향은 상위 기본 개체에 대해 트리 병합 해제 프로세스가 실행되었는지 선행 병합 해제 프로세스가 실행되었는지에 따라 달라집니다.

다음 목록에서는 계단식 병합 해제 프로세스가 실행되기 전과 후의 상위 기본 개체 레코드 및 하위 기본 개체 레코드의 구성을 보여 줍니다.

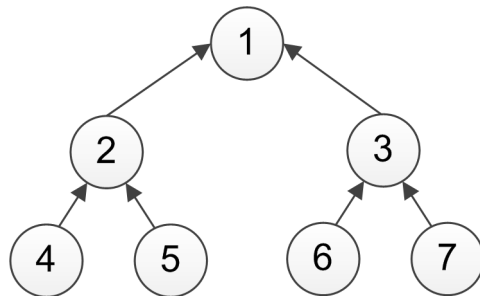
병합된 상위 기본 개체 레코드

다음 그림에서는 레코드 '2'부터 '7'까지의 데이터로 구성된 BVT(최선의 진실, Best Version of the Truth)에 해당하는 상위 기본 개체 레코드 '1'을 보여 줍니다.



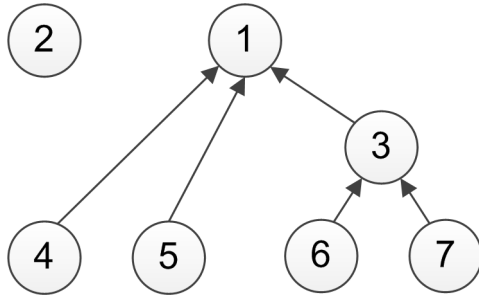
병합된 하위 기본 개체 레코드

다음 그림에서는 상위 기본 개체 레코드와 동일한 데이터 구조를 갖는 하위 기본 개체 레코드 '1'을 보여 줍니다. 일반적으로는 하위 기본 개체 레코드의 구조와 상위 기본 개체 레코드의 구조가 다릅니다.



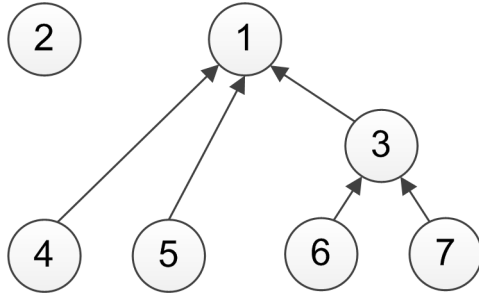
선형 병합 해제 후의 상위 기본 개체

다음 그림에서는 레코드 '2'가 선형 병합 해제 프로세스를 거친 후 별도의 기본 개체가 된 결과를 보여 줍니다.



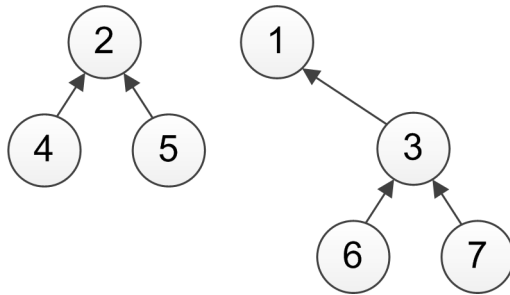
선형 병합 해제 후의 하위 기본 개체

상위 기본 개체 레코드에 대해 선형 병합 해제 프로세스를 실행하면 하위 기본 개체 레코드에도 동일한 프로세스가 실행됩니다. 다음 그림에서는 상위 기본 개체 레코드에 대해 선형 병합 해제 프로세스를 실행할 때 하위 기본 개체 레코드에 미치는 영향을 보여 줍니다.



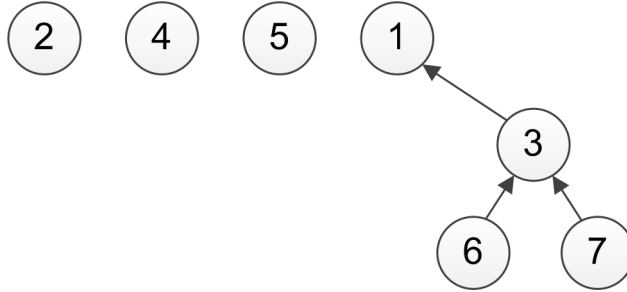
트리 병합 해제 후의 상위 기본 개체

다음 그림에서는 레코드 '2'가 트리 병합 해제 프로세스를 거친 후 트리 구조를 갖는 별도의 기본 개체가 된 결과를 보여 줍니다.



트리 병합 해제 후의 하위 기본 개체

상위 기본 개체 레코드에 대해 트리 병합 해제 프로세스를 실행할 경우 하위 기본 개체의 병합 해제된 트리에는 선형 병합 해제 프로세스가 실행됩니다. 다음 그림에서는 상위 기본 개체 레코드에 대해 트리 병합 해제 프로세스를 실행할 때 하위 기본 개체 레코드에 미치는 영향을 보여 줍니다.



여러 개의 하위 항목이 있는 상위 항목

상위 기본 개체에 여러 하위 기본 개체가 있는 상황에서는 각 하위 기본 개체에 대해 계단식 병합 해제를 명시적으로 활성화할 수 있습니다. 계단식 병합 해제를 구성하면 상위 기본 개체의 병합을 해제할 경우 연결된 모든 하위 기본 개체에서 영향을 받는 모든 레코드도 병합 해제됩니다.

계단식 병합 해제를 사용하기 위한 고려 사항

영향을 받는 레코드의 전체 병합 해제는 모든 구현에서 반드시 필요하지는 않으며, 여러 하위 레코드에 영향을 줄 수 있으므로 병합 해제 시 성능 오버헤드가 발생할 수 있습니다. 또한 이 속성을 활성화하는 것이 항상 적절한 것은 아닙니다. 한 가지 예로 **Customer**가 **Customer Type**의 하위 항목이라면 **Customer Type**이 병합 해제될 때 **Customer**가 병합 해제되지 않도록 할 수 있습니다. 그러나 대부분의 경우에는 **Customer**가 병합 해제될 때 **Customer**에 연결된 주소도 병합 해제하는 것이 좋습니다.

참고: 계단식 병합 해제를 활성화하면 하위 기본 개체에 대해 수동 병합 해제가 수행된 후 하위 레코드가 병합 해제되지 않을 수 있습니다.

병합 해제 기능을 활성화하면 하위 테이블과 하위 교차 참조 테이블에 해당 기능이 적용됩니다. 이 기능을 활성화한 후 상위 교차 참조를 병합 해제하면 원래의 하위 교차 참조도 병합 해제됩니다. 이 기능은 하위 테이블에 대해서만 작동하고 상위 테이블에는 영향을 주지 않으므로 유연성이 높습니다.

통합 설정 변경

병합 설정 탭에서 통합 설정을 변경하려면

1. 스키마 관리자에서 구성하려는 기본 개체에 대한 일치/병합 설정 세부 정보 대화 상자를 표시합니다.
2. 쓰기 잠금을 획득합니다.
3. **병합 설정** 탭을 클릭합니다.
스키마 관리자에 선택한 기본 개체에 대한 병합 설정 탭이 표시됩니다.
4. 다음 설정을 변경합니다.
 - 변경할 수 없는 Rowid 개체
 - 고유 시스템
 - 상위 항목이 병합 해제된 경우 하위 항목 병합 해제(계단식 병합 해제)
5. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

제 25 장

보류 중인 제어 테이블

보류 중인 제어 테이블은 MDM Hub이 기본 개체에 대해 자동으로 생성하는 시스템 테이블입니다. 이 테이블은 병합 태스크 작업을 대기 중인 트러스트된 값에 대한 재정의로 레코드에 대한 정보를 추적합니다. 테이블 이름의 형식은 C_<기본 개체 이름>_PCTL입니다. 보류 중인 제어 테이블에는 제어 테이블의 모든 열과 상호 작용 ID 저장장을 위한 추가 열이 포함되어 있습니다.

보류 중인 제어 테이블은 태스크 워크플로우가 병합 작업에 대해 구성되었을 때 사용됩니다. 이 테이블은 비즈니스 항목 서비스 API에서만 사용되며 SIF(서비스 통합 프레임워크) API 및 일괄 작업에서는 사용되지 않습니다.

사용자가 레코드의 트러스트된 값을 수동으로 재정의할 때 병합 프로세스가 보류 중인 제어 테이블에 레코드를 삽입합니다. 병합 작업을 보류 중인 레코드는 상호 작용 ID에 의해 보호됩니다. 트러스트가 활성화된 각 레코드 열의 보류 중인 제어 테이블에는 4개의 열이 있습니다. 사용자가 수동으로 올바른 값을 입력할 때 교차 참조 레코드가 보류 중 상태로 생성됩니다. 병합 태스크가 승인되면 보류 중인 교차 참조 레코드와 보류 중인 재정의가 승격되고 레코드가 병합됩니다.

다음 테이블에는 보류 중인 제어 테이블의 열이 설명되어 있습니다.

열 이름	설명
INTERACTION_ID	병합 태스크의 일부인 관련 트러스트 재정의의 그룹화하는 식별자입니다. 병합 태스크 작업을 완료하는 데 필요합니다.
<트러스트가 활성화된 열 이름>_LRS	기본 개체 레코드에 대한 최신 업데이트를 제공하는 소스 시스템의 마지막 행 ID입니다.
<트러스트가 활성화된 열 이름>_LUD	레코드에 대한 최종 값을 제공한 교차 참조 레코드의 마지막 업데이트 날짜입니다.
<트러스트가 활성화된 열 이름>_SRX	레코드에 대한 최종 값을 제공한 교차 참조 레코드의 식별자입니다.
<트러스트가 활성화된 열 이름>_OTS	용이한 트러스트 재정의의 위한 인코딩된 트러스트 설정입니다.

기본 개체에 대해 기록이 활성화된 경우 MDM Hub은 보류 중인 제어 테이블에 대해 별도의 기록 테이블을 유지합니다. 보류 중인 제어 테이블과 연결된 기록 테이블의 이름 형식은 C_<기본 개체 이름>_HPCT입니다.

제 26 장

게시 프로세스 구성

이 장에 포함된 항목:

- [게시 프로세스 개요, 481](#)
- [게시 프로세스에 대한 구성 단계, 482](#)
- [메시지 대기열 도구 시작, 482](#)
- [글로벌 메시지 대기열 설정 구성, 482](#)
- [메시지 대기열 서버 구성, 483](#)
- [아웃바운드 메시지 대기열 구성, 485](#)
- [JMS 메시지의 병렬 처리 구성, 487](#)
- [JMS 보안 구성, 487](#)
- [메시지 트리거 구성, 487](#)
- [메시지 대기열 풀링 비활성화, 492](#)
- [JMS 메시지 XML 참조, 492](#)
- [레거시 JMS 메시지 XML 참조, 505](#)

게시 프로세스 개요

MDM Hub 게시 프로세스를 구성하여 Hub 저장소의 데이터 변경 내용에 대한 XML 메시지를 생성하고, 메시지를 아웃바운드 JMS(Java Messaging System) 메시지 대기열에 게시할 수 있습니다. 외부 응용 프로그램에서는 MDM Hub가 JMS 메시지 대기열에 게시하는 XML 메시지를 검색할 수 있습니다.

게시 프로세스를 구성하기 전에 Hub 서버 및 처리 서버가 배포되어 있는 응용 프로그램 서버에 대해 연결 팩터리 및 JMS 메시지 대기열을 설정해야 합니다. 게시 프로세스의 성능을 개선하려면 JMS 메시지에 대한 병렬 처리를 구성합니다.

중요: SIF(서비스 통합 프레임워크)는 JMS 메시지 대기열 `siperian.sif.jms.queue`에서 MDB(메시지 구동 빈)를 사용하여 수신 비동기 SIF 요청을 처리합니다. `siperian.sif.jms.queue` JMS 메시지 대기열은 MDM Hub 설치 프로세스 중에 설정되며 모든 MDM Hub 설치에서 필수 요소입니다. MDM Hub 게시 프로세스를 위해 구성하는 JMS 메시지 대기열은 외부 응용 프로그램을 사용하여 JMS 메시지 대기열에서 JMS 메시지를 검색하는 경우에 필요합니다.

게시 프로세스에 대한 구성 단계

Informatica MDM Hub를 설치한 후 Hub 콘솔의 메시지 대기열 도구를 사용하여 Informatica MDM Hub 구현의 메시지 대기열을 구성할 수 있습니다. 아웃바운드 메시지 대기열에 이벤트를 게시하려는 경우에는 다음 테스트를 수행해야 합니다.

1. 응용 프로그램 서버에 메시지 대기열을 구성합니다.
Informatica MDM Hub 설치 프로그램에서는 메시지 대기열 및 연결 팩터리 구성을 자동으로 설정합니다. 자세한 내용은 사용 중인 플랫폼의 *Multidomain MDM 설치 가이드*를 참조하십시오.
2. 글로벌 메시지 대기열 설정을 구성합니다.
3. 하나 이상의 메시지 대기열 서버를 추가합니다.
4. 메시지 대기열 서버에 하나 이상의 메시지 대기열을 추가합니다.
5. 게시할 데이터가 있는 각 ORS에 대해 JMS 이벤트 메시지 스키마를 생성합니다.
6. 메시지 대기열에 대한 메시지 트리거를 구성합니다.

메시지 대기열을 구성한 후에는 감사 관리자를 사용하여 런타임 활동을 검토할 수 있습니다.

메시지 대기열 도구 시작

1. Hub 콘솔에서 마스터 데이터베이스에 연결합니다.
마스터 데이터베이스에 메시지 대기열이 정의되어 있습니다.
2. Hub 콘솔에서 구성 작업 영역을 확장한 후 **메시지 대기열**을 클릭합니다.
Hub 콘솔에 메시지 대기열 도구가 표시됩니다. 메시지 대기열 도구는 두 개의 창으로 구분됩니다.

창	설명
탐색 창	이 Informatica MDM Hub 구현에 대해 정의된 메시지 대기열을 트리 보기로 보여 줍니다.
속성 창	선택한 메시지 대기열의 속성을 보여 줍니다.

글로벌 메시지 대기열 설정 구성

Informatica MDM Hub 구현에 대한 글로벌 메시지 대기열 설정을 구성하려면

1. Hub 콘솔에서 메시지 대기열 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 전송 메시지에 대한 대기열을 모니터링하는 데이터 변경 모니터링에 대한 설정을 지정합니다.
4. 데이터 변경 모니터링을 활성화하거나 비활성화하려면 **데이터 변경 모니터링 상태 전환** 단추를 클릭합니다.

다음과 같은 모니터링 설정을 지정합니다.

모니터링 설정	설명
수신 제한 시간 (밀리초)	기본값은 0입니다. 대기열에서 메시지를 수신하는 데 허용되는 시간입니다.
수신 일괄 처리 크기	기본값은 100입니다. 메시지 대기열에서 한 번에 처리되고 배치되는 최대 이벤트 수입니다.
메시지 확인 간격 (밀리초)	기본값은 300000입니다. 인바운드 메시지를 폴링하거나 아웃바운드 메시지를 처리하기 전에 일시 중지할 시간입니다. 인바운드 메시지 대기열과 아웃바운드 메시지 대기열 모두에 동일한 값이 적용됩니다.
동기화 확인 간격 (밀리초)	이 설정이 구성되어 있으면 ORS 메타데이터를 정기적으로 폴링하여 ORS의 디자인 개체에 대한 후속 변경 내용이 있을 경우 XML 메시지 스키마를 다시 생성합니다. 기본적으로 이 기능은 비활성화(0으로 설정)되어 있으며 다음과 같은 경우에만 사용할 수 있습니다. 데이터 변경 모니터링이 활성화되어 있는 경우 JMS 이벤트 스키마 관리자를 사용하여 ORS 관련 XML 메시지 스키마가 생성된 경우 참고: 이 값은 메시지 확인 간격보다 크거나 같아야 합니다.

5. 변경할 속성 옆의 **편집** 단추를 클릭합니다.
6. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

메시지 대기열 서버 구성

MDM Hub 구현에 대해 메시지 대기열 서버를 구성합니다.

MDM Hub에 메시지 대기열을 정의하기 전에 MDM Hub가 메시지 대기열을 처리하는 데 사용해야 하는 메시지 대기열 서버를 구성해야 합니다. MDM Hub에 메시지 대기열 서버를 추가하기 전에 Hub 서버 및 처리 서버가 배포되어 있는 응용 프로그램 서버 인스턴스에 메시지 대기열 서버를 구성해야 합니다. MDM Hub에 메시지 대기열 서버를 정의하려면 JBoss 및 WebLogic용 메시지 대기열 서버의 연결 팩터리 이름 및 WebSphere용 메시지 대기열 서버의 서버 이름과 포트 번호가 필요합니다.

메시지 대기열 서버 속성

이 섹션에서는 메시지 대기열 서버에 대해 구성할 수 있는 설정에 대해 설명합니다.

WebLogic 및 JBoss 속성

다음과 같은 메시지 대기열 서버 속성을 구성할 수 있습니다.

속성	설명
연결 팩터리 이름	이 메시지 대기열 서버의 연결 팩터리 이름입니다.
표시 이름	Hub 콘솔에 표시되는 이 메시지 대기열 서버의 이름입니다.
설명	이 메시지 대기열 서버의 설명 정보입니다.

WebSphere 속성

IBM WebSphere 구현에는 다음과 같은 속성이 포함됩니다.

속성	설명
서버 이름	메시지 대기열이 정의된 서버의 이름입니다.
채널	메시지 대기열이 정의된 서버의 채널입니다.
포트	메시지 대기열이 정의된 서버의 포트입니다.

메시지 대기열 서버 추가

메시지 대기열에 연결하기 위한 메시지 대기열 서버를 추가합니다.

메시지 대기열 서버를 추가하려면

1. Hub 콘솔에서 메시지 대기열 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 마우스 오른쪽 단추를 클릭하고 **메시지 대기열 서버 추가**를 선택합니다.
[메시지 대기열 서버 추가] 대화 상자가 표시됩니다.
4. 메시지 대기열 서버 속성을 지정합니다.

메시지 대기열 서버 속성 편집

기존 메시지 대기열 서버의 속성을 편집하려면

1. Hub 콘솔에서 메시지 대기열 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 구성할 메시지 대기열 서버의 이름을 선택합니다.
4. 이 메시지 대기열 서버의 편집 가능한 속성을 변경합니다.
변경할 속성 옆의 **편집** 단추를 클릭합니다.
5. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

메시지 대기열 서버 삭제

기존 메시지 대기열 서버를 삭제하려면

1. Hub 콘솔에서 메시지 대기열 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 삭제하려는 메시지 대기열 서버의 이름을 마우스 오른쪽 단추로 클릭한 다음 팝업 메뉴에서 **삭제**를 선택합니다.
메시지 대기열 도구에 삭제를 확인하는 메시지가 나타납니다.
4. **예**를 클릭합니다.

아웃바운드 메시지 대기열 구성

이 섹션에서는 Informatica MDM Hub 구현에 맞게 아웃바운드 JMS 메시지 대기열을 구성하는 방법에 대해 설명합니다.

메시지 대기열 정보

Informatica MDM Hub에서 아웃바운드 JMS 메시지 대기열을 정의하려면 먼저 메시지 대기열을 처리할 메시지 대기열 서버를 정의해야 합니다. JMS에서 메시지 대기열은 XML 메시지의 준비 영역입니다. Informatica MDM Hub에서는 이 메시지 대기열에 XML 메시지를 게시합니다. 외부 응용 프로그램은 메시지 대기열에서 이러한 게시된 XML 메시지를 검색합니다.

메시지 대기열 속성

다음과 같은 메시지 대기열 속성을 구성할 수 있습니다.

속성	설명
대기열 이름	이 메시지 대기열의 이름입니다. 이 이름은 응용 프로그램 서버에서 구성된 JNDI 대기열 이름과 일치해야 합니다.
표시 이름	Hub 콘솔에 표시되는 이 메시지 대기열의 이름입니다.
설명	이 메시지 대기열의 설명 정보입니다.

메시지 대기열 서버에 메시지 대기열 추가

연결해야 하는 메시지 대기열 서버에 메시지 대기열을 추가합니다.

1. Hub 콘솔에서 메시지 대기열 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 메시지 대기열을 추가할 메시지 대기열 서버의 이름을 마우스 오른쪽 단추로 클릭하고 **메시지 대기열 추가**를 선택합니다.
[메시지 대기열 추가] 대화 상자가 표시됩니다.
4. 메시지 대기열 속성을 지정합니다.

5. **확인**을 클릭합니다.
대기열 할당을 선택하라는 메시지가 표시됩니다.
6. 다음 대기열 할당 옵션 중 하나를 선택합니다.

옵션	설명
할당되지 않은 상태로 두기	대기열이 현재 할당되어 있지 않으며 사용 중이 아닙니다. 대기열을 Informatica MDM Hub API 응답에 대한 아웃바운드 대기열로 사용하거나, 대기열이 현재 할당되어 있지 않으며 사용 중이 아님을 나타내려면 이 옵션을 선택합니다.
메시지 대기열 트리거와 함께 사용	대기열이 현재 할당되어 있고 스키마 관리자에 정의된 메시지 트리거에 사용될 수 있습니다.
레거시 XML 사용	Informatica MDM Hub 구현에서 현재 버전의 XML 메시지 형식 대신 레거시 XML 메시지 형식을 사용하도록 요구합니다.

7. **저장**을 클릭합니다.

메시지 대기열 속성 편집

기존 메시지 대기열의 속성을 편집하려면

1. **Hub** 콘솔에서 메시지 대기열 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 구성할 메시지 대기열의 이름을 선택합니다.
4. 이 메시지 대기열의 편집 가능한 속성을 변경합니다.
5. 변경할 속성 옆의 **편집** 단추를 클릭합니다.
6. 원하는 경우 대기열 할당을 변경합니다.
7. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

메시지 대기열 삭제

메시지 대기열 도구를 사용하여 메시지 대기열을 삭제합니다.

메시지 트리거가 삭제할 메시지 대기열과 연결되지 않았는지 확인합니다. 메시지 트리거가 메시지 대기열과 연결되어 있는 경우 메시지 대기열을 삭제하기 전에 메시지 트리거를 삭제합니다.

1. 구성 작업 영역에서 **메시지 대기열** 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 삭제할 메시지 대기열의 이름을 마우스 오른쪽 단추로 클릭한 다음 팝업 메뉴에서 **삭제**를 클릭합니다.
메시지 대기열 도구에 삭제를 확인하는 메시지가 나타납니다.
4. **예**를 클릭합니다.

JMS 메시지의 병렬 처리 구성

JMS 메시지의 병렬 처리를 구성하는 경우 Hub 서버가 여러 처리 서버에 일괄 방식으로 메시지 처리 로드를 분산합니다. JMS 메시지의 병렬 처리를 위한 일괄 처리 크기를 구성해야 합니다. Hub 서버 속성 파일에서 JMS 메시지의 병렬 처리를 구성합니다.

1. 다음 디렉터리에서 `cmxserver.properties` 파일을 엽니다.
<MDM Hub 설치 디렉터리>/hub/server/resources
2. JMS 메시지의 병렬 처리에 필요한 속성을 추가하고 해당 파일을 저장합니다.
다음 테이블에서는 JMS 메시지의 병렬 처리를 위한 속성에 대해 설명합니다.

속성	설명
<code>mq.data.change.threads</code>	게시 프로세스 중에 JMS 메시지 처리를 위해 사용하는 스레드의 수입니다. 기본값은 1입니다.
<code>mq.data.change.batch.size</code>	게시 프로세스를 위해 각 일괄 작업에서 처리하는 JMS 메시지의 수입니다. 기본값은 500입니다.
<code>mq.data.change.timeout</code>	JMS 메시지를 처리하는 데 허용되는 시간(초)입니다. 기본값은 120입니다.

JMS 보안 구성

메시지 대기열을 보호하도록 JMS 보안을 구성해야 합니다.

먼저 응용 프로그램 서버에 JMS 보안을 구성해야 합니다. `cmxserver.properties` 파일에서 다음 속성을 설정하여 MDM Hub가 응용 프로그램 서버에 설정된 사용자 자격 증명을 사용하도록 구성합니다.

```
<connection factory name>.qcf.username=<user name>  
<connection factory name>.qcf.password=<password>
```

메시지 트리거 구성

스키마 관리자 도구를 사용하여 MDM Hub 구현에 대한 메시지 트리거를 구성하십시오.

메시지 트리거는 MDM Hub이 외부 응용 프로그램에 전달하는 작업을 식별합니다. 규칙이 정의되어 있는 작업이 발생하면 MDM Hub이 XML 메시지를 JMS 메시지 대기열에 배치합니다. 메시지 트리거는 메시지가 배치되는 JMS 메시지 대기열을 지정합니다.

메시지 트리거 작동 방법에 대한 예제는 다음 시나리오를 참조하십시오.

1. 기본 개체에 레코드를 삽입합니다.
2. 이 삽입 작업으로 인해 메시지 트리거가 시작됩니다.
3. MDM Hub이 메시지 트리거를 평가하고 적절한 메시지 대기열에 메시지를 보냅니다.
4. 외부 응용 프로그램이 메시지 대기열을 폴링하고 메시지를 선택한 다음 처리합니다.

모든 트리거에 동일한 메시지 대기열을 사용하거나, 각 트리거에 서로 다른 메시지 대기열을 사용할 수 있습니다. 메시지 트리거를 트리거하는 작업에 대해 메시지 대기열을 구성하고 해당 기본 개체 및 작업에 대한 메시지 트리거를 정의하십시오.

메시지 트리거에 대한 이벤트 유형

다음 유형의 이벤트가 발생하면 메시지 트리거가 실행되어 메시지가 대기열에 배치될 수 있습니다.

다음 테이블에는 메시지 대기열 규칙을 정의할 수 있는 이벤트가 설명되어 있습니다.

이벤트	설명	메시지 유형 코드
새 데이터 추가	다음 방식을 통해 데이터를 추가할 때 사용됩니다. <ul style="list-style-type: none"> - 로드 프로세스 사용 - 데이터 관리자 사용 - HTTP, SOAP 및 EJB와 같은 프로토콜에서 PUT 또는 CLEANSE_PUT를 사용하는 API 동사 사용 	1
보류 중인 새 데이터 추가	레코드가 PENDING 상태로 작성됩니다. 상태 사용 기본 개체에 적용됩니다.	10
기존 데이터 업데이트	다음 방식을 통해 데이터를 업데이트할 때 사용됩니다. <ul style="list-style-type: none"> - 로드 프로세스 사용 - 데이터 관리자 사용 - HTTP, SOAP 및 EJB와 같은 프로토콜에서 PUT 또는 CLEANSE_PUT를 사용하는 API 동사 사용 트러스트 규칙으로 인해 기본 개체 열이 업데이트되지 않는 경우에는 메시지가 생성되지 않습니다. 지정된 열 중 하나 이상이 업데이트되는 경우 단일 메시지가 생성됩니다. 메시지는 전체 출력 시스템에 있는 모든 교차 참조 레코드의 데이터가 포함됩니다.	2
보류 중인 기존 데이터 업데이트	PENDING 상태의 기존 레코드가 업데이트됩니다. 상태 사용 기본 개체에 적용됩니다.	11
업데이트(XREF만 변경된 경우)	다음 시나리오에서 데이터를 업데이트할 때 사용됩니다. <ul style="list-style-type: none"> - 로드 프로세스를 통해 교차 참조만 변경된 경우 - HTTP, SOAP 및 EJB와 같은 프로토콜에서 PUT 또는 CLEANSE_PUT를 사용하는 API를 통해 교차 참조만 변경된 경우 	6
보류 중인 데이터 업데이트(XREF만 변경된 경우)	PENDING 상태의 교차 참조 레코드가 업데이트됩니다. 또한 레코드의 승격이 포함됩니다. 상태 사용 기본 개체에 적용됩니다.	12
데이터 병합	다음 방식을 통해 데이터를 병합할 때 사용됩니다. <ul style="list-style-type: none"> - 병합 관리자 사용 - HTTP, SOAP 및 EJB와 같은 프로토콜에서 API 동사 사용 - 자동 일치 및 병합 프로세스 사용 	4
데이터 병합(기본 개체가 업데이트된 경우)	다음 시나리오에서 데이터를 병합할 때 사용됩니다. <ul style="list-style-type: none"> - 기본 개체가 업데이트된 경우 - 새 교차 참조를 삽입할 행 ID별 레코드 로드 및 기본 개체가 업데이트된 경우 트러스트 규칙으로 인해 기본 개체 열이 병합되지 않는 경우 MDM Hub가 메시지를 생성하지 않습니다. 지정된 열 중 하나 이상이 업데이트되는 경우 단일 메시지가 생성됩니다. 메시지는 전체 출력 시스템에 있는 모든 교차 참조 레코드의 데이터가 포함됩니다.	7

이벤트	설명	메시지 유형 코드
데이터 병합 해제	다음 방식을 통해 데이터 병합을 해제할 때 사용됩니다. - 데이터 관리자 사용 - HTTP, SOAP 및 EJB와 같은 프로토콜에서 UNMERGE를 사용하는 API 동사 사용	8
데이터를 고유 데이터로 허용	다음 방식을 통해 레코드를 고유한 것으로 허용할 때 사용됩니다. - 병합 관리자 사용 - 일치 규칙을 구성할 때 스키마 도구 옵션을 활성화하여 사용 레코드가 일치 규칙을 통해 자동으로, 또는 데이터 스튜어드에 의해 수동으로 고유 레코드로 허용되는 경우 MDM Hub에서는 모든 출력 시스템에 대한 교차 참조 정보를 포함하여 레코드 정보를 제공하는 메시지를 생성합니다. 메시지는 대기열에 배치됩니다.	3
기본 개체 데이터 삭제	기본 개체 레코드가 일시 삭제되고 상태가 DELETED로 변경됩니다. 상태 사용 기본 개체에 적용됩니다.	14
XREF 데이터 삭제	교차 참조 레코드가 일시 삭제되고 상태가 DELETED로 변경됩니다. 상태 사용 기본 개체에 적용됩니다.	15
보류 중인 BO 데이터 삭제	PENDING 상태의 기본 개체 레코드가 영구 삭제됩니다. 상태 사용 기본 개체에 적용됩니다.	16
보류 중인 XREF 데이터 삭제	PENDING 상태의 교차 참조 레코드가 영구 삭제됩니다. 상태 사용 기본 개체에 적용됩니다.	17
복원	삭제된 교차 참조 레코드가 복원되고 상태가 ACTIVE로 변경됩니다. 상태 사용 기본 개체에 적용됩니다.	19

메시지 트리거에 대한 모범 사례

제안된 모범 사례를 사용하여 효율적으로 구현하기 위한 메시지 트리거를 작성합니다.

- 기본 개체 메시지 트리거 정의에서 메시지 대기열을 사용할 경우 MDM Hub에서 다음 메시지를 표시합니다. 메시지 대기열을 현재 메시지 트리거에서 사용하고 있습니다. 이 경우에는 메시지 대기열의 속성을 편집할 수 없습니다. 대신 다른 메시지 대기열을 작성하여 필요한 변경 작업을 수행해야 합니다.
- MDM Hub 설치 프로그램에서는 `siperian.sif.jms.queue`라는 기본 JMS 대기열을 작성합니다. 메시지 트리거를 구성할 때 이 대기열을 사용할 경우 MDM Hub에서 오류를 생성합니다.
- 메시지 트리거는 하나의 기본 개체에만 적용되며 해당 기본 개체에서 특정 작업이 직접적으로 발생할 경우에만 실행됩니다. 상위-하위 관계에 있는 두 테이블이 있는 경우에는 각 테이블에 대해 각각 명시적으로 메시지 대기열을 정의해야 합니다. MDM Hub에서 로드 INSERT 또는 PUT 같은 기본 개체에 대한 특정 변경 사항을 감지합니다. 상위 테이블의 레코드를 변경할 경우 상위 레코드에 대해서만 메시지 트리거가 실행될 수 있습니다. 상위 레코드의 변경 사항이 연결된 하위 레코드에 영향을 미칠 경우 하위 테이블에 대한 별도의 메시지 트리거를 명시적으로 구성해야 합니다.

메시지 트리거 시스템 속성

메시지 트리거를 구성할 경우 하나 이상의 트리거 시스템과 하나의 메시지 내 시스템을 선택해야 합니다.

메시지 트리거 시스템의 속성은 다음과 같습니다.

트리거

작업을 트리거하는 시스템을 선택합니다. 하나 이상의 시스템을 선택해야 합니다.

메시지 내

메시지를 표시하는 시스템을 선택합니다. 하나 이상의 시스템을 선택해야 합니다.

메시지 대기열의 메시지마다 '메시지 내' 시스템 중 하나에 있는 각 교차 참조에 대한 **pkey_src_object** 값이 포함됩니다.

예를 들어, 구현에 소스 시스템 **A, B, C**가 있습니다. 시스템 **A**를 트리거 시스템으로 선택합니다. 기본 개체 레코드에는 **A** 및 **B**에 대한 교차 참조 레코드가 있습니다. 이 기본 개체 레코드에 대해 시스템 **A**의 교차 참조를 업데이트하고자 합니다. 다음 테이블에서는 가능한 메시지 트리거 구성과 결과 메시지를 보여 줍니다.

메시지 내 시스템	결과 메시지
A	시스템 A에 대한 교차 참조가 포함된 메시지
B	시스템 B에 대한 교차 참조가 포함된 메시지
C	메시지 없음 메시지 내의 교차 참조가 없음
A 및 B	시스템 A 및 B에 대한 교차 참조가 포함된 메시지
A 및 C	시스템 A에 대한 교차 참조가 포함된 메시지
B 및 C	시스템 B에 대한 교차 참조가 포함된 메시지
A, B 및 C	시스템 A 및 B에 대한 교차 참조가 포함된 메시지

메시지 트리거 추가

1. 메시지 대기열을 메시지 트리거와 함께 사용할 수 있도록 구성합니다.
2. 스키마 관리자를 시작합니다.
3. 쓰기 잠금을 획득합니다.
4. 모니터링할 기본 개체를 확장하고 **메시지 트리거 설정** 노트를 선택합니다.
메시지 트리거가 작성되지 않은 경우 스키마 도구에서 빈 화면을 표시합니다.
5. 다음 작업 중 하나를 수행합니다.
 - 메시지 트리거가 정의되어 있지 않은 경우 **메시지 트리거 추가**를 클릭합니다.
 - 메시지 트리거가 정의된 경우 **추가**를 클릭합니다.스키마 관리자에 메시지 트리거 추가 마법사가 표시됩니다.
6. 새 메시지 트리거의 이름 및 설명을 지정합니다.
7. **다음**을 클릭합니다.
메시징 패키지를 지정하라는 메시지가 표시됩니다.
8. 메시지를 작성하는 데 사용할 패키지를 선택합니다.
9. **다음**을 클릭합니다.
대상 메시지 대기열을 지정하라는 메시지가 표시됩니다.
10. 메시지가 있는 메시지 대기열을 선택합니다.

11. **다음**을 클릭합니다.
이 메시지 트리거에 대한 규칙을 지정하라는 메시지가 표시됩니다.
12. 이 메시지 트리거에 대한 이벤트 유형을 선택합니다.
13. 이 메시지 트리거에 대한 시스템 속성을 구성합니다. "트리거" 시스템과 "메시지 내" 시스템을 각각 하나 이상 선택해야 합니다.
14. "업데이트" 옵션을 선택한 경우 **다음**을 클릭합니다. 그렇지 않은 경우 **마침**을 클릭합니다.
"업데이트" 작업을 클릭한 경우 업데이트 작업을 모니터링할 열을 선택하라는 메시지가 표시됩니다.
15. 다음 작업 중 하나를 수행합니다.
 - 이 메시지 트리거와 연결된 이벤트를 모니터링할 열을 선택합니다.
 - 모든 열에서 업데이트를 모니터링하려면 **임의의 열을 변경할 경우 메시지 트리거** 확인란을 선택합니다.
16. **마침**을 클릭합니다.

메시지 트리거 편집

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 모니터링할 기본 개체를 확장하고 **메시지 트리거 설정** 노드를 선택합니다.
4. 메시지 트리거 목록에서 구성할 메시지 트리거를 클릭합니다.
스키마 관리자에 선택한 메시지 트리거의 설정이 표시됩니다.
5. 원하는 설정을 변경합니다.
6. 변경하려는 편집 가능한 속성 옆에 있는 **편집** 단추를 클릭합니다.
7. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

메시지 트리거 삭제

1. 스키마 관리자를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 모니터링할 기본 개체를 확장하고 **메시지 트리거 설정** 노드를 선택합니다.
4. 메시지 트리거 목록에서 삭제할 메시지 트리거를 클릭합니다.
5. **삭제** 단추를 클릭합니다.
스키마 관리자에 삭제를 확인하는 메시지가 나타납니다.
6. **예**를 클릭합니다.

메시지 대기열 폴링 비활성화

다중 노드 환경에서 개별 노드에 대해 메시지 대기열 폴링을 비활성화할 수 있습니다. 메시지 대기열 폴링을 비활성화하려면 `cmxserver.properties` 및 `cmxcleanse.properties` 파일에서 `mq.data.change.monitor.thread.start` 속성을 `false`로 설정합니다.

DEBUG 모드에서 메시지 대기열 폴링 작업을 확인할 수 있습니다.

- `mq.data.change.monitor.thread.start` 값이 `false`인 노드에서는 로그에 **모니터링이 비활성화됨** 메시지가 표시됩니다.
- `mq.data.change.monitor.thread.start` 값이 `true`인 노드에서는 로그에 **데이터 변경 모니터링이 시작됨** 메시지가 표시됩니다.

기본적으로 MDM Hub EAR 파일이 배포된 모든 Java 가상 시스템에서는 메시지 대기열 폴링이 활성화됩니다. 데이터 변경 모니터링 및 메시지 게시에 다수의 폴러 스레드를 사용하는 경우 메시지가 올바른 시퀀스로 게시되지 않을 수 있습니다. 메시지가 게시되는 시퀀스를 제어하려면 단일 폴러 스레드를 사용하십시오.

JMS 메시지 XML 참조

이 섹션에서는 Informatica MDM Hub XML 메시지의 구조를 설명하고 예제 메시지를 제공합니다.

참고: Informatica MDM Hub 구현 환경에서 이 섹션에 설명된 현재 버전의 XML 메시지 형식이 아닌 레거시 XML 메시지 형식(Informatica MDM Hub XU 버전)을 사용해야 하는 경우 대신 [“레거시 JMS 메시지 XML 참조” 페이지 505](#)를 살펴보세요.

ORS 관련 XML 메시지 스키마 생성

게시 프로세스에서 XML 메시지를 생성하려면 Hub 콘솔에서 JMS 이벤트 스키마 관리자 도구를 사용하여 생성한 ORS 관련 스키마 파일(<ors-name>-siperian-mrm-event.xsd)이 필요합니다.

XML 메시지의 요소

다음 테이블에서는 XML 메시지의 요소에 대해 설명합니다.

필드	설명
루트 노드	
<siperianEvent>	XML 메시지의 루트 노드입니다.
이벤트 메타데이터	
<eventMetadata>	이벤트 메타데이터의 루트 노드입니다.
<messageId>	siperianEvent 메시지의 고유 ID입니다.

필드	설명
<eventType>	이벤트 유형으로, 다음 값 중 하나를 갖습니다. <ul style="list-style-type: none"> - 삽입 - 업데이트 - XREF 업데이트 - 고유 항목으로 허용 - 병합 - 병합 해제 - 병합 업데이트
<baseObjectUid>	이 작업의 영향을 받는 기본 개체의 UID입니다.
<packageUid>	이 작업과 연결된 패키지의 UID입니다.
<messageDate>	이 메시지가 생성된 날짜/시간입니다.
<orsId>	이 이벤트와 연결된 ORS(연산 참조 저장소)의 ID입니다.
<triggerUid>	이 메시지를 생성한 이벤트를 트리거한 규칙의 UID입니다.
이벤트 세부 정보	
<eventTypeEvent>	이벤트 세부 정보의 루트 노드입니다.
<sourceSystemName>	이 이벤트와 연결된 소스 시스템의 이름입니다.
<sourceKey>	이 이벤트와 연결된 PKEY_SRC_OBJECT의 값입니다.
<eventDate>	이벤트가 생성된 날짜/시간입니다.
<rowid>	이벤트의 영향을 받은 기본 개체 레코드의 RowID입니다.
<xrefKey>	이 이벤트의 영향을 받은 교차 참조 레코드의 루트 노드입니다.
<systemName>	이 이벤트의 영향을 받은 교차 참조 레코드의 시스템 이름입니다.
<sourceKey>	이 이벤트의 영향을 받은 교차 참조 레코드의 PKEY_SRC_OBJECT입니다.
<packageName>	이 이벤트와 연결된 보안 패키지의 이름입니다.
<columnName>	XML 파일의 요소가 패키지의 각 열을 나타냅니다. 예: rowidObject 및 consolidationInd. JMS 이벤트 스키마 관리자 도구를 사용하여 생성된 ORS 관련 XSD에 정의되어 있습니다.
<mergedRowid>	병합에서 손실되는 레코드에 대한 ROWID_OBJECT 값 목록입니다. MERGE 이벤트의 경우에 만 메시지에 이 필드가 포함됩니다.

메시지 필터링

MessageType이라는 사용자 지정 JMS 헤더를 사용하여 메시지 유형을 기준으로 수신 메시지를 필터링할 수 있습니다. 메시지 헤더에 표시되는 메시지 유형은 다음과 같습니다.

메시지 유형	설명
siperianEvent	이벤트 알림 메시지
<serviceNameReturn>	SIF(서비스 통합 프레임워크) 응답의 경우 다음에 나온 get 요청에 대한 응답의 일부와 같이 SIF 요청의 이름으로 시작됩니다. <getReturn> <message>The GET was executed successfully - retrieved 1 records</message> <recordKey> <ROWID>2</ROWID> </recordKey> ...

XML 메시지 예

이 섹션에서는 XML 메시지 예의 목록을 제공합니다.

고유 항목으로 허용 메시지

다음은 고유 항목으로 허용 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>Accept as Unique</eventType>
    <baseObjectId>BASE_OBJECT.C_CONTACT</baseObjectId>
    <packageId>PACKAGE.CONTACT_PKG</packageId>
    <orsId>localhost-mrm-CMX_ORS</orsId>
    <triggerId>MESSAGE_QUEUE_RULE.ContactUpdate</triggerId>
    <messageId>192</messageId>
    <messageDate>2008-09-10T16:33:14.000-07:00</messageDate>
  </eventMetadata>
  <acceptAsUniqueEvent>
    <sourceSystemName>Admin</sourceSystemName>
    <sourceKey>SVR1.1T1</sourceKey>
    <eventDate>2008-09-10T16:33:14.000-07:00</eventDate>
    <rowid>2</rowid>
    <xrefKey>
      <systemName>Admin</systemName>
      <sourceKey>SVR1.1T1</sourceKey>
    </xrefKey>
    <contactPkg>
      <rowidObject>2</rowidObject>
      <creator>admin</creator>
      <createDate>2008-08-13T20:28:02.000-07:00</createDate>
      <updatedBy>admin</updatedBy>
      <lastUpdateDate>2008-09-10T16:33:14.000-07:00</lastUpdateDate>
      <consolidationInd>1</consolidationInd>
      <lastRowidSystem>SYS0</lastRowidSystem>
      <dirtyInd>0</dirtyInd>
      <firstName>Joey</firstName>
      <lastName>Brown</lastName>
    </contactPkg>
  </acceptAsUniqueEvent>
</siperianEvent>
```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

AMRule 메시지

다음은 AMRule 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>AM Rule Event</eventType>
    <packageUid>PACKAGE.CONTACT_PKG</packageUid>
    <orsId>localhost-mrm-CMX_ORS</orsId>
    <interactionId>12</interactionId>
    <activityName>Changed Contact and Address </activityName>
    <triggerUid>MESSAGE_QUEUE_RULE.ContactUpdateLegacy</triggerUid>
    <messageId>291</messageId>
    <messageDate>2008-09-19T11:43:42.979-07:00</messageDate>
  </eventMetadata>
  <amRuleEvent>
    <eventDate>2008-09-19T11:43:42.979-07:00</eventDate>
    <contactPkgAmEvent>
      <amRuleUid>AM_RULE.RuleSet1|Rule1</amRuleUid>
      <contactPkg>
        <rowidObject>64 </rowidObject>
        <creator>admin</creator>
        <createDate>2008-09-08T16:24:35.000-07:00</createDate>
        <updatedBy>admin</updatedBy>
        <lastUpdateDate>2008-09-18T16:26:45.000-07:00</lastUpdateDate>
        <consolidationInd>2</consolidationInd>
        <lastRowidSystem>SYS0 </lastRowidSystem>
        <dirtyInd>1</dirtyInd>
        <firstName>Johnny</firstName>
        <lastName>Brown</lastName>
        <hubStateInd>1</hubStateInd>
      </contactPkg>
      <cContact>
        <event>
          <eventType>Update</eventType>
          <system>Admin</system>
        </event>
        <event>
          <eventType>Update XREF</eventType>
          <system>Admin</system>
        </event>
        <xrefKey>
          <systemName>CRM</systemName>
          <sourceKey>PK1265</sourceKey>
        </xrefKey>
        <xrefKey>
          <systemName>Admin</systemName>
          <sourceKey>64</sourceKey>
        </xrefKey>
      </cContact>
    </contactPkgAmEvent>
  </amRuleEvent>
</siperianEvent>
```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

BoDelete 메시지

다음은 BoDelete 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>BO Delete</eventType>
    <baseObjectUid>BASE_OBJECT.C_CONTACT</baseObjectUid>
    <packageUid>PACKAGE.CONTACT_PKG</packageUid>
    <orsId>localhost-mrm-CMX_ORS</orsId>
```

```

    <triggerUid>MESSAGE_QUEUE_RULE.ContactUpdate</triggerUid>
    <messageId>328</messageId>
    <messageDate>2008-09-19T14:35:53.000-07:00</messageDate>
  </eventMetadata>
  <boDeleteEvent>
    <sourceSystemName>Admin</sourceSystemName>
    <eventDate>2008-09-19T14:35:53.000-07:00</eventDate>
    <rowid>107</rowid>
    <xrefKey>
      <systemName>CRM</systemName>
    </xrefKey>
    <xrefKey>
      <systemName>Admin</systemName>
    </xrefKey>
    <xrefKey>
      <systemName>WEB</systemName>
    </xrefKey>
    <contactPkg>
      <rowidObject>107</rowidObject>
      <creator>sifuser</creator>
      <createDate>2008-09-19T14:35:28.000-07:00</createDate>
      <updatedBy>admin</updatedBy>
      <lastUpdateDate>2008-09-19T14:35:53.000-07:00</lastUpdateDate>
      <consolidationInd>4</consolidationInd>
      <lastRowidSystem>CRM</lastRowidSystem>
      <dirtyInd>1</dirtyInd>
      <firstName>John</firstName>
      <lastName>Smith</lastName>
      <hubStateInd>-1</hubStateInd>
    </contactPkg>
  </boDeleteEvent>
</siperianEvent>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

BoSetToDelete 메시지

다음은 BoSetToDelete 메시지의 예입니다.

```

<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>BO set to Delete</eventType>
    <baseObjectId>BASE_OBJECT.C_CONTACT</baseObjectId>
    <packageId>PACKAGE.CONTACT_PKG</packageId>
    <orsId>localhost-mrm-CMX_OR</orsId>
    <triggerId>MESSAGE_QUEUE_RULE.ContactUpdate</triggerId>
    <messageId>319</messageId>
    <messageDate>2008-09-19T14:21:03.000-07:00</messageDate>
  </eventMetadata>
  <boSetToDeleteEvent>
    <sourceSystemName>Admin</sourceSystemName>
    <eventDate>2008-09-19T14:21:03.000-07:00</eventDate>
    <rowid>102</rowid>
    <xrefKey>
      <systemName>CRM</systemName>
    </xrefKey>
    <xrefKey>
      <systemName>Admin</systemName>
    </xrefKey>
    <xrefKey>
      <systemName>WEB</systemName>
    </xrefKey>
    <contactPkg>
      <rowidObject>102</rowidObject>
      <creator>admin</creator>
      <createDate>2008-09-19T13:57:09.000-07:00</createDate>
    </contactPkg>
  </boSetToDeleteEvent>
</siperianEvent>

```

```

        <updatedBy>admin</updatedBy>
        <lastUpdateDate>2008-09-19T14:21:03.000-07:00</lastUpdateDate>
        <consolidationInd>4</consolidationInd>
        <lastRowidSystem>SYS0 </lastRowidSystem>
        <dirtyInd>1</dirtyInd>
        <hubStateInd>-1</hubStateInd>
    </contactPkg>
</boSetToDeleteEvent>
</siperianEvent>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

삭제 메시지

다음 코드는 삭제 메시지의 예입니다.

```

<?xml version="1.0" encoding="UTF-16LE"?>
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>Delete</ACTION>
    <MESSAGE_ID>11010297</MESSAGE_ID>
    <MESSAGE_DATE>2008-09-19 14:35:53.0</MESSAGE_DATE>
    <TABLE_NAME>C_CONTACT</TABLE_NAME>
    <PACKAGE>CONTACT_PKG</PACKAGE>
    <RULE_NAME>ContactUpdateLegacy</RULE_NAME>
    <RULE_ID>SVR1.28D</RULE_ID>
    <ROWID_OBJECT>107</ROWID_OBJECT>
    <DATABASE>localhost-mrm-CMX_ORS</DATABASE>
    <XREFS>
      <XREF>
        <SYSTEM>CRM</SYSTEM>
        <PKEY_SRC_OBJECT />
      </XREF>
      <XREF>
        <SYSTEM>Admin</SYSTEM>
        <PKEY_SRC_OBJECT />
      </XREF>
      <XREF>
        <SYSTEM>WEB</SYSTEM>
        <PKEY_SRC_OBJECT />
      </XREF>
    </XREFS>
  </CONTROLAREA>
  <DATAAREA>
    <DATA>
      <ROWID_OBJECT>107</ROWID_OBJECT>
      <CREATOR>sifuser</CREATOR>
      <CREATE_DATE>19 Sep 2008 14:35:28</CREATE_DATE>
      <UPDATED_BY>admin</UPDATED_BY>
      <LAST_UPDATE_DATE>19 Sep 2008 14:35:53</LAST_UPDATE_DATE>
      <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
      <DELETED_IND />
      <DELETED_BY />
      <DELETED_DATE />
      <LAST_ROWID_SYSTEM>CRM</LAST_ROWID_SYSTEM>
      <INTERACTION_ID />
      <FIRST_NAME>John</FIRST_NAME>
      <LAST_NAME>Smith</LAST_NAME>
      <HUB_STATE_IND>-1</HUB_STATE_IND>
    </DATA>
  </DATAAREA>
</SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

삽입 메시지

다음은 삽입 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>Insert</eventType>
    <baseObjectId>BASE_OBJECT.C_CONTACT</baseObjectId>
    <packageId>PACKAGE.CONTACT_PKG</packageId>
    <orsId>localhost-mrm-CMX_ORS</orsId>
    <triggerId>MESSAGE_QUEUE_RULE.ContactUpdateLegacy</triggerId>
    <messageId>114</messageId>
    <messageDate>2008-09-08T16:02:11.000-07:00</messageDate>
  </eventMetadata>
  <insertEvent>
    <sourceSystemName>CRM</sourceSystemName>
    <sourceKey>PK12658</sourceKey>
    <eventDate>2008-09-08T16:02:11.000-07:00</eventDate>
    <rowid>66</rowid>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>PK12658</sourceKey>
    </xrefKey>
    <contactPkg>
      <rowidObject>66</rowidObject>
      <creator>admin</creator>
      <createDate>2008-09-08T16:02:11.000-07:00</createDate>
      <updatedBy>admin</updatedBy>
      <lastUpdateDate>2008-09-08T16:02:11.000-07:00</lastUpdateDate>
      <consolidationInd>4</consolidationInd>
      <lastRowidSystem>CRM</lastRowidSystem>
      <dirtyInd>1</dirtyInd>
      <firstName>Joe</firstName>
      <lastName>Brown</lastName>
    </contactPkg>
  </insertEvent>
</siperianEvent>
```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

병합 메시지

다음은 병합 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>Merge</eventType>
    <baseObjectId>BASE_OBJECT.C_CONTACT</baseObjectId>
    <packageId>PACKAGE.CONTACT_PKG</packageId>
    <orsId>localhost-mrm-CMX_ORS</orsId>
    <triggerId>MESSAGE_QUEUE_RULE.ContactUpdateLegacy</triggerId>
    <messageId>130</messageId>
    <messageDate>2008-09-08T16:13:28.000-07:00</messageDate>
  </eventMetadata>
  <mergeEvent>
    <sourceSystemName>CRM</sourceSystemName>
    <sourceKey>PK126566</sourceKey>
    <eventDate>2008-09-08T16:13:28.000-07:00</eventDate>
    <rowid>65</rowid>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>PK126566</sourceKey>
    </xrefKey>
    <xrefKey>
      <systemName>Admin</systemName>
      <sourceKey>SVR1.28E</sourceKey>
    </xrefKey>
  </mergeEvent>
</siperianEvent>
```

```

</xrefKey>
<mergedRowid>62</mergedRowid>
<contactPkg>
  <rowidObject>65</rowidObject>
  <creator>admin</creator>
  <createDate>2008-09-08T15:49:17.000-07:00</createDate>
  <updatedBy>admin</updatedBy>
  <lastUpdateDate>2008-09-08T16:13:28.000-07:00</lastUpdateDate>
  <consolidationInd>4</consolidationInd>
  <lastRowidSystem>SYS0</lastRowidSystem>
  <dirtyInd>1</dirtyInd>
  <firstName>Joe</firstName>
  <lastName>Brown</lastName>
</contactPkg>
</mergeEvent>
</siperianEvent>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

병합 업데이트 메시지

다음은 병합 업데이트 메시지의 예입니다.

```

<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>Merge Update</eventType>
    <baseObjectUid>BASE_OBJECT.C_CONTACT</baseObjectUid>
    <packageUid>PACKAGE.CONTACT_PKG</packageUid>
    <orsId>localhost-mrm-CMX_ORS</orsId>
    <triggerUid>MESSAGE_QUEUE_RULE.ContactUpdate</triggerUid>
    <messageId>269</messageId>
    <messageDate>2008-09-10T17:25:42.000-07:00</messageDate>
  </eventMetadata>
  <mergeUpdateEvent>
    <sourceSystemName>CRM</sourceSystemName>
    <sourceKey>P45678</sourceKey>
    <eventDate>2008-09-10T17:25:42.000-07:00</eventDate>
    <rowid>83</rowid>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>P45678</sourceKey>
    </xrefKey>
    <mergedRowid>58</mergedRowid>
    <contactPkg>
      <rowidObject>83</rowidObject>
      <creator>admin</creator>
      <createDate>2008-09-10T16:44:56.000-07:00</createDate>
      <updatedBy>admin</updatedBy>
      <lastUpdateDate>2008-09-10T17:25:42.000-07:00</lastUpdateDate>
      <consolidationInd>1</consolidationInd>
      <lastRowidSystem>CRM</lastRowidSystem>
      <dirtyInd>1</dirtyInd>
      <firstName>Thomas</firstName>
      <lastName>Jones</lastName>
    </contactPkg>
  </mergeUpdateEvent>
</siperianEvent>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

작업 없음 메시지

다음은 [작업 없음] 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>No Action</eventType>
    <baseObjectUid>BASE_OBJECT.C_CONTACT</baseObjectUid>
    <packageUid>PACKAGE.CONTACT_PKG</packageUid>
    <orsId>localhost-mrm-CMX_ORS</orsId>
    <triggerUid>MESSAGE_QUEUE_RULE.ContactUpdate</triggerUid>
    <messageId>267</messageId>
    <messageDate>2008-09-10T17:25:42.000-07:00</messageDate>
  </eventMetadata>
  <noActionEvent>
    <sourceSystemName>CRM</sourceSystemName>
    <sourceKey>P45678</sourceKey>
    <eventDate>2008-09-10T17:25:42.000-07:00</eventDate>
    <rowid>83</rowid>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>P45678</sourceKey>
    </xrefKey>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>P45678</sourceKey>
    </xrefKey>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>P45678</sourceKey>
    </xrefKey>
    <contactPkg>
      <rowidObject>83</rowidObject>
      <creator>admin</creator>
      <createDate>2008-09-10T16:44:56.000-07:00</createDate>
      <updatedBy>admin</updatedBy>
      <lastUpdateDate>2008-09-10T17:25:42.000-07:00</lastUpdateDate>
      <consolidationInd>1</consolidationInd>
      <lastRowidSystem>CRM</lastRowidSystem>
      <dirtyInd>1</dirtyInd>
      <firstName>Thomas</firstName>
      <lastName>Jones</lastName>
    </contactPkg>
  </noActionEvent>
</siperianEvent>
```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

PendingInsert 메시지

다음은 PendingInsert 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>Pending Insert</eventType>
    <baseObjectUid>BASE_OBJECT.C_CONTACT</baseObjectUid>
    <packageUid>PACKAGE.CONTACT_PKG</packageUid>
    <orsId>localhost-mrm-CMX_ORS</orsId>
    <triggerUid>MESSAGE_QUEUE_RULE.ContactUpdate</triggerUid>
    <messageId>302</messageId>
    <messageDate>2008-09-19T13:57:10.000-07:00</messageDate>
  </eventMetadata>
  <pendingInsertEvent>
    <sourceSystemName>Admin</sourceSystemName>
    <sourceKey>SVR1.2V3</sourceKey>
    <eventDate>2008-09-19T13:57:10.000-07:00</eventDate>
```



```

<rowid>102          </rowid>
<xrefKey>
  <systemName>Admin</systemName>
  <sourceKey>SVR1.2V3</sourceKey>
</xrefKey>
<contactPkg>
  <rowidObject>102          </rowidObject>
  <creator>admin</creator>
  <createDate>2008-09-19T13:57:09.000-07:00</createDate>
  <updatedAt>admin</updatedAt>
  <lastUpdateDate>2008-09-19T13:57:09.000-07:00</lastUpdateDate>
  <consolidationInd>4</consolidationInd>
  <lastRowidSystem>SYS0          </lastRowidSystem>
  <dirtyInd>1</dirtyInd>
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <hubStateInd>0</hubStateInd>
</contactPkg>
</pendingInsertEvent>
</siperianEvent>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

PendingUpdate 메시지

다음은 PendingUpdate 메시지의 예입니다.

```

<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>Pending Update</eventType>
    <baseObjectId>BASE_OBJECT.C_CONTACT</baseObjectId>
    <packageId>PACKAGE.CONTACT_PKG</packageId>
    <orsId>localhost-mrm-CMX_ORS</orsId>
    <triggerId>MESSAGE_QUEUE_RULE.ContactUpdate</triggerId>
    <messageId>306</messageId>
    <messageDate>2008-09-19T14:01:36.000-07:00</messageDate>
  </eventMetadata>
  <pendingUpdateEvent>
    <sourceSystemName>CRM</sourceSystemName>
    <sourceKey>CPK125</sourceKey>
    <eventDate>2008-09-19T14:01:36.000-07:00</eventDate>
    <rowid>102          </rowid>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>CPK125</sourceKey>
    </xrefKey>
    <xrefKey>
      <systemName>Admin</systemName>
      <sourceKey>SVR1.2V3</sourceKey>
    </xrefKey>
    <contactPkg>
      <rowidObject>102          </rowidObject>
      <creator>admin</creator>
      <createDate>2008-09-19T13:57:09.000-07:00</createDate>
      <updatedAt>sifuser</updatedAt>
      <lastUpdateDate>2008-09-19T14:01:36.000-07:00</lastUpdateDate>
      <consolidationInd>4</consolidationInd>
      <lastRowidSystem>CRM          </lastRowidSystem>
      <dirtyInd>1</dirtyInd>
      <firstName>John</firstName>
      <lastName>Smith</lastName>
      <hubStateInd>1</hubStateInd>
    </contactPkg>
  </pendingUpdateEvent>
</siperianEvent>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

PendingUpdateXref 메시지

다음은 PendingUpdateXref 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>Pending Update XREF</eventType>
    <baseObjectId>BASE_OBJECT.C_CONTACT</baseObjectId>
    <packageId>PACKAGE.CONTACT_PKG</packageId>
    <orsId>localhost-mrm-CMX_ORs</orsId>
    <triggerId>MESSAGE_QUEUE_RULE.ContactUpdate</triggerId>
    <messageId>306</messageId>
    <messageDate>2008-09-19T14:01:36.000-07:00</messageDate>
  </eventMetadata>
  <pendingUpdateXrefEvent>
    <sourceSystemName>CRM</sourceSystemName>
    <sourceKey>CPK125</sourceKey>
    <eventDate>2008-09-19T14:01:36.000-07:00</eventDate>
    <rowId>102 </rowId>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>CPK125</sourceKey>
    </xrefKey>
    <xrefKey>
      <systemName>Admin</systemName>
      <sourceKey>SVR1.2V3</sourceKey>
    </xrefKey>
    <contactPkg>
      <rowIdObject>102 </rowIdObject>
      <creator>admin</creator>
      <createDate>2008-09-19T13:57:09.000-07:00</createDate>
      <updatedBy>sifuser</updatedBy>
      <lastUpdateDate>2008-09-19T14:01:36.000-07:00</lastUpdateDate>
      <consolidationInd>4</consolidationInd>
      <lastRowIdSystem>CRM </lastRowIdSystem>
      <dirtyInd>1</dirtyInd>
      <firstName>John</firstName>
      <lastName>Smith</lastName>
      <hubStateInd>1</hubStateInd>
    </contactPkg>
  </pendingUpdateXrefEvent>
</siperianEvent>
```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

병합 해제 메시지

다음은 병합 해제 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>UnMerge</eventType>
    <baseObjectId>BASE_OBJECT.C_CONTACT</baseObjectId>
    <packageId>PACKAGE.CONTACT_PKG</packageId>
    <orsId>localhost-mrm-CMX_ORs</orsId>
    <triggerId>MESSAGE_QUEUE_RULE.ContactUpdate</triggerId>
    <messageId>145</messageId>
    <messageDate>2008-09-08T16:24:36.000-07:00</messageDate>
  </eventMetadata>
  <unmergeEvent>
    <sourceSystemName>CRM</sourceSystemName>
  </unmergeEvent>
</siperianEvent>
```

```

<sourceKey>PK1265</sourceKey>
<eventDate>2008-09-08T16:24:36.000-07:00</eventDate>
<rowid>65</rowid>
<xrefKey>
  <systemName>CRM</systemName>
  <sourceKey>PK1265</sourceKey>
</xrefKey>
<mergedRowid>64</mergedRowid>
<contactPkg>
  <rowidObject>65</rowidObject>
  <creator>admin</creator>
  <createDate>2008-09-08T15:49:17.000-07:00</createDate>
  <updatedBy>admin</updatedBy>
  <lastUpdateDate>2008-09-08T16:24:35.000-07:00</lastUpdateDate>
  <consolidationInd>4</consolidationInd>
  <lastRowidSystem>SYS0</lastRowidSystem>
  <dirtyInd>1</dirtyInd>
  <firstName>Joe</firstName>
  <lastName>Brown</lastName>
</contactPkg>
</unmergeEvent>
</siperianEvent>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

업데이트 메시지

다음은 업데이트 메시지의 예입니다.

```

<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>Update</eventType>
    <baseObjectUid>BASE_OBJECT.C_CONTACT</baseObjectUid>
    <packageUid>PACKAGE.CONTACT_PKG</packageUid>
    <orsId>localhost-mrm-CMX_ORS</orsId>
    <triggerUid>MESSAGE_QUEUE_RULE.ContactUpdate</triggerUid>
    <messageId>120</messageId>
    <messageDate>2008-09-08T16:05:13.000-07:00</messageDate>
  </eventMetadata>
  <updateEvent>
    <sourceSystemName>CRM</sourceSystemName>
    <sourceKey>PK12658</sourceKey>
    <eventDate>2008-09-08T16:05:13.000-07:00</eventDate>
    <rowid>66</rowid>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>PK12658</sourceKey>
    </xrefKey>
    <contactPkg>
      <rowidObject>66</rowidObject>
      <creator>admin</creator>
      <createDate>2008-09-08T16:02:11.000-07:00</createDate>
      <updatedBy>admin</updatedBy>
      <lastUpdateDate>2008-09-08T16:05:13.000-07:00</lastUpdateDate>
      <consolidationInd>4</consolidationInd>
      <lastRowidSystem>CRM</lastRowidSystem>
      <dirtyInd>1</dirtyInd>
      <firstName>Joe</firstName>
      <lastName>Black</lastName>
    </contactPkg>
  </updateEvent>
</siperianEvent>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

XREF 업데이트 메시지

다음은 XREF 업데이트 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>Update XREF</eventType>
    <baseObjectUid>BASE_OBJECT.C_CONTACT</baseObjectUid>
    <packageUid>PACKAGE.CONTACT_PKG</packageUid>
    <orsId>localhost-mrm-CMX_ORS</orsId>
    <triggerUid>MESSAGE_QUEUE_RULE.ContactUpdate</triggerUid>
    <messageId>121</messageId>
    <messageDate>2008-09-08T16:05:13.000-07:00</messageDate>
  </eventMetadata>
  <updateXrefEvent>
    <sourceSystemName>CRM</sourceSystemName>
    <sourceKey>PK12658</sourceKey>
    <eventDate>2008-09-08T16:05:13.000-07:00</eventDate>
    <rowid>66          </rowid>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>PK12658</sourceKey>
    </xrefKey>
    <contactPkg>
      <rowidObject>66          </rowidObject>
      <creator>admin</creator>
      <createDate>2008-09-08T16:02:11.000-07:00</createDate>
      <updatedBy>admin</updatedBy>
      <lastUpdateDate>2008-09-08T16:05:13.000-07:00</lastUpdateDate>
      <consolidationInd>4</consolidationInd>
      <lastRowidSystem>CRM          </lastRowidSystem>
      <dirtyInd>1</dirtyInd>
      <firstName>Joe</firstName>
      <lastName>Black</lastName>
    </contactPkg>
  </updateXrefEvent>
</siperianEvent>
```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

XRefDelete 메시지

다음은 XRefDelete 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>XREF Delete</eventType>
    <baseObjectUid>BASE_OBJECT.C_CONTACT</baseObjectUid>
    <packageUid>PACKAGE.CONTACT_PKG</packageUid>
    <orsId>localhost-mrm-CMX_ORS</orsId>
    <triggerUid>MESSAGE_QUEUE_RULE.ContactUpdate</triggerUid>
    <messageId>314</messageId>
    <messageDate>2008-09-19T14:14:51.000-07:00</messageDate>
  </eventMetadata>
  <XrefDeleteEvent>
    <sourceSystemName>CRM</sourceSystemName>
    <sourceKey>CPK1256</sourceKey>
    <eventDate>2008-09-19T14:14:51.000-07:00</eventDate>
    <rowid>102          </rowid>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>CPK1256</sourceKey>
    </xrefKey>
    <contactPkg>
      <rowidObject>102          </rowidObject>
      <creator>admin</creator>
```

```

        <createDate>2008-09-19T13:57:09.000-07:00</createDate>
        <updatedBy>sifuser</updatedBy>
        <lastUpdateDate>2008-09-19T14:14:54.000-07:00</lastUpdateDate>
        <consolidationInd>4</consolidationInd>
        <lastRowidSystem>CRM                </lastRowidSystem>
        <dirtyInd>1</dirtyInd>
        <hubStateInd>1</hubStateInd>
    </contactPkg>
</XrefDeleteEvent>
</siperianEvent>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

XRefSetToDelete 메시지

다음은 XRefSetToDelete 메시지의 예입니다.

```

<?xml version="1.0" encoding="UTF-16LE"?>
<siperianEvent>
  <eventMetadata>
    <eventType>XREF set to Delete</eventType>
    <baseObjectId>BASE_OBJECT.C_CONTACT</baseObjectId>
    <packageId>PACKAGE.CONTACT_PKG</packageId>
    <orsId>localhost-mrm-CMX_OR5</orsId>
    <triggerId>MESSAGE_QUEUE_RULE.ContactUpdate</triggerId>
    <messageId>314</messageId>
    <messageDate>2008-09-19T14:14:51.000-07:00</messageDate>
  </eventMetadata>
  <XrefSetToDeleteEvent>
    <sourceSystemName>CRM</sourceSystemName>
    <sourceKey>CPK1256</sourceKey>
    <eventDate>2008-09-19T14:14:51.000-07:00</eventDate>
    <rowid>102                </rowid>
    <xrefKey>
      <systemName>CRM</systemName>
      <sourceKey>CPK1256</sourceKey>
    </xrefKey>
    <contactPkg>
      <rowidObject>102                </rowidObject>
      <creator>admin</creator>
      <createDate>2008-09-19T13:57:09.000-07:00</createDate>
      <updatedBy>sifuser</updatedBy>
      <lastUpdateDate>2008-09-19T14:14:54.000-07:00</lastUpdateDate>
      <consolidationInd>4</consolidationInd>
      <lastRowidSystem>CRM                </lastRowidSystem>
      <dirtyInd>1</dirtyInd>
      <hubStateInd>1</hubStateInd>
    </contactPkg>
  </XrefSetToDeleteEvent>
</siperianEvent>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

레거시 JMS 메시지 XML 참조

MDM Hub 구현에 레거시 JMS 메시지가 필요한 경우 현재 버전의 XML 메시지 형식 대신 레거시 XML 메시지 형식을 사용합니다. 레거시 XML 메시지를 사용하려면 메시지 대기열 도구에서 **레거시 XML 사용** 확인란을 선택합니다.

레거시 XML에 대한 메시지 필드

메시지의 데이터 영역 내용은 트리거에서 지정한 패키지에 따라 결정됩니다.

데이터 영역에는 다음과 같은 메시지 필드가 포함될 수 있습니다.

필드	설명
ACTION	작업 유형: 삽입, 업데이트, XREF 업데이트, 고유 항목으로 허용, 병합, 병합 해제 또는 병합 업데이트
MESSAGE_ID	메시지의 ID입니다. ID 값은 C_REPOS_MQ_DATA_CHANGE 테이블의 값과 동일합니다.
MESSAGE_DATE	이벤트가 발생한 시간입니다.
TABLE_NAME	이 작업의 영향을 받는 교차 참조 개체 테이블 또는 기본 개체 테이블의 이름입니다.
RULE_NAME	이 메시지를 생성한 이벤트를 트리거한 규칙 이름입니다.
RULE_ID	이 메시지를 생성한 이벤트를 트리거한 규칙의 ID입니다.
ROWID_OBJECT	이 작업의 영향을 받는 기본 개체의 고유 키입니다.
MERGED_OBJECTS	병합에서 손실되는 레코드에 대한 ROWID_OBJECT 값 목록입니다. 이 필드는 MERGE 이벤트의 경우에만 메시지에 포함됩니다.
SOURCE_XREF	UPDATE 이벤트를 트리거한 교차 참조의 SYSTEM 및 PKEY_SRC_OBJECT 값입니다. 이 필드는 UPDATE 이벤트의 경우에만 메시지에 포함됩니다.
XREFS	이 기본 개체의 출력 시스템에 있는 모든 교차 참조의 SYSTEM 및 PKEY_SRC_OBJECT 값에 대한 목록입니다.

레거시 XML에 대한 메시지 필터링

MessageType이라는 사용자 지정 JMS 헤더를 사용하여 메시지 유형을 기준으로 수신 메시지를 필터링할 수 있습니다. 메시지 헤더에 표시되는 메시지 유형은 다음과 같습니다.

메시지 유형	설명
SIP_EVENT	이벤트 알림 메시지
<serviceNameReturn>	<p>SIF(서비스 통합 프레임워크) 응답의 경우 다음에 나온 get 요청에 대한 응답의 일부와 같이 SIF 요청의 이름으로 시작됩니다.</p> <pre> <getReturn> records</message> <recordKey> <ROWID>2</ROWID> </recordKey> ... </pre>

레거시 XML에 대한 메시지 예

다음 예제 메시지를 참조로 사용할 수 있습니다.

고유 항목으로 허용 메시지

다음 코드는 고유 항목으로 허용 메시지의 예입니다.

```
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>Accept as Unique</ACTION>
    <MESSAGE_ID>11010294</MESSAGE_ID>
    <MESSAGE_DATE>2005-07-21 16:37:00.0</MESSAGE_DATE>
    <TABLE_NAME>C_CUSTOMER</TABLE_NAME>
    <RULE_NAME>CustomerRule1</RULE_NAME>
    <RULE_ID>SVR1.8E0</RULE_ID>
    <ROWID_OBJECT>74          </ROWID_OBJECT>
    <XREFS>
      <XREF>
        <SYSTEM>CRM</SYSTEM>
        <PKEY_SRC_OBJECT>196      </PKEY_SRC_OBJECT>
      </XREF>
      <XREF>
        <SYSTEM>SFA</SYSTEM>
        <PKEY_SRC_OBJECT>49      </PKEY_SRC_OBJECT>
      </XREF>
    </XREFS>
  </CONTROLAREA>
  <DATAAREA>
    <DATA>
      <ROWID_OBJECT>74          </ROWID_OBJECT>
      <CONSOLIDATION_IND>1</CONSOLIDATION_IND>
      <FIRST_NAME>Jimmy</FIRST_NAME>
      <MIDDLE_NAME>Neville</MIDDLE_NAME>
      <LAST_NAME>Darwent</LAST_NAME>
      <SUFFIX>Jr</SUFFIX>
      <GENDER>M                </GENDER>
      <BIRTH_DATE>1938-06-22</BIRTH_DATE>
      <SALUTATION>Mr</SALUTATION>
      <SSN_TAX_NUMBER>659483774</SSN_TAX_NUMBER>
      <FULL_NAME>Jimmy Darwent, Stony Brook Ny</FULL_NAME>
    </DATA>
  </DATAAREA>
</SIP_EVENT>
```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

BO 삭제 메시지

다음 코드는 BO 삭제 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>BO Delete</ACTION>
    <MESSAGE_ID>11010295</MESSAGE_ID>
    <MESSAGE_DATE>2008-09-19 14:35:53.0</MESSAGE_DATE>
    <TABLE_NAME>C_CONTACT</TABLE_NAME>
    <PACKAGE>CONTACT_PKG</PACKAGE>
    <RULE_NAME>ContactUpdateLegacy</RULE_NAME>
    <RULE_ID>SVR1.28D</RULE_ID>
    <ROWID_OBJECT>107</ROWID_OBJECT>
    <DATABASE>localhost-mrm-CMX_OR</DATABASE>
    <XREFS>
      <XREF>
        <SYSTEM>CRM</SYSTEM>
        <PKEY_SRC_OBJECT />
      </XREF>
      <XREF>
        <SYSTEM>Admin</SYSTEM>
        <PKEY_SRC_OBJECT />
      </XREF>
    </XREFS>
  </CONTROLAREA>
</SIP_EVENT>
```

```

        <XREF>
          <SYSTEM>WEB</SYSTEM>
          <PKEY_SRC_OBJECT />
        </XREF>
      </XREFS>
    </CONTROLAREA>
    <DATAAREA>
      <DATA>
        <ROWID_OBJECT>107</ROWID_OBJECT>
        <CREATOR>sifuser</CREATOR>
        <CREATE_DATE>19 Sep 2008 14:35:28</CREATE_DATE>
        <UPDATED_BY>admin</UPDATED_BY>
        <LAST_UPDATE_DATE>19 Sep 2008 14:35:53</LAST_UPDATE_DATE>
        <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
        <DELETED_IND />
        <DELETED_BY />
        <DELETED_DATE />
        <LAST_ROWID_SYSTEM>CRM</LAST_ROWID_SYSTEM>
        <INTERACTION_ID />
        <FIRST_NAME>John</FIRST_NAME>
        <LAST_NAME>Smith</LAST_NAME>
        <HUB_STATE_IND>-1</HUB_STATE_IND>
      </DATA>
    </DATAAREA>
  </SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

삭제할 BO 집합

다음 코드는 삭제할 BO 집합 메시지의 예입니다.

```

<?xml version="1.0" encoding="UTF-16LE"?>
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>BO set to Delete</ACTION>
    <MESSAGE_ID>11010296</MESSAGE_ID>
    <MESSAGE_DATE>2008-09-19 14:21:03.0</MESSAGE_DATE>
    <TABLE_NAME>C_CONTACT</TABLE_NAME>
    <PACKAGE>CONTACT_PKG</PACKAGE>
    <RULE_NAME>ContactUpdateLegacy</RULE_NAME>
    <RULE_ID>SVR1.28D</RULE_ID>
    <ROWID_OBJECT>102</ROWID_OBJECT>
    <DATABASE>localhost-mrm-CMX_OR</DATABASE>
  </XREFS>
    <XREF>
      <SYSTEM>CRM</SYSTEM>
      <PKEY_SRC_OBJECT />
    </XREF>
    <XREF>
      <SYSTEM>Admin</SYSTEM>
      <PKEY_SRC_OBJECT />
    </XREF>
    <XREF>
      <SYSTEM>WEB</SYSTEM>
      <PKEY_SRC_OBJECT />
    </XREF>
  </XREFS>
</CONTROLAREA>
<DATAAREA>
  <DATA>
    <ROWID_OBJECT>102</ROWID_OBJECT>
    <CREATOR>admin</CREATOR>
    <CREATE_DATE>19 Sep 2008 13:57:09</CREATE_DATE>
    <UPDATED_BY>admin</UPDATED_BY>
    <LAST_UPDATE_DATE>19 Sep 2008 14:21:03</LAST_UPDATE_DATE>
    <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
  </DATA>
</DATAAREA>
</SIP_EVENT>

```



```

<DELETED_IND />
<DELETED_BY />
<DELETED_DATE />
<LAST_ROWID_SYSTEM>SYS0</LAST_ROWID_SYSTEM>
<INTERACTION_ID />
<FIRST_NAME />
<LAST_NAME />
<HUB_STATE_IND>-1</HUB_STATE_IND>
</DATA>
</DATAAREA>
</SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

삭제 메시지

다음 코드는 삭제 메시지의 예입니다.

```

<?xml version="1.0" encoding="UTF-16LE"?>
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>Delete</ACTION>
    <MESSAGE_ID>11010297</MESSAGE_ID>
    <MESSAGE_DATE>2008-09-19 14:35:53.0</MESSAGE_DATE>
    <TABLE_NAME>C_CONTACT</TABLE_NAME>
    <PACKAGE>CONTACT_PKG</PACKAGE>
    <RULE_NAME>ContactUpdateLegacy</RULE_NAME>
    <RULE_ID>SVR1.28D</RULE_ID>
    <ROWID_OBJECT>107</ROWID_OBJECT>
    <DATABASE>localhost-mrm-CMX_ORS</DATABASE>
    <XREFS>
      <XREF>
        <SYSTEM>CRM</SYSTEM>
        <PKEY_SRC_OBJECT />
      </XREF>
      <XREF>
        <SYSTEM>Admin</SYSTEM>
        <PKEY_SRC_OBJECT />
      </XREF>
      <XREF>
        <SYSTEM>WEB</SYSTEM>
        <PKEY_SRC_OBJECT />
      </XREF>
    </XREFS>
  </CONTROLAREA>
  <DATAAREA>
    <DATA>
      <ROWID_OBJECT>107</ROWID_OBJECT>
      <CREATOR>siuser</CREATOR>
      <CREATE_DATE>19 Sep 2008 14:35:28</CREATE_DATE>
      <UPDATED_BY>admin</UPDATED_BY>
      <LAST_UPDATE_DATE>19 Sep 2008 14:35:53</LAST_UPDATE_DATE>
      <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
      <DELETED_IND />
      <DELETED_BY />
      <DELETED_DATE />
      <LAST_ROWID_SYSTEM>CRM</LAST_ROWID_SYSTEM>
      <INTERACTION_ID />
      <FIRST_NAME>John</FIRST_NAME>
      <LAST_NAME>Smith</LAST_NAME>
      <HUB_STATE_IND>-1</HUB_STATE_IND>
    </DATA>
  </DATAAREA>
</SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

삽입 메시지

다음 코드는 삽입 메시지의 예입니다.

```
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>Insert</ACTION>
    <MESSAGE_ID>11010298</MESSAGE_ID>
    <MESSAGE_DATE>2005-07-21 16:07:26.0</MESSAGE_DATE>
    <TABLE_NAME>C_CUSTOMER</TABLE_NAME>
    <RULE_NAME>CustomerRule1</RULE_NAME>
    <RULE_ID>SVR1.8E0</RULE_ID>
    <ROWID_OBJECT>33          </ROWID_OBJECT>
    <XREFS>
      <XREF>
        <SYSTEM>CRM</SYSTEM>
        <PKEY_SRC_OBJECT>49 </PKEY_SRC_OBJECT>
      </XREF>
    </XREFS>
  </CONTROLAREA>
  <DATAAREA>
    <DATA>
      <ROWID_OBJECT>33          </ROWID_OBJECT>
      <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
      <FIRST_NAME>James</FIRST_NAME>
      <MIDDLE_NAME>Neville</MIDDLE_NAME>
      <LAST_NAME>Darwent</LAST_NAME>
      <SUFFIX>Unknown</SUFFIX>
      <GENDER>M                </GENDER>
      <BIRTH_DATE>1938-06-22</BIRTH_DATE>
      <SALUTATION>Mr</SALUTATION>
      <SSN_TAX_NUMBER>216275400</SSN_TAX_NUMBER>
      <FULL_NAME>James Darwent,Stony Brook Ny</FULL_NAME>
    </DATA>
  </DATAAREA>
</SIP_EVENT>
```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

병합 메시지

다음 코드는 병합 메시지의 예입니다.

```
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>Merge</ACTION>
    <MESSAGE_ID>11010299</MESSAGE_ID>
    <MESSAGE_DATE>2005-07-21 16:34:28.0</MESSAGE_DATE>
    <TABLE_NAME>C_CUSTOMER</TABLE_NAME>
    <RULE_NAME>CustomerRule1</RULE_NAME>
    <RULE_ID>SVR1.8E0</RULE_ID>
    <ROWID_OBJECT>74          </ROWID_OBJECT>
    <XREFS>
      <XREF>
        <SYSTEM>CRM</SYSTEM>
        <PKEY_SRC_OBJECT>196    </PKEY_SRC_OBJECT>
      </XREF>
      <XREF>
        <SYSTEM>SFA</SYSTEM>
        <PKEY_SRC_OBJECT>49    </PKEY_SRC_OBJECT>
      </XREF>
    </XREFS>
    <MERGED_OBJECTS>
      <ROWID_OBJECT>7          </ROWID_OBJECT>
    </MERGED_OBJECTS>
  </CONTROLAREA>
  <DATAAREA>
    <DATA>
```

```

        <ROWID_OBJECT>74                </ROWID_OBJECT>
        <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
        <FIRST_NAME>Jimmy</FIRST_NAME>
        <MIDDLE_NAME>Neville</MIDDLE_NAME>
        <LAST_NAME>Darwent</LAST_NAME>
        <SUFFIX>Jr</SUFFIX>
        <GENDER>M                        </GENDER>
        <BIRTH_DATE>1938-06-22</BIRTH_DATE>
        <SALUTATION>Mr</SALUTATION>
        <SSN_TAX_NUMBER>659483774</SSN_TAX_NUMBER>
        <FULL_NAME>Jimmy Darwent, Stony Brook Ny</FULL_NAME>
    </DATA>
</DATAAREA>
</SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

병합 업데이트 메시지

다음 코드는 병합 업데이트 메시지의 예입니다.

```

<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>Merge Update</ACTION>
    <MESSAGE_ID>11010310</MESSAGE_ID>
    <MESSAGE_DATE>2005-07-21 16:34:28.0</MESSAGE_DATE>
    <TABLE_NAME>C_CUSTOMER</TABLE_NAME>
    <RULE_NAME>CustomerRule1</RULE_NAME>
    <RULE_ID>SVR1.8E0</RULE_ID>
    <ROWID_OBJECT>74                </ROWID_OBJECT>
    <XREFS>
      <XREF>
        <SYSTEM>CRM</SYSTEM>
        <PKEY_SRC_OBJECT>196        </PKEY_SRC_OBJECT>
      </XREF>
      <XREF>
        <SYSTEM>SFA</SYSTEM>
        <PKEY_SRC_OBJECT>49        </PKEY_SRC_OBJECT>
      </XREF>
    </XREFS>
    <MERGED_OBJECTS>
      <ROWID_OBJECT>7                </ROWID_OBJECT>
    </MERGED_OBJECTS>
  </CONTROLAREA>
</DATAAREA>
<DATA>
  <ROWID_OBJECT>74                </ROWID_OBJECT>
  <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
  <FIRST_NAME>Jimmy</FIRST_NAME>
  <MIDDLE_NAME>Neville</MIDDLE_NAME>
  <LAST_NAME>Darwent</LAST_NAME>
  <SUFFIX>Jr</SUFFIX>
  <GENDER>M                        </GENDER>
  <BIRTH_DATE>1938-06-22</BIRTH_DATE>
  <SALUTATION>Mr</SALUTATION>
  <SSN_TAX_NUMBER>659483774</SSN_TAX_NUMBER>
  <FULL_NAME>Jimmy Darwent, Stony Brook Ny</FULL_NAME>
</DATA>
</DATAAREA>
</SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

삽입 보류 중 메시지

다음 코드는 삽입 보류 중 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>Pending Insert</ACTION>
    <MESSAGE_ID>11010309</MESSAGE_ID>
    <MESSAGE_DATE>2008-09-19 13:57:10.0</MESSAGE_DATE>
    <TABLE_NAME>C_CONTACT</TABLE_NAME>
    <PACKAGE>CONTACT_PKG</PACKAGE>
    <RULE_NAME>ContactUpdateLegacy</RULE_NAME>
    <RULE_ID>SVR1.28D</RULE_ID>
    <ROWID_OBJECT>102</ROWID_OBJECT>
    <DATABASE>localhost-mrm-CMX_ORS</DATABASE>
    <XREFS>
      <XREF>
        <SYSTEM>Admin</SYSTEM>
        <PKEY_SRC_OBJECT>SVR1.2V3</PKEY_SRC_OBJECT>
      </XREF>
    </XREFS>
  </CONTROLAREA>
  <DATAAREA>
    <DATA>
      <ROWID_OBJECT>102</ROWID_OBJECT>
      <CREATOR>admin</CREATOR>
      <CREATE_DATE>19 Sep 2008 13:57:09</CREATE_DATE>
      <UPDATED_BY>admin</UPDATED_BY>
      <LAST_UPDATE_DATE>19 Sep 2008 13:57:09</LAST_UPDATE_DATE>
      <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
      <DELETED_IND />
      <DELETED_BY />
      <DELETED_DATE />
      <LAST_ROWID_SYSTEM>SYS0</LAST_ROWID_SYSTEM>
      <INTERACTION_ID />
      <FIRST_NAME>John</FIRST_NAME>
      <LAST_NAME>Smith</LAST_NAME>
      <HUB_STATE_IND>0</HUB_STATE_IND>
    </DATA>
  </DATAAREA>
</SIP_EVENT>
```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

업데이트 보류 중 메시지

다음 코드는 업데이트 보류 중 메시지의 예입니다.

```
<?xml version="1.0" encoding="UTF-16LE"?>
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>Pending Update</ACTION>
    <MESSAGE_ID>11010308</MESSAGE_ID>
    <MESSAGE_DATE>2008-09-19 14:01:36.0</MESSAGE_DATE>
    <TABLE_NAME>C_CONTACT</TABLE_NAME>
    <PACKAGE>CONTACT_PKG</PACKAGE>
    <RULE_NAME>ContactUpdateLegacy</RULE_NAME>
    <RULE_ID>SVR1.28D</RULE_ID>
    <ROWID_OBJECT>102</ROWID_OBJECT>
    <DATABASE>localhost-mrm-CMX_ORS</DATABASE>
    <XREFS>
      <XREF>
        <SYSTEM>CRM</SYSTEM>
        <PKEY_SRC_OBJECT>CPK125</PKEY_SRC_OBJECT>
      </XREF>
      <XREF>
        <SYSTEM>Admin</SYSTEM>
      </XREF>
    </XREFS>
  </CONTROLAREA>
  <DATAAREA>
    <DATA>
      <ROWID_OBJECT>102</ROWID_OBJECT>
      <CREATOR>admin</CREATOR>
      <CREATE_DATE>19 Sep 2008 14:01:36</CREATE_DATE>
      <UPDATED_BY>admin</UPDATED_BY>
      <LAST_UPDATE_DATE>19 Sep 2008 14:01:36</LAST_UPDATE_DATE>
      <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
      <DELETED_IND />
      <DELETED_BY />
      <DELETED_DATE />
      <LAST_ROWID_SYSTEM>SYS0</LAST_ROWID_SYSTEM>
      <INTERACTION_ID />
      <FIRST_NAME>John</FIRST_NAME>
      <LAST_NAME>Smith</LAST_NAME>
      <HUB_STATE_IND>0</HUB_STATE_IND>
    </DATA>
  </DATAAREA>
</SIP_EVENT>
```

```

        <PKEY_SRC_OBJECT>SVR1.2V3</PKEY_SRC_OBJECT>
    </XREF>
</XREFS>
</CONTROLAREA>
<DATAAREA>
    <DATA>
        <ROWID_OBJECT>102</ROWID_OBJECT>
        <CREATOR>admin</CREATOR>
        <CREATE_DATE>19 Sep 2008 13:57:09</CREATE_DATE>
        <UPDATED_BY>sifuser</UPDATED_BY>
        <LAST_UPDATE_DATE>19 Sep 2008 14:01:36</LAST_UPDATE_DATE>
        <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
        <DELETED_IND />
        <DELETED_BY />
        <DELETED_DATE />
        <LAST_ROWID_SYSTEM>CRM</LAST_ROWID_SYSTEM>
        <INTERACTION_ID />
        <FIRST_NAME>John</FIRST_NAME>
        <LAST_NAME>Smith</LAST_NAME>
        <HUB_STATE_IND>1</HUB_STATE_IND>
    </DATA>
</DATAAREA>
</SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

XREF 업데이트 보류 중 메시지

다음 코드는 XREF 업데이트 보류 중 메시지의 예입니다.

```

<?xml version="1.0" encoding="UTF-16LE"?>
<SIP_EVENT>
    <CONTROLAREA>
        <ACTION>Pending Update XREF</ACTION>
        <MESSAGE_ID>11010307</MESSAGE_ID>
        <MESSAGE_DATE>2008-09-19 14:01:36.0</MESSAGE_DATE>
        <TABLE_NAME>C_CONTACT</TABLE_NAME>
        <PACKAGE>CONTACT_ADDRESS_PKG</PACKAGE>
        <RULE_NAME>ContactAM</RULE_NAME>
        <RULE_ID>SVR1.1VU</RULE_ID>
        <ROWID_OBJECT>102</ROWID_OBJECT>
        <DATABASE>localhost-mrm-CMX_ORS</DATABASE>
        <XREFS>
            <XREF>
                <SYSTEM>CRM</SYSTEM>
                <PKEY_SRC_OBJECT>CPK125</PKEY_SRC_OBJECT>
            </XREF>
            <XREF>
                <SYSTEM>Admin</SYSTEM>
                <PKEY_SRC_OBJECT>SVR1.2V3</PKEY_SRC_OBJECT>
            </XREF>
        </XREFS>
    </CONTROLAREA>
    <DATAAREA>
        <DATA>
            <ROWID_CONTACT>102</ROWID_CONTACT>
            <CREATOR>admin</CREATOR>
            <CREATE_DATE>19 Sep 2008 13:57:09</CREATE_DATE>
            <UPDATED_BY>sifuser</UPDATED_BY>
            <LAST_UPDATE_DATE>19 Sep 2008 14:01:36</LAST_UPDATE_DATE>
            <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
            <DELETED_IND />
            <DELETED_BY />
            <DELETED_DATE />
            <LAST_ROWID_SYSTEM>CRM</LAST_ROWID_SYSTEM>
            <INTERACTION_ID />
            <FIRST_NAME>John</FIRST_NAME>

```

```

        <LAST_NAME>Smith</LAST_NAME>
        <HUB_STATE_IND>1</HUB_STATE_IND>
        <CITY />
        <STATE />
    </DATA>
</DATAAREA>
</SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

업데이트 메시지

다음 코드는 업데이트 메시지의 예입니다.

```

<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>Update</ACTION>
    <MESSAGE_ID>11010305</MESSAGE_ID>
    <MESSAGE_DATE>2005-07-21 16:44:53.0</MESSAGE_DATE>
    <TABLE_NAME>C_CUSTOMER</TABLE_NAME>
    <RULE_NAME>CustomerRule1</RULE_NAME>
    <RULE_ID>SVR1.8E0</RULE_ID>
    <ROWID_OBJECT>74          </ROWID_OBJECT>
    <SOURCE_XREF>
      <SYSTEM>Admin</SYSTEM>
      <PKEY_SRC_OBJECT>196      </PKEY_SRC_OBJECT>
    </SOURCE_XREF>
    <XREFS>
      <XREF>
        <SYSTEM>CRM</SYSTEM>
        <PKEY_SRC_OBJECT>196      </PKEY_SRC_OBJECT>
      </XREF>
      <XREF>
        <SYSTEM>SFA</SYSTEM>
        <PKEY_SRC_OBJECT>49      </PKEY_SRC_OBJECT>
      </XREF>
      <XREF>
        <SYSTEM>Admin</SYSTEM>
        <PKEY_SRC_OBJECT>74      </PKEY_SRC_OBJECT>
      </XREF>
    </XREFS>
  </CONTROLAREA>
  <DATAAREA>
    <DATA>
      <ROWID_OBJECT>74          </ROWID_OBJECT>
      <CONSOLIDATION_IND>1</CONSOLIDATION_IND>
      <FIRST_NAME>Jimmy</FIRST_NAME>
      <MIDDLE_NAME>Neville</MIDDLE_NAME>
      <LAST_NAME>Darwent</LAST_NAME>
      <SUFFIX>Jr</SUFFIX>
      <GENDER>M          </GENDER>
      <BIRTH_DATE>1938-06-22</BIRTH_DATE>
      <SALUTATION>Mr</SALUTATION>
      <SSN_TAX_NUMBER>659483773</SSN_TAX_NUMBER>
      <FULL_NAME>Jimmy Darwent, Stony Brook Ny</FULL_NAME>
    </DATA>
  </DATAAREA>
</SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

XREF 업데이트 메시지

다음 코드는 XREF 업데이트 메시지의 예입니다.

```
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>Update XREF</ACTION>
    <MESSAGE_ID>11010303</MESSAGE_ID>
    <MESSAGE_DATE>2005-07-21 16:44:53.0</MESSAGE_DATE>
    <TABLE_NAME>C_CUSTOMER</TABLE_NAME>
    <RULE_NAME>CustomerRule1</RULE_NAME>
    <RULE_ID>SVR1.8E0</RULE_ID>
    <ROWID_OBJECT>74          </ROWID_OBJECT>
    <SOURCE_XREF>
      <SYSTEM>Admin</SYSTEM>
      <PKEY_SRC_OBJECT>196      </PKEY_SRC_OBJECT>
    </SOURCE_XREF>
    <XREFS>
      <XREF>
        <SYSTEM>CRM</SYSTEM>
        <PKEY_SRC_OBJECT>196      </PKEY_SRC_OBJECT>
      </XREF>
      <XREF>
        <SYSTEM>SFA</SYSTEM>
        <PKEY_SRC_OBJECT>49      </PKEY_SRC_OBJECT>
      </XREF>
      <XREF>
        <SYSTEM>Admin</SYSTEM>
        <PKEY_SRC_OBJECT>74      </PKEY_SRC_OBJECT>
      </XREF>
    </XREFS>
  </CONTROLAREA>
  <DATAAREA>
    <DATA>
      <ROWID_OBJECT>74          </ROWID_OBJECT>
      <CONSOLIDATION_IND>1</CONSOLIDATION_IND>
      <FIRST_NAME>Jimmy</FIRST_NAME>
      <MIDDLE_NAME>Neville</MIDDLE_NAME>
      <LAST_NAME>Darwent</LAST_NAME>
      <SUFFIX>Jr</SUFFIX>
      <GENDER>M                </GENDER>
      <BIRTH_DATE>1938-06-22</BIRTH_DATE>
      <SALUTATION>Mr</SALUTATION>
      <SSN_TAX_NUMBER>659483773</SSN_TAX_NUMBER>
      <FULL_NAME>Jimmy Darwent, Stony Brook Ny</FULL_NAME>
    </DATA>
  </DATAAREA>
</SIP_EVENT>
```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

병합 해제 메시지

다음 코드는 병합 해제 메시지의 예입니다.

```
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>UnMerge</ACTION>
    <MESSAGE_ID>11010302</MESSAGE_ID>
    <MESSAGE_DATE>2006-11-07 21:37:56.0</MESSAGE_DATE>
    <TABLE_NAME>C_CONSUMER</TABLE_NAME>
    <PACKAGE>CONSUMER_PKG</PACKAGE>
    <RULE_NAME>Unmerge</RULE_NAME>
    <RULE_ID>SVR1.97S</RULE_ID>
    <ROWID_OBJECT>10</ROWID_OBJECT>
    <DATABASE>edsel-edsel2-CMX_AT</DATABASE>
    <XREFS>
      <XREF>
    </XREF>
  </CONTROLAREA>
```

```

        <SYSTEM>Retail System</SYSTEM>
        <PKEY_SRC_OBJECT>8</PKEY_SRC_OBJECT>
        </XREF>
    </XREFS>
    <MERGED_OBJECTS>
        <ROWID_OBJECT>0</ROWID_OBJECT>
    </MERGED_OBJECTS>
</CONTROLAREA>
<DATAAREA>
    <DATA>
        <ROWID_OBJECT>10</ROWID_OBJECT>
        <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
        <LAST_ROWID_SYSTEM>SVR1.7NK</LAST_ROWID_SYSTEM>
        <INTERACTION_ID />
        <CONSUMER_ID>8</CONSUMER_ID>
        <FIRST_NAME>THOMAS</FIRST_NAME>
        <MIDDLE_NAME>L</MIDDLE_NAME>
        <LAST_NAME>KIDD</LAST_NAME>
        <SUFFIX />
        <TELEPHONE>2178952323</TELEPHONE>
        <GENDER>M</GENDER>
        <DOB>1940</DOB>
    </DATA>
</DATAAREA>
</SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

XREF 삭제 메시지

다음 코드는 XREF 삭제 메시지의 예입니다.

```

<?xml version="1.0" encoding="UTF-16LE"?>
<SIP_EVENT>
    <CONTROLAREA>
        <ACTION>XREF Delete</ACTION>
        <MESSAGE_ID>11010301</MESSAGE_ID>
        <MESSAGE_DATE>2008-09-19 14:14:51.0</MESSAGE_DATE>
        <TABLE_NAME>C_CONTACT</TABLE_NAME>
        <PACKAGE>CONTACT_PKG</PACKAGE>
        <RULE_NAME>ContactUpdateLegacy</RULE_NAME>
        <RULE_ID>SVR1.28D</RULE_ID>
        <ROWID_OBJECT>102 </ROWID_OBJECT>
        <DATABASE>localhost-mrm-CMX_ORS</DATABASE>
    <XREFS>
        <XREF>
            <SYSTEM>CRM</SYSTEM>
            <PKEY_SRC_OBJECT>CPK1256</PKEY_SRC_OBJECT>
        </XREF>
    </XREFS>
</CONTROLAREA>
<DATAAREA>
    <DATA>
        <ROWID_OBJECT>102</ROWID_OBJECT>
        <CREATOR>admin</CREATOR>
        <CREATE_DATE>19 Sep 2008 13:57:09</CREATE_DATE>
        <UPDATED_BY>sifuser</UPDATED_BY>
        <LAST_UPDATE_DATE>19 Sep 2008 14:14:54</LAST_UPDATE_DATE>
        <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
        <DELETED_IND />
        <DELETED_BY />
        <DELETED_DATE />
        <LAST_ROWID_SYSTEM>CRM</LAST_ROWID_SYSTEM>
        <INTERACTION_ID />
        <FIRST_NAME />
        <LAST_NAME />
        <HUB_STATE_IND>1</HUB_STATE_IND>
    </DATA>
</DATAAREA>
</SIP_EVENT>

```



```

        </DATA>
      </DATAAREA>
    </SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

삭제할 XREF 집합

다음 코드는 삭제할 XREF 집합 메시지의 예입니다.

```

<?xml version="1.0" encoding="UTF-16LE"?>
<SIP_EVENT>
  <CONTROLAREA>
    <ACTION>XREF set to Delete</ACTION>
    <MESSAGE_ID>11010300</MESSAGE_ID>
    <MESSAGE_DATE>2008-09-19 14:14:51.0</MESSAGE_DATE>
    <TABLE_NAME>C_CONTACT</TABLE_NAME>
    <PACKAGE>CONTACT_PKG</PACKAGE>
    <RULE_NAME>ContactUpdateLegacy</RULE_NAME>
    <RULE_ID>SVR1.28D</RULE_ID>
    <ROWID_OBJECT>102</ROWID_OBJECT>
    <DATABASE>localhost-mrm-CMX_ORS</DATABASE>
    <XREFS>
      <XREF>
        <SYSTEM>CRM</SYSTEM>
        <PKEY_SRC_OBJECT>CPK1256</PKEY_SRC_OBJECT>
      </XREF>
    </XREFS>
  </CONTROLAREA>
  <DATAAREA>
    <DATA>
      <ROWID_OBJECT>102</ROWID_OBJECT>
      <CREATOR>admin</CREATOR>
      <CREATE_DATE>19 Sep 2008 13:57:09</CREATE_DATE>
      <UPDATED_BY>sifuser</UPDATED_BY>
      <LAST_UPDATE_DATE>19 Sep 2008 14:14:54</LAST_UPDATE_DATE>
      <CONSOLIDATION_IND>4</CONSOLIDATION_IND>
      <DELETED_IND />
      <DELETED_BY />
      <DELETED_DATE />
      <LAST_ROWID_SYSTEM>CRM</LAST_ROWID_SYSTEM>
      <INTERACTION_ID />
      <FIRST_NAME />
      <LAST_NAME />
      <HUB_STATE_IND>1</HUB_STATE_IND>
    </DATA>
  </DATAAREA>
</SIP_EVENT>

```

실제 메시지는 이와 정확하게 일치하지 않을 수 있습니다. 데이터에는 사용자의 데이터가 반영되고 필드에는 사용자의 패키지가 반영됩니다.

파트 V: Informatica MDM Hub 프로세스 실행

이 파트에 포함된 장:

- [일괄 작업 사용, 519](#)
- [사용자 종료, 565](#)

제 27 장

일괄 작업 사용

이 장에 포함된 항목:

- [일괄 작업 사용 개요, 519](#)
- [일괄 작업 스레드 구성, 520](#)
- [일괄 작업 실행, 521](#)
- [일괄 작업에 사용되는 지원 테이블, 522](#)
- [차례로 일괄 작업 실행, 522](#)
- [일괄 작업에 대한 모범 사례, 523](#)
- [Oracle 환경에서의 통계 수집을 위한 병렬도 제한, 523](#)
- [일괄 작업 생성, 524](#)
- [정보 전용 일괄 작업\(Hub 콘솔에서 실행되지 않음\), 525](#)
- [처리 서버 구성, 525](#)
- [일괄 처리 뷰어 도구를 사용하여 일괄 작업 실행, 526](#)
- [일괄 그룹 도구를 사용하여 일괄 작업 실행, 534](#)
- [일괄 작업 참조, 543](#)

일괄 작업 사용 개요

Hub 콘솔에서 일괄 처리 뷰어 및 일괄 그룹 도구를 사용하여 MDM Hub 일괄 작업을 구성 및 실행할 수 있습니다.

MDM Hub에서 일괄 작업은 실행 시 불연속 단위의 작업을 수행하는 프로그램입니다. 이 불연속 단위의 작업을 프로세스라고 합니다. 예를 들어, 일치 작업은 일치 프로세스를 수행합니다. MDM Hub이 일치 후보를 검색하고, 일치 규칙을 일치 후보에 적용하며, 일치를 작성한 다음, 자동 또는 수동 통합을 위해 일치를 대기열에 넣습니다. 병합 스타일 기본 개체의 경우 자동 병합 작업에서 자동 통합을 처리합니다. 수동 병합 작업에서 수동 통합을 처리합니다.

모든 MDM Hub 일괄 작업은 다중 스레드입니다. 다중 스레드 작업으로 인해 단일 프로세스의 컨텍스트 내에 여러 스레드가 있을 수 있습니다. 또한, MDM Hub 일괄 작업은 상위 기본 개체의 일치 경로에 있는 모든 하위 기본 개체에서 병렬로 실행될 수 있습니다.

일괄 작업을 사용하려면 먼저 다음 선행 작업을 수행해야 합니다.

- Informatica MDM Hub를 설치하고 Hub 저장소를 작성합니다.
- 스키마를 작성합니다.

일괄 작업 스레드 구성

다중 스레딩은 단일 프로세스 컨텍스트 내에 다중 스레드가 존재할 수 있도록 허용하는 일반적인 프로그래밍 모델입니다. 로드 작업 및 유효성 다시 검사 작업을 포함한 모든 MDM Hub 일괄 작업은 다중 스레드됩니다.

MDM Hub가 일괄 작업을 실행할 때 MDM Hub는 일괄 처리를 위해 대기된 레코드를 MDM Hub가 병렬로 처리할 수 있는 블록으로 나눕니다. `cmxserver.properties` 파일에서 각 일괄 작업에 사용할 스레드 수를 구성할 수 있습니다.

MDM Hub가 상위 기본 개체에서 로드 작업을 실행하면 MDM Hub가 해당하는 하위 기본 개체에 일괄 잠금을 적용합니다. 하위 기본 개체에 일괄 잠금이 적용되면 다른 상위 항목이 로드 또는 병합 작업을 병렬로 실행할 수 없습니다. 하위 기본 개체와 상위 기본 개체 간에 고유한 키 관계가 있는 경우 MDM Hub는 하위 기본 개체에 일괄 잠금을 적용합니다.

다중 스레드 일괄 작업 프로세스

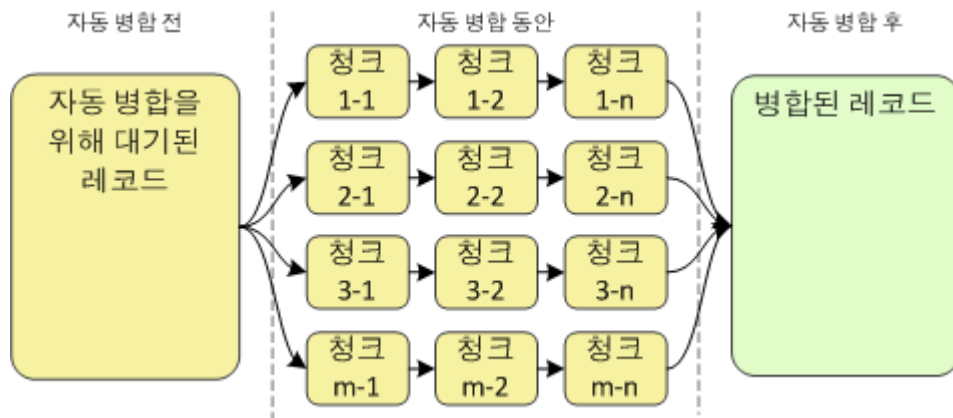
MDM Hub에서 일괄 작업을 처리할 때는 레코드를 병렬로 처리하기 위해 스레드 및 블록을 생성합니다. 일괄 작업을 구성할 때 일괄 작업으로 레코드를 처리하기 위한 스레드 수와 블록 크기를 구성합니다.

1. 일괄 작업을 실행할 때 MDM Hub는 총 레코드 수를 사용자가 구성한 블록 크기로 나눕니다.
2. MDM Hub는 사용자가 구성한 스레드 수를 기반으로 스레드를 생성합니다.
3. MDM Hub는 처리할 각 스레드에 하나의 블록을 할당합니다.
4. 스레드에서 블록 처리를 완료하고 나면 MDM Hub는 대기열의 처리되지 않은 다음 블록을 스레드에 할당합니다. 이 단계는 모든 블록이 처리될 때까지 반복됩니다.

다중 스레드 일괄 작업 예

MDM Hub에서는 사용자가 구성한 블록 크기를 사용하여 각 블록에서 처리해야 하는 레코드 수를 결정합니다. MDM Hub는 처리할 총 레코드 수를 블록 크기로 나눕니다.

일괄 처리에서 처리할 레코드 수가 3000개이고, 구성한 스레드 수가 4개이고, 블록 크기가 250인 시나리오를 가정합니다. MDM Hub에서는 총 레코드 수를 블록 크기로 나누므로, 각각 250개의 레코드가 포함된 12개의 블록이 생성됩니다. 그런 다음 MDM Hub는 처리할 각 스레드에 하나의 블록을 할당합니다. 스레드가 블록 처리를 완료하면 대기열의 다음 블록이 스레드에 할당됩니다. MDM Hub는 모든 블록이 처리될 때까지 확보된 스레드에 블록을 계속 할당합니다.



다중 스레드 일괄 처리 성능

스레드 개수가 현재 환경에서 효율적으로 처리할 수 있는 스레드 수보다 많은 경우 일괄 처리 성능이 저하될 수 있습니다. 또한 성능 및 블록 크기는 데이터베이스 환경에 따라 다릅니다.

MDM Hub에서 다중 스레드 로드를 구성하거나 다중 하위 기본 개체의 작업을 병합하여 순차적으로 실행되도록 구성해야 합니다.

다중 스레드 일괄 작업 속성

다중 스레드 일괄 작업에 대해 처리할 블록 크기 및 사용할 스레드 수를 구성해야 합니다.

cmxserver.properties 파일에서 각 일괄 작업에 사용할 스레드 수를 구성합니다. 다음 테이블에는 다중 스레드 일괄 작업에 대해 구성할 속성이 설명되어 있습니다.

속성	설명
cmx.server.automerge.threads_per_job	MDM Hub에서 자동 병합 일괄 작업을 처리하는 데 사용하는 스레드 수입니다. 기본값은 1입니다.
cmx.server.automerge.block_size	자동 병합 작업에 대해 각 블록에서 처리할 레코드 수입니다. 기본값은 250입니다.
cmx.server.batch.threads_per_job	MDM Hub에서 로드를 처리하고, BVT를 다시 계산하고, 일괄 작업의 유효성을 다시 검사하는 데 사용하는 스레드 수입니다. 기본값은 10입니다. cmx.server.batch.threads_per_job 값은 모든 처리 서버에서 일괄 처리에 사용할 수 있는 총 스레드 수보다 작거나 이와 같아야 합니다.
cmx.server.batch.load.block_size	로드 작업에 대해 각 블록에서 처리할 레코드 수입니다. 기본값은 250입니다.
cmx.server.batch.recalculate.block_size	BVT 다시 계산 및 유효성 다시 검사 작업에 대해 각 블록에서 처리할 레코드 수입니다. 기본값은 250입니다.

일괄 작업 실행

일괄 작업은 개별적으로 실행하거나, MDM Hub 콘솔에서 그룹으로 실행하거나, 서비스 통합 프레임워크 API를 사용하여 실행할 수 있습니다.

다음 도구를 사용하여 일괄 작업을 실행할 수 있습니다.

일괄 처리 뷰어 도구

MDM Hub 콘솔의 일괄 처리 뷰어 도구를 사용하여 개별 일괄 작업을 실행합니다.

일괄 그룹 도구

MDM Hub 콘솔의 일괄 그룹 도구를 사용하여 일괄 작업을 그룹으로 실행합니다. 일괄 그룹 도구를 사용하여 일괄 작업의 실행 시퀀스를 설정하거나 일괄 작업을 병렬로 실행할 수 있습니다.

개별 서비스 통합 프레임워크 API

MDM Hub 콘솔에서 사용 가능한 각 일괄 작업에는 해당하는 서비스 통합 프레임워크 API가 있습니다. API를 사용하여 개별 일괄 작업을 실행합니다.

ExecuteBatchGroup 서비스 통합 프레임워크 API

ExecuteBatchGroup 서비스 통합 프레임워크 API를 사용하여 일괄 그룹을 실행합니다.

서비스 통합 프레임워크 API에 대한 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

일괄 작업에 사용되는 지원 테이블

다음 테이블에는 Informatica MDM Hub 일괄 작업에 사용되는 다양한 지원 테이블이 나열되어 있습니다.

- 랜딩 테이블
- 준비 테이블
- Raw 테이블
- 거부 테이블
- 일치 키 테이블
- 일치 테이블
- 시스템 제어 및 기록 테이블
- XREF 테이블

차례로 일괄 작업 실행

특정 일괄 작업은 수행하기 전에 먼저 다른 일괄 작업을 완료해야 합니다. 예를 들어 모든 일괄 작업을 실행하기 전에 기본 개체의 랜딩 테이블을 채워야 합니다. 마찬가지로 기본 개체에 대해 일치 작업을 실행하기 전에 해당 준비 및 로드 작업을 실행해야 합니다. 마지막으로 기본 개체에 종속성이 포함된 경우(예를 들어 상위 테이블의 하위이거나 다른 기본 개체를 가리키는 외래 키 관계가 포함된 경우) 기본 개체가 종속된 테이블에 대해 먼저 일괄 작업을 실행해야 합니다. 사용자나 사용자 조직은 일괄 작업을 실행하기 전에 일괄 처리 및 종속성을 완료해야 함을 지정하는 관리 또는 작업 계획을 개발하는 모범 사례를 고려해야 합니다.

일괄 작업을 실행하기 전에 랜딩 테이블 채우기

Informatica MDM Hub 일괄 작업이 수행하는 태스크 중 하나는 데이터를 랜딩 테이블에서 Informatica MDM Hub의 적합한 대상 위치로 옮기는 것입니다. 따라서 Informatica MDM Hub 일괄 작업을 실행하려면 먼저 소스 시스템이나 ETL 도구가 랜딩 테이블에 데이터를 써야 합니다. 랜딩 테이블은 일괄 로드를 위한 Informatica MDM Hub의 인터페이스입니다. 데이터를 랜딩 테이블로 전송하면 Informatica MDM Hub 일괄 프로시저가 데이터를 조작한 후 해당 위치에 복사합니다. 자세한 내용은 Informatica MDM Hub 개요에서 Informatica MDM Hub 데이터 관리 프로세스에 대한 설명을 참조하십시오.

일치 작업과 이후 통합 작업

일괄 작업은 특정 시퀀스로 실행해야 합니다. 예를 들어 일치 작업은 통합 프로세스를 실행하기 전에 기본 개체에 대해 실행해야 합니다. 병합 스타일 기본 개체의 경우 자동 일치 및 병합 작업을 실행할 수 있는데, 이 경우 일치 작업이 실행된 후 기본 개체의 모든 레코드에 대한 일치 확인이 끝나거나 수동 통합 제한의 최대 레코드 개수에 도달할 때까지 자동 병합 작업이 반복적으로 실행됩니다.

상위 테이블에서 먼저 데이터 로드

모든 상위 테이블(다른 테이블이 참조하는 테이블)이 먼저 로드되어야 한다는 것이 일반적인 규칙입니다.

외래 키 관계가 있는 개체의 데이터 로드

두 테이블 사이에 외래 키 관계가 있는 경우 참조되는 테이블을 먼저 로드한 다음 참조하는 테이블을 로드해야 합니다. Informatica MDM Hub에 존재할 수 있는 외래 키 관계는 한 기본 개체(외래 키가 있는 하위)에서 다른 기본 개체(기본 키가 있는 상위)로의 키 관계입니다.

대부분의 경우 이러한 작업을 정기적으로 실행하도록 예약합니다.

일괄 작업에 대한 모범 사례

일괄 작업을 구상하고 계획할 때는 다음 문제를 고려해야 합니다.

- 스키마를 정의하십시오.
스키마는 Informatica MDM Hub의 모든 태스크에 기초가 됩니다. 스키마가 없으면 일괄 작업은 아무 것도 수행하지 못합니다.
- Oracle 환경에서는 프로세스를 지나치게 많이 사용하지 않도록 Oracle의 통계 수집 수준을 지정하는 병렬도를 제한합니다. 통계를 수집하기 위한 프로세스가 지나치게 많으면 Hub 처리에 사용할 수 있는 리소스가 부족하게 될 수 있습니다. [“Oracle 환경에서의 통계 수집을 위한 병렬도 제한” 페이지 523](#)를 참조하십시오.
- 준비 작업을 실행하기 전에 매핑을 정의하십시오.
매핑은 준비 작업에서 수행되는 변환을 정의합니다. 매핑을 정의하지 않으면 준비 작업 시 준비 프로세스에서 변환이 수행되지 않습니다.
- 일치 작업을 실행하기 전에 일치 규칙을 정의하십시오.
일치 규칙이 없으면 일치 작업 시 일치 항목이 생성되지 않습니다.
- 프로덕션 작업을 실행하기 전에 다음을 수행하십시오.
 - 작은 데이터 집합을 사용하여 테스트를 실행하십시오.
 - 정리 엔진 및 다른 구성 요소의 테스트를 실행하여 각 구성 요소가 정상적으로 작동하는지 확인하십시오.
 - 각 구성 요소를 개별적으로 테스트한 후 통합된 시스템을 전체적으로 테스트하여 전체 시스템이 정상적으로 작동하는지 확인하십시오.

Oracle 환경에서의 통계 수집을 위한 병렬도 제한

Oracle 환경에서는 Hub 프로세스 성능이 저하되지 않도록 통계 수집을 위한 병렬도를 제한해야 합니다.

병렬도를 설정하려면 데이터베이스 관리자 권한이 있는 사용자로 로그인해야 합니다.

다음 단계를 수행하여 통계 수집에 적절한 병렬도를 할당하십시오.

1. 다음 수식을 사용하여 적절한 병렬도를 계산합니다.

$$\text{APPROPRIATE PARALLEL DEGREE} = \text{CPU_COUNT} * \text{PARALLEL_THREADS_PER_CPU}$$

CPU_COUNT는 Oracle에서 사용할 수 있는 CPU의 개수입니다. PARALLEL_THREADS_PER_CPU는 일반적으로 2입니다.

참고: 서버에 CPU가 여러 개 있을 경우 CPU 코어 수보다 작거나 같은 병렬도 값을 선택합니다.

참고: 다른 응용 프로그램이 Hub와 동일한 서버에서 실행 중인 경우 Hub에 할당할 수 있는 CPU 리소스 수를 결정한 다음 이 개수를 기반으로 적절한 병렬도를 설정합니다.

2. 다음 SQL*Plus 명령을 실행하여 현재 병렬도 설정을 확인합니다.

```
select DBMS_STATS.GET_PREFS( 'DEGREE' ) from dual;
```

3. 필요한 경우 다음 SQL*Plus 명령 중 하나를 실행하여 적절한 병렬도를 설정합니다.

- Oracle 10g: DBMS_STATS.SET_PARAM ('DEGREE', <적절한 병렬도 값>);
- Oracle 11g: DBMS_STATS.SET_GLOBAL_PREFS ('DEGREE', <적절한 병렬도 값>);

4. 대형 테이블의 경우 다음 SQL 명령을 실행하여 새 병렬도 값으로 성능을 테스트합니다.

```
DBMS_STATS.GATHER_TABLE_STATS
```

5. 대기 이벤트가 없어지고 성능이 용납할 만한 수준이 될 때까지 병렬도 값을 줄이면서 3단계와 4단계를 반복합니다.

일괄 작업 생성

일괄 작업은 다음 두 가지 방법 중 하나로 생성됩니다.

- Hub 저장소를 구성할 때 자동으로 생성
- 기본 개체에 대한 트러스트 설정을 변경하는 등 Informatica MDM Hub 구성에 특정 변경 내용이 발생할 때 생성

자동으로 생성되는 일괄 작업

Hub 저장소를 구성한 경우 MDM Hub가 다음 유형의 일괄 작업을 자동으로 작성합니다.

- 자동 일치 및 병합 작업
- 자동 연결 작업
- 자동 병합 작업
- BVT 스냅샷 작업
- 외부 일치 작업
- 일치 토큰 생성 작업
- 처음에 스마트 검색 데이터 인덱싱 작업
- 로드 작업
- 수동 연결 작업
- 수동 병합 작업
- 수동 연결 해제 작업
- 수동 병합 해제 작업
- 일치 작업
- 일치 분석 작업
- 승격 작업
- 준비 작업

변경이 발생할 때 생성되는 일괄 작업

일치 및 병합 설정을 변경하거나, 속성을 설정하거나, 초기 로드 후 트러스트 설정을 활성화할 때 MDM Hub에서 다음 일괄 작업을 작성합니다.

- 일치되지 않은 레코드를 고유 레코드로 허용
- 키 일치 작업
- 일치 테이블 재설정 작업
- 유효성 다시 검사 작업(열에 대한 유효성 검사를 활성화한 경우)
- 동기화 작업

정보 전용 일괄 작업(Hub 콘솔에서 실행되지 않음)

다음 일괄 작업은 정보 제공용으로만 사용되며 Hub 콘솔에서 수동으로 실행할 수 없습니다.

- 일치되지 않은 레코드를 고유 레코드로 허용
- BVT 스냅샷 작업
- 일괄 병합 해제 작업
- 수동 연결 작업
- 수동 병합 작업
- 수동 연결 해제 작업
- 수동 병합 해제 작업
- 링크 스타일을 병합 스타일로 마이그레이션 작업
- 다중 병합 작업
- 일치 테이블 재설정 작업

다른 일괄 작업

- Hub 삭제 작업

처리 서버 구성

처리 서버는 로드, BVT 다시 계산, 유효성 다시 검사, 삭제 및 일괄 병합 해제와 같은 일괄 작업을 수행합니다. 처리 서버는 응용 프로그램 서버 환경에 배포됩니다. 일괄 작업을 수행하려면 처리 서버를 구성해야 합니다.

각 연산 참조 저장소에 대해 여러 처리 서버를 구성할 수 있습니다. 여러 호스트에 처리 서버를 배포하여 여러 CPU에 걸쳐 처리 로드를 분산하고 일괄 작업을 병렬로 실행할 수 있습니다. 또한 처리 서버는 각 인스턴스에서 여러 요청을 동시에 처리할 수 있도록 다중 스레드입니다.

관련 항목:

- [“처리 서버 속성” 페이지 329](#)
- [“처리 서버 추가” 페이지 331](#)
- [“처리 서버 속성 편집” 페이지 331](#)

일괄 처리 뷰어 도구를 사용하여 일괄 작업 실행

이 섹션에서는 Hub 콘솔의 일괄 처리 뷰어 도구를 사용하여 일괄 작업을 개별적으로 실행하는 방법에 대해 설명합니다. 일괄 작업을 그룹으로 실행하려면 [“일괄 그룹 도구를 사용하여 일괄 작업 실행” 페이지 534](#)을 참조하십시오.

일괄 처리 뷰어 도구

일괄 처리 뷰어 도구에서는 일괄 작업을 개별적으로 실행하고 작업 실행 로그를 볼 수 있습니다. 일괄 처리 뷰어는 단일 작업의 실행을 시작하거나, 트러스트 설정이 변경된 후 실행되는 동기화 작업과 같이 자주 실행할 필요가 없는 작업을 실행하는 데 유용합니다. 작업 실행 로그에는 성공, 실패 또는 경고 등의 연결된 메시지와 함께 작업 완료 상태가 표시됩니다. 일괄 처리 뷰어 도구에는 작업 통계도 표시됩니다(해당되는 경우).

참고: 일괄 처리 뷰어에서는 자동화된 예약 기능은 제공하지 않습니다.

일괄 처리 뷰어 도구 시작

일괄 처리 뷰어 도구를 시작하려면

- ▶ Hub 콘솔에서 유틸리티 작업 영역을 확장한 후 **일괄 처리 뷰어**를 클릭합니다.

Hub 콘솔에 일괄 처리 뷰어 도구가 표시됩니다.

테이블, 데이터 또는 프로시저 유형별로 그룹화

탐색 트리의 아래쪽에서 **그룹 기준** 컨트롤을 마우스 오른쪽 단추로 클릭하여 탐색 트리의 최상위 보기를 변경할 수 있습니다.

참고: 확인 표시와 함께 회색으로 처리된 항목은 현재 선택된 항목을 나타냅니다.

다음 옵션 중 하나를 선택하십시오.

그룹 기준 옵션	설명
테이블	계층에서 다음과 같은 수준으로 항목을 표시합니다. <ul style="list-style-type: none"> - 최상위 수준: 테이블 - 두 번째 수준: 프로시저 유형 - 세 번째 수준: 일괄 작업 - 네 번째 수준: 날짜/타임스탬프
Date	계층에서 다음과 같은 수준으로 항목을 표시합니다. <ul style="list-style-type: none"> - 최상위 수준: 날짜/타임스탬프 - 두 번째 수준: 날짜/타임스탬프별 일괄 작업
프로시저 유형	계층에서 다음과 같은 수준으로 항목을 표시합니다. <ul style="list-style-type: none"> - 최상위 수준: 프로시저 유형 - 두 번째 수준: 일괄 작업 - 세 번째 수준: 날짜/타임스탬프

수동으로 일괄 작업 실행

일괄 작업을 수동으로 실행하려면

1. 실행할 일괄 작업을 선택합니다.
2. 일괄 작업을 실행합니다.

일괄 작업 선택

실행할 일괄 작업을 선택하려면

1. 일괄 뷰어 도구를 시작합니다.
일괄 처리 뷰어 트리에 일괄 작업 목록이 표시됩니다. 이 목록은 프로시저 유형별로 그룹화됩니다.
2. 트리를 확장하여 실행할 일괄 작업을 표시한 후 해당 작업을 클릭하여 선택합니다.
일괄 처리 뷰어에 속성 및 명령 단추와 함께 선택한 일괄 작업에 대한 화면이 표시됩니다.

일괄 작업 속성

일괄 처리 뷰어 도구의 속성 창에서 일괄 작업 속성을 볼 수 있습니다.

Identify 테이블에는 다음 속성이 표시됩니다.

이름

일괄 작업의 이름입니다.

설명

일괄 작업의 설명입니다.

Status 테이블에는 다음 속성이 표시됩니다.

현재 상태

일괄 작업의 현재 상태입니다. 일괄 작업의 상태는 다음 중 하나일 수 있습니다.

- 실행 중
- 불완전
- 완료됨
- 실행 중이 아님
- <Batch Job> 성공
- 일괄 작업 실패에 대한 설명

일괄 작업을 실행하기 전 설정할 옵션

특정 일괄 작업 유형에는 일괄 작업을 실행하기 전에 구성할 수 있는 추가 필드가 포함됩니다.

필드	용도	설명
모든 일치 토큰 다시 생성	일치 토큰 생성 작업	일치 토큰 생성 범위를 제어합니다. 전체 기본 개체를 토큰화하거나(선택됨) 기본 개체에서 재토큰화가 필요하다고 플래그가 지정된 레코드만 토큰화(선택 취소)합니다.
강제 업데이트	로드 작업	이 옵션을 선택한 경우 로드 작업 시 강제로 새로 고침이 수행되고 레코드가 이미 로드되었는지 여부에 상관없이 준비 테이블의 레코드가 기본 개체에 로드됩니다.
일치 집합	일치 작업	이 일치 작업에 사용할 일치 규칙 집합을 선택할 수 있습니다.

일괄 작업에 대한 명령 단추

일괄 작업을 선택한 후 다음 명령 단추를 클릭할 수 있습니다.

단추	설명
일괄 실행	선택한 일괄 작업을 실행합니다.
기록 지우기	일괄 처리 뷰어의 작업 실행 기록을 지웁니다.
불완전 상태로 설정	현재 실행 중인 일괄 작업의 상태를 불완전 상태로 설정합니다.
상태 새로 고침	현재 실행 중인 일괄 작업의 상태 표시를 새로 고칩니다.

일괄 작업 실행

참고: 일괄 작업을 실행하는 동안 응용 프로그램 서버가 실행 중이어야 합니다.

일괄 처리 뷰어에서 일괄 작업을 실행하려면

1. 일괄 처리 뷰어에서 실행할 일괄 작업을 선택합니다.
2. 오른쪽 패널에서 **일괄 실행**을 클릭하거나, 왼쪽 패널에서 일괄 작업을 마우스 오른쪽 단추로 클릭하고 팝업 메뉴에서 **실행**을 선택합니다.
작업의 현재 상태가 실행 중일 경우 **일괄 실행** 단추가 비활성화됩니다. 일괄 작업을 다시 실행하려면 작업이 완료될 때까지 기다려야 합니다.

상태 새로 고침

일괄 작업이 실행되는 동안 **상태 새로 고침**을 클릭하여 상태가 변경되었는지 여부를 확인할 수 있습니다.

작업 상태를 [불완전]으로 설정

드물지만 [불완전 상태로 설정]을 클릭하여 실행 중인 작업의 상태를 변경하고 작업을 다시 실행해야 할 수 있습니다. 이 작업은 서버 재부팅이나 충돌 등과 같은 오류로 인해 일괄 작업에서 실행을 중지했지만 Informatica MDM Hub가 메타데이터의 작업 응용 프로그램 잠금으로 인해 작업이 중지되었음을 감지하지 못한 경우에만 수행합니다. 현재 상태가 **실행 중**이지만 데이터베이스, 응용 프로그램 서버 및 로그가 어떤 활동도 보이지 않을 때 이 문제를 알아차릴 수 있습니다. 이러한 상황이 발생하면 일괄 작업을 다시 실행하도록 이 단추를 클릭하여 작업 응용 프로그램 잠금을 해제합니다. 그렇지 않으면 일괄 작업을 실행할 수 없습니다. [불완전 상태로 설정]은

일괄 작업 상태를 업데이트하며, 작업을 중단하지는 않습니다. 작업 상태를 [불완전]으로 설정했으면 연결된 데이터베이스 프로세스도 중지해야 합니다.






참고: 이 옵션은 사용자 ID에 Informatica 관리자 권한이 있는 경우에만 사용할 수 있습니다.

작업 실행 로그 보기

Informatica MDM Hub에서는 일괄 작업이 실행될 때마다 작업 실행 로그가 생성됩니다.

작업 실행 상태

각 작업 실행 로그 항목에는 다음 상태 값 중 하나가 있습니다.

아이콘	설명
	일괄 작업이 현재 실행 중입니다.
	일괄 작업이 성공적으로 완료되었습니다.
	일괄 작업이 성공적으로 완료되었지만 사용할 수 있는 추가 정보가 있습니다. 예를 들어 준비 및 로드 작업의 경우 일부 레코드가 거부된 것을 나타낼 수 있습니다. 일치 작업의 경우 기본 개체가 비어 있거나 일치시킬 레코드가 더 이상 없다는 것을 나타낼 수 있습니다.
	일괄 작업이 실패했습니다.
	일괄 작업 상태를 수동으로 "실행 중"에서 "불완전"으로 변경했습니다.

일괄 작업에 대한 작업 실행 로그 보기

일괄 작업에 대한 작업 실행 로그를 보려면

1. 일괄 뷰어 도구를 시작합니다.
2. 트리를 확장하여 보려는 작업 실행 로그를 표시한 다음 해당 로그를 클릭합니다.
일괄 처리 뷰어에 선택한 작업 실행 로그의 화면이 표시됩니다.

작업 실행 로그 항목 속성

일괄 처리 뷰어에는 각 작업 실행 로그 항목에 대해 다음과 같은 정보가 표시됩니다.

필드	설명
ID	이 일괄 작업의 ID 정보입니다. C_REPOS_TABLE_OBJECT_V 테이블에 저장됩니다.
이름	이 작업 실행 로그의 이름입니다. 일괄 작업이 시작된 날짜/시간입니다.
설명	해당 일괄 작업에 대한 다음 형식의 설명입니다. <i>JobName for / from BaseObjectName</i> 예: - Load from Consumer_Credit_Stg - Match for Address

필드	설명
소스 시스템	다음 중 하나입니다. - 처리된 데이터의 소스 시스템 - 관리
소스 테이블	처리된 데이터의 소스 테이블
상태	해당 일괄 작업의 상태 정보입니다.
현재 상태	이 일괄 작업의 현재 상태. 오류가 발생한 경우 오류에 대한 정보가 표시됩니다.
메트릭스	이 일괄 작업의 메트릭스
[Various]	일괄 작업 실행 중에 수집된 통계(해당하는 경우): - 일괄 작업 메트릭스 - 자동 일치 및 병합 메트릭스 - 자동 병합 메트릭스 - 로드 작업 메트릭스 - 일치 작업 메트릭스 - 일치 분석 작업 메트릭스 - 준비 작업 메트릭스 - 승격 작업 메트릭스
시간	이 일괄 작업의 타임스탬프
시작	이 일괄 작업이 시작된 날짜/시간입니다.
종지	이 일괄 작업이 종료된 날짜/시간입니다.
경과된 시간	이 일괄 작업의 실행에 경과된 시간

일괄 작업 메트릭스 정보

Informatica MDM Hub는 일괄 작업을 실행하는 동안 다양한 통계를 수집합니다. 반환되는 실제 메트릭스는 관련 일괄 작업에 따라 달라집니다. 일괄 작업이 완료되면 C_REPOS_JOB_METRIC_TYPE에 해당 통계가 등록됩니다. 각 작업마다 여러 개의 통계가 있을 수 있습니다. 가능한 작업 메트릭스는 다음과 같습니다.

메트릭스 이름	설명
총 레코드 수	일괄 작업에 의해 처리된 레코드의 총 수입입니다.
삽입됨	일괄 작업에 의해 대상 개체에 삽입된 레코드의 수입입니다.
업데이트됨	일괄 작업에 의해 대상 개체에서 업데이트된 레코드의 수입입니다.
작업 없음	작업이 수행되지 않은 레코드(기본 개체에 이미 있는 레코드)의 수입입니다.
일치된 레코드	일괄 작업에 의해 일치된 레코드의 수입입니다.
평균 일치 항목	평균 일치 항목의 수입입니다.

메트릭스 이름	설명
업데이트된 XREF	기본 개체에 대한 교차 참조 테이블을 업데이트한 레코드의 수입입니다. 증분 로드 중에 레코드를 로드하면 레코드가 미리 통합됩니다. 레코드는 교차 참조 테이블에만 유지되고 기본 개체에는 유지되지 않습니다.
토큰화된 레코드	일괄 작업에 의해 토큰화된 레코드의 수입입니다. 스키마 도구에서 로드 시 일치 토큰 생성 확인란을 선택한 경우에만 적용됩니다.
일치 플래그가 지정된 레코드	일치 플래그가 지정된 레코드의 수입입니다.
자동 병합된 레코드	자동 병합 일괄 작업에 의해 병합된 레코드의 수입입니다.
거부된 레코드	일괄 작업에 의해 거부된 레코드의 수입입니다.
병합 해제된 소스 레코드 수	일괄 작업에 의해 병합되지 않은 소스 레코드의 수입입니다.
병합 제공자 XREF 레코드	
고유 레코드로 허용됨	일괄 작업에 의해 고유 레코드로 허용된 레코드의 수입입니다. 일치/병합 설정 구성에서 기본 개체에 대해 일치되지 않은 모든 행을 고유 행으로 허용을 활성화한 경우에만 적용됩니다.
자동 병합을 위해 대기됨	자동 일치 및 병합 작업에 의해 실행된 일치 작업 중 자동 병합을 위해 대기된 레코드의 수입입니다.
수동 병합을 위해 대기됨	수동 병합을 위해 대기된 레코드의 수입입니다. 이러한 레코드를 처리하려면 Hub 콘솔에서 병합 관리자를 사용합니다. 자세한 내용은 <i>Multidomain MDM 데이터 스튜어드 가이드</i> 를 참조하십시오.
백필 트러스트 레코드	
조회 누락/올바르지 않은 rowid_object 레코드	조회 정보가 누락되었거나 rowid_object 레코드가 잘못된 소스 레코드의 수입입니다.
보류 상태로 전환된 레코드	보류 상태로 전환된 레코드의 수입입니다.
일치하는 것으로 분석된 레코드	일치될 레코드의 수입입니다.
필요한 일치 비교	일치 비교의 수입입니다.
정리된 총 레코드 수	정리된 레코드의 수입입니다.
총 랜딩 레코드 수	랜딩 테이블에 배치된 레코드의 수입입니다.
올바르지 않게 제공된 rowid_object 레코드	rowid_object가 잘못된 레코드의 수입입니다.
자동 연결된 레코드	자동 연결된 레코드의 수입입니다.
BVT 스냅샷	BVT(최선의 진실, Best Version of the Truth)의 스냅샷입니다.
중복 일치 레코드	중복 일치된 레코드의 수입입니다.
제거된 링크	제거된 링크의 수입입니다.

메트릭스 이름	설명
유효성이 다시 검사된 레코드	유효성이 다시 검사된 레코드의 수입입니다.
새 상태로 재설정된 기본 개체 레코드	"새로운 항목" 상태로 재설정된 기본 개체 레코드의 수입입니다.
일치 링크로 변환된 링크	일치 링크로 변환된 링크의 수입입니다.
자동 승격된 레코드	자동 승격된 레코드의 수입입니다.
삭제된 XREF 레코드	삭제된 교차 참조 레코드의 수입입니다.
삭제된 레코드	삭제된 레코드의 수입입니다.
올바르지 않은 레코드	올바르지 않은 레코드의 수입입니다.
승격되지 않은 활성 레코드	승격되지 않은 활성 레코드의 수입입니다.
승격되지 않은 보호된 레코드	승격되지 않은 보호된 레코드의 수입입니다.
삭제된 기본 개체 레코드	삭제된 기본 개체 레코드의 수입입니다.
삽입된 링크 레코드	
연결되지 않은 링크 레코드	
병합된 링크 레코드	
생성된 그룹	
병합된 그룹	
처리된 일치 레코드	
처리된 링크 관리 레코드	
거부된 링크 관리 레코드	
처리하지 못한 레코드	처리되지 않은 레코드의 수입입니다.
최적 잠금 유효성 검사에 실패한 레코드	일괄 작업을 실행할 때 다른 프로세스에 의해 수정된 레코드의 수입입니다.
다시 인덱싱된 검색 데이터	인덱싱된 검색 데이터에 포함된 레코드의 수입입니다.
검색 데이터에서 제거됨	검색 데이터에서 삭제된 레코드의 수입입니다.
검색 데이터에서 삭제/검색 데이터에 추가 실패	검색 데이터에서 삭제되지 않은 레코드의 수 또는 검색 데이터에 추가되지 않은 레코드의 수입입니다.
인덱싱된 총 BO 행 수	인덱싱된 기본 개체 행의 전체 수입입니다.

거부된 레코드 보기

준비 작업의 경우 일괄 작업의 결과로 레코드가 거부 테이블에 기록되면 작업 실행 로그에 거부 항목 보기 단추가 표시됩니다.

참고: HUB_STATE_IND 값이 올바르지 않으면 레코드가 거부됩니다.

거부된 레코드와 각 레코드가 거부된 이유를 보려면

1. **거부 항목 보기** 단추를 클릭합니다.
일괄 처리 뷰어에 거부된 레코드가 테이블로 표시됩니다.
2. **닫기**를 클릭합니다.


실패한 일괄 작업 실행 처리

일괄 작업 실행이 실패했을 경우 다음 단계를 수행하십시오.

- 이 일괄 작업에 대한 실행 로그 항목을 표시합니다.
- 현재 상태 필드의 오류 텍스트를 읽어 진단 정보를 확인합니다.
- 필요한 경우 수정 작업을 수행합니다.

현재 상태를 Windows 클립보드에 복사

일괄 처리의 현재 상태를 문서, 전자 메일 등에 붙여 넣기 위해 Windows 클립보드에 복사하려면

- ▶  단추를 클릭합니다.

작업 실행 로그 항목 삭제

선택한 작업 실행 로그를 삭제하려면

- ▶ 작업 속성 페이지의 오른쪽 위에서 **삭제** 단추를 클릭합니다.

작업 실행 기록 지우기

일괄 작업을 실행한 후에 생성되는 실행된 작업 목록은 시간이 지나면서 매우 커질 수 있습니다. 정기적으로 이 목록에서 불필요한 작업 실행 로그를 제거해야 합니다.

참고: 작업 기록을 지우는 실제 절차의 단계는 보기(테이블별, 날짜별 또는 프로시저 유형별)에 따라 조금씩 달라집니다. 다음 절차에서는 테이블별 보기를 사용한다고 가정합니다.

작업 기록을 지우려면

1. 일괄 뷰어 도구를 시작합니다.
2. 일괄 뷰어에서 기본 개체 바로 아래의 트리를 확장합니다.
3. 일괄 작업 유형 아래의 트리를 확장합니다.
4. 기록을 지울 작업을 선택합니다.
5. **기록 지우기**를 클릭합니다.
6. **예**를 클릭하여 이 일괄 작업에 대한 모든 실행 기록을 삭제할 것을 확인합니다.

일괄 그룹 도구를 사용하여 일괄 작업 실행

이 섹션에서는 Hub 콘솔의 일괄 그룹 도구를 사용하여 그룹으로 일괄 작업을 실행하는 방법에 대해 설명합니다. 일괄 작업을 개별적으로 실행하려면 [“일괄 처리 뷰어 도구를 사용하여 일괄 작업 실행” 페이지 526](#)을 참조하십시오.

일괄 그룹 정보

일괄 그룹은 개별 일괄 작업(예: 준비, 로드 및 일치 작업)의 컬렉션으로, 단일 명령으로 실행할 수 있습니다. 일괄 그룹의 각 일괄 작업은 순차적으로 실행되거나 다른 작업과 병렬로 실행될 수 있습니다. 일괄 그룹 도구를 사용하여 일괄 그룹을 구성하고 실행할 수 있습니다.

일괄 그룹 도구에서 사용할 수 있는 사용자 지정 일괄 작업 및 일괄 그룹을 개발하는 방법에 대한 자세한 내용은 [“일괄 작업 참조” 페이지 543](#)를 참조하십시오.

참고: Hub 콘솔에서 매핑 등의 개체를 삭제하면 해당 개체에 종속된 준비 작업 등의 일괄 작업이 일괄 그룹 도구에 빨간색으로 강조 표시됩니다. 일괄 그룹을 다시 실행하기 전에 이 문제를 해결해야 합니다.

순차 실행 및 병렬 실행

다음과 같은 방법으로 일괄 작업을 실행할 수 있습니다.

실행 방법	설명
순차	일괄 그룹에 있는 일괄 작업을 한 번에 하나만 실행합니다.
병렬	일괄 그룹에 있는 여러 일괄 작업을 동시에 실행합니다.

실행 경로

실행 경로는 전체 일괄 그룹이 실행될 때 일괄 작업이 실행되는 시퀀스입니다.

실행 경로는 시작 노드에서 시작되어 종료 노드에서 끝납니다. 일괄 그룹 도구는 실행 시퀀스의 유효성을 검사하지 않으므로, 실행 시퀀스가 올바른지 확인하는 것은 사용자의 책임입니다. 예를 들어 준비 작업 전에 기본 개체에 대한 로드 작업을 잘못 지정했을 경우 일괄 그룹 도구가 어떠한 오류도 알리지 않습니다.

수준

일괄 그룹에서 실행 경로는 시퀀스로 실행되는 하나 이상의 수준으로 구성됩니다.

수준은 하나 이상의 일괄 작업 컬렉션입니다.

- 한 수준에 여러 개의 일괄 작업이 포함되는 경우 각 일괄 작업은 병렬로 실행됩니다.
- 한 수준에 하나의 일괄 작업만 포함되는 경우 이 일괄 작업은 개별적으로 실행됩니다.

일괄 그룹이 시퀀스의 다음 태스크로 진행하려면 해당 수준의 모든 일괄 작업이 완료되어야 합니다.

참고: 특정 수준의 모든 일괄 작업이 병렬로 실행되기 때문에 동일한 수준의 일괄 작업 사이에는 종속성이 없습니다. 예를 들어 기본 개체의 준비 작업과 로드 작업은 적절한 시퀀스로 실행되는 서로 다른 수준에 있게 됩니다.

일괄 그룹 도구 시작

일괄 그룹 도구를 시작하려면

- ▶ Hub 콘솔에서 유틸리티 작업 영역을 확장한 다음 **일괄 그룹**을 클릭합니다.

Hub 콘솔에 일괄 그룹 도구가 표시됩니다.

일괄 그룹 도구는 다음과 같은 영역으로 구성됩니다.

영역	설명
탐색 트리	일괄 그룹 및 실행 로그의 계층 목록입니다.
속성 창	속성 및 명령

일괄 그룹 구성

이 섹션에서는 일괄 그룹을 추가, 편집 및 삭제하는 방법에 대해 설명합니다.

일괄 그룹 추가

일괄 그룹을 추가하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 일괄 그룹 트리의 일괄 그룹 노드를 마우스 오른쪽 단추로 클릭하고 팝업 메뉴에서 **일괄 그룹 추가**를 선택합니다.

일괄 그룹 도구의 일괄 그룹 트리에 "새 일괄 그룹"이 추가됩니다.

실행 시퀀스는 비어 있습니다. 실행 시퀀스는 새 일괄 그룹을 추가한 후에 구성합니다.

4. 다음 정보를 지정합니다.

필드	설명
이름	이 일괄 그룹의 고유한 설명 이름을 지정합니다.
설명	이 일괄 그룹에 대한 설명을 입력합니다.

5. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

변경 내용이 저장되고 탐색 트리가 업데이트됩니다.

새 일괄 그룹에 일괄 작업을 추가하려면 [“일괄 그룹 수준에 일괄 작업 할당” 페이지 537](#)을 참조하십시오.

일괄 그룹 속성 편집

일괄 그룹 속성을 편집하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일괄 그룹 노드를 확장하여 편집할 일괄 그룹을 표시합니다.
4. 원하는 경우 다른 일괄 그룹 이름을 지정합니다.
5. 원하는 경우 다른 설명을 지정합니다.

6. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

일괄 그룹 삭제

일괄 그룹을 삭제하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일괄 그룹 노드를 확장하여 삭제할 일괄 그룹을 표시합니다.
4. 삭제할 일괄 그룹을 마우스 오른쪽 단추로 클릭하고 **일괄 그룹 삭제**를 클릭합니다.
일괄 그룹 도구에서 삭제를 확인하는 메시지를 표시합니다.
5. **예**를 클릭합니다.
삭제하는 일괄 그룹이 탐색 트리에서 제거됩니다.

일괄 그룹에 대한 수준 구성

일괄 그룹에는 순서대로 실행되는 하나 이상의 수준이 포함되어 있습니다. 이 섹션에서는 일괄 그룹의 수준을 구성하여 실행 순서를 지정하는 방법에 대해 설명합니다.

일괄 그룹에 수준 추가

일괄 그룹에 수준을 추가하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일괄 그룹 노드를 확장하여 구성할 일괄 그룹을 표시합니다.
4. 일괄 그룹 트리에서 수준을 마우스 오른쪽 단추로 클릭하고 다음 옵션 중 하나를 선택합니다.

명령	설명
상위 수준 추가	이 일괄 그룹의 선택한 항목 위에 수준을 추가합니다.
하위 수준 추가	이 일괄 그룹의 선택한 항목 아래에 수준을 추가합니다.
수준 위로 이동	이 일괄 그룹 수준을 이전 수준 위로 이동합니다.
수준 아래로 이동	이 일괄 그룹 수준을 다음 수준 아래로 이동합니다.
이 수준 제거	이 일괄 그룹 수준을 제거합니다.

일괄 그룹 도구에 "일괄 그룹에 추가할 작업 선택" 대화 상자가 표시됩니다.

5. 추가할 작업의 기본 개체를 확장합니다.
6. 추가할 작업을 선택합니다.
병렬로 실행할 작업을 선택하려면 **Ctrl** 키를 누른 상태로 선택할 각 작업을 클릭합니다.
7. **확인**을 클릭합니다. 일괄 그룹 도구의 일괄 그룹에 선택한 작업이 추가됩니다.
8. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

일괄 그룹에서 수준 제거

일괄 그룹에서 수준을 제거하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일괄 그룹 노드를 확장하여 구성할 일괄 그룹을 표시합니다.
4. 일괄 그룹에서 삭제할 수준을 마우스 오른쪽 단추로 클릭하고 **이 수준 제거**를 선택합니다.
Hub 콘솔에 삭제 확인 대화 상자가 표시됩니다.
5. **예**를 클릭합니다.
일괄 그룹 도구가 일괄 그룹에서 삭제된 수준을 제거합니다.

일괄 그룹 내에서 수준을 위로 이동하려면

일괄 그룹 내에서 수준을 위로 이동하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일괄 그룹 노드를 확장하여 구성할 일괄 그룹을 표시합니다.
4. 일괄 그룹 트리에서 위로 이동할 수준을 마우스 오른쪽 단추로 클릭하고 **수준 위로 이동**을 선택합니다.
일괄 그룹 도구의 일괄 그룹 내에서 해당 수준이 위로 이동합니다.

일괄 그룹 내에서 수준을 아래로 이동하려면

일괄 그룹 내에서 수준을 아래로 이동하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일괄 그룹 노드를 확장하여 구성할 일괄 그룹을 표시합니다.
4. 일괄 그룹 트리에서 아래로 이동할 수준을 마우스 오른쪽 단추로 클릭하고 **수준 아래로 이동**을 선택합니다.
일괄 그룹 도구의 일괄 그룹 내에서 해당 수준이 아래로 이동합니다.

일괄 그룹 수준에 일괄 작업 할당

일괄 그룹 도구에서 작업이란 Informatica MDM Hub 일괄 작업을 나타냅니다. 각 수준에는 하나 이상의 일괄 작업이 포함되어 있습니다. 한 수준에 여러 일괄 작업이 포함된 경우 이러한 모든 일괄 작업은 병렬로 실행됩니다.

일괄 그룹 수준에 일괄 작업 추가

일괄 그룹에 일괄 작업을 추가하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일괄 그룹 노드를 확장하여 구성할 일괄 그룹을 표시합니다.
4. 일괄 그룹 트리에서 작업을 추가할 수준을 마우스 오른쪽 단추로 클릭하고 **이 수준에 작업 추가...**를 선택합니다.
일괄 그룹 도구에 "일괄 그룹에 추가할 작업 선택" 대화 상자가 표시됩니다.
5. 추가할 작업의 기본 개체를 확장합니다.

6. 추가할 작업을 선택합니다.
7. 한 번에 여러 작업을 선택하려면(여러 작업을 병렬로 실행하려면) **Ctrl** 키를 누른 상태로 작업을 클릭합니다.
8. **확인**을 클릭합니다.
9. 변경 내용을 저장합니다.
 일괄 그룹 도구의 대상 수준 상자에 선택한 작업이 추가됩니다. Informatica MDM Hub에서는 그룹 수준의 모든 일괄 작업을 병렬로 실행합니다.

일괄 작업에 대한 옵션 구성

일괄 그룹을 구성할 경우 특정 종류의 일괄 작업에 대한 작업 옵션을 구성할 수 있습니다. 이러한 작업 옵션에 대한 자세한 내용은 [“일괄 작업을 실행하기 전 설정할 옵션” 페이지 528](#)을 참조하십시오.

수준에서 일괄 작업 제거

수준에서 일괄 작업을 제거하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일괄 그룹 노드를 확장하여 구성할 일괄 그룹을 표시합니다.
4. 일괄 그룹에서 삭제할 작업을 마우스 오른쪽 단추로 클릭하고 **작업 제거**를 선택합니다.
 일괄 그룹 도구에 삭제 확인 대화 상자가 표시됩니다.
5. **예**를 클릭하여 선택한 작업을 삭제합니다.
 일괄 그룹 도구가 일괄 그룹의 이 수준에서 삭제된 작업을 제거합니다.

일괄 작업을 한 수준 위로 이동하려면

일괄 작업을 한 수준 위로 이동하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일괄 그룹 노드를 확장하여 구성할 일괄 그룹을 표시합니다.
4. 일괄 그룹에서 위로 이동할 작업을 마우스 오른쪽 단추로 클릭하고 **작업 위로 이동**을 선택합니다.
 일괄 그룹 도구에서 선택한 작업이 일괄 그룹 내의 한 수준 위로 이동합니다.

일괄 작업을 한 수준 아래로 이동하려면

일괄 작업을 한 수준 아래로 이동하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일괄 그룹 노드를 확장하여 구성할 일괄 그룹을 표시합니다.
4. 일괄 그룹에서 아래로 이동할 작업을 마우스 오른쪽 단추로 클릭하고 **작업 아래로 이동**을 선택합니다.
 일괄 그룹 도구에서 선택한 작업이 일괄 그룹 내의 한 수준 아래로 이동합니다.

일괄 그룹 목록 새로 고침

일괄 그룹 목록을 새로 고치려면

- ▶ 탐색 창의 아무 위치를 마우스 오른쪽 단추로 클릭하고 **새로 고침**을 선택합니다.

일괄 그룹 도구를 사용하여 일괄 그룹 실행

이 섹션에서는 일괄 그룹 도구에서 일괄 그룹 실행을 관리하는 방법을 설명합니다.

참고: 일괄 그룹을 실행하는 동안 응용 프로그램 서버가 실행 중이어야 합니다.

참고: Hub 콘솔에서 매핑 등의 개체를 삭제하면 해당 개체에 종속된 준비 작업 등의 일괄 작업이 일괄 그룹 도구에 빨간색으로 강조 표시됩니다. 일괄 그룹을 다시 실행하기 전에 이 문제를 해결해야 합니다.

컨트롤 및 로그 화면으로 이동

컨트롤 및 로그 화면은 일괄 그룹 실행을 제어하고 해당 실행 로그를 볼 수 있는 화면입니다.

일괄 그룹에 대한 컨트롤 및 로그 화면으로 이동하려면

1. 일괄 그룹 도구를 시작합니다.
2. 일괄 그룹 트리를 확장하여 실행할 일괄 그룹을 표시합니다.
3. 일괄 그룹을 확장한 후 **컨트롤 및 로그** 노드를 클릭합니다.
일괄 그룹 도구에 이 일괄 그룹에 대한 컨트롤 및 로그 화면이 표시됩니다.

컨트롤 및 로그 화면의 구성 요소

이 화면에는 다음 구성 요소가 포함되어 있습니다.

구성 요소	설명
도구 모음	일괄 그룹 실행을 관리하기 위한 명령 단추입니다.
일괄 그룹에 대한 로그	해당 일괄 그룹에 대한 실행 로그입니다.
일괄 작업에 대한 로그	해당 일괄 그룹의 개별 일괄 작업에 대한 실행 로그입니다.

일괄 그룹에 대한 명령 단추

다음 명령 단추를 사용하여 일괄 그룹 실행을 관리할 수 있습니다.

단추	설명
실행	해당 일괄 그룹을 실행합니다.
다시 시작 상태로 설정	실패한 일괄 그룹의 실행 상태를 다시 시작 상태로 설정합니다.
불완전 상태로 설정	실행 중인 일괄 그룹의 실행 상태를 불완전 상태로 설정합니다.
선택한 항목 지우기	선택한 그룹 또는 작업 실행 로그를 제거합니다.

단추	설명
모두 지우기	모든 그룹 및 작업 실행 로그를 제거합니다.
새로 고침	해당 일괄 그룹의 화면 표시를 새로 고칩니다.

일괄 그룹 실행

일괄 그룹을 실행하려면

1. 일괄 그룹의 컨트롤 및 로그 화면으로 이동합니다.
2. 노드를 클릭한 다음 **일괄 그룹 > 실행**을 선택하거나, **실행** 단추를 클릭합니다.
일괄 그룹 도구에서 일괄 그룹이 실행되고 로그 패널이 일괄 그룹 실행 상태로 업데이트됩니다.

3. **새로 고침** 단추를 클릭하여 실행 결과를 표시합니다.

일괄 그룹 도구에 진행률 정보가 표시됩니다.







완료되면 일괄 그룹 도구에서 다음 로그에 항목을 추가합니다.

- 이 일괄 그룹의 그룹 실행 로그
- 개별 일괄 작업의 작업 실행 로그

참고: FAILED 상태인 일괄 그룹을 실행하는 경우 실제로는 실패한 인스턴스를 다시 실행하는 것이며, 상태는 최종 결과 상태로 설정되고 Hub에서 새 그룹 로그를 생성하지 않습니다. 하지만 세부 로그(아래쪽 로그 테이블)에서는 실패한 인스턴스를 다시 실행하는 것이 아니라 새 인스턴스에서 동일한 작업을 실행하게 되므로 Hub에서 여기에 표시된 새 로그를 생성합니다.

그룹 실행 상태

각 실행 로그에는 다음 상태 값 중 하나가 있습니다.

아이콘	설명
	처리 중입니다. 일괄 그룹이 현재 실행 중입니다.
	일괄 그룹 실행이 성공적으로 완료되었습니다.
	일괄 그룹 실행이 추가 정보와 함께 완료되었습니다. 예를 들어 준비 및 로드 작업의 경우 이 아이콘은 일부 레코드가 거부되었음을 나타낼 수 있습니다. 일치 작업의 경우 기본 개체가 비어 있거나 일치시킬 레코드가 더 이상 없다는 것을 나타낼 수 있습니다.
	일괄 그룹 실행이 실패했습니다.
	일괄 그룹 실행이 불안정합니다.
	일괄 그룹 실행이 다시 시작되도록 재설정되었습니다.

일괄 그룹에 대한 그룹 실행 로그 보기

일괄 그룹 도구는 일괄 그룹을 실행할 때마다 그룹 실행 로그 항목을 생성합니다.

경고: 일괄 그룹을 실행하는 동안 데이터베이스 연결 문제로 인해 작업이 실패하면 실패한 작업은 작업 제어 테이블에 나타나지 않습니다. 실패는 cmxserver 로그에 표시됩니다.

다음 테이블에는 각 로그 항목의 속성이 설명되어 있습니다.

필드	설명
상태	이 일괄 작업의 현재 상태. 일괄 그룹 실행에 실패한 경우 문제에 대한 설명이 표시됩니다.
시작	이 일괄 작업이 시작된 날짜/시간입니다.
종료	이 일괄 작업이 종료된 날짜/시간입니다.
메시지	일괄 그룹 실행과 관련된 메시지입니다.

일괄 작업에 대한 작업 실행 로그 보기

일괄 그룹 도구는 일괄 그룹 내의 일괄 작업을 실행할 때마다 작업 실행 로그 항목을 생성합니다.

각 로그 항목에는 다음과 같은 속성이 있습니다.

필드	설명
작업 이름	해당 일괄 작업의 이름입니다.
상태	이 일괄 작업의 현재 상태.
시작	이 일괄 작업이 시작된 날짜/시간입니다.
끝	이 일괄 작업이 종료된 날짜/시간입니다.
메시지	일괄 그룹 실행과 관련된 메시지입니다.

참고: 완료된 일괄 작업에 대한 메트릭스를 보려면 일괄 처리 뷰어를 사용합니다.

실행이 실패한 일괄 그룹 다시 시작

일괄 그룹 실행이 실패한 경우 실패를 야기할 수 있는 문제를 모두 해결한 후 일괄 그룹을 처음부터 다시 시작할 수 있습니다.

일괄 그룹을 다시 실행하려면

1. [내 일괄 그룹에 대한 로그] 목록에서 실패한 일괄 그룹에 대한 실행 로그 항목을 선택합니다.
2. **다시 시작 상태로 설정**을 클릭합니다.

일괄 그룹 도구가 이 일괄 작업의 상태를 [다시 시작]으로 바꿉니다.

3. 실패를 야기할 수 있는 모든 문제를 해결한 후 일괄 그룹을 다시 실행합니다.

일괄 그룹 도구가 일괄 그룹을 실행하고 새 실행 로그 항목을 생성합니다.

참고: 일괄 그룹이 실패하고 내 일괄 그룹 로그 목록에서 **다시 시작 상태로 설정** 단추 또는 **불완전 상태로 설정** 단추를 클릭하지 않으면 Informatica MDM Hub가 이전에 실패한 수준에서 일괄 작업을 다시 시작합니다.

불완전한 일괄 그룹 실행 처리

아주 드문 상황에서 실행 중인 일괄 그룹의 상태를 변경할 수도 있습니다.

- 일괄 그룹 상태가 아직 실행 중이라고 표시될 경우 불완전 상태로 설정을 클릭하고 일괄 그룹을 다시 실행할 수 있습니다. 서버 재부팅 또는 충돌 등의 오류로 인해 일괄 그룹이 실행을 중지했지만 Informatica MDM Hub가 메타데이터의 작업 응용 프로그램 잠금으로 인해 일괄 그룹이 중지되었음을 감지하지 못한 경우에만 이 작업을 수행합니다.

현재 상태가 실행 중이지만 데이터베이스, 응용 프로그램 서버 및 로그가 어떤 활동도 보이지 않을 때 이 문제를 알아차릴 수 있습니다. 이 문제가 발생하면 일괄 그룹을 다시 실행할 수 있도록 이 단추를 클릭하여 작업 응용 프로그램 잠금을 지워야 합니다. 그렇지 않으면 작업 그룹을 실행할 수 없습니다. 상태를 불완전 상태로 설정하면 일괄 그룹의 상태가 바로 업데이트됩니다. 일괄 그룹 내의 모든 일괄 작업도 마찬가지로 업데이트됩니다. 하지만 처리가 종료되지는 않습니다.

작업 상태가 불완전 상태일 경우에는 작업 상태를 다시 시작 상태로 설정할 수 없습니다.

- 작업 상태가 실패일 경우에는 다시 시작 상태로 설정을 클릭할 수 있습니다. 작업 상태가 다시 시작 상태일 경우에는 작업 상태를 불완전 상태로 설정할 수 없습니다.

상태를 변경하면 일괄 그룹이 완료되는 동안 계속해서 다른 작업을 수행할 수 있습니다.

실행 중인 일괄 그룹의 상태를 불완전 상태로 설정하려면

- 내 일괄 그룹 로그 목록에서 불완전으로 표시할 실행 중인 일괄 그룹의 실행 로그 항목을 선택합니다.
- 불완전 상태로 설정**을 클릭합니다.
일괄 작업의 상태가 불완전으로 변경됩니다.
- 일괄 그룹을 다시 실행합니다.

참고: 일괄 그룹이 실패하고 내 일괄 그룹 로그 목록에서 다시 시작 상태로 설정 단추 또는 불완전 상태로 설정 단추를 클릭하지 않으면 Informatica MDM Hub가 이전에 실패한 수준에서 일괄 작업을 다시 시작합니다.

거부된 레코드 보기

준비 작업 또는 로드 작업을 실행하는 중 일괄 그룹을 실행한 결과로 레코드가 거부 항목 테이블에 기록되면 작업 실행 로그에서 거부 항목 보기 단추가 활성화됩니다.

거부된 레코드를 보려면

- 거부 항목 보기** 단추를 클릭합니다.
일괄 그룹 도구에 거부 항목 창이 표시됩니다.
- 거부된 레코드를 필요한 대로 탐색하고 검사합니다.
- 닫기**를 클릭합니다.

상태별로 실행 로그 필터링

상태별 로그 노드 아래에서 해당 노드를 클릭하여 실행 상태를 기준으로 모든 일괄 그룹의 기록 로그를 볼 수 있습니다.

상태별로 실행 로그를 필터링하려면

- 일괄 그룹 도구를 시작합니다.
- 일괄 그룹 트리에서 상태별 로그 노드를 확장합니다.
일괄 그룹 도구에 로그 상태 목록이 표시됩니다.

3. 로그 패널 위쪽에서 검토할 특정 일괄 그룹 로그 항목을 클릭합니다.

Informatica MDM Hub 패널 아래쪽에 해당 일괄 그룹에 대한 자세한 작업 실행 로그가 표시됩니다.

참고: 배치 그룹 로그를 선택하고 **선택 항목 지우기** 단추를 클릭하여 일괄 그룹 로그를 삭제할 수 있습니다. 패널에 표시된 모든 로그를 삭제하려면 **모두 지우기** 단추를 클릭합니다.

일괄 그룹 삭제

일괄 그룹을 삭제하려면

1. 일괄 그룹 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 일괄 그룹 노드를 확장하여 삭제할 일괄 그룹을 표시합니다.
4. 일괄 그룹에서 제거할 작업을 마우스 오른쪽 단추로 클릭하고 **일괄 그룹 삭제**를 선택합니다(또는 **일괄 그룹 > 일괄 그룹 삭제** 선택).

일괄 작업 참조

이 섹션에서는 Informatica MDM Hub의 각 일괄 작업에 대해 설명합니다.

일괄 작업의 사전순 목록

일괄 작업	설명
일치되지 않은 레코드를 고유 레코드로 허용	일치 프로세스를 거쳤지만 일치하는 데이터가 없는 레코드의 경우 통합 표시기를 1(통합됨)로 설정합니다. 이 값은 레코드가 고유하므로 통합할 필요가 없음을 의미합니다.
자동 연결 작업	일치 프로세스 동안 자동으로 연결할 수 있는 것으로 확인되고 자동 연결로 플래그가 지정 (Automerge_ind=1)된 레코드를 자동으로 연결합니다.
자동 일치 및 병합 작업	일치하는 레코드가 더 이상 없거나 수동으로 통합할 수 있는 일치 항목 수가 구성된 임계값을 초과할 때까지 일치 작업과 이후 자동 병합 작업으로 구성되는 연속 주기를 실행합니다. 병합 스타일 기본 개체에서만 사용됩니다.
자동 병합 작업	일치 프로세스 동안 자동으로 병합할 수 있는 것으로 확인되고 자동 병합으로 플래그가 지정 (Automerge_ind=1)된 레코드를 자동으로 병합합니다. 병합 스타일 기본 개체에서만 사용됩니다.
외부 일치 작업	현재 일치 설정에 따라 결과를 생성하여 "외부에서 관리/준비되는" 레코드를 기존의 기본 개체와 일치시킵니다. 이때 기본 개체의 데이터가 실제로 수정되지는 않습니다.
일치 토큰 생성 작업	현재 일치 설정에 따라 일치 토큰을 생성하여 데이터를 일치시킬 준비를 합니다. 일치 토큰은 일치 후보를 식별하는 데 사용되는 열을 인코딩하는 문자열입니다.
Hub 삭제 작업	기본 개체/XREF 수준의 입력을 기반으로 하여 Hub에서 데이터를 삭제합니다.
처음에 스마트 검색 데이터 인덱싱 작업	비즈니스 항목 유형의 검색 가능한 필드의 모든 값에 대한 인덱스를 생성합니다. 이 인덱스는 검색 시 검색 가능한 필드에 포함된 데이터를 검색할 때 사용됩니다.

일괄 작업	설명
키 일치 작업	둘 이상의 소스가 동일한 기본 키를 사용하는 경우 이러한 소스의 레코드를 일치시킵니다. 새 레코드를 서로 비교하고 기존 레코드와 비교한 다음, 일치 규칙에 정의된 대로 소스 레코드 키의 비교 방법에 따라 잠재적 일치 항목을 식별합니다.
로드 작업	준비 테이블의 레코드를 Hub 저장소에서 해당하는 대상 기본 개체에 복사합니다. 로드 프로세스 중에는 레코드에 현재 트러스트 및 유효성 검사 규칙을 적용합니다.
수동 병합 작업	병합 관리자 도구에서 수동으로 병합된 레코드의 로그를 표시합니다. 병합 스타일 기본 개체에서만 사용됩니다.
수동 병합 해제 작업	병합 관리자 도구에서 수동으로 병합이 해제된 레코드에 대한 로그를 표시합니다.
일치 작업	현재 일치 규칙을 기반으로 하여 기본 개체에서 중복 레코드를 찾습니다.
일치 분석 작업	검색을 수행하여 일치 통계를 수집하되 실제로 일치 프로세스를 수행하지는 않습니다. 매우 큰 일치 요구 사항에 대해 잠재적 일치 항목이 있는 데이터 영역이 발견될 경우 Informatica MDM Hub에서는 레코드를 보류 상태로 전환하여 일치 프로세스를 진행하기 전에 데이터 스튜어드가 데이터를 수동으로 검토할 수 있도록 합니다.
중복 데이터에 대한 일치 작업	중복 레코드의 비율이 높은 데이터에 대해 새 레코드를 서로 간에, 또 기존 레코드와 비교한 다음 정확히 중복되는 항목을 식별합니다. 정확히 중복되는 항목의 최대 수는 이 기본 개체에 대한 중복 일치 임계값 설정에 따라 달라집니다.
다중 병합 작업	하나의 작업에서 여러 레코드를 병합할 수 있습니다.
승격 작업	XREF 테이블의 PROMOTE_IND 열을 읽고 이 열의 값이 1인 모든 행을 활성 상태로 변경합니다.
BO 다시 계산 작업	모든 기본 개체 또는 ROWID_OBJECT_TABLE 매개 변수로 지정한 기본 개체를 다시 계산합니다.
BVT 다시 계산 작업	지정된 ROWID_OBJECT에 대한 BVT를 다시 계산합니다.
일치 테이블 재설정 작업	일치된 모든 레코드가 일치를 위해 대기하도록 재설정된 작업의 로그를 표시합니다.
유효성 다시 검사 작업	로드 프로세스 중 초기 유효성 검사 이후 수정된 레코드에 대한 유효성 검사 논리/규칙을 실행합니다.
준비 작업	랜딩 테이블의 레코드를 준비 테이블에 복사합니다. 실행 중 현재 정리 설정에 따라 데이터를 정리합니다.
동기화 작업	기본 개체에 대한 메타데이터를 업데이트합니다. 기본 개체가 로드되었으나 아직 병합되기 전이고 해당 기본 개체의 열에 대해 트러스트를 활성화하는 등의 후속 트러스트 구성 변경 작업이 수행된 후에 사용됩니다. 이 작업은 이 기본 개체에 대한 데이터를 병합하기 전에 실행되어야 합니다.

일치되지 않은 레코드를 고유 레코드로 허용

"일치되지 않은 레코드를 고유 레코드로 허용" 일괄 작업에서는 일치 항목이 없는 레코드의 상태를 변경합니다. 이 일괄 작업에서는 일치 항목이 없는 레코드의 통합 표시기를 "1"로 설정합니다. 값 "1"은 MDM Hub에서 레코드를 통합할 필요가 없음을 나타냅니다. 수동 병합 작업에서도 대상 레코드에 일치 항목이 없으면 해당 레코드의 통합 표시기를 "1"로 설정합니다. 자동 병합 일괄 작업에서는 통합 표시기가 1인 레코드를 고유 레코드로 간주합니다.

MDM Hub에서는 일치되지 않은 모든 행을 고유 행으로 허용이 활성화되어 있는 경우 병합 일괄 작업 후 "일치되지 않은 레코드를 고유 레코드로 허용" 일괄 작업을 생성합니다.

참고: 일괄 처리 뷰어에서는 "일치되지 않은 레코드를 고유 레코드로 허용" 일괄 작업을 실행할 수 없습니다.

자동 일치 및 병합 작업

자동 일치 및 병합 일괄 작업은 일치시킬 레코드가 더 이상 없거나 수동 통합 제한의 최대 레코드 수에 도달할 때까지 일치 작업과 그 뒤의 자동 병합 작업으로 구성되는 연속 주기를 실행합니다.

일치 일괄 처리 크기 매개 변수는 이 프로세스에서 일치 및 병합 주기를 마칠 때까지의 주기당 레코드 수를 제어합니다.

중요: 테이블 간 또는 테이블 내 일치 경로의 레코드 간 관계를 정의하는 데 사용되는 기본 개체에는 자동 일치 및 병합 작업을 실행하지 마십시오. 그러면 관계 데이터가 변경되어 레코드 간의 연관이 손실될 수 있습니다.

응용 프로그램 서버 다시 시작 후 표시되는 두 번째 작업

자동 일치 및 병합 작업을 실행할 경우 이 작업은 상태에 하나의 작업이 표시된 상태에서 성공적으로 완료됩니다. 하지만 응용 프로그램 서버를 중지했다가 다시 시작한 후 일괄 처리 뷰어로 돌아온 경우 잠시 후 경고와 함께 두 번째 작업(일치 작업 아래에 표시됨)이 표시됩니다. 두 번째 작업은 기본 개체가 비어 있거나 일치시킬 더 이상의 레코드가 없음을 확인하기 위한 것입니다.

자동 일치 및 병합 매트릭스

자동 일치 및 병합 작업을 실행하고 나면 일괄 처리 뷰어의 작업 실행 로그에 다음 매트릭스(해당되는 경우)가 표시됩니다.

메트릭	설명
일치된 레코드	자동 일치 및 병합 작업에 의해 일치된 레코드의 수입입니다.
토큰화된 레코드	자동 일치 및 병합 작업 전에 토큰화된 레코드의 수입입니다.
자동 병합된 레코드	자동 일치 및 병합 작업에 의해 병합된 레코드의 수입입니다.
고유 레코드로 허용됨	자동 일치 및 병합 작업에 의해 고유 레코드로 허용된 레코드의 수입입니다. 일치/병합 설정 구성에서 기본 개체에 대해 "일치되지 않은 모든 행을 고유 행으로 허용"을 활성화("예"로 설정)한 경우에만 적용됩니다.
자동 병합을 위해 대기됨	자동 일치 및 병합 작업에 의해 실행된 일치 작업 중 자동 병합을 위해 대기된 레코드의 수입입니다.
수동 병합을 위해 대기됨	수동 병합을 위해 대기된 레코드의 수입입니다. 이러한 레코드를 처리하려면 Hub 콘솔에서 병합 관리자를 사용합니다. 자세한 내용은 <i>Multidomain MDM 데이터 스튜어드 가이드</i> 를 참조하십시오.

자동 병합 작업

병합 스타일 기본 개체의 경우에 한해, 일치 작업을 실행한 후 자동 병합 작업을 실행하여 일치 프로세스 중에 자동 병합 가능한 항목으로 확인된 레코드를 자동으로 병합할 수 있습니다. 자동 병합 작업을 실행하면 MATCH 테이블에서 자동 병합 플래그(Automerger_ind=1)가 지정된 모든 일치 항목이 처리됩니다.

참고: 상태 사용 개체의 경우, PENDING(소스 및 대상 레코드) 또는 DELETED 상태의 레코드는 자동으로 병합되지 않습니다. 레코드를 삭제하면 일치 테이블에서 해당 레코드가 제거되며 consolidation_ind는 4로 재설정됩니다.

자동 병합 작업과 자동 일치 및 병합

자동 일치 및 병합 일괄 작업은 일치시킬 레코드가 더 이상 없거나 수동 통합 제한의 최대 레코드 수에 도달할 때까지 일치 작업과 그 뒤의 자동 병합 작업으로 구성되는 연속 주기를 실행합니다.

자동 병합 작업 및 트러스트가 활성화된 열

트러스트가 활성화된 열이 다수인 경우에는 자동 병합 작업이 실패합니다. 작업이 실패하게 되는 정확한 열 수는 변수이며, 열 이름의 길이와 트러스트가 활성화된 열의 개수에 따라 달라집니다. 허용 가능한 최대 길이인 26자에 근접한 열 이름은 긴 이름으로 간주됩니다. 이 문제를 방지하려면 트러스트가 활성화된 열의 개수를 40개 미만으로 유지하거나 열 이름의 길이를 짧게 하십시오.

자동 병합 메트릭스

자동 병합 작업을 실행하고 나면 일괄 처리 뷰어의 작업 실행 로그에 다음 메트릭스(해당되는 경우)가 표시됩니다.

메트릭스	설명
자동 병합된 레코드	자동 병합 작업에 의해 자동 병합된 레코드의 수입니다.
고유 레코드로 허용됨	자동 병합 작업에 의해 고유 레코드로 허용된 레코드의 수입니다. 일치/병합 설정 구성에서 기본 개체에 대해 일치되지 않은 모든 행을 고유 행으로 허용 을 활성화(예로 설정)한 경우에만 적용됩니다.

일괄 병합 해제

이전 프로세스를 통해 병합되었던 레코드를 병합 해제할 수 있습니다. ExecuteBatchUnmerge SIF API를 사용하여 레코드를 일괄 병합 해제합니다.

다음 작업을 통해 통합되었던 레코드를 일괄 병합 해제할 수 있습니다.

- 자동 병합 일괄 작업
- 수동 병합
- 수동 레코드 편집
- ROWID_OBJECT별 로드
- 놓기 SIF API를 사용하여 교차 참조 레코드 삽입 또는 업데이트

ExecuteBatchUnmerge SIF API는 상위 레코드 또는 관련 하위 테이블과 관련된 병합 해제된 개체에 대한 일치 레코드를 제거하지 않습니다. 테이블 TEST_OUTPUT_TBL의 병합 해제된 레코드 목록을 사용하여 이러한 일치 레코드를 제거합니다. MDM Hub는 병합 해제된 레코드와 수동으로 추가된 레코드를 구분합니다. 기본적으로 MDM Hub는 수동으로 추가된 레코드에 0 값을 할당합니다. MDM Hub는 TEST_OUTPUT_TBL 테이블의 EXPLODE_NODE_IND 열에 병합 해제된 레코드에 대해 0을 제외한 값을 할당합니다.

또한 EXPLODE_NODE_IND 열을 입력 테이블에 추가할 수 있습니다. EXPLODE_NODE_IND를 1로 설정하면 MDM Hub에서 전체 기본 개체를 병합 해제하려고 합니다.

ExecuteBatchUnmerge SIF API에 대한 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

BVT 스냅샷 작업

기본 개체 테이블에서 BVT(최선의 진실, *Best Version of the Truth*)는 소스 레코드에서 제공된 최적의 데이터 셀과 통합된 레코드입니다.

참고: 상태 사용 기본 개체의 경우에 한해 BVT 논리는 HUB_STATE_IND를 사용하여 데이터를 제공하지 않는 기본 개체, 즉 HUB_STATE_IND가 -1 또는 0(PENDING 또는 DELETED 상태)인 기본 개체를 무시합니다. 온라인으로 BUILD_BVT를 호출할 경우에는 INCLUDE_PENDING_IND 매개 변수를 제공하십시오.

가능한 시나리오는 다음과 같습니다.

1. 이 매개 변수가 0인 경우 ACTIVE 기본 개체 레코드만 포함됩니다.
2. 이 매개 변수가 1인 경우 ACTIVE 및 PENDING인 기본 개체 레코드가 포함됩니다.
3. 이 매개 변수가 2인 경우 "what-if" 기능을 제공하기 위해 ACTIVE 및 PENDING인 XREF 레코드를 기준으로 계산이 수행됩니다.
4. 이 매개 변수가 3인 경우 XREF 기준의 현재 BVT를 제공하기 위해 ACTIVE 상태의 XREF 레코드를 기준으로 계산이 수행되며 이는 첫 번째 시나리오와 다를 수 있습니다.

외부 일치 작업

외부 일치 작업은 외부에서 준비한 레코드를 기본 개체와 비교하여 현재 일치 설정을 기반으로 한 결과를 반환합니다. 외부 일치 작업은 입력 테이블의 데이터를 기본 개체에 로드하거나 기본 개체 데이터에 어떤 식으로든 영향을 미치지 않습니다. 표준 일치 작업을 실행하기 전에 외부 일치를 사용하여 데이터를 미리 테스트하고, 일치 규칙을 테스트하고, 결과를 검사할 수 있습니다.

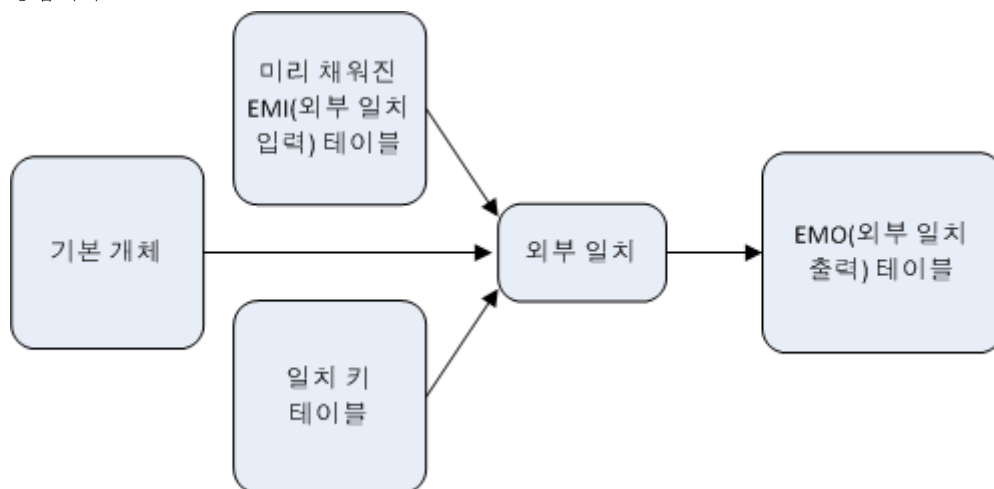
외부 일치 작업을 사용하여 유사 항목 일치 및 정확히 일치 규칙과 유사 항목 일치 및 정확히 일치 기본 개체를 둘 다 처리할 수 있습니다.

참고: 연결되는 실제 열이 들어 있는 정확히 일치 열의 경우 각 열 끝에 공백이 필요합니다. 예를 들어 "John"과 "Smith"가 연결된 경우 "John Smith" 일치만 반환됩니다.

외부 일치 작업은 일괄 작업으로 실행됩니다. 외부 응용 프로그램이 호출할 수 있는 해당하는 SIF 요청이 없습니다.

외부 일치 작업에 사용되는 입력 및 출력 테이블

기본 개체 및 관련 일치 키 테이블 외에도 외부 일치 작업에서는 다음 그래픽에 표시된 입력 및 출력 테이블을 사용합니다.



EMI(외부 일치 입력) 테이블

각 기본 개체에는 외부 일치 작업과 관련된 EMI(외부 일치 입력) 테이블이 있습니다. 이 테이블에서 사용하는 이름 지정 패턴은 다음과 같습니다.

*C_BaseObject*_EMI

여기서 *BaseObject*는 이 외부 일치 작업과 연결된 기본 개체의 이름입니다.

기본 개체를 생성하면 스키마 관리자가 관련 EMI 테이블을 생성하고 다음 시스템 열을 추가합니다.

열 이름	데이터 유형	크기	Null이 아님	설명
SOURCE_KEY	VARCHAR	50		레코드를 고유하게 식별하고 <i>C_BaseObject</i> _EMI 테이블의 레코드에 매핑하기 위해 3열 복합 기본 키의 일부로 사용됩니다.
SOURCE_NAME	VARCHAR	50		레코드를 고유하게 식별하고 <i>C_BaseObject</i> _EMI 테이블의 레코드에 매핑하기 위해 3열 복합 기본 키의 일부로 사용됩니다.
FILE_NAME	VARCHAR	50		레코드를 고유하게 식별하고 <i>C_BaseObject</i> _EMI 테이블의 레코드에 매핑하기 위해 3열 복합 기본 키의 일부로 사용됩니다.

EMI 테이블을 채울 때 시스템 열 중 하나 이상에 데이터가 포함되어야 합니다. 열 이름은 제한적이지 않습니다. 복합 3열 기본 키가 고유하기만 하면 어떤 식별 데이터도 포함될 수 있습니다.

또한 특정 열(예: *Person_Name* 또는 *Exact_Cust_ID*)에 대한 일치 규칙을 구성할 때 스키마 관리자가 해당 열을 *C_BaseObject*_EMI 테이블에 추가합니다.

참고: 스키마 관리자에서 외부 일치 테이블의 열을 보려면 **외부 일치 테이블** 노드를 확장하십시오.

EMI 테이블의 레코드는 일치 작업에 사용되는 일치 일괄 처리와 비슷합니다. 일치 일괄 처리에는 나머지 기본 개체 레코드와 일치하는 레코드 집합이 포함되어 있습니다. 일치 작업의 경우 일치 일괄 레코드가 기본 개체에 있지만 외부 일치의 경우 이러한 레코드가 별도의 입력 테이블에 있다는 점이 다릅니다.

외부 일치 출력 테이블

각 기본 개체에는 외부 일치 일괄 작업에 대한 출력 데이터가 포함된 외부 일치 출력 테이블이 있습니다. MDM Hub는 외부 일치 일괄 작업이 실행되기 전에 외부 일치 출력 테이블을 삭제한 다음 다시 작성합니다.

MDM Hub에서는 외부 일치 출력 테이블에 *C_<base_object_name>_EMI*라는 이름을 지정합니다. 여기서 *base_object_name*은 외부 일치 일괄 작업과 연결되어 있는 기본 개체의 이름입니다.

외부 일치 일괄 작업에서는 외부 일치 출력 테이블을 일치 쌍으로 채웁니다. 외부 일치 작업에서는 일치 테이블을 채우지 않습니다. 외부 일치 출력 테이블의 각 행은 일치된 레코드 쌍을 나타냅니다. 쌍의 한 레코드는 외부 일치 입력 테이블에서 가져온 것이고 쌍의 다른 레코드는 기본 개체에서 가져온 것입니다. 기본 키 (*SOURCE_NAME* 및 *FILE_NAME*과 연결된 *SOURCE_KEY*)는 외부 일치 입력 테이블의 레코드를 식별합니다. *ROWID_OBJECT_MATCHED* 값은 기본 개체의 레코드를 식별합니다.

외부 일치 출력 테이블에는 다음과 같은 열이 포함됩니다.

열 이름	MDM Hub 데이터 유형(크기)	Null 가능 여부	설명
SOURCE_KEY	VARCHAR (50)	예	<i>C_<base_object_name>_EMI</i> 테이블의 소스 레코드에 다시 매핑합니다.
SOURCE_NAME	VARCHAR (50)	예	<i>C_<base_object_name>_EMI</i> 테이블의 소스 레코드에 다시 매핑합니다.
FILE_NAME	VARCHAR (50)	예	<i>C_base_object_name_EMI</i> 테이블의 소스 레코드에 다시 매핑합니다.
ROWID_OBJECT_MATCHED	CHAR(14)	아니요	기본 개체에서 외부 일치 입력 테이블의 레코드와 일치하는 레코드의 ROWID_OBJECT입니다.
ROWID_MATCH_RULE	CHAR(14)	예	외부 일치 일괄 작업에서 일치를 확인하는 데 사용한 일치 규칙을 식별합니다.
AUTOMERGE_IND	NUMBER (38)	아니요	일치 프로세스 중에 레코드가 자동 통합 대상인지 여부를 지정합니다. AUTOMERGE_IND는 다음 값 중 하나입니다. <ul style="list-style-type: none"> - 0: 레코드가 자동 통합 대상이 아닙니다. - 1: 레코드가 자동 통합 대상입니다. - 2: 레코드가 자동 일치 대상이지만 레코드 중 하나 이상이 PENDING 상태입니다. 상태 관리를 활성화하고 "보류 중인 레코드에서 일치 활성화"를 활성화하면 값 2가 발생합니다. PENDING 레코드로 그룹을 빌드하지 마십시오. PENDING 레코드는 개별 일치 항목으로 유지합니다. 자동 병합 일괄 작업에서는 AUTOMERGE_IND 값이 1인 모든 레코드를 처리합니다.
CREATOR	VARCHAR2 (50)	예	레코드의 생성을 담당하는 사용자 또는 프로세스입니다.
CREATE_DATE	TIMESTAMP	예	레코드가 생성된 날짜입니다.

입력 테이블 채우기

외부 일치 작업을 실행하기 전에 기본 개체의 레코드에 대해 일치할 레코드로 EMI 테이블을 채워야 합니다. EMI 테이블에 데이터를 로드하는 프로세스는 MDM Hub 외부에서 실행됩니다. 사용자 데이터베이스 환경에서 작동하는 데이터 로드 도구를 사용하십시오.

EMI 테이블을 채울 때 *_EMI* 테이블에서 다시 링크할 수 있도록 하나 이상의 시스템 열에 대한 데이터를 제공해야 합니다. 또한 *C_BaseObject_EMI* 테이블에는 고유한 소스 키를 갖고 있고 다른 테이블에 대한 외래 키가 없는 플랫 레코드가 포함되어야 합니다.

입력 테이블의 데이터 유형 변환

MDM Hub에서는 기본 개체에 날짜 데이터 유형을 가진 열이 연결된 EMI 테이블에서 VARCHAR 데이터 유형 열로 표시됩니다. MDM Hub에서 데이터 비교를 위해 날짜를 문자로 변환하기 때문에 이 문제가 발생합니다. EMI 테이블을 채우기 전에 기본 개체에서 날짜 데이터 유형을 변환할 경우 이 문제를 방지할 수 있습니다.

1. 변환할 기본 개체의 레코드를 식별합니다. 열 및 연결된 rowid_object의 날짜 값을 기록해 둡니다.

2. 쿼리 도구를 사용하여 기본 개체 레코드에 해당하는 **STRP** 테이블에서 날짜 데이터 유형의 레코드를 선택합니다.
쿼리에서 **SSA_DATA** 필드의 날짜 및 타임스탬프와 유사한 데이터를 가진 레코드를 반환합니다.
3. 날짜 형식 마스크 **YYYY/MM/DD HH24:MI:SS**를 사용하여 기본 개체의 날짜 값을 **VARCHAR** 값으로 변환합니다.

데이터베이스	변환 명령
Oracle 및 IBM DB2	<code>(to_char(date_column, 'YYYY/MM/DD HH24:MI:SS'))</code>
Microsoft SQL Server	<code>(convert(VARCHAR(10), date_column, 111) + ' ' + convert(VARCHAR(8), date_column, 108))</code>

4. 외부 일치를 위해 데이터를 **EMI** 테이블에 로드합니다.
5. 외부 일치 작업을 실행합니다.
외부 일치 작업에서 날짜 데이터 유형에 대한 일치 항목을 반환합니다.

외부 일치 작업 실행

1. **MDM Hub** 외부에서 실행되는 데이터 로드 프로세스를 사용하여 **C_BaseObject_EMI** 테이블에 데이터를 채웁니다.
2. **Hub** 콘솔에서 다음 도구 중 하나를 시작합니다.
 - 일괄 처리 뷰어
 - 일괄 그룹
3. 기본 개체에 대해 외부 일치 작업을 선택합니다.
4. 외부 일치에 사용할 일치 규칙 집합을 선택합니다.
5. 외부 일치 작업을 실행합니다.
 - 외부 일치 작업은 **C_BaseObject_EMI** 테이블의 모든 레코드를 기본 개체의 레코드와 일치시킵니다. 입력 또는 출력 테이블의 통합 표시기에 대한 개념은 없습니다.
 - 일치 그룹 빌드는 결과에 대해 실행되지 않습니다.
6. 데이터 관리 도구를 사용하여 **C_BaseObject_EMO** 테이블의 결과를 검사합니다.
7. 결과를 저장하려면 외부 일치 작업을 다시 실행하기 전에 데이터 백업 복사본을 만듭니다.

참고: 외부 일치 작업을 실행할 때마다 **MDM Hub**에서 **C_BaseObject_EMO** 테이블을 제거하고 다시 작성합니다.

일치 토큰 생성 작업

일치 토큰 생성 작업은 토큰화 프로세스를 실행합니다. 이 프로세스는 일치 토큰을 생성하고 일치 프로세스의 후속 작업에서 일치 후보를 식별하는 데 사용할 수 있도록 기본 개체와 연결된 일치 키 테이블에 일치 토큰을 저장합니다. 일치 토큰 생성 작업은 정확히 일치하는 기본 개체에는 적용되지 않고 유사 항목 일치 기본 개체에만 적용됩니다.

일치 프로세스를 사용하려면 일치 키 테이블의 일치 토큰이 최신 상태여야 합니다. 로드 프로세스 중에 레코드가 추가되거나 업데이트된 경우와 같이 일치 토큰을 업데이트해야 하는 경우 일치 작업을 시작할 때 일치 프로세스

가 자동으로 토큰화 프로세스를 실행합니다. 일치 프로세스가 빠르게 실행되게 하려면 일치 프로세스 실행 전에 다음 방법 중 하나를 사용하여 토큰화 프로세스를 별도로 실행하는 것이 좋습니다.

- 일치 토큰 생성 작업을 수동으로 실행
- 로드 프로세스 완료 후 자동으로 실행되도록 토큰화 프로세스 구성

상태 사용 기본 개체에 대한 토큰화 프로세스

상태 사용 기본 개체에 한해 토큰화 프로세스는 DELETED 상태의 레코드를 건너뜁니다.

이러한 레코드는 토큰화 API를 통해 토큰화할 수 있지만 일괄 처리에서는 무시됩니다. PENDING 상태의 레코드는 MATCH_PENDING_IND(기본적으로 해제됨)를 설정하여 기본 개체별로 일치시킬 수 있습니다.

일치 토큰 생성 작업의 범위 설정

일치 토큰 생성 작업을 실행하기 전에 모든 레코드에 대해 토큰을 생성할지, 아니면 더티 테이블에 ROWID_OBJECT 값이 있는 새 레코드와 변경된 레코드에 대해서만 토큰을 생성할지 결정해야 합니다.

- 기본 개체의 모든 레코드에 대한 일치 토큰을 생성하려면 **모든 일치 토큰 다시 생성** 확인란을 선택합니다.
- 기본 개체의 새 레코드 또는 업데이트된 레코드에 대한 일치 토큰을 생성하려면 **모든 일치 토큰 다시 생성** 확인란을 선택 취소합니다.

일치 토큰이 생성된 후 기본 개체에 대한 일치 작업을 실행할 수 있습니다.

팁: 일치 토큰 생성 프로세스에서 ssa.ssaname3.jssan3cl 클래스를 초기화할 수 없으며 시스템 관리자에게 문의하라는 내용의 오류가 반환됩니다.

1. SSA-NAME3에 대한 DLL(동적 연결 라이브러리) 파일이 포함된 <MDM 설치 디렉터리>/hub/cleanse/lib 디렉터리에 대한 경로가 PATH 환경 변수에 포함되어 있는지 확인합니다.
2. Microsoft Visual Studio 2015용 Visual C++ 재배포 가능 패키지가 MDM Hub의 검색 및 일치를 수행하는 처리 서버에 설치되어 있는지 확인합니다.
3. Microsoft Visual Studio 2015용 Visual C++ 재배포 가능 패키지가 설치되어 있는 경우 Dependency Walker(depends.exe) 같은 종속성 검사기를 사용하여 jssan3cl.dll을 로드하고 Visual C++ 재배포 가능 패키지가 성공적으로 적용되었는지 확인합니다.

팁: Visual Studio 2015용 Visual C++ 재배포 가능 패키지를 사용하려면 Windows Server에 운영 체제 패치가 설치되어 있어야 합니다. Visual C++ 재배포 가능 패키지를 설치하기 전에 운영 체제 요구 사항을 확인하십시오. 예를 들어 기존 버전인 Windows Server 2012에서는 Visual C++ 재배포 가능 패키지를 설치하기 전에 약 100개(전체적으로 약 2GB)의 패치를 운영 체제에 적용해야 합니다.

스마트 검색 데이터 초기 인덱스 작업

스마트 검색 데이터 초기 인덱스 작업은 비즈니스 항목에서 검색 가능한 필드의 모든 값에 대한 인덱스를 생성합니다. 이 인덱스는 검색 시 검색 가능한 필드에 포함된 데이터를 검색할 때 사용됩니다.

스마트 검색 데이터 초기 인덱스 작업을 실행하여 검색 가능한 비즈니스 항목에서 레코드를 추출하고 이 레코드를 인덱스에 추가합니다. 데이터를 한 번 이상 인덱싱한 후 로드 작업을 실행하면, 로드 작업에서 내부적으로 스마트 검색 데이터 초기 인덱스 작업을 실행하여 새로 추가되고 업데이트된 데이터를 인덱싱합니다.

스마트 검색 데이터 초기 인덱스 작업은, 작업에서 모든 레코드에 대한 인덱싱 요청을 대기열에 추가한 후에 레코드를 비동기적으로 인덱싱하고 작업의 성공적인 완료를 보고합니다. 검색 요청은 인덱싱 요청이 성공적으로 완료된 후에만 예상된 결과를 반환합니다. 인덱싱 요청이 완료되는 데는 몇 분이 소요될 수 있습니다.

모든 레코드를 인덱싱한 후에 레코드를 추가하거나 업데이트할 경우 새 비즈니스 항목 또는 업데이트된 비즈니스 항목을 인덱싱해야 합니다. 기본 개체 레코드를 삭제할 경우 일부 인덱스가 만료되고 관련이 없어질 수 있습니다.

니다. 처음에 스마트 검색 데이터 인덱싱 일괄 작업을 실행하여 데이터를 다시 인덱싱하고, 만료된 인덱스를 제거하고, 검색 요청의 성능을 개선할 수 있습니다.

정수 필드 값이 19자리를 넘으면 스마트 검색 데이터 초기 인덱싱 작업이 값을 인덱싱하지 않습니다. 날짜 및 시간 필드 값에 시간대 세부 정보가 포함되지 않으면 스마트 검색 데이터 초기 인덱싱 작업은 시간을 로컬 시간대로 간주합니다. 이 작업은 해당 시간을 UTC로 변환하고 UTC 시간을 인덱싱합니다.

처음에 스마트 검색 데이터 인덱싱 메트릭스

다음 테이블에서는 처음에 스마트 검색 데이터 인덱싱 작업을 성공적으로 실행한 후 일괄 처리 뷰어에 표시되는 메트릭스를 설명합니다.

메트릭스	설명
삽입됨	인덱스에 추가된 총 레코드 수를 나타냅니다.
거부된 레코드	거부된 총 레코드 수를 나타냅니다. HUB_STATE_IND 값이 올바르지 않으면 작업에서 레코드를 거부합니다.
삭제된 BO 레코드	삭제된 상태에 있는 총 레코드 수를 나타냅니다. HUB_STATE_IND 값이 -1이면 레코드는 삭제된 상태에 있는 것입니다.

키 일치 작업

기본 키 일치 규칙과 함께만 사용되는 키 일치 작업은 동일한 기본 키 값을 사용하는 둘 이상의 소스 시스템의 레코드를 대상으로 일치 프로세스를 실행합니다.

키 일치 작업에서는 새 레코드를 서로 비교하고 기존 레코드와 비교한 다음, 기본 키 일치 규칙에 정의된 대로 소스 레코드 키의 비교에 따라 잠재적 일치 항목을 식별합니다.

키 일치 작업은 기본 개체의 기본 키 일치 규칙이 생성되었거나 스키마 관리자의 일치/병합 설정 구성에서 변경되었을 경우 자동으로 생성됩니다.

로드 작업

로드 작업은 준비 테이블에서 Hub 저장소의 해당 대상 기본 개체로 데이터를 이동합니다. 또한 트러스트된 열이 정의된 기본 개체에 대한 트러스트 값을 계산하고 유효성 검사 규칙(정의된 경우)을 적용하여 최종 트러스트 값을 결정합니다.

로드 작업과 상태 사용 기본 개체

상태 사용 기본 개체의 경우 로드 일괄 처리에서 모든 상태의 레코드를 로드할 수 있습니다. 상태는 준비 테이블에서 입력 열로 지정됩니다. 입력 상태는 매핑 보기에서 랜딩 테이블 열로 지정하거나 파생될 수 있습니다. 입력 상태가 매핑에서 지정되지 않으면 ACTIVE로 간주됩니다.

다음 테이블에는 입력 상태가 기존 XREF의 상태에 영향을 주는 방식이 설명되어 있습니다.

	기존 XREF 상태:	ACTIVE	PENDING	DELETED	XREF 없음 (rowid 기준 로드)	기본 개체 없음
수신 XREF 상태:						
ACTIVE		업데이트	업데이트 + 승격	업데이트 + 복원	삽입	삽입
PENDING		업데이트 보류 중	업데이트 보류 중	업데이트 보류 중 + 복원	업데이트 보류 중	삽입 보류 중
DELETED		일시 삭제	영구 삭제	영구 삭제	오류	오류
정의되지 않음		Active로 간주	Pending으로 간주	Deleted로 간주	Active로 간주	Active로 간주

참고: HUB_STATE_IND 값이 올바르지 않으면 레코드가 거부됩니다.

다음 테이블에서는 Informatica MDM Hub가 특정 작업에 대한 로드 및 Put 중에 레코드 상태를 기준으로 상태 사용 기본 개체의 레코드를 처리하는 방식에 대한 매트릭스를 제공합니다.

	수신 레코드 상태	기존 레코드 상태	참고
XREF 레코드를 업데이트하는 경우:	ACTIVE	ACTIVE	
	DELETED	ACTIVE	
	PENDING	PENDING	
	ACTIVE	PENDING	
	DELETED	DELETED	
	PENDING	DELETED	
	DELETED		기본 개체 rowid 삭제 레코드가 들어오면 Informatica MDM Hub는 ROWID_SYSTEM에 관계없이 기본 개체와 모든 XREF 레코드를 DELETED 상태로 업데이트합니다.
XREF 레코드를 삽입하는 경우:	PENDING	ACTIVE	쌍의 두 번째 레코드가 생성됩니다.
	ACTIVE	레코드 없음	
	PENDING	레코드 없음	
XREF 레코드를 삭제하는 경우:	ACTIVE	PENDING(쌍을 이루는 레코드)	쌍의 ACTIVE 레코드가 삭제된 다음 PENDING 레코드가 업데이트됩니다. 쌍을 이루는 레코드는 PKEY_SRC_OBJECT 및 ROWID_SYSTEM이 동일한 두 개의 레코드입니다.

	수신 레코드 상태	기존 레코드 상태	참고
	DELETED	PENDING	
Informatica MDM Hub가 오류를 표시하는 경우:	PENDING	ACTIVE(쌍을 이루는 레코드)	쌍을 이루는 레코드는 PKEY_SRC_OBJECT 및 ROWID_SYSTEM이 동일한 두 개의 레코드입니다.

추가 참고:

- 업데이트형 로드의 경우 수신 상태가 지정되지 않으면 수신 상태는 현재 상태와 동일한 것으로 간주됩니다. 예를 들어 수신 상태가 null이고 업데이트할 XREF 또는 기본 개체의 기존 상태가 PENDING이면 수신 상태는 null 대신 PENDING으로 간주됩니다.

로드 작업 실행 규칙

로드 작업에 적용되는 규칙은 다음과 같습니다.

- 로드 작업에서 사용된 준비 테이블을 로드하는 준비 작업이 성공적으로 완료된 경우에만 로드 작업을 실행합니다.
- 하위 테이블에 대해 로드 작업을 실행하기 전에 상위 테이블에 대해 로드 작업을 실행합니다.
- 하위 개체의 조화가 정의되지 않은 경우(조화 테이블 및 열이 채워지지 않은 경우) 데이터를 성공적으로 로드 하려면 로드 작업을 실행하기 전에 하위 개체에 대해 준비 작업을 반복해야 합니다.
- 같은 기본 개체에 대해 한 번에 하나의 로드 작업만 실행할 수 있습니다. 같은 기본 개체에 대해 로드 작업을 동시에 여러 개 실행할 수 없습니다.
- 상위 기본 개체가 공통의 하위 기본 개체를 공유하는 경우 여러 상위 기본 개체에 대한 로드 작업을 동시에 실행할 수 없습니다.

로드 작업에서 강제 업데이트

로드 작업을 실행하기 전에 **강제 업데이트** 확인란을 사용하여 로드 작업이 준비 테이블에서 대상 기본 개체로 데이터를 로드하는 방식을 구성할 수 있습니다. 기본적으로 Informatica MDM Hub는 준비 테이블에 있는 각 레코드의 마지막 업데이트 날짜를 검사하여 아직 레코드를 로드하지 않았는지 확인합니다. 이 동작을 재정의하려면 **강제 업데이트** 확인란을 선택합니다. 그러면 마지막 업데이트 날짜가 무시되고 새로 고침이 강제되어 각 레코드가 준비 테이블에서 이미 로드되었는지 여부에 관계없이 로드됩니다. 하지만 이 방법은 신중하게 사용해야 합니다. 로드할 데이터의 양에 따라 강제 업데이트는 처리 시간이 매우 오래 걸릴 수 있습니다.

로드 작업의 상태 관리 재정의 시스템

하나의 소스 시스템을 상태 관리 재정의 시스템으로 구성할 수 있습니다. 상태 관리 재정의 시스템에는 삭제된 상태의 레코드가 있을 수 있습니다. 로드 작업 중에 상태 관리 재정의 시스템의 삭제된 레코드는 기본 개체 레코드의 셀 값을 재정의할 수 있습니다.

로드 작업 중에 상태 관리 재정의 시스템을 사용하여 기본 개체 레코드의 셀 값을 재정의하려면 다음 조건을 충족해야 합니다.

- 랜딩 테이블의 HUB_STATE_IND 열이 준비 테이블의 HUB_STATE_IND 열에 매핑되어야 합니다.
- 랜딩 테이블의 DELETED_IND 열이 준비 테이블의 DELETED_IND 열에 매핑되어야 합니다.
- Hub 상태 표시기가 삭제된 상태에 있어야 합니다.
- 준비 테이블의 DELETED_IND 열 값이 -999999999여야 합니다.

로드 작업 중 일치 토큰 생성

스키마 도구에서 기본 개체의 고급 속성을 구성할 때 **로드 시 일치 토큰 생성** 확인란을 선택하여 레코드가 기본 개체에 로드되고 나면 로드 작업 중에 일치 토큰을 생성할 수 있습니다.

기본적으로 이 확인란은 선택 취소되어 있으며, 이 경우 일치 토큰은 일치 프로세스 중에 생성됩니다.

시스템 열에 대한 로드 일괄 작업

로드 일괄 작업의 효과는 시스템 열에 따라 달라집니다.

다음 테이블에서는 로드 일괄 작업의 삽입 작업과 업데이트 작업이 시스템 열에 미치는 영향에 대해 설명합니다.

시스템 열	로드 일괄 작업의 작업 유형	시스템 열에 대한 효과
CREATE_DATE	삽입	로드 일괄 작업에서 이러한 열을 준비 테이블의 데이터로 채웁니다. 준비 테이블 열이 null 값을 가진 경우 로드 일괄 작업에서 이러한 열을 열 기본값으로 채웁니다.
CREATE_DATE	업데이트	기본 개체의 열은 원래 값을 유지합니다.
CREATOR	삽입	로드 일괄 작업에서 이러한 열을 준비 테이블의 데이터로 채웁니다. 준비 테이블 열이 null 값을 가진 경우 로드 일괄 작업에서 이러한 열을 열 기본값으로 채웁니다.
CREATOR	업데이트	기본 개체의 열은 원래 값을 유지합니다.
UPDATE_BY	삽입	로드 일괄 작업에서 이러한 열을 준비 테이블의 데이터로 채웁니다. 준비 테이블 열이 null 값을 가진 경우 로드 일괄 작업에서 이러한 열을 열 기본값으로 채웁니다.
UPDATE_BY	업데이트	기본 개체의 열은 원래 값을 유지합니다. 로드 일괄 작업에서 교차 참조 테이블의 열을 준비 테이블의 데이터로 채웁니다.
LAST_UPDATE_DATE	삽입	로드 일괄 작업에서 기본 개체 테이블 및 교차 참조 테이블의 LAST_UPDATE_DATE 열을 SYSDATE로 채웁니다.
LAST_UPDATE_DATE	업데이트	로드 일괄 작업에서 기본 개체 테이블 및 교차 참조 테이블의 LAST_UPDATE_DATE 열을 SYSDATE로 채웁니다.
SRC_LUD	삽입	로드 일괄 작업에서 SRC_LUD 열을 LAST_UPDATE_DATE 열의 값으로 채웁니다.
SRC_LUD	업데이트	로드 일괄 작업에서 SRC_LUD 열을 LAST_UPDATE_DATE 열의 값으로 채웁니다.
DELETED_IND	삽입	로드 일괄 작업에서 이러한 열을 준비 테이블의 데이터로 채웁니다. 준비 테이블 열이 null 값을 가진 경우 로드 일괄 작업에서 이러한 열을 열 기본값으로 채웁니다.
DELETED_IND	업데이트	기본 개체의 열은 원래 값을 유지합니다.

시스템 열	로드 일괄 작업의 작업 유형	시스템 열에 대한 효과
DELETED_BY	삽입	로드 일괄 작업에서 이러한 열을 준비 테이블의 데이터로 채웁니다. 준비 테이블 열이 null 값을 가진 경우 로드 일괄 작업에서 이러한 열을 열 기본값으로 채웁니다.
DELETED_BY	업데이트	기본 개체의 열은 원래 값을 유지합니다.
DELETED_DATE	삽입	로드 일괄 작업에서 이러한 열을 준비 테이블의 데이터로 채웁니다. 준비 테이블 열이 null 값을 가진 경우 로드 일괄 작업에서 이러한 열을 열 기본값으로 채웁니다.
DELETED_DATE	업데이트	기본 개체의 열은 원래 값을 유지합니다.

로드 작업 메트릭스

로드 작업을 실행한 후 일괄 처리 뷰어에서 작업 실행 로그에 다음과 같은 메트릭스(해당되는 경우)가 표시됩니다.

메트릭스	설명
총 레코드 수	로드 작업에서 처리된 레코드의 수입니다.
삽입됨	로드 작업에서 대상 개체로 삽입된 레코드의 수입니다.
업데이트됨	로드 작업이 대상 개체에서 업데이트한 레코드의 수입니다.
작업 없음	작업이 수행되지 않은 레코드(기본 개체에 이미 있는 레코드)의 수입니다.
업데이트된 XREF	이 기본 개체에 대한 교차 참조 테이블을 업데이트한 레코드의 수입니다. 증분 로드 중에 레코드를 로드할 경우 해당 레코드는 이미 통합되어 XREF에만 있고 기본 개체에는 없습니다.
토큰화된 레코드	로드 작업에서 토큰화된 레코드의 수입니다. 스키마 도구에서 "로드 시 일치 토큰 생성" 확인란을 선택한 경우에만 적용됩니다.
병합 제공자 XREF 레코드	다른 rowid_object로 병합된 업데이트된 교차 참조 레코드의 수입니다. 업데이트된 교차 참조 레코드의 총 수와 업데이트된 기본 개체 레코드의 수 사이의 차이를 나타냅니다.
조회 누락/올바르지 않은 rowid_object 레코드	조회 정보가 누락되었거나 rowid_object 레코드가 잘못된 소스 레코드의 수입니다.

수동 병합 작업

일치 작업을 실행한 후 데이터 스튜어드는 병합 관리자를 사용하여 일치 작업에서 수동 병합을 위해 대기열에 넣은 레코드를 처리할 수 있습니다.

수동 병합 작업은 일괄 처리 뷰어가 아닌 병합 관리자에서 실행합니다. 일괄 처리 뷰어에서는 병합 관리자에서 실행된 수동 병합 작업의 작업 실행 로그를 조사하는 것만 가능합니다.

수동 통합의 최대 일치 항목 수

스키마 관리자에서 수동 통합을 위한 최대 일치 항목 수를 구성하여 데이터 스튜어드가 처리해야 하는 수동 병합 수를 제한할 수 있습니다. 이 제한에 도달하면 일치 항목 수가 줄어들 때까지 일치 작업, 자동 일치 및 병합 작업이 실행되지 않습니다.

병합 관리자에서 수동 병합 작업 실행

수동 병합 작업을 시작하면 병합 관리자에서 진행률 표시기가 포함된 대화 상자가 표시됩니다. 수동 병합은 완료되기까지 시간이 어느 정도 걸릴 수 있습니다. 처리 중에 문제가 발생하면 완료될 때 오류 메시지가 표시됩니다. 이 오류는 일괄 처리 뷰어의 수동 병합 작업에 대한 작업 실행 로그에도 표시됩니다.

병합 관리자의 프로세스 대화 상자에는 **프로세스를 불완전한 것으로 표시**라는 단추가 있는데, 이 단추는 수동 병합 작업의 상태를 업데이트하지만 수동 병합 작업을 중단하지는 않습니다. 이 단추를 클릭하면 병합 프로세스가 백그라운드에서 계속됩니다. 이 시점에 일괄 처리 뷰어에는 이 프로세스에 대한 항목이 나타납니다. 프로세스가 완료되면 성공이나 실패가 보고됩니다. 병합 관리자에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

수동 병합 해제 작업

병합 스타일 기본 개체의 경우에만 수동 병합 작업이 실행된 후 데이터 스튜어드가 데이터 관리자를 사용하여 수동으로 병합된 레코드를 수동으로 병합 해제할 수 있습니다. 수동 병합 해제 작업은 일괄 처리 뷰어가 아닌 데이터 관리자에서 실행됩니다. 일괄 처리 뷰어에서는 데이터 관리자에서 실행된 수동 병합 해제 작업에 대한 작업 실행 로그만 검사할 수 있습니다. 데이터 관리자에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

데이터 관리자에서 수동 병합 해제 작업 실행

수동 병합 해제 작업을 시작하면 데이터 관리자에 진행률을 표시하는 대화 상자가 나타납니다. 수동 병합 해제를 완료하려면 약간의 시간이 걸릴 수 있으며, 특히 관련 레코드가 많은 구성 레코드의 조합인 경우 오랜 시간이 걸릴 수 있습니다. 처리하는 동안 문제가 발생하면 완료 시 오류 메시지가 표시됩니다. 이 오류는 일괄 처리 뷰어의 수동 병합 해제 작업 실행 로그에도 표시됩니다.

데이터 관리자의 프로세스 대화 상자에는 수동 병합 해제 작업의 상태를 업데이트하는 **프로세스를 불완전한 것으로 표시** 단추가 있습니다. 이 단추로는 수동 병합 해제 작업이 중단되지 않습니다. 이 단추를 클릭하면 병합 해제 프로세스가 백그라운드에서 계속 실행됩니다. 이 시점에 일괄 처리 뷰어에는 이 프로세스에 대한 항목이 나타납니다. 프로세스가 완료되면 성공이나 실패가 보고됩니다.

일치 작업

일치 작업은 기본 개체에 대한 검색 키를 생성하고 데이터를 검색하여 일치 후보를 찾고 일치 후보에 일치 규칙을 적용합니다. 그런 다음 일치를 생성하여 자동 또는 수동 통합을 위한 대기열에 넣습니다.

ORS에서 새 기본 개체를 작성하면 MDM Hub가 이 개체의 일치 작업을 작성합니다. 각 일치 작업에서는 기본 개체의 새 레코드 또는 업데이트된 레코드를 기본 개체의 모든 레코드와 비교합니다.

일치 작업을 실행한 후 일치된 행에는 자동 및 수동 통합을 위한 플래그가 지정됩니다. MDM Hub는 자동 병합 또는 자동 연결을 통합하는 작업을 작성합니다. 레코드에 수동 통합 플래그가 지정되면 데이터 스튜어드가 병합 관리자를 사용하여 수동 통합을 수행해야 합니다. 수동 통합에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

일치 작업은 스키마 관리자의 일치/병합 설정 노드에서 구성할 수 있습니다.

테이블 간 일치 경로 또는 테이블 내 일치 경로에서 레코드 간의 관계를 정의하는 데 사용되는 기본 개체에는 일치 작업을 실행하지 마십시오. 그러면 관계 데이터가 변경되어 레코드 간의 연관이 손실될 수 있습니다.

참고: 환경에 Windows를 실행하는 응용 프로그램 서버와 Linux를 실행하는 데이터베이스 서버가 있는 경우 일치 작업이 응답하지 않을 수 있습니다.

일치 테이블

기본 개체에 대해 Informatica MDM Hub 일치 작업이 실행되면 해당 일치 테이블이 일치된 레코드 쌍으로 채워집니다.

일치 테이블 이름은 보통 *Base_Object_MTCH*로 지정됩니다.

일치 작업과 상태 사용 기본 개체

다음 테이블에는 상태 사용 기본 개체의 수신 상태를 기준으로 일치 일괄 처리 동작에 대한 정보가 설명되어 있습니다.

소스 기본 개체 상태	대상 기본 개체 상태	작업 결과
ACTIVE	ACTIVE	레코드가 일치를 위해 분석됩니다.
PENDING	ACTIVE	PENDING 상태의 레코드가 일괄 일치에서 무시되는지 여부는 테이블 수준 매개 변수입니다. 설정하면 일괄 일치에 지정된 기본 개체에 대한 PENDING 상태의 레코드가 포함됩니다. 하지만 PENDING 상태의 레코드는 일치에서 소스 레코드만 될 수 있습니다.
DELETED	모든 상태	DELETED 상태의 레코드는 일괄 일치에서 무시됩니다.
ANY	PENDING	PENDING 상태의 레코드는 일치의 대상이 될 수 없습니다.

참고: BMG(일치 그룹 빌드)의 경우에는 PENDING 상태의 레코드를 포함하는 그룹을 빌드하지 마십시오. PENDING 상태의 레코드는 개별 일치 항목으로 유지합니다. PENDING 상태의 일치 항목은 *automerge_ind=2*가 됩니다.

자동 일치 및 병합 작업

병합 스타일 기본 개체의 경우에만 기본 개체에 대해 자동 일치 및 병합 작업을 실행할 수 있습니다.

자동 일치 및 병합 일괄 작업은 일치시킬 레코드가 더 이상 없거나 수동 통합 제한의 최대 레코드 수에 도달할 때까지 일치 작업과 그 뒤의 자동 병합 작업으로 구성되는 연속 주기를 실행합니다.

일괄 작업 제한 설정

기본 개체의 일괄 작업에서는 기본 개체의 모든 레코드를 기본 개체의 다른 모든 레코드와 비교하지는 않습니다.

대신 스키마 도구에서 다음을 지정합니다.

- 일치 작업이 실행될 때마다 작업에서 일치시킬 레코드 수
- 수동 통합에 허용되는 일치 항목 수

이 기능을 통해 데이터 스튜어드가 처리해야 하는 수동 통합 수를 제한할 수 있습니다. 이 제한에 도달하면 수동 통합에 대해 준비된 일치 항목 수가 줄어들 때까지 일치 작업이 실행되지 않습니다.

일치 규칙 집합 선택

일치 작업 시 작업을 실행하기 전에 일치 항목을 평가하는 데 사용할 일치 규칙 집합을 선택할 수 있습니다.

이 기본 개체의 기본 일치 규칙 집합이 자동으로 선택됩니다. 다른 일치 규칙 집합을 선택하려면 드롭다운 목록을 클릭하여 이 기본 개체에 정의된 다른 일치 규칙 집합 중 원하는 집합을 선택하면 됩니다.

일치 작업 매트릭스

일치 작업을 실행한 후 일괄 처리 뷰어의 작업 실행 로그에 다음과 같은 매트릭스가 표시됩니다(해당하는 경우).

메트릭스	설명
일치된 레코드	일치 작업에 의해 일치된 레코드의 수입입니다.
토큰화된 레코드	일치 작업에 의해 토큰화된 레코드의 수입입니다.
자동 병합을 위해 대기됨	자동 병합을 위해 일치 작업에서 대기열에 보관된 레코드의 수입입니다. 자동 병합 작업을 사용하여 이러한 레코드를 처리합니다.
수동 병합을 위해 대기됨	수동 병합을 위해 일치 작업에서 대기열에 보관된 레코드의 수입입니다. 이러한 레코드를 처리하려면 Hub 콘솔에서 병합 관리자를 사용합니다.

일치 분석 작업

일치 분석 작업은 검색을 수행하여 메트릭스를 수집하지만 실제 일치를 수행하지는 않습니다.

대량의 일치 요구 가능성이 있는 데이터 영역(핫스팟)이 발견될 경우 Informatica MDM Hub는 해당 레코드를 대기 상태로 전환하여 과도 일치를 방지합니다. 대기 중인 레코드는 통합 표시기가 9로 지정되어 데이터 스튜디오가 일치 및 통합을 진행하기 전에 데이터 관리자 도구에서 수동으로 데이터를 검토할 수 있도록 합니다. 일치 분석 작업은 일치 규칙을 조정하는 데 주로 사용되거나, 기본 개체의 데이터가 "과도하게 일치"되는지 또는 과도 일치를 유발하는 큰 데이터 교집합("핫스팟")을 포함하는지 여부를 확인하는 데 간단히 사용됩니다.

일치 분석 작업의 종속성

각 일치 분석 작업은 기본 개체에서 토큰화되어 일치를 위해 대기된 신규/업데이트된 레코드에 종속됩니다. 테이블 간 일치가 활성화된 기본 개체의 경우 일치 분석 작업은 모든 하위 테이블에 대한 데이터 토큰화 작업의 완료 성공 여부에도 종속되는데, 이러한 토큰화 작업은 다시 하위 테이블에 대한 로드 작업의 성공 여부에 종속됩니다.

대기 중 레코드 수 제한

일치 분석 작업이 대기 상태로 전환하는 레코드의 수를 제한할 수 있습니다. 기본적으로 제한이 설정되어 있지 않습니다. 제한을 구성하려면 `cmxcleanse.properties` 파일을 편집하여 다음 설정을 추가합니다.

```
cmx.server.match.threshold_to_move_range_to_hold = n
```

여기서 `n`은 일치 분석 작업이 대기 상태로 전환할 수 있는 최대 레코드 수입입니다. `cmxcleanse.properties` 파일에 대한 자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

일치 분석 작업 매트릭스

일치 분석 작업을 실행한 후 일괄 처리 뷰어의 작업 실행 로그에 다음과 같은 매트릭스가 표시됩니다(해당하는 경우).

메트릭스	설명
토큰화된 레코드	일치 분석 작업에 의해 토큰화된 레코드의 수입입니다.
보류 상태로 전환된 레코드	과도 일치를 방지하기 위해 "보류" 상태(통합 표시기 = 9)로 전환된 레코드의 수입입니다. 이러한 레코드는 대개 데이터의 핫스팟을 나타내며 일치 프로세스에서 제외됩니다. 데이터 스튜어드가 데이터 관리자에서 보류 상태를 제거할 수 있습니다.
일치하는 것으로 분석된 레코드	일치를 위해 분석된 레코드의 수입입니다.
필요한 일치 비교	이 기본 개체를 처리하는 데 필요한 실제 일치 수입입니다.

실행 로그의 매트릭스

메트릭스	설명
보류 상태로 전환된 레코드	보류로 전환된 레코드 수
일치하는 것으로 분석된 레코드	일치를 위해 분석된 레코드 수
필요한 일치 비교	이 기본 개체를 처리하는 데 필요한 실제 일치 항목 수

통계

통계	설명
최상위 10개 범위 수	지정된 검색 범위에서 최상위 10개의 레코드입니다.
최상위 10개 범위 비교 수	지정된 범위에 대해 수행해야 하는 최상위 10개의 일치 비교입니다.
보류 상태로 전환된 총 레코드 수	보류 상태로 전환된 레코드 수입입니다.
보류 상태로 전환된 총 일치 항목 수	이 레코드를 보류 상태로 전환해야 하는 총 일치 항목 수입입니다.
처리하는 데 필요한 총 범위 수	기본 개체의 모든 일치 항목을 처리하는 데 필요한 범위 수입입니다.
총 후보 수	이 기본 개체에 대해 모든 일치 항목을 처리하는 데 필요한 총 일치 후보 수입입니다.
분석 시간	분석을 실행하는 데 필요한 시간입니다.

중복 데이터에 대한 일치 작업

중복 데이터에 대한 일치 작업에서는 일치하는 것으로 간주할 정확히 일치하는 중복 항목을 검색합니다.

정확히 일치하는 중복 항목의 최대 개수는 스키마 관리자에서 각 기본 개체의 중복 일치 임계값 속성에 정의된 기본 개체 열을 기반으로 합니다.

참고: 중복 일치 임계값이 1로 설정되어 있고 기본 개체에서 같지 않은 일치가 활성화된 경우에는 일괄 처리 뷰어에 중복 데이터에 대한 일치 작업이 표시되지 않습니다.

중복 데이터에 대한 일치를 수행하려면

1. 로드 작업을 마친 후 즉시 중복 데이터에 대한 일치 작업을 실행합니다.
2. 중복 데이터에 대한 일치 작업이 완료되면 자동 병합 작업을 실행하여 중복 데이터에 대한 일치 작업에서 발견된 중복 항목을 처리합니다.
3. 자동 병합 작업이 완료되면 일반적인 일치 및 병합 프로세스를 실행합니다(일치 작업을 실행한 다음 자동 병합 작업 실행 또는 자동 일치 및 병합 작업 실행).

다중 병합 작업

다중 병합 작업을 통해 여러 레코드를 단일 작업으로 병합할 수 있습니다. 기본적으로 전체 레코드 집합이 하나의 일괄 처리로 병합되도록 통합됩니다. 이 일괄 작업은 SIF MultiMergeRequest 요청을 호출하는 외부 응용 프로그램에 의해서만 시작됩니다. 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

승격 작업

상태 사용 개체의 경우 승격 작업이 XREF 테이블에서 PROMOTE_IND 열을 읽으며 열 값이 1인 모든 행에 대해 시스템 상태를 ACTIVE로 바꿉니다.

승격 작업이 실행된 후 Informatica MDM Hub는 PROMOTE_IND를 재설정합니다.

참고: 레코드가 승격되지 않은 경우에는 승격 일괄 처리 동안 레코드의 PROMOTE_IND 열이 0으로 바뀌지 않습니다.

다음은 승격 일괄 작업에 대한 동작 세부 정보입니다.

승격 전 XREF 상태	승격 전 기본 개체 상태	XREF의 Hub 작업	BO의 Hub 작업	BVT 새로 고침 여부	결과 BO 상태	작업 결과
PENDING	ACTIVE	승격	업데이트	예	ACTIVE	Informatica MDM Hub가 보류 중인 XREF를 승격하고 승격된 XREF를 포함하도록 BVT를 다시 계산합니다.
PENDING	PENDING	승격	승격	예	ACTIVE	Informatica MDM Hub가 보류 중인 XREF 및 기본 개체를 승격합니다. 그러면 승격된 XREF를 바탕으로 BVT가 계산됩니다.

승격 전 XREF 상태	승격 전 기본 개체 상태	XREF의 Hub 작업	BO의 Hub 작업	BVT 새로 고침 여부	결과 BO 상태	작업 결과
DELETED	이 작업은 기본 개체 레코드 상태와 상관없이 똑같이 작동합니다.	없음	없음	아니오	이 작업에 따라 결과 기본 개체 레코드의 상태가 변경되지 않습니다.	Informatica MDM Hub가 승격 일괄 작업에서 DELETED 레코드를 무시합니다. 이 시나리오는 승격 일괄 처리를 실행하기 전에 승격하도록 플래그가 지정된 레코드가 삭제된 경우에만 발생할 수 있습니다.
ACTIVE	이 작업은 기본 개체 레코드 상태와 상관없이 똑같이 작동합니다.	없음	없음	아니오	이 작업에 따라 결과 기본 개체 레코드의 상태가 변경되지 않습니다.	Informatica MDM Hub가 승격 일괄 작업에서 ACTIVE 레코드를 무시합니다. 이 시나리오는 승격 일괄 처리를 실행하기 전에 승격하도록 플래그가 지정된 레코드가 ACTIVE로 설정된 경우에만 발생할 수 있습니다.

참고: 승격 및 삭제 작업은 직접 하위 레코드에 계단식으로 배열됩니다.

Hub 콘솔을 사용하여 승격 작업 실행

승격 작업을 실행하려면

1. Hub 콘솔에서 다음 도구 중 하나를 시작합니다.
 - 일괄 처리 뷰어
 - 일괄 그룹
2. 원하는 기본 개체에 대한 승격 작업을 선택합니다.
3. 승격 작업을 실행합니다.
4. 승격 작업 결과를 표시합니다.

Informatica MDM Hub에 승격 작업 결과가 표시됩니다.

승격 작업 매트릭스

승격 작업을 실행하고 나면 일괄 처리 뷰어의 작업 실행 로그에 다음과 같은 매트릭스(해당되는 경우)가 표시됩니다.

메트릭스	설명
자동 승격된 레코드	승격 작업에 따라 승격된 레코드 수입입니다.
삭제된 XREF 레코드	승격 작업에 따라 삭제된 XREF 레코드 수입입니다.
승격되지 않은 활성 레코드	승격되지 않은 ACTIVE 상태의 레코드 수입입니다.
승격되지 않은 보호된 레코드	승격되지 않은 보호된 레코드 수입입니다.

승격 작업을 실행하고 나면 일괄 처리 뷰어의 작업 요약 페이지에서 해당 통계를 볼 수 있습니다.

기본 개체 다시 계산 작업

다시 계산할 기본 개체를 식별하는 데 있어 ROWID_OBJECT_TABLE 매개 변수를 사용하지 않은 경우 모든 기본 개체에 대해 BVT(최선의 진실, Best Version of the Truth)가 다시 계산됩니다.

트러스트 설정이나 유효성 검사 설정을 변경한 후 기본 개체 다시 계산 작업을 실행합니다. MDM Hub에서는 메타데이터가 동기화된 경우에도 트러스트 설정이나 유효성 검사 설정을 변경한 후 BVT(최선의 진실, Best Version of the Truth)를 다시 계산하지 않습니다. 트러스트 설정이나 유효성 검사 설정 변경 후 BVT(최선의 진실, Best Version of the Truth)를 다시 계산하지 않으면 BVT(최선의 진실, Best Version of the Truth)가 구식 값이 될 수 있습니다.

ROWID_OBJECT_TABLE 매개 변수를 사용하거나 사용하지 않은 상태에서 기본 개체 다시 계산 작업을 실행할 수 있습니다. ROWID_OBJECT_TABLE 매개 변수를 사용하여 작업을 실행하면 MDM Hub가 테이블/인라인 보기의 ROWID_OBJECT 열에서 식별된 모든 기본 개체에 대해 BVT(최선의 진실, Best Version of the Truth)를 다시 계산합니다. 인라인 보기에는 괄호가 필요합니다. ROWID_OBJECT_TABLE 매개 변수를 사용하지 않은 상태에서 작업을 실행하면 MDM Hub가 기본 개체의 모든 레코드에 대해 BVT(최선의 진실, Best Version of the Truth)를 다시 계산합니다. MDM Hub는 MATCH_BATCH_SIZE의 일괄 처리 크기의 레코드나 테이블의 레코드 수에 대한 1/4 중 더 작은 것을 다시 계산합니다.

BVT 다시 계산 작업

지정된 ROWID_OBJECT에 대한 BVT(최선의 진실, Best Version of the Truth)를 다시 계산합니다.

트러스트 설정이나 유효성 검사 설정을 변경한 후 BVT 다시 계산 작업을 실행합니다. MDM Hub에서는 메타데이터가 동기화된 경우에도 트러스트 설정이나 유효성 검사 설정을 변경한 후 BVT(최선의 진실, Best Version of the Truth)를 다시 계산하지 않습니다. 트러스트 설정이나 유효성 검사 설정을 변경한 후 기본 개체를 다시 계산하지 않으면 BVT(최선의 진실, Best Version of the Truth)가 구식 값이 될 수 있습니다.

일치 테이블 재설정 작업

일치 테이블 재설정 작업은 일치 작업을 실행한 후 다음 조건이 있으면 자동으로 생성됩니다. 레코드에서 consolidation_ind가 2로 업데이트된 후 일치 규칙 변경

일치 후에 일치 규칙을 변경하면 일치 항목을 재설정하라는 메시지가 표시됩니다. 일치 항목을 재설정하면 일치 테이블의 모든 항목이 삭제됩니다. 또한 일치 테이블 재설정 작업으로 consolidation_ind = 2가 consolidation_ind = 4로 재설정됩니다.

변경 내용을 스키마 일치 열에 저장하면 기존 일치 항목을 재설정하고 일괄 처리 뷰어에서 일치 테이블 재설정 작업을 생성하라는 메시지가 표시된 메시지 상자가 나타납니다. 예를 클릭해야 합니다.

참고: 기존 일치 항목을 재설정하지 않은 경우 Informatica MDM Hub에서 일치 작업을 실행하기 전에 일치 토كن을 다시 생성해야 하므로 다음 일치 작업을 실행하는 데 있어 더 많은 시간이 걸립니다.

참고: 이 작업은 일괄 처리 뷰어에서 실행할 수 없습니다.

유효성 다시 검사 작업

유효성 다시 검사 작업은 로드 프로세스 동안 초기 유효성 검사 이후 수정된 레코드에 대해 유효성 검사 논리/규칙을 실행합니다. 레코드 변경으로 초기 로드 프로세스의 유효성 검사 단계가 게시되면 유효성 다시 검사를 실행할 수 있습니다. 레코드가 변경되지 않은 경우에는 레코드가 업데이트되지 않습니다. 일부 레코드가 변경되었고 기존 유효성 검사 규칙에 의해 발견된 경우에는 매트릭스에 결과가 표시됩니다.

참고: 유효성 다시 검사 작업은 초기 로드 후 그리고 유효성 검사 규칙 설정이 포함된 기본 개체에 대한 병합 전 유효성 검사가 열에 대해 활성화된 경우에만 실행할 수 있습니다.

기본 개체의 경우 일괄 처리 뷰어를 사용하여 유효성 다시 검사가 수동으로 실행됩니다.

준비 작업

준비 작업은 데이터를 랜딩 테이블에서 준비 테이블로 옮기며, 테이블 간 Informatica MDM Hub 매핑에서 구성된 정리를 수행합니다.

준비 작업에는 실행할 수 있는 병렬 정리 작업이 포함됩니다. 준비 상태는 준비하는 동안 연결된 처리 서버를 나 타냅니다.

준비가 활성화된 기본 개체의 경우 HUB_STATE_IND 값이 올바르지 않으면 레코드가 거부됩니다.

참고: 준비 작업이 회색으로 표시된 경우에는 준비 테이블, 열 매핑 또는 정리 함수의 변경으로 인해 매핑이 올바르지 않은 것입니다. 매핑 도구를 사용하여 특정 매핑을 열고 확인한 후 저장하십시오.

준비 작업 지침

준비 작업 일괄 작업을 실행할 경우

- 준비 작업에서 사용된 랜딩 테이블을 로드해야 하는 ETL 프로세스가 성공적으로 완료된 경우에만 준비 작업을 실행합니다.
- 준비 작업 간 종속성이 없는지 확인합니다.
- 작업을 실행하도록 설정된 처리 서버가 여러 개 있는 경우 여러 준비 작업을 동시에 실행할 수 있습니다.

준비 작업 매트릭스

준비 작업을 실행하고 나면 일괄 처리 뷰어의 작업 실행 로그에 다음과 같은 매트릭스가 표시됩니다.

메트릭스	설명
총 레코드 수	준비 작업에서 처리하는 레코드 수입입니다.
삽입됨	준비 작업에 의해 대상 개체에 삽입되는 레코드 수입입니다.
거부됨	준비 작업에서 거부하는 레코드 수입입니다.

동기화 작업

동기화 작업은 기본 개체에 대한 메타데이터를 업데이트합니다. 동기화 작업은 스키마 트러스트 설정을 변경한 후 그리고 로드 작업을 실행하기 전에 실행합니다.

기본 개체를 저장할 경우 다음 조건이 충족되면 MDM Hub에서 동기화 작업을 실행하도록 요청합니다.

- 기본 개체에는 데이터가 있습니다.
- 기본 개체에는 새로 추가된 트러스트가 활성화된 열이 있습니다.

동기화 작업을 실행하려면 일괄 처리 뷰어로 이동한 후 실행할 기본 개체에 대한 동기화 작업을 찾습니다. 그런 다음 **일괄 실행** 단추를 클릭하여 작업을 실행합니다. 초기 로드가 발생하면 MDM Hub이 트러스트가 활성화된 열이 있는 기본 개체에 대한 메타데이터를 업데이트합니다. 동기화 작업을 실행한 후에 로드 작업을 실행할 수 있습니다.

기본 개체에 정의된 트러스트 활성화 열이 지나치게 많거나 열 이름이 지나치게 길면 동기화 작업에서 오류가 발생합니다. 허용 가능한 최대 길이인 26자에 근접한 열 이름은 지나치게 긴 이름으로 간주됩니다. 작업에서 오류가 발생하지 않도록 하려면 열 이름의 길이를 짧게 하고 트러스트가 활성화된 열의 개수를 48개 미만으로 유지해야 합니다. 또는 기본 개체를 저장하기 전에 모든 트러스트 및 유효성 검사 열을 활성화하면, 동기화 작업을 실행하지 않아도 됩니다.

제 28 장

사용자 종료

이 장에 포함된 항목:

- [사용자 종료 개요, 565](#)
- [사용자 종료 처리, 566](#)
- [사용자 종료 JAR 파일, 567](#)
- [UserExitContext 클래스, 568](#)
- [준비 프로세스 사용자 종료, 569](#)
- [로드 프로세스 사용자 종료, 572](#)
- [일치 프로세스 사용자 종료, 574](#)
- [병합 프로세스 사용자 종료, 576](#)
- [병합 해제 프로세스 사용자 종료, 577](#)
- [태스크 관리 사용자 종료, 579](#)
- [사용자 종료 내에서 서비스 통합 프레임워크 API 사용, 580](#)
- [사용자 종료 구현 지침, 583](#)

사용자 종료 개요

사용자 종료는 MDM Hub의 기능을 확장하기 위해 SIF(서비스 통합 프레임워크) API 프로세스 또는 일괄 처리의 특정 시점에서 실행되도록 사용자가 개발한 사용자 지정 Java 코드입니다. MDM Hub는 사용자가 개발한 코드를 실행하기 위해 일반 MDM Hub 처리를 종료합니다.

예를 들어, 사후 랜딩 사용자 종료를 사용하여 델타 검색 프로세스 전에 주소에 대한 사전 정리를 수행할 수 있습니다.

단일 JAR 파일로 모든 사용자 종료를 MDM Hub에 업로드합니다. 각 연산 참조 저장소에 하나의 JAR 파일을 업로드할 수 있습니다.

MDM Hub는 사용자 종료 호출 시 입력 매개 변수 값을 제공합니다. 사용자 종료가 불필요하게 성능을 저하시키지 않도록 하려면 사용자 종료 구현 지침을 따르십시오.

다음과 같은 MDM Hub 프로세스에서 사용자 종료를 구현할 수 있습니다.

준비 프로세스

준비 프로세스에서는 랜딩 테이블의 데이터를 기본 개체와 연결된 준비 테이블로 이동합니다. 준비 프로세스 동안 다음 사용자 종료로 실행할 수 있습니다.

- 사후 랜딩 사용자 종료. ETL 프로세스를 통해 랜딩 테이블을 채운 후 사후 랜딩 사용자 종료를 사용하여 랜딩 테이블의 데이터를 세밀하게 조정합니다.
- 사전 준비 사용자 종료. 사전 준비 사용자 종료를 사용하여 델타 프로세스의 사용자 지정 처리를 수행합니다.
- 사후 준비 사용자 종료. 사후 준비 사용자 종료를 사용하여 준비 작업 마지막 부분에서 사용자 지정 처리를 수행합니다.

로드 프로세스

로드 프로세스에서는 준비 테이블의 데이터를 기본 개체 테이블로 이동합니다. 로드 프로세스 동안 다음 사용자 종료로 실행할 수 있습니다.

- 사후 로드 사용자 종료. 로드 프로세스 후에 사후 로드 사용자 종료를 사용하여 사용자 지정 처리를 수행합니다.

일치 프로세스

일치 프로세스에서는 잠재적 중복 항목인 기본 개체 레코드를 식별합니다. 일치 프로세스 동안 다음 사용자 종료로 실행할 수 있습니다.

- 사전 일치 사용자 종료. 일치 프로세스 전에 사전 일치 사용자 종료를 사용하여 사용자 지정 처리를 수행합니다.
- 사후 일치 사용자 종료. 사후 일치 사용자 종료를 사용하여 일치 테이블에 대해 사용자 지정 처리를 수행합니다.

병합 프로세스

병합 프로세스에서는 중복 기본 개체 레코드를 단일 마스터 기본 개체 레코드로 통합합니다. 병합 프로세스 동안 다음 사용자 종료로 실행합니다.

- 사후 병합 사용자 종료. 병합 프로세스 후에 사후 병합 사용자 종료를 사용하여 사용자 지정 처리를 수행합니다.

병합 해제 프로세스

병합 해제 프로세스에서는 단일 마스터 기본 개체 레코드를 레코드가 병합되기 전에 있었던 개별 기본 개체 레코드로 병합 해제합니다. 병합 해제 프로세스 동안 다음 사용자 종료로 실행할 수 있습니다.

- 사전 병합 해제 사용자 종료. 병합 해제 프로세스 전에 사전 병합 해제 사용자 종료를 사용하여 사용자 지정 처리를 수행합니다.
- 사후 병합 해제 사용자 종료. 병합 해제 프로세스 후에 사후 병합 해제 사용자 종료를 사용하여 사용자 지정 처리를 수행합니다.

사용자 종료 처리

MDM Hub는 사용자 종료로 블록, 일괄 처리 또는 SIF API 트랜잭션의 일부로 처리합니다(해당하는 경우). 사용자 종료에서 예외가 발생하는 경우 MDM Hub는 블록, 일괄 처리 또는 SIF API 호출에 대한 처리를 롤백합니다. SIF API와 일괄 처리는 동일한 사용자 종료로 호출합니다.

일괄 처리 동안 사용자 종료로 실행되면 사용자 종료는 사후 로드 및 사후 병합 사용자 종료를 제외하고는 일괄 작업이 실행되기 전이나 실행된 후에 실행됩니다. 사후 로드 및 사후 병합 사용자 종료는 MDM Hub가 각 블록을 처리한 후 실행됩니다. 처리 서버를 구성할 때 `cmxserver.properties` 파일에서 블록 크기를 구성합니다.

사용자 종료 JAR 파일

사용자 종료 인터페이스 클래스를 기반으로 사용자 지정 사용자 종료 클래스를 개발할 수 있습니다. 사용자 지정 사용자 종료 클래스를 별도의 JAR 파일로 구현한 다음 JAR 파일을 MDM Hub에 업로드하여 사용자 종료를 등록합니다.

MDM Hub 설치에는 <MDM Hub 설치 디렉터리>/hub/server/lib에 있는 mdm-ue.jar이라는 JAR 파일이 포함됩니다. mdm-ue.jar 파일에는 각 사용자 종료에 대한 Java 인터페이스 클래스가 포함되어 있습니다.

Hub 콘솔을 사용하여 사용자 지정 코드가 포함된 사용자 지정 JAR 파일을 MDM Hub에 업로드합니다. 각 연산 참조 저장소에 대해 하나의 사용자 종료 JAR 파일을 업로드할 수 있습니다. MDM Hub에서 사용자 종료 구현을 변경하려면 먼저 MDM Hub에서 사용자 종료를 제거합니다. 그런 다음 최신 구현이 포함된 사용자 종료가 포함되어 있는 JAR 파일을 업로드합니다.

사용자 종료 JAR 파일 구현

사용자 종료 JAR 파일을 구현하려면 사용자 지정 Java 코드를 작성한 다음 코드를 JAR 파일로 패키징합니다.

1. Java 개발 도구로 사용자 종료 인터페이스를 구현할 클래스를 작성합니다.
2. 구현된 사용자 지정 사용자 종료 클래스를 JAR 파일로 내보냅니다. Java 개발 도구나 다음과 같은 Java 명령을 사용하여 이 작업을 수행할 수도 있습니다.

```
<java_project_folder>\bin>jar -cf <user_specified_JAR_file_name> .\com\userexit\*.class
```

사용자 종료를 MDM Hub로 업로드

연산 참조 저장소에 사용자 종료를 업로드하려면 Hub 콘솔에서 사용자 개체 레지스트리를 사용합니다.

1. 사용자 종료를 업로드할 연산 참조 저장소에 연결되어 있는지 확인합니다.
2. 유틸리티 작업 영역에서 **사용자 개체 레지스트리**를 선택합니다.
3. 쓰기 잠금을 획득합니다.
4. 탐색 창에서 **사용자 종료**를 선택합니다.
5. **추가** 단추를 클릭합니다.
6. **사용자 종료 추가** 창에서 **찾아보기**를 클릭합니다.
7. **열기** 창에서 사용자 종료가 포함된 JAR 파일을 찾습니다. **열기**를 클릭합니다.
8. **사용자 종료 추가** 창에서 필요에 따라 설명을 입력한 다음 **확인**을 클릭합니다.

속성 창의 사용자 종료 테이블에 사용자 종료가 표시됩니다.

MDM Hub에서 사용자 종료 제거

MDM Hub에서 모든 사용자 종료를 삭제하려면 사용자 개체 레지스트리에서 사용자 종료를 제거합니다. 개별 사용자 종료를 제거할 수는 없습니다.

1. 유틸리티 작업 영역에서 **사용자 개체 레지스트리**를 선택합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 창에서 **사용자 종료**를 선택합니다.
4. 속성 창에서 사용자 종료 테이블을 선택한 다음 **제거** 단추를 클릭합니다. **확인**을 클릭합니다.

UserExitContext 클래스

UserExitContext 클래스에는 MDM Hub가 사용자 종료에 대해 사용하는 매개 변수가 포함되어 있습니다.

UserExitContext 클래스는 다음 매개 변수를 사용자 종료에 전달합니다.

batchJobRowid

일괄 작업의 작업 ID입니다. UserExitContext 클래스는 일괄 처리 동안 BatchJobRowid 매개 변수를 모든 사용자 종료에 전달합니다.

connection

MDM Hub 프로세스에서 사용하는 데이터베이스 연결입니다.

stagingTableName

로드 작업의 소스 테이블입니다. UserExitContext 클래스는 로드 및 준비 프로세스 동안 stagingTableName 매개 변수를 전달합니다.

tableName

MDM Hub 프로세스에서 사용하는 테이블의 이름입니다.

다음 코드는 UserExitContext 클래스를 보여 줍니다.

```
package com.informatica.mdm.userexit;

import java.sql.Connection;

/**
 * Represents the hub context that is passed to the user exit interfaces.
 * This is a placeholder for data that is applicable to all user exits.
 */
public class UserExitContext {
    String jobRowid;
    String tableName;
    String stagingTableName;
    Connection conn;

    /**
     * See the corresponding {@link #setBatchJobRowid(String) setter} method for details.
     *
     * @return the rowid of the batch job
     */
    public String getBatchJobRowid() {
        return this.jobRowid;
    }

    /**
     * See the corresponding {@link #setTableName(String) setter} method for details.
     *
     * @return the name of the table used in the hub process
     */
    public String getTableName() {
        return this.tableName;
    }

    /**
     * See the corresponding {@link #setDBConnection(String) setter} method for details.
     *
     * @return the database connection used in the hub process
     */
    public Connection getDBConnection() {
        return this.conn;
    }

    /**
     * Set the rowid of the batch job in the context. This is applicable to batch jobs only.
     */
}
```

```

    *
    * @param batchJobRowid the rowid of the batch job
    */
    public void setBatchJobRowid(String batchJobRowid) {
        this.jobRowid = batchJobRowid;
    }

    /**
     * Set the name of the table used in the hub process.
     *
     * @param tableName the name of the table
     */
    public void setTableName(String tableName) {
        this.tablName = tableName;
    }

    /**
     * See the corresponding {@link #setStagingTableName(String) setter} method for details.
     *
     * @return the name of the staging table used in the hub load and stage process
     */
    public String getStagingTableName() {
        return this.stagingTablName;
    }

    /**
     * Set the name of the staging table used in the context. This is applicable to load and stage
     process.
     *
     * @param stagingTableName the name of the staging table
     */
    public void setStagingTableName(String stagingTableName) {
        this.stagingTablName = stagingTableName;
    }

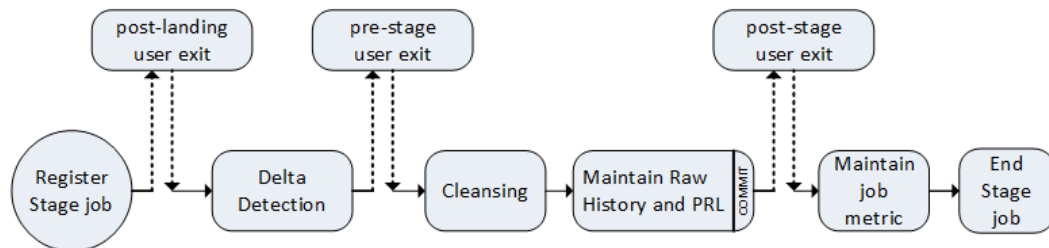
    /**
     * Set the database connection in the context.
     *
     * @param connection the database connection used in the hub process
     */
    public void setDBConnection(Connection connection) {
        this.conn = connection;
    }
}

```

준비 프로세스 사용자 종료

준비 프로세스에서는 사후 랜딩, 사전 준비 및 사후 준비 사용자 종료를 호출할 수 있습니다.

다음 이미지는 준비 프로세스 및 준비 프로세스에서 호출할 수 있는 사용자 종료를 보여 줍니다.



사용자 종료는 다음 시퀀스로 준비 프로세스 내에서 실행됩니다.

1. MDM Hub에서 준비 작업을 등록합니다.
2. 사후 랜딩 사용자 종료가 실행됩니다.
3. MDM Hub에서 델타 검색을 수행합니다.
4. 사전 준비 사용자 종료가 실행됩니다.
5. MDM Hub에서 데이터 정리를 수행합니다.
6. MDM Hub에서 준비 테이블을 채웁니다.
7. 감사 추적을 활성화하면 MDM Hub에서 Raw 테이블을 채웁니다.
8. MDM Hub에서 준비 테이블 변경 사항을 커밋합니다.
9. 사후 준비 사용자 종료가 실행됩니다.
10. 준비 작업이 종료됩니다.

사후 랜딩 사용자 종료

MDM Hub는 MDM Hub가 준비 작업을 등록한 후 사후 랜딩 사용자 종료를 호출합니다.

ETL 프로세스를 통해 랜딩 테이블을 채운 후 사후 랜딩 사용자 종료를 사용하여 랜딩 테이블의 데이터를 세밀하게 조정합니다. 델타 검색 전에 사후 랜딩 사용자 종료를 사용하여 랜딩 테이블에 대해 사용자 지정 처리를 수행할 수 있습니다. 예를 들어, 영구 삭제 검색을 수행하고 제어 문자를 인쇄 가능한 문자로 바꾸거나 주소에 대해 사전 정리 처리를 수행할 수 있습니다.

사후 랜딩 사용자 종료 인터페이스

사후 랜딩 사용자 종료를 구현할 때 적절한 인터페이스 이름, 메서드 및 매개 변수를 사용합니다.

인터페이스 이름

사후 랜딩 사용자 종료에서는 다음과 같은 정규화된 인터페이스 이름을 사용합니다.

```
com.informatica.mdm.userexit.PostLandingUserExit
```

메서드

사후 랜딩 사용자 종료에서는 다음과 같은 메서드를 사용합니다.

```
void processUserExit(UserExitContext userExitContext, String stagingTableName, String landingTableName, String previousLandingTableName) throws Exception;
```

합의 유형의 영구 삭제 검색을 사용하는 경우 메서드에 다음 논리를 추가합니다.

```
ConsensusFlagUpdate consensusProcess = new ConsensusFlagUpdate(userExitContext.getBatchJobRowid(), stagingTableName);
consensusProcess.startConsensusFlagUpdate(userExitContext.getDBConnection());
```

매개 변수

사후 랜딩 사용자 종료에서는 다음과 같은 매개 변수를 사용합니다.

landingTableName

준비 작업의 소스 테이블입니다.

previousLandingTableName

이전 랜딩 테이블 이름으로서, 이전 준비 작업 실행 시 준비 테이블에 매핑된 소스 데이터 사본을 포함하고 있습니다.

stagingTableName

준비 작업의 대상 테이블입니다.

userExitContext

매개 변수를 사용자 종료에 전달합니다.

사전 준비 사용자 종료

MDM Hub는 데이터를 준비 테이블에 로드하기 전에 사전 준비 사용자 종료를 호출합니다.

사전 준비 사용자 종료를 사용하여 델타 프로세스의 사용자 지정 처리를 수행합니다. 사전 준비 사용자 종료를 사용하여 델타 수가 미리 정의된 허용 가능한 제한을 초과하는지 여부를 확인할 수 있습니다. 예를 들어, 사용자 종료를 사용하여 소스 시스템의 델타 수가 500,000보다 큰 경우 준비 프로세스를 중지할 수 있습니다.

사전 준비 사용자 종료 인터페이스

사전 준비 사용자 종료를 구현할 때 적절한 인터페이스 이름, 메서드 및 매개 변수를 사용합니다.

인터페이스 이름

사전 준비 사용자 종료에서는 다음과 같은 정규화된 인터페이스 이름을 사용합니다.

```
com.informatica.mdm.userexit.PreStageUserExit
```

메서드

사전 준비 사용자 종료에서는 다음과 같은 메서드를 사용합니다.

```
void processUserExit(UserExitContext userExitContext, String stagingTableName, String landingTableName, String deltaTableName) throws Exception;
```

매개 변수

사전 준비 사용자 종료에서는 다음과 같은 매개 변수를 사용합니다.

deltaTableName

델타 테이블 이름입니다. 델타 테이블에는 MDM Hub에서 델타로 식별하는 레코드가 포함됩니다.

landingTableName

준비 작업의 소스 테이블입니다.

stagingTableName

준비 작업의 대상 테이블입니다.

userExitContext

매개 변수를 사용자 종료에 전달합니다.

사후 준비 사용자 종료

MDM Hub는 MDM Hub가 데이터를 준비 테이블에 로드한 후 사후 준비 사용자 종료를 호출합니다.

사후 준비 사용자 종료를 사용하여 준비 작업 마지막 부분에서 사용자 지정 처리를 수행합니다. 사후 준비 사용자 종료를 사용하여 준비 작업에서 거부된 레코드를 처리할 수 있습니다. 예를 들어, 알려진 중요하지 않은 조건 때문에 MDM Hub가 거부한 레코드를 삭제하도록 사용자 종료를 구성할 수 있습니다.

사후 준비 사용자 종료 인터페이스

사후 준비 사용자 종료를 구현할 때 적절한 인터페이스 이름, 메서드 및 매개 변수를 사용합니다.

인터페이스 이름

사후 준비 사용자 종료에서는 다음과 같은 정규화된 인터페이스 이름을 사용합니다.

```
com.informatica.mdm.userexit.PostStageUserExit
```

메서드

사후 준비 사용자 종료에서는 다음과 같은 메서드를 사용합니다.

```
void processUserExit(UserExitContext userExitContext, String stagingTableName, String landingTableName, String previousLandingTableName) throws Exception;
```

영구 삭제 검색을 사용하는 경우 메서드에 다음 논리를 추가합니다.

```
HardDeleteDetection hdd = new HardDeleteDetection(rowidJob, stagingTableName);  
hdd.startHardDeleteDetection(userExitContext.getDBConnection());
```

매개 변수

사후 준비 사용자 종료에서는 다음과 같은 매개 변수를 사용합니다.

landingTableName

준비 작업의 소스 테이블입니다.

previousLandingTableName

이전 랜딩 테이블 이름으로서, 이전 준비 작업 실행 시 준비 테이블에 매핑된 소스 데이터 사본을 포함하고 있습니다.

stagingTableName

준비 작업의 대상 테이블입니다.

userExitContext

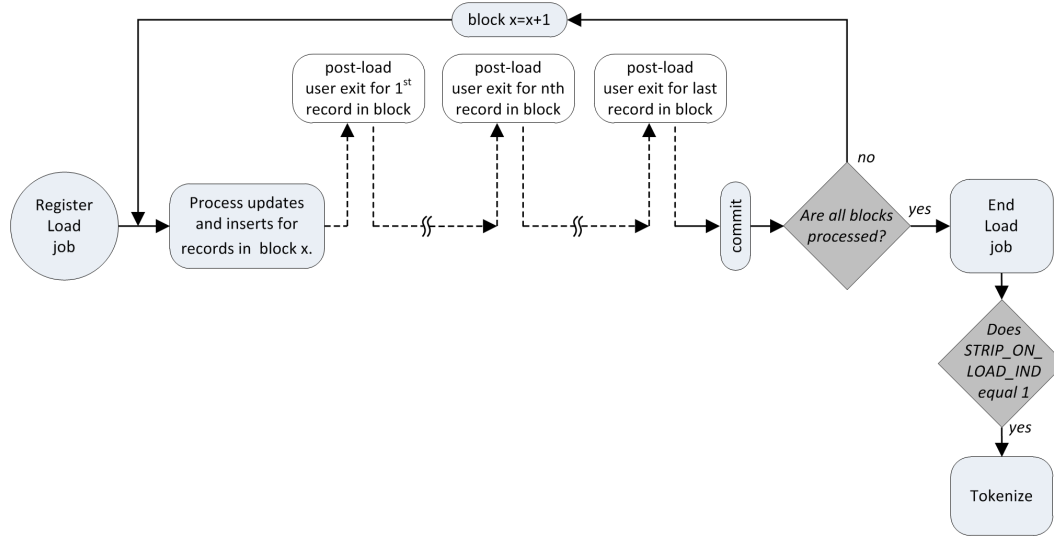
매개 변수를 사용자 종료에 전달합니다.

로드 프로세스 사용자 종료

로드 프로세스에서는 사후 로드 사용자 종료를 호출할 수 있습니다.

로드된 작업 수가 0보다 큰 경우 사후 로드 사용자 종료가 호출됩니다. 사후 로드 사용자 종료는 일괄 처리가 종료될 때가 아닌 MDM Hub가 각 블록을 처리한 후 실행됩니다. 사후 로드 사용자 종료는 MDM Hub가 블록의 모든 레코드에 대한 업데이트와 삽입을 처리한 후 블록의 각 레코드에 대해 실행됩니다.

다음 이미지는 로드 프로세스 및 로드 프로세스에서 호출할 수 있는 사용자 종료를 보여 줍니다.



사후 로드 사용자 종료는 다음 시퀀스로 로드 프로세스 내에서 실행됩니다.

1. MDM Hub에서 로드 작업을 등록합니다.
2. MDM Hub에서 첫 번째 블록에 대한 레코드를 업데이트하거나 삽입합니다.
3. 사후 로드 사용자 종료는 블록의 각 레코드에 대해 실행됩니다.
4. MDM Hub에서 변경 사항을 커밋합니다.
5. MDM Hub에 로드할 블록이 더 있는 경우 다음 블록을 처리하기 위해 프로세스가 2단계로 돌아갑니다.
6. MDM Hub에서 모든 블록을 로드한 후 로드 작업이 종료됩니다.
7. `strip_on_load_ind`가 1이면 토큰화 작업에서 유사 항목 일치를 수행하는 데 필요한 일치 토큰을 작성합니다.

사후 로드 사용자 종료

로드 프로세스 후에 MDM Hub에서 사후 로드 사용자 종료를 호출합니다.

로드 프로세스 후에 사후 로드 사용자 종료를 사용하여 사용자 지정 처리를 수행합니다.

참고: 사후 로드 사용자 종료가 반복적으로 호출되지 않도록 하려면 **MultiMerge SIF API**와 사후 로드 사용자 종료를 함께 사용하지 마십시오. **농기 API**와 사후 로드 사용자 종료를 함께 사용할 경우 사후 로드 사용자 종료가 반복적으로 호출되지 않도록 하려면 **Put API**의 **BypassPostLoadUE** 매개 변수를 **true**로 설정합니다.

사후 로드 사용자 종료 인터페이스

사후 로드 사용자 종료를 구현할 때 적절한 인터페이스 이름, 메서드 및 매개 변수를 사용합니다.

인터페이스 이름

사후 로드 사용자 종료에서는 다음과 같은 정규화된 인터페이스 이름을 사용합니다.

`com.informatica.mdm.userexit.PostLoadUserExit`

메서드

사후 로드 사용자 종료에서는 다음과 같은 메서드를 사용합니다.

```
void processUserExit(UserExitContext userExitContext, ActionType actionType,
    Map<String, Object>baseObjectDataMap, Map<String, Object> xrefDataMap,
    List<Map<String, Object>> xrefDataMapList) throws Exception;
```

매개 변수

사후 로드 사용자 종료에서는 다음과 같은 매개 변수를 사용합니다.

actionType

로드 프로세스에서 레코드를 삽입했는지 아니면 레코드를 업데이트했는지를 나타냅니다.

baseObjectDataMap

로드 프로세스에서 업데이트하거나 삽입한 기본 개체의 데이터가 포함되어 있습니다.

userExitContext

매개 변수를 사용자 종료에 전달합니다.

xrefDataMap

업데이트 또는 삽입에 제공된 교차 참조 레코드의 데이터가 포함되어 있습니다.

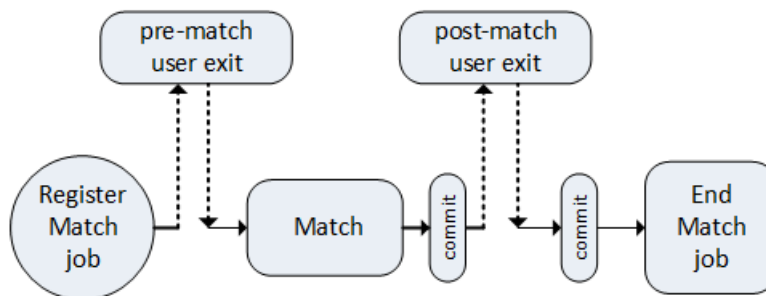
xrefDataMapList

유효 기간이 겹쳐 MDM Hub가 수정한 교차 참조 레코드의 데이터가 포함되어 있습니다. MDM Hub는 시간 표시 막대가 활성화된 기본 개체에 대해 **xrefDataMapList** 매개 변수를 채웁니다.

일치 프로세스 사용자 종료

일치 프로세스에서는 사전 일치 및 사후 일치 사용자 종료를 호출할 수 있습니다.

다음 이미지는 일치 프로세스 및 일치 프로세스에서 호출할 수 있는 사용자 종료를 보여 줍니다.



사용자 종료는 다음 시퀀스로 일치 프로세스 내에서 실행됩니다.

1. MDM Hub에서 일치 작업을 등록합니다.
2. 사전 일치 사용자 종료가 실행됩니다.
3. MDM Hub에서 일치 작업을 실행합니다.
4. MDM Hub에서 일치 변경 사항을 커밋합니다.
5. 사후 일치 사용자 종료가 실행됩니다.
6. MDM Hub에서 일치 변경 사항을 커밋합니다.
7. 일치 작업이 종료됩니다.

사전 일치 사용자 종료

일치 프로세스 전에 MDM Hub가 사전 일치 사용자 종료를 호출합니다.

일치 프로세스 전에 사전 일치 사용자 종료를 사용하여 사용자 지정 처리를 수행합니다.

사전 일치 사용자 종료 인터페이스

사전 일치 사용자 종료를 구현할 때 적절한 인터페이스 이름, 메서드 및 매개 변수를 사용합니다.

인터페이스 이름

사전 일치 사용자 종료에서는 다음과 같은 정규화된 인터페이스 이름을 사용합니다.

```
com.informatica.mdm.userexit.PreMatchUserExit
```

메서드

사전 일치 사용자 종료에서는 다음과 같은 메서드를 사용합니다.

```
void processUserExit(UserExitContext userExitContext, String matchSetName) throws Exception;
```

매개 변수

사전 일치 사용자 종료에서는 다음과 같은 매개 변수를 사용합니다.

matchSetName

일치 규칙 집합의 이름입니다.

userExitContext

매개 변수를 사용자 종료에 전달합니다.

사후 일치 사용자 종료

일치 프로세스 후에 MDM Hub가 사후 일치 사용자 종료를 호출합니다.

사후 일치 사용자 종료를 사용하여 일치 테이블에 대해 사용자 지정 처리를 수행합니다. 예를 들어, 사후 일치 사용자 종료를 사용하여 일치 대기열의 일치 항목을 조작할 수 있습니다.

사후 일치 사용자 종료 인터페이스

사후 일치 사용자 종료를 구현할 때 적절한 인터페이스 이름, 메서드 및 매개 변수를 사용합니다.

인터페이스

사후 일치 사용자 종료에서는 다음과 같은 정규화된 인터페이스 이름을 사용합니다.

```
com.informatica.mdm.userexit.PostMatchUserExit
```

메서드

사후 일치 사용자 종료에서는 다음과 같은 메서드를 사용합니다.

```
void processUserExit(UserExitContext userExitContext, String matchSetName) throws Exception;
```

매개 변수

사후 일치 사용자 종료에서는 다음과 같은 매개 변수를 사용합니다.

matchSetName

MDM Hub가 일치 항목을 찾는 데 사용한 일치 규칙 집합의 이름입니다.

userExitContext

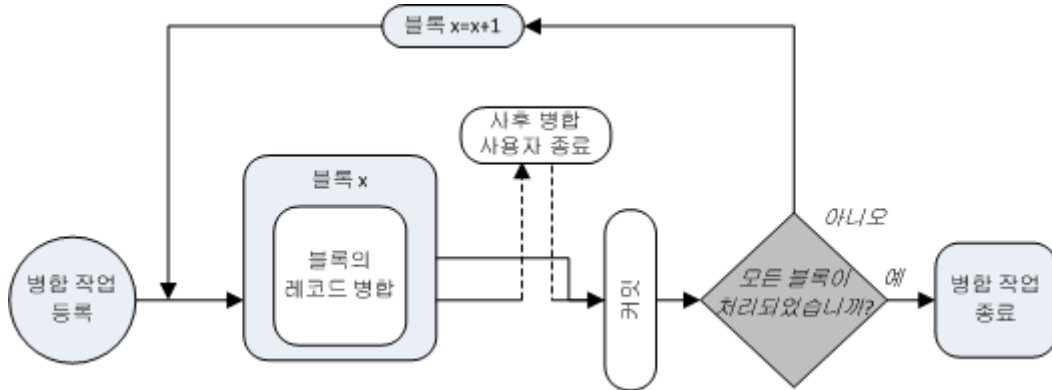
매개 변수를 사용자 종료에 전달합니다.

병합 프로세스 사용자 종료

병합 프로세스에서는 사후 병합 사용자 종료를 호출할 수 있습니다.

병합 SIF API 및 다중 병합 SIF API는 병합 프로세스가 완료된 후 사후 병합 사용자 종료를 호출합니다.

다음 이미지는 일괄 병합 프로세스 및 병합 프로세스에서 호출할 수 있는 사용자 종료를 보여 줍니다.



사용자 종료는 다음 시퀀스로 일괄 병합 프로세스 내에서 실행됩니다.

1. MDM Hub에서 병합 작업을 등록합니다.
2. MDM Hub가 블록에서 일치된 레코드를 병합합니다.
3. 사후 병합 사용자 종료는 해당 블록에서 영향을 받은 모든 레코드에 대해 실행됩니다.
4. MDM Hub에서 변경 사항을 커밋합니다.
5. MDM Hub에서 나머지 블록에 대해 2단계에서 4단계를 반복해서 실행합니다.
6. MDM Hub에서 모든 블록을 처리한 후 병합 작업이 종료됩니다.

사후 병합 사용자 종료

병합 프로세스 후에 MDM Hub에서 사후 병합 사용자 종료를 호출합니다.

병합 프로세스 후에 사후 병합 사용자 종료를 사용하여 사용자 지정 처리를 수행합니다. 예를 들어, 사후 병합 사용자 종료를 사용하여 상위 레코드 일치 및 병합의 영향을 받는 하위 레코드를 일치시키고 병합할 수 있습니다.

참고: 사후 병합 사용자 종료는 반복적으로 호출되지 않도록 하려면 MultiMerge SIF API와 사후 병합 사용자 종료를 함께 사용하지 마십시오.

사후 병합 사용자 종료 인터페이스

사후 병합 사용자 종료를 구현할 때 적절한 인터페이스 이름, 메서드 및 매개 변수를 사용합니다.

인터페이스 이름

사후 병합 사용자 종료에서는 다음과 같은 정규화된 인터페이스 이름을 사용합니다.

com.informatica.mdm.userexit.PostMergeUserExit

메서드

사후 병합 사용자 종료에서는 다음과 같은 메서드를 사용합니다.

```
void processUserExit(UserExitContext userExitContext, Map<String, List<String>> baseObjectRowIds) throws Exception;
```

매개 변수

사후 병합 사용자 종료에서는 다음과 같은 매개 변수를 사용합니다.

baseObjectRowIds

병합에 관련된 기본 개체 행 ID 목록입니다. 첫 번째 항목은 대상 기본 개체 레코드입니다. 목록의 나머지 항목은 대상에 병합된 소스 기본 개체입니다.

Map<String, List<String>>

키로 대상 행 ID와 매핑합니다.

userExitContext

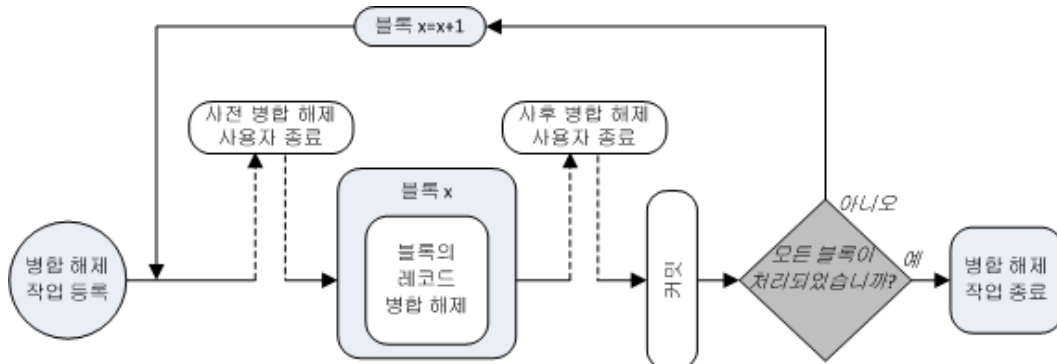
매개 변수를 사용자 종료에 전달합니다.

병합 해제 프로세스 사용자 종료

병합 해제 프로세스에서는 사전 병합 해제 및 사후 병합 해제 사용자 종료를 호출할 수 있습니다.

SIF API 병합 해제에는 병합 해제 프로세스가 시작되기 전에 사전 병합 해제 사용자 종료를 호출합니다. SIF API 병합 해제에는 병합 해제 프로세스가 완료된 후 사후 병합 해제 사용자 종료를 호출합니다.

다음 이미지는 일괄 병합 해제 프로세스 및 일괄 병합 해제 프로세스에서 호출할 수 있는 사용자 종료를 보여 줍니다.



사용자 종료는 다음 시퀀스로 일괄 병합 해제 프로세스 내에서 실행됩니다.

1. MDM Hub가 병합 해제 작업을 등록합니다.
2. 사전 병합 해제 사용자 종료가 실행됩니다.
3. MDM Hub는 첫 번째 블록의 레코드를 병합 해제합니다.
4. 사후 병합 해제 사용자 종료가 실행됩니다.
5. MDM Hub가 변경 내용을 커밋합니다.
6. MDM Hub는 나머지 블록에 대해 2단계에서 5단계를 반복합니다.
7. MDM Hub가 모든 블록을 처리한 후 병합 해제 작업이 종료됩니다.

사전 병합 해제 사용자 종료

병합 해제 프로세스 전에 MDM Hub에서 사전 병합 해제 사용자 종료를 호출합니다.

병합 해제 프로세스 전에 사전 병합 해제 사용자 종료를 사용하여 사용자 지정 처리를 수행합니다.

참고: 사전 병합 해제 사용자 종료는 반복적으로 호출되지 않도록 하려면 **Unmerge SIF API**와 사후 병합 해제 사용자 종료를 함께 사용하지 마십시오.

사전 병합 해제 사용자 종료 인터페이스

사전 병합 해제 사용자 종료를 구현할 때 적절한 인터페이스 이름, 메서드 및 매개 변수를 사용하십시오.

인터페이스 이름

사전 병합 해제 사용자 종료에서는 다음과 같은 정규화된 인터페이스 이름을 사용합니다.

```
com.informatica.mdm.userexit.PreUnmergeUserExit
```

메서드

사전 병합 해제 사용자 종료에서는 다음과 같은 메서드를 사용합니다.

```
void processUserExit(UserExitContext userExitContext, Set<UnmergeKey> unmergeKeys) throws Exception;
```

매개 변수

사전 병합 해제 사용자 종료에서는 다음과 같은 매개 변수를 사용합니다.

pkeySourceObject

소스 개체의 기본 키입니다. **UnmergeKeys**에서 블록의 각 레코드에 대해 전달했습니다.

rowidSystem

병합 해제에서 처리하는 기본 개체에 대한 교차 참조 레코드의 시스템 행 ID입니다. **UnmergeKeys**에서 블록의 각 레코드에 대해 전달했습니다.

userExitContext

매개 변수를 사용자 종료에 전달합니다.

사후 병합 해제 사용자 종료

병합 해제 프로세스 후에 MDM Hub에서 사후 병합 해제 사용자 종료를 호출합니다.

병합 해제 프로세스 후에 사후 병합 해제 사용자 종료를 사용하여 사용자 지정 처리를 수행합니다.

사후 병합 해제 사용자 종료 인터페이스

사후 병합 해제 사용자 종료를 구현할 때 적절한 인터페이스 이름, 메서드 및 매개 변수를 사용합니다.

인터페이스 이름

사후 병합 해제 사용자 종료에서는 다음과 같은 정규화된 인터페이스 이름을 사용합니다.

```
com.informatica.mdm.userexit.PostUnmergeUserExit
```

메서드

사후 병합 해제 사용자 종료에서는 다음과 같은 메서드를 사용합니다.

```
void processUserExit(UserExitContext userExitContext, Set<PostUnmergeResponse> responses) throws Exception;
```

매개 변수

사후 병합 해제 사용자 종료에서는 다음과 같은 매개 변수를 사용합니다.

boTableName

병합 해제된 레코드가 포함된 기본 개체 테이블의 이름입니다. `PostUnmergeResponse`에서 블록의 각 레코드에 대해 전달했습니다.

rowidObject

복구된 기본 개체 레코드의 행 ID입니다. `PostUnmergeResponse`에서 블록의 각 레코드에 대해 전달했습니다.

userExitContext

매개 변수를 사용자 종료에 전달합니다.

태스크 관리 사용자 종료

태스크 관리를 위해 `AssignTasks` 사용자 종료 및 `GetAssignableUsersForTask` 사용자 종료를 사용할 수 있습니다.

AssignTasks 사용자 종료 인터페이스

`AssignTasks` 사용자 종료를 구현할 때 적절한 인터페이스 이름, 메서드 및 매개 변수를 사용합니다.

인터페이스 이름

`AssignTasks` 사용자 종료에서는 다음과 같은 정규화된 인터페이스 이름을 사용합니다.

`com.informatica.mdm.userexit.AssignTasksUserExit`

메서드

`AssignTasks` 사용자 종료에서는 다음과 같은 메서드를 사용합니다.

`void processAssignTasks(UserExitContext userExitContext, int maxTasks)`

매개 변수

`AssignTasks` 사용자 종료에서는 다음과 같은 매개 변수를 사용합니다.

userExitContext

매개 변수를 사용자 종료에 전달합니다.

maxTasks

각 사용자에게 할당할 최대 태스크 수입니다.

GetAssignableUsersForTask 사용자 종료 인터페이스

GetAssignableUsersForTask 사용자 종료를 구현할 때 적절한 인터페이스 이름, 메서드 및 매개 변수를 사용합니다.

인터페이스 이름

GetAssignableUsersForTask 사용자 종료에서는 다음과 같은 정규화된 인터페이스 이름을 사용합니다.

```
com.informatica.mdm.userexit.GetAssignableUsersForTaskUserExit
```

메서드

GetAssignableUsersForTask 사용자 종료에서는 다음과 같은 메서드를 사용합니다.

```
public List<String> processGetAssignableUsersForTask(String taskType, String rowidSubjectArea,
UserExitContext userExitContext)
```

매개 변수

GetAssignableUsersForTask 사용자 종료에서는 다음과 같은 매개 변수를 사용합니다.

rowidSubjectArea

제목 영역의 행 ID입니다.

taskType

태스크의 유형입니다.

userExitContext

매개 변수를 사용자 종료에 전달합니다.

사용자 종료 내에서 서비스 통합 프레임워크 API 사용

SIF(서비스 통합 프레임워크) API를 호출하는 사용자 종료를 생성할 수 있습니다. 사용자 종료에서는 사용자 종료 코드에 구현된 SIF API를 호출하는 SIF 클라이언트를 사용합니다.

SIF 클라이언트를 생성하려면 `mdm-ue.jar` 파일에 포함된 `UserExitSifClient` Java 클래스를 사용합니다. 사용자 종료 구현 내에서만 SIF 클라이언트를 사용하여 SIF API를 호출할 수 있습니다. 데이터베이스 연결 정보를 입력 매개 변수로 사용합니다.

사용자 종료와 SIF API 호출은 동일한 데이터베이스 연결을 사용하므로 동일한 트랜잭션에 포함됩니다. 사용자 종료에서 SIF API를 호출할 때 이 프로세스는 트랜잭션이 되므로 아직 커밋되지 않은 변경 내용을 볼 수 있습니다. 오류가 발생할 경우에는 전체 프로세스를 롤백할 수 있습니다.

서비스 통합 프레임워크 API를 호출하기 위한 사용자 종료 생성

사용자 지정 코드를 추가하여 SIF(서비스 통합 프레임워크) API를 호출하는 사용자 종료를 생성할 수 있습니다. SIF API를 호출할 수 있는 SIF 클라이언트를 생성하려면 코드를 추가해야 합니다.

1. Eclipse와 같은 Java 통합 개발 환경에서 Java 프로젝트를 생성합니다.
2. Java 프로젝트에 다음 MDM Hub JAR 파일을 추가합니다.
 - `mdm-ue.jar`
 - `siperian-api.jar`

- log4j-1.2.16.jar

jar 파일은 다음 디렉터리에 있습니다.

UNIX의 경우. <infamdm_install_dir>/hub/server/lib

Windows의 경우. <infamdm_install_dir>\hub\server\lib

3. mdm-ue.jar 파일에 사용자 종료 인터페이스를 구현하는 사용자 종료의 Java 클래스를 생성합니다.
4. SIF API를 호출하는 사용자 지정 코드를 추가합니다.
 - a. mdm-ue.jar 파일에 있는 UserExitSifClient 클래스 파일을 사용하여 사용자 종료 클라이언트를 생성합니다.
 - b. 호출해야 하는 SIF API 요청을 정의합니다.
 - c. 사용자 종료 SIF 클라이언트를 사용하여 SIF API를 호출하는 코드를 지정합니다.
5. Java 클래스를 컴파일하고 해당 클래스 파일을 사용자 지정 사용자 종료 JAR 파일로 패키징합니다.
6. Hub 콘솔의 **사용자 개체 레지스트리** 도구를 사용하여 사용자 지정 사용자 종료 JAR 파일을 MDM Hub에 업로드합니다.

사용자 종료 예

조직에서 레코드에 대해 유사 항목 일치치를 수행해야 한다고 가정합니다. 유사 항목 일치치를 수행하기 전에 일치 토큰을 생성한 후 이를 기본 개체와 연결된 일치 키 테이블에 저장해야 합니다.

일치 토큰을 생성하려면 **Tokenize API**를 호출해야 합니다. **Tokenize API**를 호출하는 사용자 종료에서 유사 항목 일치치를 수행해야 하는 레코드에 대해 일치 토큰을 생성하도록 구성할 수 있습니다.

다음 샘플 사용자 종료 코드에서는 SIF 클라이언트를 사용하여 **Tokenize API**를 호출함으로써 레코드에 대한 일치 토큰을 생성합니다.

```
private String ORS_ID = "orclmain-MDM_SAMPLE";
private UserExitSifClient sifClient;

@Override
public void processUserExit(UserExitContext arg0, ActionType arg1,
    Map<String, Object> arg2, Map<String, Object> arg3,
    List<Map<String, Object>> arg4) throws Exception {

    // Begin custom user exit code ...
    log.info("##### - Starting PostLoad User Exit");

    // Get the ROWID_OBJECT value of the record that was loaded.
    String rowidObject = (String) arg3.get("ROWID_OBJECT");

    // Initialize user exit SIF Client.
    sifClient = new UserExitSifClient(arg0.getDBConnection(), ORS_ID);

    // Tokenize the record that was loaded.
    TokenizeRequest r = new TokenizeRequest();

    // Specify a user that should call the SIF API
    r.setUsername("userExitSifUser");

    r.setOrsId(ORS_ID);

    // Specify the base object that must be tokenized.
    r.setSiperianObjectUid(SiperianObjectType.BASE_OBJECT.makeUid("C_PARTY"));

    // Specify the record that must be tokenized.
    RecordKey rkey=new RecordKey();
    rkey.setRowid(rowidObject);
    r.setRecordKey(rkey);
    r.setActionType("UPDATE");
```

```
// Call Tokenize SIF API.
TokenizeResponse response = (TokenizeResponse)sifClient.process(r);

// Print out response message
log.info("TokenizeReponse=" + response.getMessage());

// When making subsequent SIF API requests, SIF client can be reused.
// It does not need to be initialized again.

} // End processUserExit
```

서비스 통합 프레임워크 API

사용자 종료 JAR 파일에 포함된 UserExitSifClient Java 클래스를 사용하여 SIF(서비스 통합 프레임워크) API를 호출할 수 있습니다.

다음 표에서는 사용자 종료가 호출할 수 있는 SIF API에 대해 설명합니다.

SIF API	설명
CanUnmergeRecords	지정된 교차 참조 레코드를 통합된 기본 개체 레코드에서 병합 해제할 수 있는지 여부를 결정합니다.
CleansePut	기본 개체를 삽입하거나 키로 식별된 단일 레코드로 업데이트합니다.
Cleanse	요청에서 지정된 입력 레코드를 선택된 MDM Hub 정리 함수에 지정된 출력 형식으로 변환합니다.
CleanTable	연산 참조 저장소 및 연결된 모든 테이블에서 데이터를 제거합니다.
CreateTask	태스크를 생성합니다.
Delete	기본 개체에서 레코드를 제거합니다. deleteBORecord 플래그를 지정하면 Delete API는 소스 키와 시스템 이름만 지정된 경우에도 기본 개체 레코드를 삭제합니다.
FlagForAutomerger	일치 테이블에서 레코드에 자동 병합 플래그를 지정합니다. 일치 테이블에서 레코드의 자동 병합 표시기 값이 1이면 다음에 자동 병합 프로세스가 실행될 때 레코드가 병합됩니다. 레코드가 없으면 FlagForAutomerger API는 일치 테이블에 레코드를 생성하고 자동 병합 표시기를 1로 설정합니다.
GetAssignableUsersForTasks	태스크를 할당할 수 있는 사용자의 목록을 검색합니다.
GetEffectivePeriods	기본 개체 레코드에 대한 집계 유효 기간을 검색합니다.
GetMatchedRecords	레코드에 대한 일치 후보를 검색합니다.
GetMergeHistory	기본 개체 레코드의 병합 기록을 나타내는 트리를 검색합니다.
Get	알려진 키를 사용하여 패키지에서 단일 레코드를 검색합니다.
GetSearchResults	검색 쿼리에서 찾은 레코드 수가 검색 쿼리에서 반환되기를 위한 레코드 수보다 많을 경우 추가 데이터를 검색합니다.
GetTasks	태스크 및 태스크 세부 정보의 목록을 검색합니다.
GetTrustScore	기본 개체 레코드의 열에 대한 현재 트러스트 점수를 검색합니다.

SIF API	설명
GetXrefForEffectiveDate	유효 날짜에 대한 여러 교차 참조 레코드를 검색합니다.
Merge	동일한 개체를 나타내는 두 개의 기본 개체 레코드를 병합합니다.
MultiMerge	동일한 개체를 나타내는 여러 개의 기본 개체 레코드를 병합합니다. 병합된 레코드에 대한 필드 수준 재정의를 지정할 수 있습니다. 중요: 사후 로드 및 사후 병합 사용자 종료에 이 API를 사용할 경우 MDM Hub는 반복적 호출을 생성합니다. MultiMerge API를 호출하려면 다른 사용자 종료를 사용합니다.
PreviewBvt	지정된 레코드 집합이 병합되거나 보류 중인 업데이트가 발생할 때 기본 개체 레코드가 어떻게 나타나는지를 보여 주는 미리 보기를 생성합니다.
PromotePendingXrefs	교차 참조 레코드를 승격하거나 교차 참조 레코드에 승격 플래그를 지정합니다.
Put	키로 식별된 단일 레코드를 기본 개체에 삽입하거나 업데이트합니다. 중요: 사후 로드 사용자 종료에 이 API를 사용할 경우 Informatica MDM Hub는 반복적 호출을 생성합니다. 사후 로드 사용자 종료를 바이패스하려면 PUT API의 BypassPostLoadUE 매개 변수를 true로 설정합니다.
RemoveMatchedRecords	일치 테이블에서 기본 개체 레코드와 연결된 일치 항목을 제거합니다.
Restore	교차 참조 레코드를 복원합니다.
SearchHmQuery	계층 관리자 항목 및 관계를 검색합니다.
SearchMatch	일치 열 및 규칙 정의를 기반으로 패키지의 레코드를 검색합니다.
SearchQuery	패키지에서 지정된 조건을 충족하는 레코드 집합을 검색합니다.
SetRecordState	지정한 키로 식별되는 기본 개체 레코드에 대해 통합 표시기를 설정합니다.
Tokenize	MDM Hub가 업데이트하거나 삽입하는 기본 개체 레코드에 대해 일치 토큰을 생성합니다.
Unmerge	기본 개체 레코드를 병합 해제합니다. 중요: 사전 병합 해제 사용자 종료에 이 API를 사용할 경우 MDM Hub는 반복적 호출을 생성합니다. Unmerge API를 호출하려면 다른 사용자 종료를 사용합니다.
UpdateMatchRecord	일치 테이블에서 레코드를 생성하거나 업데이트합니다.
UpdateTask	태스크를 업데이트합니다.

사용자 종료 구현 지침

사용자 종료는 MDM Hub 성능이 불필요하게 저하되지 않는 방식으로 구현합니다.

사용자 종료를 구현할 경우 다음 지침을 고려하십시오.

사용자 종료가 성능에 미치는 영향 고려

사용자 종료를 구현하면 MDM Hub 성능에 영향을 주게 됩니다. 원하는 작업을 수행하기 위해 사용자 종료를 사용해야 하는지 아니면 사용자 종료를 사용하지 않고도 동일한 작업을 비동기식으로 수행할 수 있는지를 고려합니다.

예를 들어 데이터를 로드할 때 보류 중인 레코드에 대한 태스크를 작성하고자 할 수 있습니다. 이 경우 사용자 종료를 사용하여 태스크를 작성할 수도 있지만 반드시 사용자 종료를 사용할 필요는 없습니다. 사용자 종료를 사용하면 프로세스에 병목 현상이 발생하고 불필요하게 성능이 저하됩니다. 이러한 태스크는 프로세스의 해당 시점에서 중요하지 않기 때문에 나중에 작성해도 됩니다.

데이터베이스 연결을 사용하여 데이터베이스 쿼리 또는 업데이트

Java 사용자 종료 컨텍스트는 데이터베이스 연결을 제공합니다. JDBC를 직접 호출하거나 SIF API를 호출하여 데이터베이스를 쿼리하거나 업데이트하려면 이 데이터베이스 연결을 사용합니다.

사용자 종료에서 SIF API를 호출할 경우 일괄 처리 성능에 미치는 영향 고려

사용자 종료에서 SIF API를 호출할 경우 일괄 처리 성능에 미치는 영향을 고려해야 합니다.

가능한 경우 SIF API 사용

동등한 SIF API를 사용할 수 있는 경우에는 MDM Hub 테이블을 직접 사용하지 마십시오.

사용자 종료에서 변경한 내용을 명시적으로 커밋하거나 롤백하면 안 됨

데이터베이스 연결을 사용하여 SIF API를 호출하거나 JDBC를 직접 호출할 경우 해당 호출은 사용자 종료 호출자와 동일한 트랜잭션에 참여합니다. 사용자 종료 호출자는 트랜잭션의 변경 내용을 유지 관리합니다. 따라서 사용자 종료에 전달된 연결에서 명시적으로 커밋 또는 롤백을 호출하면 안 됩니다.

MDM Hub에서는 사용자 종료에 대해 인증 또는 권한 부여가 수행되지 않음

사용자 종료는 MDM Hub 개체에 대한 전체 액세스 권한으로 실행됩니다. 사용자 종료는 컨텍스트에 관계없이 MDM Hub 개체에 대한 전체 액세스 권한으로 실행됩니다. 사용자 종료 SIF 클라이언트를 사용하여 지원되는 SIF API의 하위 집합을 호출할 수 있습니다. 지원되는 SIF API만 사용자 종료 호출자와 동일한 트랜잭션에 참여할 수 있습니다. 사용자 종료 SIF 클라이언트를 사용하여 지원되지 않는 SIF API에 액세스할 수는 없습니다.

파트 VI: 응용 프로그램 액세스 구성

이 파트에 포함된 장:

- [ORS 관련 API, 586](#)
- [ORS 관련 메시지 스키마, 591](#)
- [등록된 사용자 지정 코드 보기, 595](#)
- [Informatica MDM Hub 서비스 및 이벤트 감사, 599](#)

제 29 장

ORS 관련 API

이 장에 포함된 항목:

- [ORS 관련 API 개요, 586](#)
- [성능 고려 사항, 587](#)
- [지원되는 리포지토리 개체, 587](#)
- [보관 테이블, 588](#)
- [ORS 관련 SIF API 생성 및 배포, 589](#)
- [ORS 관련 SIF API 이름 바꾸기, 589](#)
- [ORS 관련 클라이언트 JAR 파일 다운로드, 589](#)
- [ORS 관련 클라이언트 JAR 파일과 SIF SDK 사용, 590](#)
- [ORS 관련 SIF API 제거, 590](#)

ORS 관련 API 개요

연산 참조 저장소의 기본 개체, 패키지 및 정리 함수와 같은 특정 개체에 대한 API를 생성할 수 있습니다. Hub 콘솔의 SIF 관리자 도구를 사용하여 ORS 관련 API를 생성할 수 있습니다.

ORS 관련 API를 생성하기 전에 연산 참조 저장소의 기본 개체와 패키지를 구성합니다. ORS 관련 API는 클라이언트 JAR 파일을 통해 **SiperianClient** 및 웹 서비스로 사용할 수 있습니다. ORS 관련 API를 생성할 때 MDM Hub는 버전 ID를 ORS 관련 API 클라이언트 JAR 파일과 연결합니다.

ORS 관련 API는 특정 연산 참조 저장소 개체에서 작동합니다. 예를 들어 일반 API는 지정한 데이터베이스 레코드에 데이터를 배치할 수 있습니다. ORS 관련 API는 이름 및 전자 메일 주소와 동일한 데이터를 식별한 후 연산 참조 저장소에 정의된 대로 해당 필드를 고객 레코드에 배치합니다.

SIF(서비스 통합 프레임워크) SDK를 통해 ORS 관련 API를 사용하거나 ORS 관련 API를 SOAP 웹 서비스로 사용합니다. SIF SDK를 사용하는 경우 Java 개발 키트 및 Apache Jakarta Ant 빌드 시스템을 설치해야 합니다.

성능 고려 사항

ORS 관련 API 생성의 성능은 선택하는 개체의 수에 따라 다릅니다. 최적의 성능을 위해 API를 생성해야 하는 개체만 선택합니다.

참고: 연결된 SIF API Javadoc에 대한 힙 공간이 고갈되지 않도록 하려면 힙 크기를 늘려야 할 수 있습니다. 기본 힙 크기는 256M입니다. `cmxserver.properties` 파일의 `sif.jvm.heap.size` 매개 변수를 설정하여 이 기본값을 재정의할 수도 있습니다.

지원되는 리포지토리 개체

보호되는 특정 리포지토리 개체에 대해 SIF API를 생성할 수 있습니다. 개체를 보호하려면 Hub 콘솔에서 보안 리소스 도구를 사용합니다.

다음 리포지토리 개체에 대해 ORS 관련 SIF API를 생성할 수 있습니다.

- 기본 개체
- 패키지
- 매핑
- 정리 함수
- 일치 열
- 일치 규칙 집합

참고: 일치 열 및 일치 규칙 집합에 대해 API를 생성하는 경우에는 연결된 패키지를 선택해야 합니다. 연결된 패키지를 선택하지 않으면 MDM Hub이 일치 열 및 일치 규칙 집합에 대해 ORS 관련 API를 생성하지 않습니다.

ORS 관련 SIF API 속성

Hub 콘솔에서 SIF 관리자 유틸리티를 사용하여 ORS 관련 SIF API의 속성을 구성합니다.

ORS 관련 SIF API의 다음 속성을 구성할 수 있습니다.

논리 이름

ORS의 논리 이름입니다.

논리 이름은 편집할 수 있습니다. 편집한 이름이 고유한지 확인합니다. ORS 관련 SIF API의 논리 이름을 편집한 후에는 ORS 관련 SIF API를 다시 생성하고 배포합니다.

Java 이름

ORS의 Java 이름입니다.

ORS 관련 SIF API의 클라이언트 JAR 파일 이름에는 Java 이름이 들어갑니다. 논리 이름을 편집하여 Java 이름을 변경합니다. 편집한 논리 이름이 고유한지 확인합니다.

WSDL URL

ORS 관련 SIF API 배포 시 MDM Hub이 생성하는 WSDL 파일의 URL입니다.

API 생성 시간

ORS 관련 SIF API를 생성한 날짜 및 시간입니다. 형식은 mm/dd/yy hh:mm tt입니다.

버전 ID

MDM Hub이 생성 및 배포하는 ORS 관련 SIF API의 고유 ID입니다.

MDM Hub은 다음 요소에서 버전 ID를 사용합니다.

- 엔터프라이즈 관리자 도구의 **환경 보고서** 및 **ORS 데이터베이스** 탭의 속성입니다.
- 클라이언트 JAR 파일의 이름입니다.
- 클라이언트 JAR의 MANIFEST.MF 파일입니다.

리포지토리 개체 상태

리포지토리 개체의 상태는 리포지토리 개체에 대해 MDM Hub이 ORS 관련 SIF API를 생성 및 배포할 수 있는지를 결정합니다.

Hub 콘솔의 SIF 관리자 유틸리티에서는 **선택한 비동기 개체** 테이블의 **상태** 열에 리포지토리 개체의 상태가 표시됩니다. 개체를 업데이트한 후 개체의 상태를 새로 고칠 수 있습니다. 개체의 상태를 새로 고치려면 SIF 관리자 유틸리티의 **SIF API 관리자** 탭에서 **개체 상태 새로 고침**을 클릭합니다.

리포지토리 개체의 상태는 다음 중 하나일 수 있습니다.

새로 작성

개체가 새 개체이며 해당 개체에 대해 SIF API가 생성되고 배포되지 않았음을 나타냅니다. 개체에 대한 ORS 관련 SIF API를 생성 및 배포하면 상태가 최신 상태로 변경됩니다.

최신 상태

SIF API 생성 이후 개체가 변경되지 않았으며 개체가 최신 상태임을 나타냅니다.

동기화되지 않음

SIF API 생성 이후 개체가 변경되었으며 개체가 비동기임을 나타냅니다. SIF API를 다시 생성하여 상태를 최신 상태로 변경합니다.

안전하지 않음

개체가 보안 상태가 아니며 해당 개체에 대해 SIF API를 생성할 수 없음을 나타냅니다. Hub 콘솔의 보안 리소스 도구에서 보안되지 않음 상태인 개체는 개인 리소스로 표시됩니다.

삭제됨

개체가 삭제되었으며 해당 개체에 대해 API를 생성할 수 없음을 나타냅니다. Hub 콘솔에서 모델 작업 영역의 도구를 사용하여 개체를 삭제한 경우 개체의 상태는 삭제됨이 됩니다. ORS 관련 SIF API를 생성 및 배포하면 삭제됨 상태의 개체가 제거됩니다.

보관 테이블

CMX_DATA 테이블스페이스에 저장된 C_REPAR_SIF_ORS_CONFIG 테이블에서 생성한 모든 ORS 관련 SIF API를 보관할 수 있습니다. ORS 관련 SIF API의 버전 ID를 사용하여 보관을 식별합니다.

보관 테이블의 레코드에는 blob 데이터가 포함되어 있습니다. 이 데이터의 크기는 문자 기반 레코드보다 클 수 있으며 시간에 따라 늘어날 수 있습니다. 데이터베이스 관리자는 정기적으로 보관 테이블을 보관하거나 제거하여 데이터베이스를 정리할 수 있습니다.

ORS 관련 SIF API 생성 및 배포

Hub 콘솔의 SIF 관리자 유틸리티를 사용하여 보안 리포지토리 개체에 대한 ORS 관련 SIF API를 생성 및 배포합니다. 특정 리포지토리 개체를 선택하여 ORS 관련 SIF API를 생성할 수 있습니다.

ORS 관련 SIF API를 생성하고 배포하려면 MDM Hub이 응용 프로그램 서버가 설치된 컴퓨터의 Java 컴파일러에 액세스해야 합니다. ORS 관련 SIF API를 생성 및 배포하기 전에 ORS의 기본 개체와 패키지를 구성했는지 확인합니다.

1. Hub 콘솔을 시작하고 ORS에 연결합니다.
2. 유틸리티 작업 영역을 확장한 후 **SIF 관리자**를 클릭합니다.
SIF 관리자 유틸리티가 나타납니다.
3. 쓰기 잠금을 획득하려면 **쓰기 잠금** 메뉴에서 **잠금 획득**을 클릭합니다.
4. 리포지토리 개체의 상태를 업데이트하려면 **SIF API 관리자** 탭에서 **개체 상태 새로 고침**을 클릭합니다.
리포지토리 개체의 상태는 **선택한 비동기 개체** 테이블에 업데이트됩니다.
5. SIF API를 생성 및 배포할 리포지토리 개체를 선택합니다.
6. **ORS 관련 SIF API 생성 및 배포**를 클릭합니다.
ORS 관련 클라이언트 JAR 파일 및 WSDL이 생성됩니다.

ORS 관련 SIF API 이름 바꾸기

Hub 콘솔에서 SIF 관리자 유틸리티를 사용하여 ORS 관련 SIF API의 이름을 바꿉니다.

1. SIF 관리자 유틸리티의 **쓰기 잠금** 메뉴에서 **잠금 획득**을 클릭합니다.
2. **SIF API 관리자** 탭의 **논리 이름** 상자에서 **편집** 단추를 클릭하고 논리 이름을 편집합니다.
3. **허용** 단추를 클릭합니다.
논리 이름이 저장되고 Java 이름이 논리 이름과 일치하도록 업데이트됩니다.
4. **ORS 관련 SIF API 생성 및 배포**를 클릭합니다.
ORS 관련 클라이언트 JAR 파일 및 WSDL 파일이 다시 생성됩니다.

ORS 관련 클라이언트 JAR 파일 다운로드

특정 리포지토리 개체에 대해 ORS 관련 SIF API를 생성한 후 SiperianClient 클래스 및 SIF API 참조 설명서가 포함된 클라이언트 JAR 파일을 다운로드합니다.

1. SIF 관리자 유틸리티의 **SIF API 관리자** 탭에서 **클라이언트 JAR 파일 다운로드**를 클릭합니다.
클라이언트 JAR 파일을 저장할 디렉토리를 선택하십시오 대화 상자가 나타납니다.
2. 클라이언트 JAR 파일을 저장할 디렉토리를 선택하고 **저장**을 클릭합니다.
<Java Name>Client_<Version ID>.jar 파일이 다운로드되며 선택한 디렉토리에 저장됩니다.

ORS 관련 클라이언트 JAR 파일과 SIF SDK 사용

ORS 관련 클라이언트 JAR 파일과 SIF SDK를 사용할 수 있습니다.

1. IDE(통합 개발 환경)를 사용하고 있고 웹 서비스를 작성할 프로젝트 파일이 있는 경우 다운로드한 클라이언트 JAR 파일을 빌드 클래스 경로에 추가합니다.
2. 다음 디렉터리에서 `build.xml` 파일을 엽니다.
 - Windows의 경우. <리소스 키트 설치 디렉터리>\hub\resourcekit\sdk\sifsdk
 - UNIX의 경우. <리소스 키트 설치 디렉터리>/hub/resourcekit/sdk/sifsdk
3. `build.xml` 파일을 사용자 지정하여 `build_war` 매크로에 다운로드한 클라이언트 JAR 파일이 들어가도록 합니다.
4. `build.xml` 파일을 저장하고 닫습니다.

ORS 관련 SIF API 제거

ORS 관련 SIF API를 제거하려면 Hub 콘솔에서 SIF 관리자 유틸리티를 사용합니다.

1. SIF 관리자 유틸리티의 **쓰기 잠금** 메뉴에서 **잠금 획득**을 클릭합니다.
2. **SIF API 관리자** 탭에서 **ORS 관련 API 제거**를 클릭합니다.
ORS 관련 클라이언트 JAR 파일 및 WSDL이 제거됩니다.

제 30 장

ORS 관련 메시지 스키마

이 장에 포함된 항목:

- [ORS 관련 메시지 스키마 개요, 591](#)
- [JMS 이벤트 스키마 관리자 도구 정보, 591](#)
- [JMS 이벤트 스키마 관리자 도구 시작, 592](#)
- [SIF 관리자 도구 시작, 592](#)
- [ORS 관련 스키마 생성 및 배포, 593](#)

ORS 관련 메시지 스키마 개요

Hub 콘솔의 SIF 관리자 도구를 사용하여 ORS 관련 JMS 이벤트 메시지 스키마를 생성할 수 있습니다. SIF 관리자 도구의 동기화되지 않은 개체 섹션에서는 메시지 스키마를 생성할 수 있는 개체를 표시합니다.

ORS 관련 JMS 이벤트 메시지 스키마 생성의 성능은 MDM Hub가 ORS 관련 JMS 이벤트 메시지 스키마를 생성 및 배포하는 데 사용하는 개체의 수에 따라 다릅니다.

ORS 관련 JMS 이벤트 메시지 스키마는 URL을 통해 다운로드하거나 액세스할 수 있는 XSD 파일로 사용할 수 있습니다. 레거시 이벤트 XML 메시지 스키마를 사용하려는 경우 ORS 관련 JMS 이벤트 메시지 스키마를 생성할 필요가 없습니다. SIF SDK를 사용하는 경우 Java 개발 키트 및 Apache Jakarta Ant 빌드 시스템을 설치해야 합니다.

JMS 이벤트 스키마 관리자 도구 정보

JMS 이벤트 스키마 관리자는 Hub에서 JMS 메시지를 생성하는 데 사용되는 메시지 구조를 정의하는 XML 스키마를 사용합니다.

이 XML 스키마는 Informatica MDM Hub 리소스 키트의 일부로 포함되어 있습니다. ORS 관련 스키마는 URL을 통해 사용하거나 파일로 다운로드할 수 있습니다.

참고: JMS 이벤트 스키마를 생성하려면 하나 이상의 보안 패키지나 원격 패키지가 정의되어 있어야 합니다.

중요: 두 개의 데이터베이스에서 동일한 스키마(예: CMX_ORS)를 사용하는 경우 구성이 처음 저장될 때 JMS 이벤트의 논리 이름(스키마 이름과 동일함)이 중복되게 됩니다. 따라서 초기 논리 이름으로는 스키마 이름 대신 고유한 데이터베이스 표시 이름을 사용하여 SIF API와 연관되도록 해야 합니다. 스키마를 생성하기 전에 논리 이름을 변경해야 합니다.

또한 각 ORS에는 공통 XSD 파일(siperian-mrm-events.xsd)의 요소를 사용하는 ORS 관련 XSD 파일이 있습니다. ORS 관련 XSD의 이름은 <ors-name>-siperian-mrm-event.xsd로 지정됩니다. XSD는 스키마의 각 패키지 및 원격 패키지에 대한 두 개의 개체를 정의합니다.

개체 이름	설명
[packageName]Event	EventMetadata 및 [packageName] 유형의 요소를 포함하는 복합 유형입니다.
[packageName]Record	패키지 및 해당 필드를 나타내는 복합 유형입니다. SipMetadata 유형의 요소도 포함되어 있습니다. 이 복합 유형은 Informatica MDM Hub SIF(서비스 통합 프레임워크)에 정의된 패키지 레코드 구조와 유사합니다.

참고: 레거시 XML 이벤트 메시지 개체를 사용할 경우에는 ORS 관련 메시지 개체를 생성하지 않아도 됩니다.

JMS 이벤트 스키마 관리자 도구 시작

JMS 이벤트 스키마 관리자 도구를 시작하려면

1. Hub 콘솔에서 연산 참조 저장소(연산 참조 저장소)에 연결합니다.
2. Informatica 유틸리티 작업 영역을 확장한 후 **SIF 관리자**를 클릭합니다.
3. **JMS 이벤트 스키마 관리자** 탭을 클릭합니다.

Hub 콘솔에 JMS 이벤트 스키마 관리자 도구가 표시됩니다.

JMS 이벤트 스키마 관리자 도구에는 다음과 같은 영역이 표시됩니다.

영역	설명
JMS ORS 관련 이벤트 메시지 스키마	<p>ORS에 대한 이벤트 메시지 스키마를 표시합니다.</p> <p>이 기능을 사용하여 현재 ORS에 대해 ORS 관련 JMS 이벤트 메시지를 생성 및 배포할 수 있습니다. 논리 이름은 배포 구성 요소의 이름을 지정하는 데 사용됩니다. 스키마는 URL을 사용하여 다운로드하거나 액세스할 수 있습니다.</p> <p>참고: 레거시 XML 이벤트 메시지 개체를 사용할 경우에는 ORS 관련 메시지 개체를 생성하지 않아도 됩니다.</p>
동기화되지 않은 개체	생성된 API와 동기화되지 않은, 스키마의 데이터베이스 개체를 표시합니다.

SIF 관리자 도구 시작

Hub 콘솔의 유틸리티 작업 영역에서 SIF 관리자 도구를 시작할 수 있습니다.

1. Hub 콘솔을 시작하고 ORS(연산 참조 저장소)에 연결합니다.
2. 유틸리티 작업 영역을 확장한 후 **SIF 관리자**를 클릭합니다.

SIF 관리자 도구가 나타납니다.

ORS 관련 스키마 생성 및 배포

Java 소프트웨어 개발 키트(SDK)의 `tools.jar`에 컴파일러가 들어 있습니다.

이 작업을 수행하려면 응용 프로그램 서버 시스템의 Java 컴파일러에 액세스해야 합니다. Java 소프트웨어 개발 키트(SDK)의 `tools.jar`에 컴파일러가 들어 있습니다. JRE(Java Runtime Environment)에는 컴파일러가 포함되어 있지 않습니다. SDK를 사용할 수 없는 경우 응용 프로그램 서버의 클래스 경로에 `tools.jar` 파일을 추가해야 합니다.

중요: 두 개의 데이터베이스에서 동일한 스키마(예: `CMX_ORS`)를 사용하는 경우 구성이 처음 저장될 때 JMS 이벤트의 논리 이름(스키마 이름과 동일함)이 중복되게 됩니다. 따라서 초기 논리 이름으로는 스키마 이름 대신 고유한 데이터베이스 표시 이름을 사용하여 SIF API와 일관되도록 해야 합니다. 스키마를 생성하기 전에 논리 이름을 변경해야 합니다.

다음 절차에서는 ORS의 기본 개체, 패키지 및 매핑을 이미 구성했다고 가정합니다. 나중에 이러한 파일을 변경하는 경우 ORS 관련 스키마를 다시 생성하십시오.

또한, JMS 이벤트 스키마를 생성하려면 하나 이상의 보안 패키지나 원격 패키지가 필요합니다.

ORS 관련 스키마를 생성 및 배포하려면

1. JMS 이벤트 스키마 관리자를 시작합니다.
Hub 콘솔에 JMS 이벤트 스키마 관리자 도구가 표시됩니다.
2. 이벤트 스키마의 논리 이름 필드에 값을 입력합니다.
스키마를 변경하기 위해서는 쓰기 잠금이 필요합니다.
3. **ORS 관련 스키마 생성 및 배포**를 클릭합니다.

참고: 스키마를 생성하려면 하나 이상의 보안 패키지나 원격 패키지가 구성되어 있어야 합니다. 생성할 보안 개체가 없는 경우 Informatica MDM Hub에서 런타임 오류 메시지가 생성됩니다.

XSD 파일 다운로드

XSD 파일은 XML 파일의 구조를 정의하며 XML 파일의 유효성 검사에도 사용됩니다.

예를 들어 XML 파일에 XSD에 대한 참조가 포함된 경우 XML 유효성 검사 도구를 사용하여 XML의 태그가 XSD에 정의된 정의를 따르는지 확인할 수 있습니다.

XSD 파일을 다운로드하려면

1. JMS 이벤트 스키마 관리자를 시작합니다.
Hub 콘솔에 JMS 이벤트 스키마 관리자 도구가 표시됩니다.
2. 쓰기 잠금을 획득합니다.
스키마를 변경하기 위해서는 쓰기 잠금이 필요합니다.
3. **XSD 파일 다운로드**를 클릭합니다.
또는 스키마 URL에 지정된 URL을 사용하여 XSD 파일에 액세스할 수 있습니다.

동기화되지 않은 개체 찾기

시스템의 변경 내용을 반영하기 위해 이벤트 스키마를 다시 생성해야 하는지 여부를 결정하려면 동기화되지 않은 개체 찾기를 사용합니다.

JMS 이벤트 스키마 관리자에 마지막 스키마 생성 이후 변경된 패키지 및 원격 패키지 목록이 표시됩니다.

참고: 동기화되지 않은 개체 기능에서는 생성된 API를 스키마의 데이터베이스 개체와 비교하므로 동기화되지 않은 개체를 찾으려면 두 항목이 모두 있어야 합니다.

동기화되지 않은 개체를 찾으려면

1. JMS 이벤트 스키마 관리자를 시작합니다.

Hub 콘솔에 JMS 이벤트 스키마 관리자 도구가 표시됩니다.

2. 쓰기 잠금을 획득합니다.

스키마를 변경하기 위해서는 쓰기 잠금이 필요합니다.

3. 동기화되지 않은 개체 찾기를 클릭합니다.

JMS 이벤트 스키마 관리자의 아래쪽 패널에 동기화되지 않은 모든 개체가 표시됩니다.

참고: 동기화되지 않은 개체의 영향을 평가한 후 스키마 재생성 여부를 결정할 수 있습니다. 일반적으로 Hub와 상호 작용하는 외부 구성 요소는 생성된 스키마의 특정 버전과 함께 작동하도록 작성됩니다. 스키마를 다시 생성한 경우 이러한 외부 구성 요소가 더 이상 작동하지 않을 수 있습니다.

JMS 이벤트 스키마 관리자가 동기화되지 않은 개체를 반환하는 경우 ORS 관련 스키마 생성 및 배포를 클릭하여 이벤트 스키마를 다시 생성합니다.

동기화되지 않은 개체에 대한 자동 검색

Informatica MDM Hub에서 동기화되지 않은 개체를 정기적으로 검색하고 필요할 경우 스키마를 다시 생성하도록 구성할 수 있습니다.

이 자동 폴링 기능은 지정된 폴링 간격(밀리초)을 자동으로 적용하는 데이터 변경 모니터링 스레드 내에서 작동합니다. 이 시간은 메시지 대기열 도구에서 메시지 확인 간격을 사용하여 지정합니다. 모니터링 스레드가 활성 상태인 경우 이 자동 서비스는 먼저 동기화 확인 간격이 경과되었는지 여부를 확인한 다음, 동기화 확인 간격이 경과되었으면 동기화 확인을 수행하고 필요한 경우 이벤트 스키마를 다시 생성합니다.

Hub에서 동기화되지 않은 개체를 정기적으로 검색하도록 구성하려면

1. JMS 이벤트 스키마 관리자에서 생성할 스키마의 논리 이름을 설정합니다.

참고: 이 단계를 건너뛰면 Hub에서는 스키마 생성을 구성하라는 경고를 서버 로그에 생성합니다.

2. 데이터 변경 모니터링 메시지에 대해 대기열 상태를 활성화합니다.

3. 루트 노드인 "메시지 대기열"을 선택하고 동기화 확인 간격(밀리초)을 설정합니다.

동기화 자동 폴링 기능은 실제로는 메시지 확인 간격의 영향을 받으므로 동기화 확인 간격을 메시지 확인 간격보다 크거나 같은 값으로 설정해야 합니다.

참고: 동기화 확인을 비활성화하려면 동기화 확인 간격을 0으로 설정하면 됩니다.

제 31 장

등록된 사용자 지정 코드 보기

이 장에 포함된 항목:

- [개요, 595](#)
- [사용자 개체, 595](#)
- [사용자 개체 레지스트리 도구 시작, 596](#)
- [사용자 종료 보기, 596](#)
- [사용자 지정 Java 정리 함수 보기, 596](#)
- [사용자 지정 단추 함수 보기, 597](#)

개요

이 장에서는 사용자 개체 레지스트리 도구를 사용하여 등록된 사용자 지정 코드를 보는 방법에 대해 설명합니다.

사용자 개체

사용자 개체는 MDM Hub 기능을 확장하기 위해 Informatica MDM Hub에 등록할 수 있는 사용자 지정 함수입니다.

사용자 개체 레지스트리 도구는 MDM Hub용으로 개발한 다음 여기에 등록하는 사용자 개체를 추적하는 읽기 전용 도구입니다. 사용자 개체 레지스트리에서는 다음과 같은 사용자 개체에 액세스할 수 있습니다.

사용자 종료

미리 정의된 고정 매개 변수 집합을 포함하는 암호화되지 않은 사용자 지정 Java 코드입니다. 사용자 종료는 일괄 작업 동안 특정 시점에서 실행됩니다. 각 기본 개체에 대해 사용자 종료를 다르게 구성할 수 있습니다.

사용자 지정 Java 정리 함수

표준 정리 라이브러리를 보완하기 위해 사용자 지정 논리를 사용하는 Java 정리 함수입니다. 사용자 지정 Java 정리 함수는 Informatica MDM Hub가 데이터베이스에 BLOB(Binary Large Object)으로 저장하는 JAR 파일입니다.

사용자 지정 단추 함수

데이터 관리자, 병합 관리자 및 계층 관리자에 추가 아이콘 및 논리를 제공하는 사용자 인터페이스 함수입니다.

사용자 개체 레지스트리 도구 시작

사용자 개체 레지스트리 도구를 시작하려면

1. Hub 콘솔에서 **ORS**(연산 참조 저장소)에 연결합니다.
2. Informatica 유틸리티 작업 영역을 확장한 후 **사용자 개체 레지스트리**를 클릭합니다.
Hub 콘솔에 사용자 개체 레지스트리 도구가 표시됩니다.

사용자 개체 레지스트리 도구에는 다음 영역이 표시됩니다.

열	설명
등록된 사용자 개체 유형	선택한 ORS에 등록된 사용자 개체의 계층 트리, 다음 범주로 구성됩니다. <ul style="list-style-type: none">- 사용자 종료.- 사용자 지정 Java 정리 함수.- 사용자 지정 단추 함수.
사용자 개체 속성	선택한 사용자 개체의 속성입니다.

사용자 종료 보기

이 섹션에서는 사용자 개체 레지스트리 도구에서 사용자 종료를 보는 방법에 대해 설명합니다.

사용자 종료 정보

사용자 종료는 MDM Hub의 기능을 확장하기 위한 SIF API 프로세스 또는 일괄 처리의 특정 시점에서 실행되는 Java 코드로 구성됩니다.

사용자 종료는 사용자 지정 작업을 사후 로드, 사후 병합 및 사후 일치와 같은 Hub 서버 프로세스와 통합하는 메커니즘을 제공하는 Informatica MDM Hub 백엔드 프로세스에 의해 트리거됩니다.

사용자 종료 보기

사용자 개체 레지스트리 도구에서 Informatica MDM Hub 사용자 종료를 보려면

1. 사용자 개체 레지스트리 도구를 시작합니다.
2. 사용자 개체 목록에서 **사용자 종료**를 선택합니다.
사용자 개체 레지스트리 도구에 사용자 종료가 표시됩니다.

사용자 지정 Java 정리 함수 보기

이 섹션에서는 사용자 개체 레지스트리 도구에서 등록된 사용자 지정 Java 정리 함수를 보는 방법에 대해 설명합니다.

사용자 지정 Java 정리 함수 정보

사용자 개체 레지스트리에서는 사용자 라이브러리가 아니라 **Java** 라이브러리에 추가된 사용자 지정 정리 함수에 대한 세부 정보를 제공합니다.

Informatica MDM Hub에서는 데이터를 정리하는 정리 함수를 작성하여 실행할 수 있습니다. 정리 함수는 표준화 또는 확인 목적으로 레코드의 데이터 값에 적용되는 함수입니다. 예를 들어 데이터에 인사말 열이 있는 경우 정리 함수를 사용하여 나타나는 모든 "Doctor"를 "Dr."로 표준화할 수 있습니다. 정리 함수를 연속적으로 적용할 수도 있고, 단순히 준비 테이블의 열에 출력 값을 할당할 수도 있습니다.

사용자 지정 Java 정리 함수를 등록하는 방법

정리 함수는 **Hub** 콘솔의 정리 함수 도구를 사용하여 구성합니다.

등록된 사용자 지정 Java 정리 함수 보기

사용자 개체 레지스트리 도구에서 등록된 사용자 지정 **Java** 정리 함수를 보려면

1. 사용자 개체 레지스트리 도구를 시작합니다.
2. 사용자 개체 목록에서 **사용자 지정 Java 정리 함수**를 선택합니다.

사용자 개체 레지스트리 도구에 등록된 사용자 지정 **Java** 정리 함수가 표시됩니다.

사용자 지정 단추 함수 보기

이 섹션에서는 사용자 개체 레지스트리 도구에서 등록된 사용자 지정 단추 함수를 보는 방법에 대해 설명합니다.

사용자 지정 단추 함수 정보

Informatica MDM Hub 구현에서는 **Hub** 콘솔 사용자에게 **Informatica MDM Hub** 구현을 확장하는 데 사용할 수 있는 사용자 지정 단추를 제공할 수 있습니다. 사용자는 사용자 지정 단추를 사용하여 데이터 검색 또는 결과 계산 같은 특정 외부 서비스를 호출하고, 워크플로우 실행 같은 특수화된 작업과 기타 태스크를 수행할 수 있습니다. 사용자 지정 단추는 **Hub** 콘솔의 병합 관리자, 데이터 관리자 및 계층 관리자 도구에 추가할 수 있습니다.

서버 및 클라이언트 기반 사용자 지정 함수는 사용자 개체 레지스트리에 표시됩니다.

사용자 지정 단추 함수를 등록하는 방법

Informatica MDM Hub 구현에서 **Hub** 콘솔에 사용자 지정 단추를 추가하려면 다음 태스크를 완료합니다.

1. 요청 및 응답 메시지의 형식과 매개 변수 등 호출할 외부 서비스에 대한 세부 정보를 확인합니다.
2. 사용자 지정 단추로 실행할 비즈니스 논리를 작성하여 패키지합니다.
3. **Hub** 콘솔의 해당하는 도구에 표시되도록 패키지를 배포합니다.

등록된 사용자 지정 단추 함수 보기

사용자 개체 레지스트리 도구에서 등록된 사용자 지정 단추 함수를 보려면

1. 사용자 개체 레지스트리 도구를 시작합니다.

2. **사용자 지정 단추 함수**를 선택합니다.

사용자 개체 레지스트리 도구에 등록된 사용자 지정 단추 함수가 표시됩니다.

제 32 장

Informatica MDM Hub 서비스 및 이벤트 감사

이 장에 포함된 항목:

- [개요, 599](#)
- [통합 감사 정보, 599](#)
- [감사 관리자 시작, 601](#)
- [SIF API 요청 감사, 603](#)
- [메시지 대기열 감사, 603](#)
- [오류 감사, 604](#)
- [감사 로그 사용, 604](#)

개요

이 장에서는 Hub 콘솔에서 감사 및 디버깅을 설정하는 방법에 대해 설명합니다.

통합 감사 정보

MDM Hub에는 다양한 구성 요소에서의 활동을 추적하는 MDM Hub 로그, 응용 프로그램 서버 로그, 데이터베이스 서버 로그 등의 여러 로그 파일이 있습니다.

이 장에서 다루는 감사는 MDM Hub과 외부 시스템 간의 데이터 교환과 관련된 활동을 추적하는 통합 감사로 설명될 수 있습니다. 다른 유형의 로그 파일에 대한 자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

감사는 환경의 각 연산 참조 저장소에 대해 개별적으로 구성됩니다.

감사 가능 이벤트

외부 응용 프로그램과 통합할 경우 구현이 복잡해지는 경우가 많습니다.

여러 응용 프로그램이 서로 상호 작용하고, 동기 또는 비동기적으로 데이터를 교환하고, 데이터 상호 변환을 사용하며, 다양한 비즈니스 규칙을 적용하여 응용 프로그램 간의 비즈니스 프로세스를 실행합니다.

응용 프로그램 개발자와 시스템 통합자가 응용 프로그램 통합의 세부 정보를 파악할 수 있도록 Informatica MDM Hub에서는 다음과 같은 경우에 항상 감사 내역을 생성하는 기능을 제공합니다.

- 외부 응용 프로그램이 SIF(서비스 통합 프레임워크) 요청을 호출하여 Informatica MDM Hub와 상호 작용할 경우. 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.
- Informatica MDM Hub가 데이터 변경 내용을 다른 시스템에 배포할 목적으로 JMS를 사용하여 메시지 대기열로 메시지를 보낼 경우

Informatica MDM Hub 감사 메커니즘은 선택적이며 구성 가능합니다. 감사 메커니즘은 감사 사용 SIF 요청의 호출을 추적하고, 언제 어떤 작업이 수행되었는지에 대한 데이터를 수집하고, 특정 작업이 실행된 이유에 대한 컨텍스트 정보를 제공합니다. 또한 나중에 TOAD나 호환되는 다른 외부 데이터 관리 도구를 사용하여 볼 수 있는 감사 로그 테이블(C_REPOS_AUDIT)에 감사 정보를 저장합니다.

참고: 감사는 메타데이터 캐싱이 활성화(설정)되어 있는 비활성화(해제)되어 있는 관계없이 적용됩니다.

감사 관리자 도구

감사는 Hub 콘솔의 감사 관리자 도구를 사용하여 구성됩니다.

감사 관리자에서 관리자는 다음을 선택할 수 있습니다.

- 감사할 SIF 요청 및 시스템(관리 시스템, 정의된 소스 시스템 또는 시스템 선택 안 함)
- 아웃바운드 메시지가 JMS 대기열로 보내질 때 감사할 메시지 대기열(메시지 트리거와 함께 사용하도록 할당된 대기열)

요청 및 응답의 XML 캡처

특정 SIF 요청 또는 JMS 이벤트를 완벽하게 디버깅하기 위해 필요한 경우 사용자가 감사 로그의 요청 및 응답 XML을 캡처할 수 있습니다. 이는 특히 쓰기 작업에 유용합니다.

이 상세 수준의 감사 기능은 광범위한 정보를 수집하기 때문에 성능을 저하시킬 수 있으므로 디버깅 목적으로만 사용하고 프로덕션 환경에서 지속적으로 사용하지 않는 것이 좋습니다.

감사는 명시적으로 활성화되어야 함

기본적으로 SIF 요청 및 이벤트의 감사는 비활성화되어 있습니다.

감사 관리자 도구를 사용하여 감사하려는 각 SIF 요청 및 이벤트에 대해 감사 기능을 명시적으로 활성화해야 합니다.

감사는 인증 후에 수행됨

모든 SIF 요청 호출은 호출과 연관된 사용자 자격 증명이 Hub 서버에서 인증된 후에 감사될 수 있습니다.

따라서 실패한 로그인 시도는 감사되지 않습니다. 예를 들어 타사 응용 프로그램에서 SIF 요청을 호출하려고 하면서 잘못된 로그인 자격 증명을 제공할 경우 해당 정보는 C_REPOS_AUDIT 테이블에 캡처되지 않습니다. 감사는 인증이 성공한 후에만 시작됩니다.

감사는 올바른 형식의 유효한 XML을 사용한 호출에 대해 수행됨

올바른 형식의 유효한 XML을 사용한 SIF 요청 호출만 감사됩니다. 잘못된 XML이나 올바른 형식이 아닌 XML을 사용한 SIF 요청은 감사되지 않습니다.

암호 변경 내용 감사

Informatica MDM Hub 암호 변경 서비스를 호출할 경우 사용자의 기본 데이터베이스에 따라 SIF 요청의 감사 여부가 결정됩니다.

- 사용자의 기본 데이터베이스가 ORS(연산 참조 저장소)이면 Informatica MDM Hub 암호 변경 서비스가 감사됩니다.
- 사용자의 기본 데이터베이스가 마스터 데이터베이스이면 암호 변경 서비스 호출이 감사되지 않습니다.

감사 관리자 시작

감사 관리자를 시작하려면

- Hub 콘솔에서 유틸리티 작업 영역까지 스크롤한 후 **감사 관리자**를 클릭합니다.
Hub 콘솔에 감사 관리자가 표시됩니다.

감사 관리자는 두 개의 창으로 구분됩니다.

창	설명
탐색 창	다음 정보를 트리 보기로 표시합니다. <ul style="list-style-type: none">- 이 Informatica MDM Hub 구현에 대한 감사 유형- 감사할 시스템- 감사할 메시지 대기열
속성 창	선택한 감사 유형 또는 시스템의 속성을 표시합니다.

감사 가능한 API 요청 및 메시지 대기열

감사 관리자의 탐색 창에는 다음 유형의 감사 항목 목록과 사용 가능한 시스템이 표시됩니다.

유형	설명
API 요청	외부 응용 프로그램에서 SIF(서비스 통합 프레임워크) SDK(소프트웨어 개발 키트)를 사용하여 호출한 요청입니다.
메시지 대기열	메시지 트리거에 사용되는 메시지 대기열입니다. 참고: 메시지 대기열은 CMX_SYSTEM 수준에서 정의됩니다. 이러한 설정은 해당 ORS(연산 참조 저장소)에 대한 메시지에만 적용됩니다.

감사할 시스템

감사 관리자에는 감사할 각 항목 유형에 대해 감사할 수 있는 시스템의 목록과 해당 시스템에 연결된 SIF 요청이 표시됩니다.

시스템	설명
시스템 없음	특정 시스템과 연결되어 있지 않거나 반드시 연결될 필요는 없는 서비스(예: 병합 작업)입니다.
관리	관리 시스템과 연결되어 있는 서비스입니다.
정의된 소스 시스템	미리 정의된 소스 시스템과 연결되어 있는 서비스입니다.

참고: 동일한 API 요청 또는 메시지 대기열이 여러 소스 시스템에서 나타날 수 있습니다(예: 해당 소스 시스템 중 하나에서 API 요청 또는 메시지 대기열의 사용이 선택적인 경우).



감사 속성

감사할 항목을 선택하면 감사 관리자의 속성 창에 다음과 같이 설정을 구성할 수 있는 속성이 표시됩니다.

참고: 감사를 구성하는 데는 쓰기 잠금이 필요하지 않습니다.

필드	설명
시스템 이름	선택한 시스템의 이름입니다. 읽기 전용.
설명	선택한 시스템에 대한 설명입니다. 읽기 전용.
API 요청	감사할 수 있는 API 요청의 목록입니다.
메시지 대기열	감사할 수 있는 메시지 대기열의 목록입니다.
감사 활성화 여부	<p>감사는 기본적으로 활성화되어 있지 않습니다.</p> <ul style="list-style-type: none"> 항목에 대한 감사를 활성화하려면 이 속성을 선택합니다. 항목에 대한 감사를 비활성화하려면 이 속성을 선택 취소합니다.
XML 포함 여부	<p>이 확인란은 해당 항목에 대한 감사가 활성화되어 있는 경우에만 사용할 수 있습니다. 캡처 XML은 기본적으로 로그에 포함되지 않습니다.</p> <ul style="list-style-type: none"> 해당 항목에 대한 감사 로그에 XML을 포함하려면 이 속성을 선택합니다. 해당 항목에 대한 감사 로그에서 XML을 제외하려면 이 속성을 선택 취소합니다. <p>참고: 암호는 감사 로그에 저장되지 않습니다. 암호가 XML 스트림에 있는 경우(암호화 여부는 관계없음) Informatica MDM Hub에서는 암호를 별표로 바꿉니다.</p> <pre> ...<get> <username>admin</username> <password> <encrypted>>false</encrypted> <password>*****</password> </password> ... </pre> <p>중요: 이 옵션을 선택하면 감사 로그 파일의 크기가 매우 빠른 속도로 증가할 수 있습니다.</p>

"감사 활성화 여부" 및 "XML 포함 여부" 확인란의 경우 다음 단추를 사용할 수 있습니다.

단추	이름	설명
	모두 선택	목록의 모든 항목을 선택합니다.
	모두 지우기	목록의 선택된 항목을 모두 선택 취소합니다.

SIF API 요청 감사

외부 응용 프로그램에서의 SIF(서비스 통합 프레임워크) 요청을 감사할 수 있습니다.

특정 SIF API 요청에 대한 감사가 활성화되어 있으면 Informatica MDM Hub는 각 SIF 요청 호출 및 응답을 감사 로그에 캡처합니다.

SIF API 요청에 대한 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

SIF API 요청을 감사하려면

1. 감사 관리자를 시작합니다.
2. 탐색 트리에서 API 요청 아래의 시스템을 선택합니다.
모든 시스템에 대한 글로벌 감사 설정을 구성하려면 **시스템 없음**을 선택합니다.
편집 창에 선택한 시스템에 대한 구성 가능한 API 요청이 표시됩니다.
3. 감사할 각 SIF 요청에 대해 **감사 활성화 여부** 확인란을 선택합니다.
4. 특정 API 요청에 대해 감사가 활성화되어 있는 경우 해당 API 요청과 연결된 XML도 감사 로그에 포함하려면 **XML 포함 여부** 확인란을 선택합니다.
5. **저장** 단추를 클릭하여 변경 내용을 저장합니다.
참고: 저장된 설정은 최대 60초 후에 Hub 서버에 적용될 수 있습니다.

메시지 대기열 감사

메시지 트리거가 할당된 메시지 대기열에 대해 감사를 구성할 수 있습니다.

구성된 메시지 트리거가 없는 메시지 대기열은 감사할 수 없습니다.

메시지 대기열을 감사하려면

1. 감사 관리자를 시작합니다.
2. 탐색 트리에서 **메시지 대기열** 아래의 시스템을 선택합니다.
편집 창에 선택한 시스템에 대한 구성 가능한 메시지 대기열이 표시됩니다.
3. 감사할 각 메시지 대기열에 대해 **감사 활성화 여부** 확인란을 선택합니다.
4. 특정 메시지 대기열에 대해 감사가 활성화되어 있는 경우 해당 메시지 대기열과 연결된 XML도 감사 로그에 포함하려면 **XML 포함 여부** 확인란을 선택합니다.

5. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

참고: 저장된 설정은 최대 60초 후에 Hub 서버에 적용될 수 있습니다.

오류 감사

웹 서비스에서 오류 메커니즘을 트리거하는 SIF 요청 호출에 대해 구문 오류, 런타임 오류 등의 오류 정보를 캡처할 수 있습니다.

SIF 요청과 연결된 모든 오류에 대해 감사를 활성화할 수 있습니다.

오류 감사는 전역적으로 활성화하는 기능입니다. 즉, 특정 SIF 요청에 대해 현재 감사가 활성화되어 있지 않더라도 해당 SIF 요청 호출 중에 오류가 발생하면 관련 이벤트가 감사 로그에 캡처됩니다.

글로벌 오류 감사 구성

오류를 감사하려면

1. 감사 관리자를 시작합니다.
2. 탐색 트리에서 **API 요청**을 선택하여 SIF 오류에 대한 감사를 구성합니다.
편집 창에 오류에 대한 구성 페이지가 표시됩니다.
3. 다음 작업 중 하나를 수행합니다.
 - 오류를 감사하려면 **감사 활성화 여부** 확인란을 선택합니다.
 - 오류 감사를 중지하려면 **감사 활성화 여부** 확인란의 선택을 취소합니다.
4. **감사 활성화 여부**를 선택한 경우 오류와 연결된 XML도 감사 로그에 포함하려면 **XML 포함 여부** 확인란을 선택합니다.

참고: 감사 활성화 여부만 선택할 경우 Informatica MDM Hub는 C_REPOS_AUDIT에서 관련 감사 정보를 제공합니다.

XML 포함 여부도 선택할 경우 Informatica MDM Hub는 C_REPOS_AUDIT에 감사에 대한 세부 로그 데이터를 포함하는 DATA_XML이라는 추가 열을 포함합니다.

두 확인란을 모두 선택할 경우 사용자가 데이터 관리자에서 삽입, 업데이트 또는 삭제 작업을 실행하거나 연결된 일괄 작업을 실행하면 Informatica MDM Hub는 C_REPOS_AUDIT의 DATA_XML에 감사 데이터를 포함합니다.

5. **저장** 단추를 클릭하여 변경 내용을 저장합니다.

감사 로그 사용

SIF 요청 및 이벤트에 대한 감사를 구성한 후에는 채워진 감사 로그 테이블(C_REPOS_AUDIT)을 분석, 예외 보고, 디버깅 등에 필요한 대로 사용할 수 있습니다.

감사 로그 정보

C_REPOS_AUDIT 테이블은 ORS(연산 참조 저장소)에 저장됩니다.

특정 SIF 요청 또는 이벤트에 대해 감사가 활성화되어 있으면 Informatica MDM Hub에서 해당 SIF 요청이 호출되거나 해당 이벤트가 트리거될 때마다 감사 메커니즘이 관련 정보를 캡처하여 C_REPOS_AUDIT 테이블에 저장합니다.

참고: 외부 응용 프로그램에서는 SIF 감사 요청을 사용하여 새 레코드를 C_REPOS_AUDIT 테이블에 삽입할 수 있습니다. Hub가 기록할 수 있는 것보다 더 많은 정보를 포함하거나 수준이 더 높은 Informatica MDM Hub의 레코드와 관련된 작업을 보고하는 데는 이 요청을 사용합니다. 예를 들어 복잡한 개체를 Hub 개체로 변환 및 분해하기 전에 해당 개체에 대한 업데이트를 감사합니다. 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

감사 로그 테이블

감사 로그 정보는 C_REPOS_AUDIT라는 감사 로그 테이블에 저장됩니다.

다음 테이블에서는 감사 로그 테이블 C_REPOS_AUDIT의 스키마를 보여 줍니다.

이름	Oracle 데이터 유형	IBM DB2 데이터 유형	Microsoft SQL Server 데이터 유형	설명
ROWID_AUDIT	CHAR(14)	CHARACTER(14)	NCHAR(14)	해당 레코드의 고유 ID입니다. 기본 키입니다.
CREATE_DATE	TIMESTAMP	TIMESTAMP	DATETIME2	레코드 생성 날짜입니다. 기본적으로 시스템 날짜로 설정됩니다.
CREATOR	VARCHAR2(50)	VARCHAR(50)	NVARCHAR(50)	감사 이벤트와 연결된 사용자입니다.
LAST_UPDATE_DATE	TIMESTAMP	TIMESTAMP	DATETIME2	CREATE_DATE와 동일합니다.
UPDATED_BY	VARCHAR2(50)	VARCHAR(50)	NVARCHAR(50)	CREATOR와 동일합니다.
COMPONENT	VARCHAR2(50)	VARCHAR(50)	NVARCHAR(50)	관련 구성 요소입니다. - SIF.sif.api
ACTION	VARCHAR2(50)	VARCHAR(50)	NVARCHAR(50)	다음 중 하나입니다. - SIF 요청 이름 - 메시지 대기열 이름
STATUS	VARCHAR2(50)	VARCHAR(50)	NVARCHAR(50)	다음 값 중 하나입니다. - 디버그 - 정보 - 경고 - 오류 - 치명적
ROWID_OBJECT	CHAR(14)	CHARACTER(14)	NCHAR(14)	rowid_object입니다(알려진 경우).

이름	Oracle 데이터 유형	IBM DB2 데이터 유형	Microsoft SQL Server 데이터 유형	설명
DATA_XML	CLOB	CLOB	NVARCHAR (max)	감사 가능 이벤트와 연결된 XML(요청, 응답 또는 JMS 메시지)입니다. XML 포함 여부 옵션이 활성화(선택)되어 있는 경우에만 채워집니다. 참고: 암호는 감사 로그에 저장되지 않습니다. 암호가 XML 스트림에 있는 경우 (암호화 여부는 관계없음) Informatica MDM Hub에서는 암호를 "*****" 텍스트로 바꿉니다.
CONTEXT_XML	CLOB	CLOB	NVARCHAR (max)	구성 데이터, 호출된 URL, 일치 규칙의 실행 추적 등 상황에 맞는 정보를 포함할 수 있는 XML입니다. 호출된 SIF 요청에 대해 감사가 활성화되지 않은 경우 오류가 발생하면 요청 XML이 항상 이 열에 put되어 캡처됩니다. XML 포함 여부 옵션이 활성화(선택)되어 있는 경우에만 채워집니다.
ROWID_AUDIT_PREVIOUS	CHAR(14)	CHARACTER(14)	NCHAR(14)	관련된 이전 항목의 ROWID_AUDIT에 대한 참조입니다. 예를 들어 응답 항목을 해당 요청 항목에 연결합니다.
INTERACTION_ID	NUMBER(19)	BIGINT(8)	NUMERIC (19,0)	상호 작용 ID입니다. INTERACTION_ID는 선택적이므로 NULL일 수 있습니다.
USERNAME	VARCHAR2(50)	VARCHAR(50)	NVARCHAR (50)	SIF 요청을 호출한 사용자입니다. 메시지 대기열의 경우 Null입니다.
FROM_SYSTEM	VARCHAR2(50)	VARCHAR(50)	NVARCHAR (50)	SIF 요청의 소스 시스템이거나 메시지 대기열의 관리자입니다.
TO_SYSTEM	VARCHAR2(50)	VARCHAR(50)	NVARCHAR (50)	감사된 이벤트와 관련된 시스템입니다. 예를 들어 Hub에 대한 API 요청에서는 이 항목을 "Admin"으로 설정하며, 응답은 해당 시스템이거나 null(알 수 없는 경우)입니다. 응답의 경우에는 그 반대입니다.

이름	Oracle 데이터 유형	IBM DB2 데이터 유형	Microsoft SQL Server 데이터 유형	설명
TABLE_NAME	VARCHAR2(100)	VARCHAR(100)	NVARCHAR (100)	이 감사된 이벤트와 연결된 Hub 저장소의 테이블입니다.
CONTEXT	VARCHAR2(255)	VARCHAR(255)	NVARCHAR (255)	메타데이터입니다. 예: pkeySource Hub에서의 감사일 경우 이 값은 null이지만 SIF API를 통해 수행된 감사의 값을 가질 수도 있습니다.

감사 로그 보기

Informatica MDM Hub에 포함되지 않은 외부 데이터 관리 도구를 사용하여 감사 로그를 볼 수 있습니다.

로그 파일을 보는 데 사용하는 데이터 관리 도구에 해당 기능이 있는 경우 감사 수준(디버그 수준 또는 정보 수준 항목만 표시할 경우), 시간(과거 항목을 표시할 경우), 작업 성공/실패(오류 항목만 표시할 경우) 등을 기준으로 항목을 필터링하여 보려는 항목에 초점을 맞출 수 있습니다.

다음 SQL 문은 Oracle 및 IBM DB2의 경우에 대한 예를 보여 줍니다.

```
SELECT ROWID_AUDIT, FROM_SYSTEM, TO_SYSTEM, USERNAME, COMPONENT, ACTION, STATUS, TABLE_NAME,
ROWID_OBJECT, ROWID_AUDIT_PREVIOUS, DATA_XML,
CREATE_DATE FROM C_REPOS_AUDIT
WHERE CREATE_DATE >= TO_DATE('07/06/2006 12:23:00', 'MM/DD/YYYY HH24:MI:SS')
ORDER BY CREATE_DATE
```

다음 SQL 문은 Microsoft SQL Server의 경우에 대한 예를 보여 줍니다.

```
SELECT ROWID_AUDIT, FROM_SYSTEM, TO_SYSTEM, USERNAME, COMPONENT, ACTION, STATUS, TABLE_NAME,
ROWID_OBJECT, ROWID_AUDIT_PREVIOUS, DATA_XML,
CREATE_DATE FROM C_REPOS_AUDIT
WHERE CREATE_DATE >= CAST('01-JAN-2012' AS DATETIME)
ORDER BY CREATE_DATE
```

정기적 감사 로그 제거

감사 로그 테이블은 특히 XML 요청 및 응답 정보를 캡처할 때([XML 포함] 옵션이 활성화된 경우) 급속도로 매우 커질 수 있습니다.

데이터베이스 관리 시스템에서 제공된 도구를 사용할 경우 특정 필터와 일치하는 레코드(예: 생성된 지 60분이 넘는 항목)를 정기적으로 삭제하는 예약 작업을 설정하는 것이 좋습니다.

다음 SQL 문은 Oracle 및 IBM DB2의 경우에 대한 예를 보여 줍니다.

```
DELETE FROM C_REPOS_AUDIT WHERE CREATE_DATE < (SYSDATE - 1) AND STATUS='INFO'
```

다음 SQL 문은 Microsoft SQL Server의 경우에 대한 예를 보여 줍니다.

```
DELETE FROM C_REPOS_AUDIT WHERE CREATE_DATE < SYSDATETIME() AND STATUS='INFO';
```

부록 A

MDM Hub 속성

이 부록에 포함된 항목:

- [MDM Hub 속성 개요, 608](#)
- [Hub 서버 속성, 608](#)
- [샘플 Hub 서버 속성 파일, 625](#)
- [처리 서버 속성, 627](#)
- [샘플 처리 서버 속성 파일, 634](#)
- [연산 참조 저장소 속성, 635](#)

MDM Hub 속성 개요

MDM Hub 속성 파일에는 Hub 서버 및 처리 서버 구성 설정이 포함되어 있습니다.

텍스트 편집기를 사용하여 속성 파일을 보거나 편집할 수 있습니다. Hub 서버 및 처리 서버를 처음 설치할 때 설치 프로그램에서 속성 파일의 일부 속성 값을 설정합니다. 설치 후 속성을 편집하여 MDM Hub의 동작을 변경할 수 있습니다. 예를 들어 Hub 서버에 대한 데이터 암호화를 구성하려면 `encryption.plugin.jar` 속성에서 데이터 암호화 JAR의 경로 및 파일 이름을 지정해야 합니다.

일부 속성은 선택 사항입니다. 즉, 이러한 파일은 MDM Hub 속성 파일에 수동으로 추가해야 합니다. 예를 들어 WebSphere 클러스터링을 활성화하려면 `cluster.flag` 속성을 Hub 서버 속성에 추가한 후 값을 `true`로 설정해야 합니다.

속성 파일을 편집한 후 변경 사항을 적용하려면 응용 프로그램 서버를 다시 시작합니다.

Hub 서버 속성

`cmxserver.properties` 파일에서 Hub 서버 속성을 구성합니다.

`cmxserver.properties` 파일은 다음 디렉터리에 있습니다.

<MDM Hub 설치 디렉터리>/hub/server/resources

Hub 서버 환경 속성

다음 속성은 Hub 서버의 위치 및 응용 프로그램 서버와 데이터베이스의 연결 세부 정보를 설정합니다.

cmx.home

Hub 서버의 설치 디렉터리입니다. 이 속성은 Hub 서버 설치 과정에서 설정됩니다.

cmx.appserver.hostname

MDM Hub를 배포할 EJB 클러스터의 이름입니다. 다음 형식으로 클러스터 서버의 호스트 이름을 지정합니다.

`<host_name>.<도메인>.com`

클러스터된 환경에 Hub 서버를 배포하는 방법에 대한 자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

cmx.appserver.naming.protocol

응용 프로그램 서버 유형의 이름 지정 프로토콜입니다. 기본값은 Websphere가 iiop, JBoss 5가 jnp, JBoss 7이 원격, WebLogic이 t3입니다. 이 속성은 Hub 서버 설치 과정에서 설정됩니다.

cmx.appserver.rmi.port

응용 프로그램 서버 포트입니다. 기본값은 Websphere가 2809, WebLogic이 7001, JBoss 5가 1099, JBoss 7이 4447입니다. 클러스터된 환경에 Hub 서버를 배포하는 방법에 대한 자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

cmx.appserver.type

응용 프로그램 서버의 유형입니다. 이 속성의 값은 JBoss, WebSphere 또는 WebLogic 중 하나입니다. 이 속성은 Hub 서버 설치 과정에서 설정됩니다.

cmx.server.attachment.temp.ttl_minutes

TEMP 저장소에 생성된 파일이 만료되기까지 남은 시간(분)입니다. 파일이 만료되지 않게 하려면 0으로 설정합니다. 기본값은 60입니다.

cmx.server.masterdatabase.type

MDM Hub 마스터 데이터베이스의 유형입니다. 이 속성의 값은 DB2, Oracle 또는 MSSQL 중 하나입니다. 이 속성은 Hub 서버 설치 과정에서 설정됩니다.

cmx.server.masterdatabase.schemaname

IBM DB2 환경에만 필요합니다. MDM Hub 마스터 데이터베이스 이름이 cmx_system이 아닌 경우 속성을 사용하여 이름을 지정합니다. 기본값은 cmx_system입니다.

cookie-secure

Data Director 세션 쿠키를 보호합니다. 보안 IDD 세션 쿠키를 허용하려면 cookie-secure 플래그의 주석 처리를 제거하고 값을 true로 설정합니다. 기본값은 false입니다.

변경 내용이 적용되도록 Hub 콘솔을 다시 시작합니다.

http-only

HTTP에 대해서만 Data Director 세션 쿠키를 보호합니다. 세션 쿠키를 허용하려면 http-only 플래그의 주석 처리를 제거하고 값을 true로 설정합니다. 기본값은 false입니다.

변경 내용이 적용되도록 Hub 콘솔을 다시 시작합니다.

로컬

Hub 서버 및 Hub 콘솔의 로컬입니다. Hub 서버를 처음 설치할 때 이 속성의 값이 설정됩니다.

JBoss용 응용 프로그램 서버 속성

Hub 서버는 JBoss 응용 프로그램 서버에서 실행될 경우 다음 속성을 사용합니다.

cmx.appserver.version

응용 프로그램 서버의 JBoss 버전입니다. 이 속성의 값은 5 또는 7입니다. 이 속성은 Hub 서버 설치 과정에서 설정됩니다.

cmx.jboss7.management.port

JBoss 원시 관리 포트입니다. JBoss의 기본값은 9990입니다. 이 속성은 Hub 서버 설치 과정에서 설정됩니다.

cmx.jboss7.security.enabled

JBoss EJB 보안을 활성화합니다. JBoss EJB 보안을 활성화하려면 true로 설정합니다. JBoss EJB 보안을 비활성화하려면 false로 설정합니다. 기본값은 true입니다. JBoss 보안 구성에 대한 자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

cmx.server.ejb3

JBoss 7에만 사용됩니다. 응용 프로그램 서버 EJB3 조회를 허용하려면 true로 설정합니다. 기본값은 false입니다. 이 속성은 Hub 서버 설치 과정에서 설정됩니다.

ejb-client-version

선택 사항입니다. 수동으로 추가해야 합니다. EJB 클라이언트 버전을 지정합니다. 기본 JBoss EJB 클라이언트를 사용하지 않으려는 경우 ejb-client-version을 사용하여 다른 EJB 클라이언트를 지정하십시오. Hub 콘솔에 대해 EJB 클라이언트를 구성하는 방법에 대한 자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

jboss.cluster

JBoss 7에만 사용됩니다. Hub 서버를 위해 EJB 서버가 클러스터되는지 여부를 지정합니다. EJB 클러스터링을 허용하려면 true로 설정합니다. 클러스터된 EJB 서버가 없으면 false로 설정합니다. 기본값은 false입니다.

WebSphere용 응용 프로그램 서버 속성

Hub 서버는 WebSphere 응용 프로그램 서버에서 실행될 경우 다음 속성을 사용합니다.

cluster.flag

선택 사항입니다. 수동으로 추가해야 합니다. WebSphere에만 사용됩니다. 클러스터링 허용 여부를 지정합니다. 클러스터링을 허용하려면 true로 설정합니다. 클러스터링을 허용하지 않으려면 false로 설정합니다. 기본값은 false입니다.

cmx.appserver.password

WebSphere 관리자의 암호입니다. WebSphere 관리 보안이 활성화되면 이 속성을 사용할 수 있게 됩니다.

cmx.appserver.username

WebSphere 관리자의 사용자 이름입니다. WebSphere 관리 보안이 활성화되면 이 속성을 사용할 수 있게 됩니다.

cmx.appserver.soap.connector.port

WebSphere에만 사용됩니다. SOAP 커넥터 포트입니다. WebSphere의 경우 기본값은 8880입니다. 클러스터된 환경에 Hub 서버를 배포하는 데 대한 자세한 내용은 WebSphere용 *Multidomain MDM 설치 가이드*를 참조하십시오.

cmx.websphere.security.enabled

WebSphere 보안이 활성화되는지 여부를 지정합니다. WebSphere 관리 보안을 활성화하려면 true 또는 yes로 설정합니다. 기본값은 아니요입니다. WebSphere 관리 보안 활성화에 대한 자세한 내용은 *Multidomain MDM 업그레이드 가이드*를 참조하십시오.

cmx.websphere.security.sas.config.name

WebSphere에만 사용됩니다. sas.client.props 파일의 사용자 지정 이름입니다. 보안 EJB 조회를 사용하는 환경에 사용됩니다.

기본값은 sas.client.props입니다.

cmx.websphere.security.sas.config.url

WebSphere에만 사용됩니다. sas.client.props 파일의 위치입니다. 보안 EJB 조회를 사용하는 환경에 사용됩니다.

기본값은 https://yourdomain.com:9443/cmx/filesx/Security/WebSphere/sas.client.props입니다.

cmx.websphere.security.ssl.config.name

WebSphere에만 사용됩니다. ssl.client.props 파일의 사용자 지정 이름입니다. 보안 EJB 조회를 사용하는 환경에 사용됩니다.

기본값은 ssl.client.props입니다.

cmx.websphere.security.ssl.config.url

WebSphere에만 사용됩니다. ssl.client.props 파일의 위치입니다. 보안 EJB 조회를 사용하는 환경에 사용됩니다.

기본값은 https://yourdomain.com:9443/cmx/filesx/Security/WebSphere/ssl.client.props입니다.

was.jms.log.dir

선택 사항입니다. 수동으로 추가해야 합니다. WebSphere에만 사용됩니다. SIB의 로그 디렉터리 위치, 즉 WebSphere 리소스를 지정합니다.

was.jms.permanent_store.dir

선택 사항입니다. 수동으로 추가해야 합니다. WebSphere에만 사용됩니다. SIB의 영구 저장소 디렉터리 위치, 즉 WebSphere 리소스를 지정합니다.

was.jms.temp_store.dir

선택 사항입니다. 수동으로 추가해야 합니다. WebSphere에만 사용됩니다. SIB의 임시 저장소 디렉터리 위치, 즉 WebSphere 리소스를 지정합니다.

데이터베이스 속성

데이터베이스에 대해 다음과 같은 속성을 설정할 수 있습니다.

cmx.server.loadWorker.max.joins.optimization

선택 사항입니다. 수동으로 추가해야 합니다. IBM DB2에만 해당됩니다. 쿼리에 사용되는 최대 조인 수를 지정합니다. 로드 작업의 조회 테이블이 12개 이상인 쿼리를 DB2에서 실행하는 데 지나치게 오래 걸리는 경우에는 값을 20으로 설정합니다. 기본값은 30입니다.

일반 속성

다음 속성은 Hub 서버 프로세스의 동작을 설정합니다.

cmx.outbound.bypass.multixref.insystem

선택 사항입니다. 수동으로 설정해야 합니다. true로 설정하면 SIF API에서 여러 교차 참조 레코드가 있는 기본 개체를 업데이트할 때 Hub 서버에서 메시지를 생성하지 않습니다. 기본값은 false입니다.

cmx.server.datalayer.cleanse.execution

정리 작업을 실행할 위치를 지정합니다. 응용 프로그램 서버에서 정리 작업을 실행하려면 LOCAL로 설정합니다. 데이터베이스 서버에서 정리 작업을 실행하려면 DATABASE로 설정합니다. 기본값은 LOCAL입니다. 정리 엔진 통합에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

cmx.server.datalayer.cleansse.working_files

정리 작업을 수행하는 동안 생성된 임시 파일을 저장할지 여부를 지정합니다. 임시 파일을 문제 해결 또는 감사 목적으로 사용할 수 있습니다. 임시 작업 파일을 삭제하려면 **FALSE**로 설정합니다. 임시 작업 파일을 저장하려면 **KEEP**으로 설정합니다. 기본값은 **KEEP**입니다. 정리 엔진 통합에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

cmx.server.datalayer.cleansse.working_files.location

정리 작업의 작업 파일 위치입니다. **MDM Hub**가 **Hub** 서버 초기화 루틴에서 이 속성을 사용합니다. 이 속성은 **Hub** 서버 설치 과정에서 설정됩니다. 이 속성을 변경하지 마십시오. 정리 엔진 통합에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

cmx.server.encryptionMethod

선택 사항입니다. 수동으로 추가해야 합니다. **SSL** 암호화를 활성화하려면 **SSL**로 설정합니다.

cmx.server.load.nonsmos.sourcesystem.enddate.like.smos

SMOS(상태 관리 재정의의 시스템)가 아닌 시스템의 관계 종료 날짜를 **SMOS**와 동일한 날짜로 설정합니다. 관계 종료 날짜를 **SMOS**의 날짜와 동일한 날짜로 설정하려면 **true**로 설정합니다.

cmx.server.met.max_send_size

리포지토리 관리자에서 보낼 수 있는 최대 파일 크기(바이트)입니다. 기본값은 **9000000**입니다.

cmx.server.met.promotion_snapshot

선택 사항입니다. 수동으로 추가해야 합니다. **.meta** 파일 생성을 활성화하려면 **true**로 설정합니다. **.meta** 파일 생성을 비활성화하려면 **false**로 설정합니다. 기본값은 **true**입니다.

cmx.server.multi_data_set_schema

선택 사항입니다. 수동으로 추가해야 합니다. 상위 레코드 및 모든 해당하는 하위 레코드를 포함하도록 메시지 트리거 **XML** 메시지를 활성화하려면 **true**로 설정합니다. 상위 레코드 및 모든 해당하는 하위 레코드를 포함하도록 메시지 트리거 **XML** 메시지를 비활성화하려면 **false**로 설정합니다. 기본값은 **false**입니다.

cmx.server.poller.monitor.interval

모든 서버에 대한 폴링 간격(초)입니다. 서버 폴링을 비활성화하려면 **0**으로 설정합니다. 기본값은 **30**입니다.

cmx.server.put.autopopulate.missing.user.columns.bo.list

수동으로 추가해야 합니다. 기본 개체 열에 대해 **Null** 가능 속성이 비활성화된 경우 속성을 설정합니다.

열에 대해 **Null** 가능 속성이 비활성화된 경우, 레코드가 **Data Director**에서 업데이트되거나 **SIF PUT** 작업 중에 업데이트된 업데이트된 필드에 대한 값만 포함한 교차 참조 레코드가 생성됩니다. **Null**이 아니어야 하는 필드를 포함하여 다른 모든 교차 참조 레코드 필드는 **Null** 값을 갖습니다. 레코드에 **BVT**(최선의 진실, **Best Version of the Truth**) 계산을 진행하는 동안 **Null** 값은 **Null**이 아니어야 하는 필드에도 우선 적용될 수 있어서 오류가 발생합니다.

Null이 아니어야 하는 필드를 **BVT** 계산에서 고려하려면 **Null** 가능 속성이 비활성화된 열이 있는 기본 개체의 이름 목록(덱표로 구분된 목록)으로 속성 값을 설정하십시오. 속성이 설정되면 **MDM Hub**가 연결된 기본 개체의 값이 있는 교차 참조 레코드의 **Null** 값을 업데이트합니다. 따라서 **BVT** 계산 시 **Null** 값은 **Null**이 아니어야 하는 필드에 우선 적용되지 않습니다.

cmx.server.selective.bvt.enabled

선택 사항입니다. 수동으로 추가해야 합니다. **MDM Hub**가 **SIF** 요청의 일부인 필드에만 **BVT** 계산을 적용하도록 지정합니다. **MDM Hub**가 **SIF** 요청에 지정된 필드만 업데이트하도록 하려면 **true**로 설정합니다. 기본값은 **false**입니다.

cmx.server.validateServerCertificate

선택 사항입니다. 수동으로 추가해야 합니다. 서버 인증서 유효성 검사를 해제하려면 **false**로 설정합니다. 기본값은 **true**입니다.

com.informatica.mdm.message.queue.max.bytes.threshold

선택 사항입니다. 수동으로 추가해야 합니다. 메시지 대기열로 전송되는 메시지의 최대 제한(바이트)을 지정합니다. 메시지가 지정된 크기를 초과하면 메시지가 전송되지 않고 메시지 상태가 **Failed**로 설정됩니다.

com.informatica.mdm.sifapi.xref.edit.sys0.only

선택 사항입니다. 수동으로 추가해야 합니다. 관리 소스 시스템을 통해서만 편집 교차 참조 레코드를 생성하려면 **true**로 설정합니다. 모든 소스 시스템을 통해서 편집 교차 참조 레코드를 생성하려면 **false**로 설정합니다. 기본값은 **true**입니다.

중요: Hub 서버와 처리 서버 모두에 대해 속성을 설정해야 하며 속성 값이 두 속성 파일에서 동일해야 합니다.

<연결 팩터리 이름>.qcf.password

선택 사항입니다. 수동으로 추가해야 합니다. 응용 프로그램 서버에서 설정한 암호를 사용하여 **JMS** 보안을 구성하도록 **MDM Hub**를 구성합니다.

<연결 팩터리 이름>.qcf.username

선택 사항입니다. 수동으로 추가해야 합니다. 응용 프로그램 서버에서 설정한 사용자 이름을 사용하여 **JMS** 보안을 구성하도록 **MDM Hub**를 구성합니다. 메시지 대기열 보안에 대한 자세한 내용은 [“JMS 보안 구성” 페이지 487](#)을 참조하십시오.

databaseld.password

선택 사항입니다. 수동으로 추가해야 합니다. 암호 암호화 도구와 함께 사용할 암호화된 암호를 지정합니다. 암호 암호화 도구 사용에 대한 자세한 내용은 *Multidomain MDM 리소스 키트 가이드*를 참조하십시오.

databaseld.username

선택 사항입니다. 수동으로 추가해야 합니다. 암호 암호화 도구와 함께 사용할 사용자 이름을 지정합니다. 암호 암호화 도구 사용에 대한 자세한 내용은 *Multidomain MDM 리소스 키트 가이드*를 참조하십시오.

encryption.plugin.jar

데이터 암호화 **JAR** 파일의 경로 및 파일 이름입니다. Hub 서버의 데이터 암호화 구성에 대한 자세한 내용은 [“3단계. Hub 서버에 대한 데이터 암호화 구성” 페이지 182](#)을 참조하십시오.

mq.data.change.monitor.thread.start

다중 노드 환경에서 개별 노드에 메시지 대기열 폴링을 사용할지 여부를 지정합니다. 메시지 대기열 폴링을 비활성화하려면 **false**로 설정합니다. 기본값은 **MDM Hub EAR** 파일이 배포된 모든 **Java** 가상 시스템에서 **true**입니다.

searchQuery.buildBvtTemp.MaxRowCount

선택 사항입니다. 수동으로 추가해야 합니다. **BVT**를 계산하는 동안 **GetOneHop API**에서 사용하는 최대 레코드 수를 지정합니다. 기본값은 **5000**입니다. **GetOneHop**에 대한 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

sif.api.hm.flyover.max.record.count

선택 사항입니다. 수동으로 추가해야 합니다. 계층 보기 관계 테이블에 표시되는 관계 레코드 수를 제한하려면 최대 레코드 개수를 설정합니다. 기본값은 **10000**입니다.

서버 속성을 업데이트한 후에는 스키마의 유효성을 검사하고 **Data Director** 응용 프로그램을 재배포해야 합니다. 계층 보기 관계 테이블 레코드에 대한 자세한 내용은 *Multidomain MDM Data Director 구현 가이드*를 참조하십시오.

sif.jvm.heap.size

선택 사항입니다. 수동으로 추가해야 합니다. **API**의 기본 힙 크기(**MB**)를 설정합니다. 기본값은 **256**입니다.

sif.search.result.query.temptableTimeToLive.seconds

선택 사항입니다. 수동으로 추가해야 합니다. GetOneHop에만 사용됩니다. 검색 쿼리를 실행하는 동안 임시 테이블에 데이터가 존재하는 시간(초)을 지정합니다. 기본값은 30입니다. GetOneHop에 대한 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

sip.hm.entity.font.size

선택 사항입니다. 수동으로 추가해야 합니다. 계층 관리자의 글꼴 크기를 설정합니다. 6부터 100 사이의 값을 설정할 수 있습니다. 계층 관리자 속성 설정에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

sip.hm.entity.max.width

선택 사항입니다. 수동으로 추가해야 합니다. 계층 관리자의 최대 항목 상자 너비를 설정합니다. 20부터 5000 사이의 값을 설정할 수 있습니다. 계층 관리자 속성 설정에 대한 자세한 내용은 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

sip.lookup.dropdown.limit

데이터 관리자 도구와 병합 관리자 도구의 메뉴에 표시되는 항목 수입니다. 이 속성에는 최소 또는 최대 제한이 없습니다. 기본값은 100입니다.

cmx.match.training.confidence.threshold

선택 사항입니다. 프로비저닝 도구에서 일치 규칙 집합을 생성하는 데 필요한 최소 일치 신뢰도 점수입니다. 기본값은 85입니다.

cmx.match.training.data.encoding

선택 사항입니다. 프로비저닝 도구에서 일치 학습을 위한 인코딩을 구성합니다. 일치 학습에 대한 인코딩을 활성화하려면 1로 설정합니다. 기본값은 0입니다.

cmx.server.match.server_encoding 속성에서 사용하는 동일한 값을 사용하고 있는지 확인합니다.

일괄 처리 속성

다음 속성은 일괄 프로세스에 영향을 미칩니다.

cmx.server.automerge.block_size

자동 병합 일괄 처리의 블록 크기입니다. 기본값은 250입니다.

cmx.server.automerge.threads_per_job

자동 병합 일괄 처리의 스레드 수입니다. 권장하는 최대값은 $n-1$ 입니다. 여기서 n 은 Hub 서버에 사용 가능한 CPU 수입니다. 기본값은 1입니다.

cmx.server.batch.acceptunmatchedrecordsasunique.block_size

각 블록에서 일치되지 않은 레코드를 고유 레코드로 허용 일괄 작업에 대해 처리할 최대 레코드 수입니다. 기본값은 250입니다.

cmx.server.batch.acceptunmatchedrecordsasunique.threads_per_job

MDM Hub에서 일치되지 않은 레코드를 고유 레코드로 허용 일괄 작업을 처리할 때 사용하는 스레드 수입니다. 기본값은 20입니다.

cmx.server.batch.batchunmerge.block_size

일괄 병합 해제 프로세스의 블록 크기입니다. 기본값은 250입니다.

cmx.server.batch.load.block_size

로드 작업에 대해 각 블록에서 처리할 최대 레코드 수입니다. 기본값은 250입니다.

cmx.server.batch.recalculate.block_size

BVT 다시 계산 및 유효성 다시 검사 작업에 대해 각 블록에서 처리할 최대 레코드 수입니다. 기본값은 250입니다.

cmx.server.batch.threads_per_job

MDM Hub에서 로드를 처리하고, BVT를 다시 계산하고, 자동 병합 일괄 작업을 제외한 일괄 작업의 유효성을 다시 검사하는 데 사용하는 스레드 수입니다. 또한 MDM Hub에서 일괄 병합 해제 프로세스에 사용하는 스레드 수를 지정합니다.

권장하는 최대값은 $n-1$ 입니다. 여기서 n 은 Hub 서버에 사용 가능한 CPU 수입니다. 기본값은 10입니다.

cmx.server.batch.max_concurrent_job_groups

처리할 최대 동시 가져오기 작업 그룹 수입니다. 기본값은 10입니다.

cmx.server.jobControl.noOfDays

선택 사항입니다. 수동으로 추가해야 합니다. Hub 콘솔의 일괄 그룹 도구에서 일괄 그룹 작업 로그에 대해 처리할 기록의 일 수입니다. 기본값은 -1이고 로그에 모든 기록 세부 정보를 포함하는 것을 나타냅니다.

cmx.server.strp_clean.execution_mode

선택 사항입니다. 수동으로 추가해야 합니다. 일치 키 테이블에 대한 백그라운드 정리 프로세스의 작업 범위를 구성합니다.

작업 범위의 경우 다음 값 중 하나를 사용합니다.

- ALL. 등록된 모든 연산 참조 저장소의 모든 일치 키 테이블에서 `invalid_ind=1`인 일치 토큰을 검색합니다.
- CONFIGURED_ORs. 지정한 모든 연산 참조 저장소의 모든 일치 키 테이블에서 `invalid_ind=1`인 일치 토큰을 검색합니다. 작업 범위를 CONFIGURED_ORs로 설정한 경우 `cmx.server.strp_clean.ors` 속성을 `cmxserver.properties` 파일에 추가합니다.
- CONFIGURED_STRP. 특정 연산 참조 저장소의 특정 기본 개체에 대한 일치 키 테이블에서 `invalid_ind=1`인 일치 토큰을 검색합니다. 작업 범위를 CONFIGURED_STRP로 설정한 경우 `cmx.server.strp_clean.strp` 속성을 `cmxserver.properties` 파일에 추가합니다.

cmx.server.strp_clean.ors

선택 사항입니다. 수동으로 추가해야 합니다. 백그라운드 정리 프로세스를 실행하여 유효하지 않은 일치 토큰을 삭제해야 하는 연산 참조 저장소의 이름을 지정합니다. 예를 들어 `cmx_ors1` 및 `cmx_ors2`의 모든 일치 키 테이블에서 `invalid_ind=1`인 일치 토큰을 삭제하려면 `cmx.server.strp_clean.ors=cmx_ors1,cmx_ors2`를 추가합니다.

cmx.server.strp_clean.strp

선택 사항입니다. 수동으로 추가해야 합니다. 백그라운드 정리 프로세스를 실행하여 일치 키 테이블을 정리해야 하는 연산 참조 저장소 및 기본 개체 조합을 지정합니다. 예를 들어 `cmx_ors1`의 BO1 및 `cmx_ors2`의 BO2에 대한 일치 키 테이블에서 `invalid_ind=1`인 일치 토큰을 삭제하려면 `cmx.server.strp_clean.strp=cmx_ors1.C_BO1,cmx_ors2.C_BO2`를 추가합니다.

cmx.server.strp_clean.delete_records_count

선택 사항입니다. 수동으로 추가해야 합니다. 일치 키 테이블에서 정리할 레코드 수를 지정합니다.

cmx.server.strp_clean.retry_sec

선택 사항입니다. 수동으로 추가해야 합니다. MDM Hub가 일치 키 테이블에서 유효하지 않은 일치 토큰이 있는 레코드를 검색할 기간(초)을 지정합니다. 기본값은 60입니다.

cmx.server.strp_clean.threads_count

선택 사항입니다. 수동으로 추가해야 합니다. MDM Hub가 일치 키 테이블에서 유효하지 않은 일치 토큰이 있는 레코드를 검색할 때 사용할 스레드 수를 지정합니다. 기본값은 20입니다.

mq.data.change.threads

게시 프로세스 중에 JMS 메시지 처리를 위해 사용하는 스레드의 수입니다. 기본값은 1입니다.

mq.data.change.batch.size

게시 프로세스를 위해 각 일괄 작업에서 처리하는 JMS 메시지의 수입니다. 기본값은 500입니다.

mq.data.change.timeout

JMS 메시지를 처리하는 데 허용되는 시간(초)입니다. 기본값은 120입니다.

보안 관리자 속성

다음 속성은 보안 관리자에 영향을 줍니다.

cmx.server.clock.tick_interval

클록 틱당 시간(밀리초)입니다. 기본값은 60000입니다.

cmx.server.provider.userprofile.cacheable

데이터를 캐시할 수 있는지 여부를 지정합니다. 데이터를 캐싱하면 성능 개선에 도움이 됩니다. 데이터 캐싱을 활성화하려면 **true**로 설정합니다. 데이터 캐싱을 비활성화하려면 **false**로 설정합니다. 기본값은 **true**입니다.

cmx.server.provider.userprofile.expiration

캐시된 데이터가 만료될 때까지 존속되는 기간(밀리초)입니다. 기본값은 60000입니다.

cmx.server.provider.userprofile.lifespan

캐시된 데이터가 만료될 때까지 존속되는 기간(밀리초)입니다. 기본값은 60000입니다.

cmx.server.sam.cache.resources.refresh_interval

데이터베이스에서 SAM(보안 액세스 관리자) 리소스 데이터를 다시 로드하는 간격(클록 틱 수)입니다. 기본값은 5입니다. 새로 고침 간격을 변경하는 데 대한 자세한 내용은 *Multidomain MDM 보안 가이드*를 참조하십시오. 클록 틱당 시간(밀리초)을 지정하려면 **cmx.server.clock.tick_interval** 속성을 사용합니다.

cmx.server.sam.cache.user_profile.refresh_interval

데이터베이스에서 사용자 프로필의 SAM 리소스 데이터를 다시 로드하는 간격(클록 틱 수)입니다. 기본값은 30입니다. 클록 틱당 시간(밀리초)을 지정하려면 **cmx.server.clock.tick_interval** 속성을 사용합니다.

Oracle 데이터베이스 사용자 종료

다음 속성은 Oracle 데이터베이스에 사용할 수 있습니다. 이러한 속성을 사용하려면 **cmxserver.properties** 파일 및 **cmxcleanse.properties** 파일 둘 모두에 속성을 추가해야 합니다.

cmx.server.dbuserexit.load.PostLoadUserExit

선택 사항입니다. MDM Hub가 로드 프로세스 후 데이터베이스 **postload** 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 **true**로 설정합니다. 기본값은 **false**입니다. PL/SQL 사용자 종료 활성화에 대한 자세한 내용은 현재 환경에 대한 *Multidomain MDM 업그레이드 가이드*를 참조하십시오.

cmx.server.dbuserexit.put.PostLoadUserExit

선택 사항입니다. MDM Hub가 Put 요청을 수행한 후에 데이터베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 **true**로 설정합니다. 기본값은 **false**입니다.

cmx.server.dbuserexit.PostMergeUserExit

선택 사항입니다. MDM Hub가 병합 요청 또는 자동 병합 일괄 작업을 수행한 후에 데이터베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 **true**로 설정합니다. 기본값은 **false**입니다.

cmx.server.dbuserexit.PreUnmergeUserExit

선택 사항입니다. MDM Hub가 병합 해제 요청 또는 병합 해제 일괄 작업을 수행하기 전에 데이터베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 **true**로 설정합니다. 기본값은 **false**입니다.

cmx.server.dbuserexit.PostUnmergeUserExit

선택 사항입니다. MDM Hub가 병합 해제 요청 또는 병합 해제 일괄 작업을 수행한 후에 데이터베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 **true**로 설정합니다. 기본값은 **false**입니다.

cmx.server.dbuserexit.PreUserMergeAssignment

선택 사항입니다. MDM Hub가 병합 해제된 레코드를 검토 목적으로 할당하기 전에 데이터베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 **true**로 설정합니다. 기본값은 **false**입니다.

cmx.server.dbuserexit.AssignTask

선택 사항입니다. MDM Hub가 사용자에게 태스크를 할당하기 전에 데이터베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 **true**로 설정합니다. 기본값은 **false**입니다.

cmx.server.dbuserexit.GetAssignableUserForTask

선택 사항입니다. MDM Hub가 사용자에게 태스크를 할당하기 전에 데이터베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 **true**로 설정합니다. 기본값은 **false**입니다.

Data Director 일반 속성

다음 속성은 Data Director의 동작에 영향을 줍니다.

참고: 다음 서버 속성을 업데이트한 후에는 스키마의 유효성을 검사하고 **IDD 응용 프로그램을 재배포해야 합니다.**

com.siperian.dsapp.mde.common.idd2cocs.Many2ManyChild.name.version

수동으로 추가해야 합니다. MDM 관리자가 비즈니스 항목 스키마를 생성하는 경우 일부 하위 수준 제목 영역 이름의 대문자가 소문자로 변경됩니다. 대/소문자를 유지하려면 속성을 **10.2**로 설정합니다.

case.insensitive.search

true로 설정할 경우 기본 개체의 개별 열에 대해 대/소문자를 구분하지 않도록 하는 특성을 설정하여 Data Director에서 대/소문자를 구분하지 않는 쿼리 검색을 활성화할 수 있습니다. 이 설정이 활성화된 각 열에 대해 새 인덱스가 생성됩니다. 인덱스 관리는 자체적으로 성능 오버헤드가 발생하므로 이 속성은 주의해서 사용해야 합니다. 기본값은 **false**입니다.

cmx.bdd.redirect_to_login_after_logout

선택 사항입니다. 수동으로 추가해야 합니다. Data Director에서만 Google SSO 인증용으로 사용됩니다. 사용자가 로그아웃할 경우 Data Director가 로그인 화면으로 돌아가도록 구성하려면 **true**로 설정합니다. 사용자가 로그아웃할 경우 Data Director가 기본 로그아웃 화면으로 리디렉션되도록 구성하려면 **false**로 설정합니다. 기본값은 **false**입니다.

cmx.bdd.server.traffic.compression_enabled

Data Director 서버 트래픽 압축이 활성화되는지 여부를 지정합니다. 트래픽을 압축하면 성능 개선에 도움이 됩니다. 압축을 활성화하려면 **true**로 설정합니다. 압축을 비활성화하려면 **false**로 설정합니다. 기본값은 **true**입니다.

cmx.dataview.enabled

MDM 관리자가 제목 영역 모델을 구현하는 경우 IDD 사용자는 **데이터** 탭을 사용하여 레코드를 검색, 편집 및 관리합니다. 이 옵션은 **데이터** 탭 및 관련 요소가 IDD 응용 프로그램에 표시되는지 여부를 지정합니다.

새 설치의 경우 기본값은 **false**입니다. 업그레이드의 경우 이 속성이 설정되어 있었다면 업그레이드 전 값이 유지됩니다. 이 속성을 설정하지 않으면 기본값이 **true**입니다.

cmx.dataview.enabled=true인 경우 다음 사용자 인터페이스가 요소가 IDD 응용 프로그램에 표시됩니다.

- **새로 만들기** 탭 - 제목 영역이 있는 **새로 만들기** 창이 열립니다.
- **데이터** 탭 - 다음과 같은 임시 인터페이스를 포함합니다.
 - 레코드를 편집하고 관리하기 위한 제목 영역 레코드 보기가 있는 데이터 작업 공간 탭
 - 검색 쿼리 및 검색 쿼리 결과가 있는 검색 탭
 - 태스크를 관리하기 위한 태스크 탭
- 다른 보기의 메뉴에 제공되는 **데이터** 보기에 대한 링크
- 사용자 지정 탭(구성된 경우)

cmx.dataview.enabled 및 cmx.e360.view.enabled 속성이 true로 설정되었을 때 제목 영역이 있는 Data Director에 대해 교차 참조 보기, 기록 보기 및 일치 레코드 보기를 활성화하려면 cmx.e360.match_xref.view.enabled 속성을 false로 설정합니다.

MDM 관리자가 Entity 360 프레임워크를 구현하는 경우 Data Director 사용자는 **검색** 상자를 사용하여 레코드를 찾고 항목 탭을 사용하여 마스터 데이터를 편집 및 관리합니다. 이 경우, 유사한 기능으로 인한 사용자의 혼동을 방지하기 위해 **데이터** 탭 및 관련 요소를 숨길 수 있습니다. 예를 들어 cmx.e360.view.enabled=true를 설정하면 cmx.dataview.enabled=false를 설정합니다.

cmx.bdd.enable_url_authentication

선택 사항입니다. 수동으로 추가해야 합니다. Data Director에서 URL 인증을 활성화합니다. 활성화된 경우 사용자는 Data Director에서 로그인 시 자신의 사용자 이름과 암호를 URL에 전달합니다. 인증을 활성화하려면 true로 설정합니다. 인증을 해제하려면 false로 설정합니다. 기본값은 false입니다.

cmx.bdd.password_blowfish_encrypted

선택 사항입니다. 수동으로 추가해야 합니다. Data Director의 URL에 대해 인증이 활성화된 경우 사용자 암호에 대한 Blowfish 암호화를 활성화합니다. 활성화된 경우, 암호가 Data Director의 URL에서 노출되지 않습니다. 암호화를 활성화하려면 true로 설정합니다. 암호화를 해제하려면 false로 설정합니다. 기본값은 false입니다.

cmx.display.deployed.invalid.met.app

연산 참조 저장소의 메타데이터가 올바르지 않으면 배포된 응용 프로그램 목록이 Data Director에 표시되지 않습니다. 서로 다른 올바른 연산 참조 저장소를 사용하는 응용 프로그램도 사용할 수 없습니다. 배포된 응용 프로그램 목록을 표시하려면 이 속성을 추가하고 true로 설정합니다.

cmx.e360.view.enabled

MDM 관리자가 Entity 360 프레임워크를 구현하는 경우 IDD 사용자는 **검색** 상자를 사용하여 레코드를 찾고 항목 탭을 사용하여 레코드를 편집 및 관리합니다. 새 설치의 경우 기본값은 true입니다. 업그레이드의 경우 이 속성이 설정되어 있었다면 업그레이드 전 값이 유지됩니다. 이 속성을 설정하지 않으면 기본값이 false입니다.

cmx.e360.view.enabled=true인 경우 다음 사용자 인터페이스 요소가 Data Director 응용 프로그램에 표시됩니다.

- **새로 만들기** 탭 - 비즈니스 항목이 있는 **새로 만들기** 창이 열립니다.
- 태스크를 관리하기 위한 **태스크 관리자** 탭
- 검색 결과가 있는 **검색** 탭
- 비즈니스 항목 레코드를 편집하고 관리하기 위한 항목 탭. 항목 탭은 새 비즈니스 항목 레코드를 추가하거나 검색 결과에서 비즈니스 항목 레코드를 열면 나타납니다. 탭의 레이블은 작업 공간에서 열리는 작업을 기반으로 동적으로 표시됩니다.
- 다른 보기의 메뉴에 제공되는 **항목 보기**에 대한 링크

- 사용자 지정 탭(구성된 경우)

cmx.dataview.taskmanager.enabled

선택 사항입니다. `cmx.e360.view.enabled` 속성이 `false`로 설정된 경우에만 적용됩니다. 제목 영역을 사용하는 Data Director 응용 프로그램에서 태스크 관리자를 표시할지 여부를 나타냅니다. 태스크 관리자를 표시하려면 `true`로 설정합니다. 기본값은 `false`입니다.

`cmx.dataview.taskmanager.enabled`가 `false`로 설정되어 있고 프로비저닝 도구에서 Data Director 응용 프로그램에 대한 홈 페이지를 생성하지 않은 경우 응용 프로그램은 레거시 시작 페이지를 표시합니다.

cmx.e360.match_xref.view.enabled

비즈니스 항목이 있는 Data Director에 대해 교차 참조 보기 및 일치 레코드 보기를 활성화할지 여부를 지정합니다. 보기를 활성화하려면 `true`로 설정합니다. 기본값은 `true`입니다.

`cmx.dataview.enabled` 및 `cmx.e360.view.enabled` 속성이 `true`로 설정되었을 때 제목 영역이 있는 Data Director에 대해 교차 참조 보기, 기록 보기 및 일치 레코드 보기를 활성화하려면 `cmx.e360.match_xref.view.enabled` 속성을 `false`로 설정합니다.

cmx.server.override_orstitle

선택 사항입니다. 수동으로 추가해야 합니다. Data Director에 로그인할 때 표시할 현재 태스크의 기본 설정된 기본 제목을 지정합니다. `cmxserver.properties` 파일에서 `cmx.server.override_orstitle` 속성을 기본 설정된 제목으로 설정합니다.

예를 들어 속성을 All Subject Areas로 설정한 경우 화면에 제목이 **Tasks for All Subject Areas**로 표시됩니다.

cmx.server.be-import.task-limit

태스크 승인 워크플로우의 트리거를 위해 사용자가 가져올 수 있는 최대 레코드 수를 지정합니다.

`cmx.server.be-import.task-limit` 값을 기본 설정 최대 수로 설정합니다. 예를 들어 사용자가 최대 10,000개의 레코드를 가져오고 태스크 승인 워크플로우를 트리거하려면 `cmx.server.be-import.task-limit=10000`을 설정합니다. 사용자가 10,000개를 초과하는 레코드를 가져오려고 하면 태스크 승인 워크플로우가 트리거되지 않고 오류가 표시됩니다.

cmx.server.find-replace.record-limit

사용자가 대량 작업으로 교체할 수 있는 레코드의 최대 수를 지정합니다. `cmx.server.find-replace.record-limit` 속성을 기본 설정 최대 수로 설정합니다. 예를 들어 속성을 `cmx.server.find-replace.record-limit=10000`으로 설정하는 경우 사용자는 최대 10,000개의 레코드를 찾아서 교체할 수 있습니다.

cmx.server.find-replace.task-limit

태스크 승인 워크플로우를 트리거하는 교체된 레코드의 최대 수를 지정합니다. `cmx.server.find-replace.task-limit` 속성을 기본 설정 최대 수로 설정합니다. 예를 들어 속성을 `cmx.server.find-replace.task-limit=500`으로 설정하는 경우 사용자가 최대 500개의 레코드를 교체하면 태스크 승인 워크플로우가 트리거됩니다. 사용자가 500개를 초과하는 레코드를 교체하려고 하면 오류가 표시됩니다.

cmx.server.find-replace.entity-record-limit

사용자가 스마트 검색 화면에서 찾기 및 교체 화면으로 복사할 수 있는 레코드의 최대 수를 지정합니다. `cmx.server.find-replace.entity-record-limit` 속성을 기본 설정 최대 수로 설정합니다. 예를 들어 속성을 `cmx.server.find-replace.entity-record-limit=1000`으로 설정하는 경우 사용자는 스마트 검색 화면에서 찾기 및 교체 화면으로 최대 1,000개의 레코드를 복사할 수 있습니다. 사용자가 1,000개를 초과하는 레코드를 복사하려고 하면 오류가 표시됩니다.

cmx.file_import.job_group.ttl

파일 가져오기 작업 그룹이 삭제되기 전에 MDM Hub에 저장되는 최대 시간을 지정합니다. 기본값은 180day입니다. 값 뒤에 접미사를 추가해야 합니다. 접미사 옵션으로는 day, hour, min 또는 sec이 있습니다. 예를

들어 속성을 `cmx.file_import.job_group.ttl=180day`로 설정하는 경우 가져오기 작업 그룹은 MDM Hub에 180일 동안 저장됩니다.

`cmx.file_import.job_group_control.ttl`

파일 가져오기 작업 그룹 제어 로그가 삭제되기 전에 MDM Hub에 저장되는 최대 시간을 지정합니다. 기본값은 30day입니다. 값 뒤에 접미사를 추가해야 합니다. 접미사 옵션으로는 **day**, **hour**, **min** 또는 **sec**이 있습니다. 예를 들어 속성을 `cmx.file_import.job_group_control.ttl=30day`로 설정하는 경우 파일 가져오기 작업 그룹 제어 로그는 MDM Hub에 30일 동안 저장됩니다.

`cmx.file_import.mapping.temp.ttl`

파일 가져오기 매핑이 삭제되기 전에 MDM Hub 임시 저장소에 저장되는 최대 시간을 지정합니다. 기본값은 20min입니다. 값 뒤에 접미사를 추가해야 합니다. 접미사 옵션으로는 **day**, **hour**, **min** 또는 **sec**이 있습니다. 예를 들어 속성을 `cmx.file_import.mapping.temp.ttl=20min`으로 설정하는 경우 가져오기 작업 매핑은 MDM Hub에 20분 동안 저장됩니다.

`cmx.file_import.mapping.system.ttl`

파일 가져오기 매핑이 삭제되기 전에 MDM Hub 영구 저장소에 저장되는 최대 시간을 지정합니다. 기본값은 20day입니다. 값 뒤에 접미사를 추가해야 합니다. 접미사 옵션으로는 **day**, **hour**, **min** 또는 **sec**이 있습니다. 예를 들어 속성을 `cmx.file_import.mapping.system.ttl=20day`로 설정하는 경우 가져오기 작업 매핑은 MDM Hub에 20일 동안 저장됩니다.

`cmx.file.allowed_file_extensions`

Data Director 응용 프로그램의 레코드 또는 태스크에 연결할 수 있는 파일의 확장자를 나열합니다. 기본적으로 **.pdf** 및 **.jpg** 파일을 연결할 수 있습니다. 여러 확장자를 지정하려면 각 값을 쉼표로 구분하십시오.

예: `cmx.file.allowed_file_extensions=pdf,jpg,png,txt,zip,exe`

`cmx.file.max_file_size_mb`

Data Director 응용 프로그램에 연결할 수 있는 파일의 크기 제한을 지정합니다.

참고: 제목 영역 데이터 모델을 사용하는 Data Director 응용 프로그램의 정적 크기 제한은 20MB입니다. 20MB를 초과하는 크기 제한을 지정하는 경우 제목 영역 데이터 모델을 사용하는 Data Director 응용 프로그램에서 20MB의 정적 크기 제한이 유지됩니다. 비즈니스 항목 데이터 모델을 사용하는 Data Director 응용 프로그램에는 `cmx.file.max_file_size_mb` 속성에 정의된 크기 제한이 유지됩니다.

Data Director 검색 속성

다음 속성은 Data Director의 검색 동작에 영향을 줍니다.

`ex.max.conn.per.host`

호스트에 연결하려는 Elasticsearch 노드의 최대 수를 설정합니다. 호스트의 Elasticsearch 클러스터 노드 수로 설정합니다.

`ex.max.threads`

Elasticsearch 클러스터의 각 노드에 사용할 Apache 비동기식 비차단 수신기의 최대 스레드 수를 설정합니다. 기본값은 1입니다.

이 값은 Informatica 글로벌 고객 지원 센터에서 제안한 경우에만 변경합니다.

`es.index.refresh.interval`

처음에 스마트 검색 데이터 인덱싱 일괄 작업이 실행된 후 Elasticsearch에서 데이터 변경 내용을 커밋하는 간격(초)을 설정합니다. 이 시간 간격 후에 데이터를 검색에 사용할 수 있습니다. 기본값은 30입니다.

이 속성은 초기 인덱싱 중에 발생하는 높은 인덱싱 볼륨에 영향을 미칩니다. 이 값은 Informatica 글로벌 고객 지원 센터에서 제안한 경우에만 변경합니다.

cmx.task.search.records.return

사용자가 비즈니스 항목이 있는 **Data Director**의 태스크 관리자에서 태스크를 검색할 때 **Elasticsearch** 페이지 매기기를 제어합니다. 기본값은 1000입니다.

이 값은 **Informatica** 글로벌 고객 지원 센터에서 제안한 경우에만 변경합니다.

cmx.server.batch.smartsearch.initial.block_size

스마트 검색 데이터 초기 인덱스 일괄 작업이 각 블록에서 처리할 수 있는 최대 레코드 수입입니다. 기본값은 250입니다. 대규모 데이터 집합을 인덱싱할 때는 레코드 수를 늘리십시오. 권장되는 최대값은 1000입니다.

cmx.server.match.max_time_searcher

선택 사항입니다. 수동으로 추가해야 합니다. 한 검색에 허용되는 최대 실행 시간을 지정합니다. 기본값은 99999999초입니다.

cmx.server.remove_duplicates_in_search_query_results

중복 레코드를 검색 쿼리 결과에 표시할지 여부를 지정합니다. 중복 레코드를 검색 쿼리 결과에 표시하려면 **true**로 설정합니다. 검색 쿼리 결과에서 중복 레코드를 숨기려면 **false**로 설정합니다. 기본값은 **false**입니다.

cmx.server.enrichcopager.thread_pool

속성을 수동으로 추가합니다. **EnrichCoPager** 속성이 스레드 풀에서 병렬 **ReadCO** 작업을 수행하기 위해 사용하는 스레드 수를 설정합니다. 기본값은 30입니다. 스레드 수를 1로 설정하면 속성이 비활성화됩니다.

cmx.server.enrichcopager.min_rec_for_multithreading

EnrichCoPager 속성에서 다중 스레드를 사용하기 전에 반환할 최소 레코드 수를 설정합니다. 기본값은 2입니다.

cmx.ss.enabled

검색을 활성화할지 여부를 나타냅니다. 새 설치의 경우 기본값은 **true**입니다. 업그레이드의 경우 이 속성이 설정되어 있었다면 업그레이드 전 값이 유지됩니다. 이 속성을 설정하지 않으면 기본값이 **false**입니다.

search.provisioning.numshards

선택 사항입니다. **Elasticsearch** 환경에 생성할 **shard** 수입입니다. 최대 **shard** 수 및 총 노드 수에 따라 값이 달라집니다. 예를 들어, 최대 **shard** 수가 1이고 노드 수가 3인 경우 **shard**를 3개 작성할 수 있습니다. 기본값은 **Hub** 서버의 총 수입입니다.

search.provisioning.numreplicas

선택 사항입니다. 서로 다른 노드에 생성할 **Elasticsearch** 엔진 문서의 사본 수입입니다. 서로 다른 노드의 **shard**에 여러 개의 문서 사본을 생성하려면 복제 계수를 사용합니다. 하나 이상의 노드가 예기치 않게 종료된 경우 고가용성을 달성하려면 여러 개의 문서 사본이 필요합니다. 예를 들어 복제 계수가 2라면 두 개의 노드에서 두 개의 문서 사본을 얻을 수 있습니다. **Elasticsearch**의 경우 기본값은 0입니다.

sif.search.result.drop.batch.interval.milliseconds

선택 사항입니다. 수동으로 추가해야 합니다. **SearchResultManager** 데몬이 각 검색 결과 정리 일괄 작업을 처리한 후 일시 중지하는 간격(밀리초)을 지정합니다. 기본값은 0입니다.

sif.search.result.drop.batch.record.count

선택 사항입니다. 수동으로 추가해야 합니다. **SearchResultManager** 데몬이 한 번에 정리하는 캐시된 검색의 수를 지정합니다. 기본값은 200입니다.

sif.search.result.query.timeToLive.seconds

선택 사항입니다. 수동으로 추가해야 합니다. 사용되지 않은 쿼리가 캐시된 상태로 남아 있는 시간(초)을 지정합니다. 기본값은 900입니다.

sif.search.result.refresh.interval.seconds

선택 사항입니다. 수동으로 추가해야 합니다. 캐시된 검색의 정리 프로세스를 실행한 후 **SearchResultManager** 데몬이 일시 중지되는 간격(초)을 지정합니다. 기본값은 1입니다. SIF 검색 API 구성에 대한 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

ssl.keyStore

응용 프로그램 서버의 HTTPS 포트를 사용하여 Hub 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 키 저장소 파일의 절대 경로 및 파일 이름입니다.

ssl.keyStore.password

응용 프로그램 서버의 HTTPS 포트를 사용하여 Hub 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 키 저장소 파일에 대한 일반 텍스트 암호입니다.

ssl.trustStore

응용 프로그램 서버의 HTTPS 포트를 사용하여 Hub 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 트러스트 저장소 파일의 절대 경로 및 파일 이름입니다.

ssl.trustStore.password

응용 프로그램 서버의 HTTPS 포트를 사용하여 Hub 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 트러스트 저장소 파일에 대한 일반 텍스트 암호입니다.

계층 REST API 속성

다음 속성은 비즈니스 항목 서비스의 계층 REST API에 영향을 줍니다.

cmx.server.hierarchy.max-search-depth

계층 REST API를 사용하여 계층 경로를 찾을 때 검색되는 최대 깊이입니다. 기본값은 100입니다.

cmx.server.hierarchy.max-search-width

계층 REST API를 사용하여 내보낼 때 포함할 검색된 계층의 최대 너비입니다. 기본값은 1000000입니다.

com.informatica.mdm.bulk.relationship.changes.limit

대량 태스크 관리 REST API를 사용할 때 요청의 최대 변경 수입니다. 기본값은 1000입니다.

Data Director 태스크, 워크플로우 및 보고서 속성

다음 속성은 태스크, 검토 프로세스 워크플로우 및 보고서에 영향을 줍니다.

activevos.jndi

선택 사항입니다. 수동으로 추가해야 합니다. 보고 서비스를 ActiveVOS에 연결할 JNDI 조회 문자열을 지정합니다. ActiveVOS EAR 파일을 편집하여 JNDI 조회 문자열을 사용자 지정한 경우 사용자 지정 JNDI 조회 문자열을 지정하려면 이 속성을 사용합니다. 기본 JNDI 조회 문자열은 `java:/jdbc/ActiveVOS`입니다.

activevos.merge.workflow.service.name

Informatica ActiveVOS에 대한 MDM 서비스 호출의 이름을 지정해야 합니다. 이 속성은 기본적으로 정의되어 있지 않습니다. 속성이 정의되어 있지 않으면 자동 병합 태스크가 생성되지 않습니다.

activevos.workflow.startup.timeout.seconds

Informatica ActiveVOS가 태스크를 생성하고 태스크 ID를 반환할 때까지 대기하는 시간(초)입니다. 기본값은 10입니다.

cmx.e360.BPMProcess.view.enabled

선택 사항입니다. 수동으로 추가해야 합니다. 태스크 관리자에서 ActiveVOS abAdmin 역할이 할당된 사용자에게 대해 태스크에 연결된 워크플로우 다이어그램을 표시할지 여부를 나타냅니다. 워크플로우 다이어그램을 표시하려면 `true`로 설정합니다. 기본값은 `false`입니다.

cmx.e360.BPMProcess.view.autologout.seconds

선택 사항입니다. 수동으로 추가해야 합니다. 태스크 관리자에서 워크플로우 다이어그램에 액세스할 때 ActiveVOS 세션을 활성 상태로 유지할 시간(초)입니다. 지정된 기간이 지나면 세션이 종료됩니다. 기본값은 30입니다.

cmx.server.task.grouppotentialmergebyruleid

선택 사항입니다. 수동으로 추가해야 합니다. 여러 일치 항목이 생성되는 수동 일치 태스크에서 여러 태스크를 동일한 ROWID를 사용하여 생성하도록 지정합니다. 각 일치 항목에 대한 단일 태스크를 생성하려면 false로 설정합니다. 기본값은 true입니다.

sip.task.assignment.interval

자동 태스크 할당 간격(분)입니다. 자동 태스크 할당을 비활성화하려면 0으로 설정합니다. 기본값은 0입니다.

sip.task.assignment.start.delay

MDM Hub가 초기화된 후 자동 태스크 할당이 시작될 때까지 기다리는 시간(단위: 분)입니다. 지연을 구성하지 않으면 사용자가 태스크를 생성할 때 오류가 발생할 수 있습니다. 기본값은 10분입니다.

sip.task.digest.interval

태스크 알림 간격(시간)입니다. 태스크 알림을 비활성화하려면 0으로 설정합니다. 기본값은 0입니다.

sip.task.maximum.assignment

자동 태스크 할당이 활성화된 경우 각 사용자에게 자동으로 할당되는 태스크의 수입니다. 기본값은 25입니다.

task.creation.batch.size

선택 사항입니다. 수동으로 추가해야 합니다. 자동 태스크 할당 프로세스가 반복될 때마다 각 일치 테이블에서 처리할 최대 레코드 수를 설정합니다. 기본값은 1000입니다.

병합 태스크 속성 구성에 대한 자세한 내용은 *Multidomain MDM 비즈니스 프로세스 관리자 어댑터 SDK 구현 가이드*를 참조하십시오.

task.creation.maximum

선택 사항입니다. 수동으로 추가해야 합니다. MDM Hub에서 각 일치 테이블에 대해 생성하는 최대 태스크 수를 설정합니다. 기본값은 50입니다. 태스크 수가 이 값을 초과하면 일치 테이블에 관련된 수의 태스크를 닫기 전까지 해당 일치 테이블의 레코드에 대해 병합 태스크를 더 이상 생성할 수 없습니다.

Siperian 워크플로우 엔진 메일 서버 속성

Siperian 워크플로우 엔진을 사용할 경우 다음 속성은 태스크 알림에 사용되는 메일 서버의 동작에 영향을 줍니다.

mail.smtp.auth

지정된 메일 서버에서 전송 메시지에 대한 인증을 필요로 하는지 여부를 결정합니다. MDM Hub 메일 서버를 사용할 경우 mail.smtp.auth를 true로 설정하십시오. 기본값은 false입니다.

태스크 알림 전자 메일 구성에 대한 자세한 내용은 *Multidomain MDM Data Director 구현 가이드*를 참조하십시오.

mail.smtp.host

태스크 알림 전자 메일의 메일 서버 이름입니다. 서버 속성을 업데이트한 후에는 스키마의 유효성을 검사하고 Data Director 응용 프로그램을 재배포해야 합니다. 태스크 알림 전자 메일 구성에 대한 자세한 내용은 *Multidomain MDM Data Director 구현 가이드*를 참조하십시오.

mail.smtp.password

지정된 `mail.smtp.user`의 암호입니다. `mail.smtp.auth`가 `true`인 경우 `mail.smtp.password`에 대한 값을 설정하십시오. 태스크 알림 전자 메일 구성에 대한 자세한 내용은 *Multidomain MDM Data Director 구현 가이드*를 참조하십시오.

mail.smtp.port

메일 서버의 포트 번호입니다. 기본값은 25입니다. 태스크 알림 전자 메일 구성에 대한 자세한 내용은 *Multidomain MDM Data Director 구현 가이드*를 참조하십시오.

mail.smtp.sender

태스크 알림 전자 메일 보낸 사람의 전자 메일 주소를 지정합니다. 태스크 알림 전자 메일 구성에 대한 자세한 내용은 *Multidomain MDM Data Director 구현 가이드*를 참조하십시오.

mail.smtp.user

전송 메일 서버의 사용자 이름입니다. `mail.smtp.auth`가 `true`인 경우 `mail.smtp.user`에 대한 값을 설정하십시오. 태스크 알림 전자 메일 구성에 대한 자세한 내용은 *Multidomain MDM Data Director 구현 가이드*를 참조하십시오.

암호 해시 및 사용자 지정 해시 속성

다음 속성은 암호 해시 및 사용자 지정 해시 알고리즘에 영향을 미칩니다.

password.security.hash.algo

MDM Hub에서 암호를 암호화하는 데 사용되는 해시 알고리즘(`ALGO_NAME`)을 결정합니다. 이 속성은 Hub 서버 설치 과정에서 설정됩니다. **SHA3** 해시 알고리즘을 사용하려면 **SHA3**으로 설정합니다. 사용자 지정 해시 알고리즘에 대해 특수 문자 또는 공백을 포함하지 않는 이름을 설정합니다.

해시 알고리즘 구성에 대한 자세한 내용은 *Multidomain MDM 보안 가이드* 항목을 참조하십시오.

password.security.hash.algo.<ALGO_NAME>.class

`password.security.hash.algo` 속성에 지정된 해시 알고리즘의 기초적인 구현 클래스를 포함합니다. 이 속성은 Hub 서버 설치 과정에서 설정됩니다.

password.security.hash.algo.property.<param-name>

선택 사항입니다. 구성된 해시 알고리즘의 사용자 지정 속성을 지정합니다. 기본적으로 **SHA3** 해시 알고리즘의 크기 속성을 지정합니다. 224, 256, 384, 512 중 하나의 값으로 설정합니다. 기본값은 512입니다.

com.informatica.mdm.security.certificate.provider.class

MDM Hub에서 기본 인증서 공급자를 `com.siperian.sam.security.certificate.PKIUtilDefaultImpl`로 설정합니다. 이 속성은 Hub 서버 설치 과정에서 설정됩니다.

informatica.security.unique.id

암호 해시에 사용되는 고객 해시 키입니다. 구분자 없이 최대 32개 16진수 문자 시퀀스를 포함하는 해시 키를 사용하는 것이 좋습니다. 고객 해시 키 사용에 대한 자세한 내용은 *Multidomain MDM 보안 가이드* 항목을 참조하십시오.

중요: 고객 해시 키의 비밀을 보호하십시오. 고객 해시 키 값을 분실하면 모든 암호를 재설정해야 합니다.

보안 구성 유틸리티 속성

보안 구성 유틸리티를 사용하려면 다음 속성을 설정합니다.

mdm.mail.server.host

MDM 관리자가 사용하는 전자 메일 클라이언트의 **SMTP** 서버 호스트로 설정합니다. 예: `smtp.gmail.com`. 암호를 재설정하면 보안 구성 유틸리티가 사용자 계정과 연결된 전자 메일 주소에 새 임시 암호를 보냅니다.

mdm.mail.server.port

MDM 관리자가 사용하는 전자 메일 클라이언트에서 사용하는 포트로 설정합니다.

mdm.mail.server.user

MDM 관리자의 전자 메일 주소로 설정합니다. 예: MDM_Hub_admin@gmail.com.

mdm.mail.server.password

MDM 관리자의 전자 메일 주소에 대한 암호를 입력합니다.

mdm.mail.server.smtpauth

SMTP 인증을 활성화하려면 true로 설정합니다. Gmail SMTP 서버에 연결하는 데 필요합니다.

mdm.mail.server.ttls

TLS 인증을 활성화하려면 true로 설정합니다. Gmail SMTP 서버에 연결하는 데 필요합니다.

샘플 Hub 서버 속성 파일

Hub 서버 속성 파일을 cmxserver.properties라고 합니다.

다음 예제에서는 일반적인 cmxserver.properties 파일의 콘텐츠를 보여 줍니다.

```
# Installation directory
cmx.home=C:/infamdm/Hub_971_DB2/server

# Master database settings
cmx.server.masterdatabase.type=DB2

# Server settings
# Application server type: jboss, websphere or weblogic
cmx.appserver.type=jboss
cmx.appserver.version=7
#Should application server use ejb3 lookup (Jboss7 supports only ejb3 lookup mechanism )
cmx.server.ejb3=true

# Application server hostname. Optional property to be used for deploying MDM into EJB cluster
#cmx.appserver.hostname=clustername

# The following port number depends on appserver type
# default setting: 2809 for websphere, 1099 for jboss5, 4447 for jboss7 7001 for weblogic
cmx.appserver.rmi.port=4447
# default setting: iiop for websphere, jnp for jboss5, remote for jboss7, t3 for weblogic
cmx.appserver.naming.protocol=remote
# default setting: 8880 for websphere only (this is not being used in jboss and weblogic
cmx.appserver.soap.connector.port=
# default setting: 'No' for websphere only (this is not being used in jboss and weblogic
cmx.websphere.security.enabled=
## You can customize location of sas.client.props and ssl.client.props which are used for secured ejb
lookup
#cmx.websphere.security.sas.config.url=https://yourdomain.com:9443/cmx/filesx/Security/Websphere/
sas.client.props
#cmx.websphere.security.ssl.config.url=https://yourdomain.com:9443/cmx/filesx/Security/Websphere/
ssl.client.props
## Or you can just customize file names (default values are sas.client.props and ssl.client.props)
#cmx.websphere.security.sas.config.name=sas.client.props
#cmx.websphere.security.ssl.config.name=ssl.client.props

# enable JBoss EJB security support
#cmx.jboss7.security.enabled=true

# DO NOT EDIT SETTINGS BELOW
cmx.server.datalayer.cleanser.execution=SERVER
```

```

cmx.server.datalayer.cleanse.working_files.location=C:/infamdm/Hub_971_DB2/server/logs
cmx.server.datalayer.cleanse.working_files=LOCAL

# SAM properties
cmx.server.sam.cache.resources.refresh_interval=5
cmx.server.sam.cache.user_profile.refresh_interval=1
cmx.server.clock.tick_interval=60000
cmx.server.provider.userprofile.cacheable=true
cmx.server.provider.userprofile.expiration=60000
cmx.server.provider.userprofile.lifespan=60000

# Setting for dropdown limit
sip.lookup.dropdown.limit=100

#
# Task settings
#
# Number of Hours between task notifications. 0 means that notifications are disabled.
sip.task.digest.interval=0
# Number of Minutes between automated task assignments. 0 means that assignment is disabled.
sip.task.assignment.interval=0
# Maximum number of tasks automatically assigned to each user
sip.task.maximum.assignment=25

#
# Mail server settings for task notification emails
#
mail.smtp.host=
mail.smtp.port=25
mail.smtp.auth=false
mail.smtp.sender=siperian_task_notification@siperian.com
# Use the following if your smtp server requires authentication.
#mail.smtp.user=
#mail.smtp.password=

# interval sleeping between polling all servers in seconds, default=10, 0 will disable
cmx.server.poller.monitor.interval=30

#MET properties
cmx.server.met.max_send_size=9000000

# BDD traffic compression option
cmx.bdd.server.traffic.compression_enabled=true

# Sif property to remove duplicates from the search query results
cmx.server.remove_duplicates_in_search_query_results=false

# The Case Insensitive Search feature can be disabled by setting this property to false.
case.insensitive.search=false

# Locale for hub server and hub console
locale=en

# cookie secure flag and cookie httpOnly flag
# In JBoss, both of these flags will be used.
# In WebLogic, cookie-http-only flag is set to true by default, so only cookie-secure flag will be used here.
# in WebLogic, setting httpOnly will have no effect.
# in webSphere, these setting should be done thorough websphere console under Session Management
# in deployed siperian-mrm.ear.
#cookie-secure=false
#http-only=false

#Property for batch job processing. The number of threads will be used to distribute blocks of a batch job to batch servers.
cmx.server.batch.threads_per_job=10

#Block size for Load job.
cmx.server.batch.load.block_size=250
#Block size for Recalculate and Revalidate job.

```

```

cmx.server.batch.recalculate.block_size=250

#Properties for Automerger batch job (number of threads to use and block size)
cmx.server.automerger.threads_per_job=1
cmx.server.automerger.block_size=250

#Properties for Active VOS BPM integration
# Name of the merge operation in active vos
activevos.merge.workflow.operation.name=start
# Name of the service for all mdm service calls to ActiveVOS
activevos.merge.workflow.service.name=Merge
#The wait time for ActiveVOS to create task for the process and return task ID
activevos.workflow.startup.timeout.seconds=10
encryption.plugin.jar=C:\Temp\informatica_dataencryption.jar

```

처리 서버 속성

cmxcleanse.properties 파일에서 처리 서버 속성을 구성할 수 있습니다.

cmxcleanse.properties 파일은 다음 디렉터리에 있습니다.

<MDM Hub 설치 디렉터리>/hub/cleanse/resources

cmx.server.datalayer.cleansse.working_files.location

처리 서버 파일의 설치 디렉터리입니다. 이 속성은 처리 서버 설치 과정에서 설정됩니다. 정리 엔진 통합에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

cmx.server.datalayer.cleansse.working_files

정리 작업을 수행하는 동안 생성된 임시 파일을 저장할지 여부를 지정합니다. 임시 파일을 문제 해결 또는 감사 목적으로 사용할 수 있습니다. 임시 작업 파일을 삭제하려면 FALSE로 설정합니다. 임시 작업 파일을 저장하려면 KEEP으로 설정합니다. 기본값은 KEEP입니다. 정리 엔진 통합에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

cmx.server.datalayer.cleansse.execution

정리 작업을 실행할 위치를 지정합니다. 응용 프로그램 서버에서 정리 작업을 실행하려면 LOCAL로 설정합니다. 데이터베이스 서버에서 정리 작업을 실행하려면 DATABASE로 설정합니다. 기본값은 LOCAL입니다. 정리 엔진 통합에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

cmx.home

처리 서버의 설치 디렉터리입니다. 이 속성은 처리 서버 설치 과정에서 설정됩니다.

cmx.appserver.type

응용 프로그램 서버의 유형입니다. 이 속성의 값은 JBoss, WebSphere 또는 WebLogic 중 하나입니다. 이 속성은 처리 서버 설치 과정에서 설정됩니다.

cmx.appserver.version

응용 프로그램 서버의 JBoss 버전입니다. 이 속성의 값은 5 또는 7입니다. 이 속성은 처리 서버 설치 과정에서 설정됩니다.

cmx.appserver.soap.connector.port

WebSphere에만 사용됩니다. SOAP 커넥터 포트입니다. WebSphere의 경우 기본값은 8880입니다.

cmx.websphere.security.enabled

WebSphere 보안이 활성화되는지 여부를 지정합니다. WebSphere 관리 보안을 활성화하려면 true 또는 yes로 설정합니다. 기본값은 No입니다.

cmx.jboss7.management.port

JBoss 관리 포트입니다. JBoss의 기본값은 9990입니다. 이 속성은 처리 서버의 설치 과정에서 설정됩니다.

cmx.server.load.nonsmos.sourcesystem.enddate.like.smos

SMOS(상태 관리 재정의 시스템)가 아닌 시스템의 관계 종료 날짜를 SMOS와 동일한 날짜로 설정합니다. 관계 종료 날짜를 SMOS의 날짜와 동일한 날짜로 설정하려면 true로 설정합니다.

cmx.server.match.lwm

선택 사항입니다. 수동으로 추가해야 합니다. 경량 일치 기능을 제어합니다. 일치하는 레코드에 대한 전체 평가와 함께 경량 일치 기능을 활성화하려면 Y로 설정합니다. 일치하는 레코드에 대한 전체 평가 없이 경량 일치 기능을 활성화하려면 ONLY로 설정합니다. 기본값은 N입니다.

cmx.server.match.lwm_param 및 cmx.server.match.stats 속성과 함께 이 속성을 사용합니다.

cmx.server.match.lwm_param

선택 사항입니다. 수동으로 추가해야 합니다. cmx.server.match.lwm 속성을 Y 또는 ONLY로 설정해야 합니다. SSA-NAME3 컨트롤에 대한 속성 값을 다음 형식으로 설정합니다.

LWM=Y LWM_FIELDS=<field1>,<weight1>[,...,<fieldn>,<weightn>] LWM_LIMIT=<Reject>[,<Accept>]

cmx.server.match.stats

선택 사항입니다. 수동으로 추가해야 합니다. cmx.server.match.lwm 속성을 Y 또는 ONLY로 설정해야 합니다.

cmx.server.match.server_encoding

일치 처리에 대한 인코딩을 구성합니다. 일치 처리에 대한 인코딩을 허용하려면 1로 설정합니다. 기본값은 0입니다.

cmx.server.match.max_records_per_ranger_node

일치 Ranger 노드당 레코드 수입니다. 일치 Ranger 노드당 레코드 수가 많을수록 메모리를 더 많이 사용합니다. 일치 Ranger 노드당 최적 레코드 수는 처리 서버의 메모리 및 처리 능력에 따라 다릅니다. 기본값은 3000입니다.

cmx.server.match.max_return_records_searcher

유사 항목 검색 작업 중에 검색 스레드 점수를 지정할 후보 레코드 수를 제한합니다. 수동으로 추가해야 합니다. 기본값은 -1입니다.

유사 항목 검색 작업의 속도를 개선해야 하거나 CPU 사용량이 많은 경우 이 속성을 설정합니다. MDM Hub는 ORS(연산 참조 저장소)의 GETLIST 제한 속성 값을 고려하여 검색 스레드를 중지할 시기를 결정합니다. GETLIST 제한 속성은 Hub 콘솔에서 데이터베이스 도구를 사용하여 구성할 수 있습니다.

cmx.server.match.max_return_records_searcher 속성 값을 설정하면 유사 항목 검색 작업이 더 빠르게 완료될 수 있습니다. 다음 조건 중 하나가 충족되면 검색 스레드가 중지됩니다.

- 점수가 지정된 후보 레코드의 수가 cmx.server.match.max_return_records_searcher 속성에 설정된 값에 도달합니다.
- 일치하는 레코드의 수가 GETLIST 제한 속성에 설정된 값에 도달합니다.

속성을 설정하지 않거나 기본값인 -1을 사용하면 유사 항목 검색 작업이

cmx.server.match.max_return_records_searcher 속성을 무시하고 GETLIST 제한 속성을 따릅니다. 다음 조건 중 하나가 충족되면 검색 스레드가 중지됩니다.

- 일치하는 레코드의 수가 GETLIST 제한 속성에 설정된 값에 도달합니다.
- 점수를 지정할 후보 레코드가 남아 있지 않습니다.

com.informatica.mdm.sifapi.xref.edit.sys0.only

선택 사항입니다. 수동으로 추가해야 합니다. 관리 소스 시스템을 통해서만 편집 교차 참조 레코드를 생성하려면 **true**로 설정합니다. 모든 소스 시스템을 통해서 편집 교차 참조 레코드를 생성하려면 **false**로 설정합니다. 기본값은 **true**입니다.

중요: Hub 서버와 처리 서버 모두에 대해 속성을 설정해야 하며 속성 값이 두 속성 파일에서 동일해야 합니다.

cmx.ss.enabled

검색을 활성화할지 여부를 나타냅니다. 새 설치의 경우 기본값은 **true**입니다. 업그레이드의 경우 이 속성이 설정되어 있었다면 업그레이드 전 값이 유지됩니다. 이 속성을 설정하지 않으면 기본값이 **false**입니다.

JBoss 6.4.0에만 해당합니다. JBoss 6.4.0을 사용하는 환경에서 검색을 활성화하면 **cmx.server.match.file_load**를 **false**로 설정해야 합니다. 이 설정은 처리 서버가 기본 데이터베이스 유틸리티 대신 JDBC 업로더를 사용하여 일치 항목을 찾으도록 지정합니다.

cleanse.library.addressDoctor.property.SetConfigFile

Informatica Address Verification 구성 파일 경로입니다. 예를 들어 C:/infamdm/Hub/cleanse/resources/AddressDoctor/5/SetConfig.xml입니다. 정리 엔진 통합에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

cleanse.library.addressDoctor.property.ParametersFile

Informatica Address Verification 매개 변수 파일 경로입니다. 예를 들어 C:/infamdm/Hub/cleanse/resources/AddressDoctor/5/Parameters.xml입니다.

cleanse.library.addressDoctor.property.DefaultCorrectionType

Informatica Address Verification 수정 유형이며, PARAMETERS_DEFAULT로 설정해야 합니다.

cleanse.library.trilliumDir.property.config.file.1

Trillium Director 정리 라이브러리 구성 파일 1 파일 경로입니다. 예를 들어 C:/infamdm/Hub/cleanse/resources/Trillium/samples/director/td_default_config_Global.txt입니다. 정리 엔진 통합에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

cleanse.library.trilliumDir.property.config.file.2

Trillium Director 정리 라이브러리 구성 파일 2 파일 경로입니다. 예를 들어 C:/infamdm/Hub/cleanse/resources/Trillium/samples/director/td11_default_config_US_detail.txt입니다.

cleanse.library.trilliumDir.property.config.file.3

Trillium Director 정리 라이브러리 구성 파일 3 파일 경로입니다. 예를 들어 C:/infamdm/Hub/cleanse/resources/Trillium/samples/director/td11_default_config_US_summary.txt입니다.

cleanse.library.trilliumDir11.property.config.file.1

Trillium Director 11 정리 라이브러리 구성 파일 1 파일 경로입니다. 예를 들어 C:/infamdm/Hub/cleanse/resources/TrilliumDirector11/samples/director/td11_default_config_Global.txt입니다. 정리 엔진 통합에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

cleanse.library.trilliumDir11.property.config.file.2

Trillium Director 11 정리 라이브러리 구성 파일 2 파일 경로입니다. 예를 들어 C:/infamdm/Hub/cleanse/resources/TrilliumDirector11/samples/director/td11_default_config_US_detail.txt입니다.

cleanse.library.trilliumDir11.property.config.file.3

Trillium Director 11 정리 라이브러리 구성 파일 3 파일 경로입니다. 예를 들어 C:/infamdm/Hub/cleanse/resources/TrilliumDirector11/samples/director/td11_default_config_US_summary.txt입니다.

cleanse.library.trilliumDir.property.set_maximum_retry_count

선택 사항입니다. MDM Hub가 Trillium 서버에 연결하여 레코드 처리를 시도하는 최대 횟수를 설정합니다. 기본값은 5입니다. 네트워크 연결 재시도 횟수를 높이는 방법에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

cleanse.library.group1EntServer.property.config.file

Group1Software Enterprise Server 구성 파일입니다. 이 속성은 처리 서버 설치 과정에서 설정됩니다.

cleanse.library.group1CDQ.property.config.file

Group1Software CDQ Server 구성 파일입니다. 이 속성은 처리 서버 설치 과정에서 설정됩니다.

cleanse.library.firstLogicDirect.property.config.file

FirstLogicDirect 구성 파일입니다. 이 속성은 처리 서버 설치 과정에서 설정됩니다.

cmx.server.match.distributed_match

선택 사항입니다. 수동으로 추가해야 합니다. 처리 서버에서 정리 작업 및 유사 항목 일치 프로세스를 위한 분산 처리 환경에 참여할지 지정합니다. 기본값은 1이며, 이 경우 활성화됩니다. 분산 처리를 비활성화하려면 0으로 설정합니다.

여러 처리 서버 구성에 대한 자세한 내용은 *Multidomain MDM 설치 가이드*를 참조하십시오.

cmx.server.cleans.min_size_for_distribution

선택 사항입니다. 수동으로 추가해야 합니다. 처리 서버 간에 작업을 분배할 수 있는 크기를 지정합니다. 기본값은 1000입니다.

cmx.server.java_jdbc_loader

선택 사항입니다. 수동으로 추가해야 합니다. SQL 로더 대신 JDBC 파일 로더를 사용할지 지정합니다. JDBC 파일 로더를 사용하려면 true로 설정합니다. 기본값은 false입니다.

cmx.server.tokenize.file_load

선택 사항입니다. 수동으로 추가해야 합니다. 토큰화에 대해 데이터를 데이터베이스로 로드하는 데 중간 파일을 사용할지 여부를 지정합니다. 중간 파일을 사용하여 데이터를 로드하려면 true로 설정합니다. 데이터를 직접 로드하려면 false로 설정합니다. Oracle 및 IBM DB2 환경의 경우 기본값은 true이고, 중간 파일을 사용하면 성능이 개선됩니다. Microsoft SQL Server 환경의 경우 기본값은 false입니다.

cmx.server.tokenize.loader_batch_size

선택 사항입니다. 수동으로 추가해야 합니다. 토큰화 프로세스의 직접 로드 중에 데이터베이스로 보낼 수 있는 최대 삽입 문 수입니다. 기본값은 1000입니다.

cmx.server.match.file_load

선택 사항입니다. 수동으로 추가해야 합니다. 일치에 대해 데이터를 데이터베이스로 로드하는 데 중간 파일을 사용할지 여부를 지정합니다. 중간 파일을 사용하여 데이터를 로드하려면 true로 설정합니다. 데이터를 직접 로드하려면 false로 설정합니다. Oracle 및 IBM DB2 환경의 경우 기본값은 true입니다. 외부 일치에 대해 구성된 Microsoft SQL Server 환경 및 IBM DB2 환경의 경우 기본값은 false입니다.

참고: cmx.server.match.file_load 속성이 false로 설정된 경우 정리 로그의 일치 항목 수가 일괄 처리 뷰어와 다를 수 있습니다. 일치 항목 수가 다를 경우 일괄 처리 뷰어에 나열된 일치 항목 수를 참조합니다.

cmx.server.match.loader_batch_size

선택 사항입니다. 수동으로 추가해야 합니다. 일치 프로세스의 직접 로드 중에 데이터베이스로 보낼 수 있는 최대 삽입 문 수입니다. 기본값은 1000입니다.

cmx.server.match.exact_match_fuzzy_bo_api

선택 사항입니다. 수동으로 추가해야 합니다. 유사 항목 기본 개체에 대해 정확히 일치를 수행하려면 1로 설정합니다. 유사 항목 기본 개체에 대해 정확히 일치를 비활성화하려면 0으로 설정합니다. 기본값은 0입니다.

변경 내용이 적용되도록 응용 프로그램 서버를 다시 시작합니다. 유사 항목 기본 개체에 대해 정확히 일치를 구성하는 방법에 대한 자세한 내용은 *Multidomain MDM 서비스 통합 프레임워크 가이드*를 참조하십시오.

encryption.plugin.jar

선택 사항입니다. 수동으로 추가해야 합니다. 데이터 암호화 JAR 파일의 경로 및 파일 이름입니다. 데이터 암호화 구성에 대한 자세한 내용은 [“3단계. Hub 서버에 대한 데이터 암호화 구성” 페이지 182](#)을 참조하십시오.

cmx.server.bmg.use_longs

선택 사항입니다. 수동으로 추가해야 합니다. 처리 서버에서 긴 ROWID_OBJECT 값을 사용하도록 활성화하려면 1로 설정합니다. 처리 서버에서 긴 ROWID_OBJECT 값을 사용하지 못하도록 비활성화하려면 0으로 설정합니다. 기본값은 0입니다.

cmx.server.match.threshold_to_move_range_to_hold

선택 사항입니다. 수동으로 추가해야 합니다. 일치 분석 작업이 대기 상태로 전환할 수 있는 최대 레코드 수를 설정합니다. 기본값은 1000000입니다.

cmx.server.dbuserexit.load.PostLoadUserExit

선택 사항입니다. cmxserver.properties 파일 및 cmxcleanse.properties 파일 둘 모두에 수동으로 추가해야 합니다. Oracle에만 사용됩니다. MDM Hub가 로드 프로세스 후 데이터베이스 postload 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 true로 설정합니다. 기본값은 false입니다. PL/SQL 사용자 종료 활성화에 대한 자세한 내용은 현재 환경에 대한 *Multidomain MDM 업그레이드 가이드*를 참조하십시오.

cmx.server.dbuserexit.PostLandingUserExit

선택 사항입니다. 수동으로 추가해야 합니다. Oracle에만 사용됩니다. MDM Hub에서 사후 랜딩 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 true로 설정합니다. 기본값은 false입니다.

PL/SQL 사용자 종료 활성화에 대한 자세한 내용은 *Multidomain MDM 업그레이드 가이드*를 참조하십시오.

cmx.server.dbuserexit.PreStageUserExit

선택 사항입니다. 수동으로 추가해야 합니다. Oracle에만 사용됩니다. 준비 요청을 실행하기 전에 MDM Hub에서 날짜베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 true로 설정합니다. 기본값은 false입니다.

cmx.server.dbuserexit.PostStageUserExit

선택 사항입니다. 수동으로 추가해야 합니다. Oracle에만 사용됩니다. 준비 요청을 실행한 후에 MDM Hub에서 날짜베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 true로 설정합니다. 기본값은 false입니다.

cmx.server.dbuserexit.PreMatchUserExit

선택 사항입니다. 수동으로 추가해야 합니다. Oracle에만 사용됩니다. 일치 요청을 실행하기 전에 MDM Hub에서 날짜베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 true로 설정합니다. 기본값은 false입니다.

cmx.server.dbuserexit.PostMatchUserExit

선택 사항입니다. 수동으로 추가해야 합니다. Oracle에만 사용됩니다. 일치 요청을 실행한 후에 MDM Hub에서 날짜베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 true로 설정합니다. 기본값은 false입니다.

cmx.server.dbuserexit.PostMergeUserExit

선택 사항입니다. 수동으로 추가해야 합니다. Oracle에만 사용됩니다. 병합 요청을 실행한 후에 MDM Hub에서 날짜베이스 사용자 종료를 호출하는지 여부를 지정합니다. 이 속성을 활성화하려면 true로 설정합니다. 기본값은 false입니다.

cluster.flag

선택 사항입니다. 수동으로 추가해야 합니다. WebSphere에만 사용됩니다. 클러스터링 허용 여부를 지정합니다. 클러스터링을 허용하려면 true로 설정합니다. 클러스터링을 허용하지 않으려면 false로 설정합니다. 기본값은 false입니다.

cmx.server.cleanser.number_of_recs_batch

선택 사항입니다. 수동으로 추가해야 합니다. 일괄 처리에 포함될 수 있는 정리할 최대 레코드 수를 설정합니다. 기본값은 50입니다.

처리 서버에서 런타임 동작을 구성하는 방법에 대한 자세한 내용은 *Multidomain MDM 정리 어댑터 가이드*를 참조하십시오.

cmx.server.match.searcher_search_level

선택 사항입니다. 수동으로 추가해야 합니다. Data Director에서 확장 검색의 검색 수준을 설정합니다. Narrow, Typical, Exhaustive 또는 Extreme 중에 값을 선택할 수 있습니다. 기본값은 Narrow입니다.

서버 속성을 업데이트한 후에는 스키마의 유효성을 검사하고 Data Director 응용 프로그램을 재배포해야 합니다. 검색 수준에 대한 자세한 내용은 “[검색 수준](#)” 페이지 398을 참조하십시오. 확장 검색 구성에 대한 자세한 내용은 *Multidomain MDM Data Director 구현 가이드*를 참조하십시오.

cmx.server.match.searcher.database.worker.multithreaded

선택 사항입니다. 수동으로 추가해야 합니다. true로 설정하면 검색 범위를 처리하는 데 다중 병렬 스레드가 사용되고 SearchMatch API의 성능이 최적화됩니다. 기본적으로 다중 스레드 범위 처리는 비활성화되어 있습니다.

cmx.server.match.searcher.database.worker.multithreaded 속성을 설정하는 경우
cmx.server.match.searcher_thread_count 속성을 구성하여 스레드 수를 설정해야 합니다.

cmx.server.match.searcher.dbfiltered.max.key.size

선택 사항입니다. SearchMatch API의 성능을 최적화하기 위해 DBFILTERED 임계값을 지정합니다. SearchMatch 레코드에 cmx.server.match.searcher.dbfiltered.max.key.size 속성의 값보다 작거나 같은 SSA_KEY가 있는 경우 DBFILETRED 기능이 호출됩니다.

cmx.server.match.searcher.resultset.size

SearchMatch 데이터베이스 쿼리의 결과 집합 크기를 지정합니다.

cmx.server.match.searcher_thread_count

선택 사항입니다. 수동으로 추가해야 합니다. SearchMatch API의 스레드 수를 구성합니다. 기본값은 1입니다. SearchMatch API에 스레드를 한 개 사용하려면 1로 설정합니다.

cmx.server.match.searcher_thread_count 속성을 기본값이 아닌 값으로 설정하는 경우
cmx.server.match.searcher.database.worker.multithreaded 속성을 true로 설정해야 합니다.

SearchMatch API 성능 최적화에 대한 자세한 내용은 Informatica 내 지원 포털에서 다음 H2L을 참조하십시오.

- *Multidomain MDM for IBM DB2 성능 조정*(<https://mysupport.informatica.com/docs/DOC-11208>)
- *Multidomain MDM for Oracle 성능 조정*(<https://mysupport.informatica.com/docs/DOC-11207>)
- *Multidomain MDM for Microsoft SQL Server 성능 조정*
(<https://mysupport.informatica.com/docs/DOC-11149>)

ex.max.conn.per.host

호스트에 연결하려는 Elasticsearch 노드의 최대 수를 설정합니다. 호스트의 Elasticsearch 클러스터 노드 수로 설정합니다.

ex.max.threads

Elasticsearch 클러스터의 각 노드에 사용할 Apache 비동기식 비차단 수신기의 최대 스레드 수를 설정합니다. 기본값은 1입니다.

이 값은 Informatica 글로벌 고객 지원 센터에서 제안한 경우에만 변경합니다.

search.provisioning.numshards

선택 사항입니다. Elasticsearch 환경에 생성할 shard 수입니다. 최대 shard 수 및 총 노드 수에 따라 값이 달라집니다. 예를 들어, 최대 shard 수가 1이고 노드 수가 3인 경우 shard를 3개 생성할 수 있습니다. 기본값은 검색을 활성화한 처리 서버의 총 수입니다.

search.provisioning.numreplicas

선택 사항입니다. 서로 다른 노드에 생성할 Elasticsearch 엔진 문서의 사본 수입니다. 서로 다른 노드의 shard에 여러 개의 문서 사본을 생성하려면 복제 계수를 사용합니다. 하나 이상의 노드가 예기치 않게 종료된 경우 고가용성을 달성하려면 여러 개의 문서 사본이 필요합니다. 예를 들어 복제 계수가 2라면 두 개의 노드에서 두 개의 문서 사본을 얻을 수 있습니다. Elasticsearch의 경우 기본값은 0입니다.

MAX_INITIAL_RESULT_SIZE_TO_CONSIDER

선택 사항입니다. 속성을 수동으로 추가합니다. Data Director 응용 프로그램에 표시할 검색 결과의 총 개수입니다. 권장되는 최대값은 250입니다. 기본값은 130입니다. 130보다 높은 값은 Data Director 응용 프로그램의 성능에 영향을 줍니다.

mdm.smartsearch.cache.ttl

선택 사항입니다. 속성을 수동으로 추가합니다. 캐시된 결과가 만료되기 전에 존속시키기 위해 비즈니스 항목 검색 웹 서비스가 요청하는 캐시된 검색 결과에 대한 기간(밀리초)입니다. 기본값은 60000입니다.

min_rec_for_multithreading

MDM Hub에서 일괄 작업에 다중 스레드 일괄 작업을 적용할 때 사용하는 최소 일괄 처리 크기입니다. 자동 병합, 병합 해제, 로드, 처음에 스마트 검색 데이터 인덱싱, 준비, 배포 일치 및 토큰화 프로세스 유형의 일괄 작업에 적용됩니다. 기본값은 1000입니다.

mq.data.change.monitor.thread.start

다중 노드 환경에서 개별 노드에 메시지 대기열 풀링을 사용할지 여부를 지정합니다. 메시지 대기열 풀링을 비활성화하려면 false로 설정합니다. 기본값은 MDM Hub EAR 파일이 배포된 모든 Java 가상 시스템에서 true입니다.

ssl.keyStore

응용 프로그램 서버의 HTTPS 포트를 사용하여 처리 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 키 저장소 파일의 절대 경로 및 파일 이름입니다.

ssl.keyStore.password

응용 프로그램 서버의 HTTPS 포트를 사용하여 처리 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 키 저장소 파일에 대한 일반 텍스트 암호입니다.

ssl.trustStore

응용 프로그램 서버의 HTTPS 포트를 사용하여 처리 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 트러스트 저장소 파일의 절대 경로 및 파일 이름입니다.

ssl.trustStore.password

응용 프로그램 서버의 HTTPS 포트를 사용하여 처리 서버를 구성하는 경우 필수입니다. 속성을 수동으로 추가합니다. 트러스트 저장소 파일에 대한 일반 텍스트 암호입니다.

cmx.websphere.security.ssl.config.url

응용 프로그램 서버의 HTTPS 포트를 사용하여 처리 서버를 구성하는 경우 필수입니다. WebSphere에만 사용됩니다. 속성을 수동으로 추가합니다. 파일 이름을 포함한 ssl.client.props 파일의 절대 경로입니다.

cmx.outbound.bypass.multixref.insystem

선택 사항입니다. 수동으로 설정해야 합니다. **true**로 설정하면 일괄 작업에서 여러 교차 참조 레코드가 있는 기본 개체를 업데이트할 때 프로세스 서버에 메시지를 생성하지 않습니다. 기본값은 **false**입니다.

cmx.server.stage.sqlldr.charset

선택 사항입니다. **SQL***로더를 사용하여 데이터를 업로드하는 경우에 업로드된 데이터가 손상되었으면 이 속성을 데이터와 일치하는 문자 집합(예: **AL32UTF8**)로 설정하십시오. 준비 작업을 실행할 경우 준비 작업은 지정된 문자 집합을 사용하여 **SQL***로더의 제어 파일을 생성합니다. 그런 다음 데이터를 다시 로드할 수 있습니다. 기본값은 **UTF8**입니다.

cmx.server.stripDML.blockSize

MDM Hub가 각 블록에서 처리하는 레코드 수입입니다. 기본값은 **100**입니다.

cmx.server.stripDML.noOfThreadsForDelete

MDM Hub가 일치 키 테이블에서 레코드를 삭제할 때 사용하는 스레드 수입입니다. 기본값은 **30**입니다.

cmx.server.stripDML.noOfThreadsForInsert

MDM Hub가 레코드를 일치 키 테이블에 삽입할 때 사용하는 스레드 수입입니다. 기본값은 **50**입니다.

cmx.server.stripDML.noOfThreadsForUpdate

MDM Hub가 일치 키 테이블의 레코드를 업데이트할 때 사용하는 스레드 수입입니다. 기본값은 **30**입니다.

cmx.server.stripDML.useDeleteInsertLock

선택 사항입니다. 수동으로 설정해야 합니다. MDM Hub가 일치 작업 중에 토큰화를 실행하거나 많은 수의 레코드가 있는 기본 개체에 대한 토큰화 API 호출을 실행하게 하려면 **true**로 설정하십시오. 기본값은 **false**입니다.

cmx.server.stripDML.useUpdate

선택 사항입니다. 수동으로 설정해야 합니다. IBM DB2에만 해당됩니다. 재토큰화 중에 IBM DB2 환경의 성능을 개선하려면 **true**로 설정합니다. 기본값은 **false**입니다.

샘플 처리 서버 속성 파일

처리 서버 속성 파일을 **cmxcleanse.properties**라고 부릅니다.

다음 예제에서는 일반적인 **cmxcleanse.properties** 파일의 콘텐츠를 보여 줍니다.

```
# Cleanse properties
#
cmx.server.datalayer.cleanser.working_files.location=C:/infamdm/Hub_971_DB2/cleanser/tmp
cmx.server.datalayer.cleanser.working_files=KEEP
cmx.server.datalayer.cleanser.execution=LOCAL

# Server settings
# Installation directory
cmx.home=C:/infamdm/Hub_971_DB2/cleanser
# Application server type: jboss, tomcat, websphere or weblogic
cmx.appserver.type=jboss
cmx.appserver.version=7

# default setting: 8880 for websphere only (this is not being used in jboss and weblogic
cmx.appserver.soap.connector.port=
cmx.websphere.security.enabled=

# Match Properties
cmx.server.match.server_encoding=0
```

```
# limit memory usage by managing the number of records per match ranger node
cmx.server.match.max_records_per_ranger_node=3000

# Cleanse Properties

# Informatica Address Verification Properties
cleanse.library.addressDoctor.property.SetConfigFile=C:/infamdm/Hub_971_DB2/cleanse/resources/
AddressDoctor/5/SetConfig.xml
cleanse.library.addressDoctor.property.ParametersFile=C:/infamdm/Hub_971_DB2/cleanse/resources/
AddressDoctor/5/Parameters.xml
cleanse.library.addressDoctor.property.DefaultCorrectionType=PARAMETERS_DEFAULT

# Trillium Director Properties
cleanse.library.trilliumDir.property.config.file.1=
cleanse.library.trilliumDir.property.config.file.2=
cleanse.library.trilliumDir.property.config.file.3=

# Trillium Director 11+ Properties
cleanse.library.trilliumDir11.property.config.file.1=C:/infamdm/Hub_971_DB2/cleanse/resources/
TrilliumDirector11/samples/director/td11_default_config_Global.txt
cleanse.library.trilliumDir11.property.config.file.2=C:/infamdm/Hub_971_DB2/cleanse/resources/
TrilliumDirector11/samples/director/td11_default_config_US_detail.txt
cleanse.library.trilliumDir11.property.config.file.3=C:/infamdm/Hub_971_DB2/cleanse/resources/
TrilliumDirector11/samples/director/td11_default_config_US_summary.txt

# Group1Software Enterprise Server Properties
cleanse.library.group1EntServer.property.config.file=

# Group1Software CDQ Server Properties
cleanse.library.group1CDQ.property.config.file=

#FirstLogicDirect Properties
cleanse.library.firstLogicDirect.property.config.file=
encryption.plugin.jar=C:\Temp\informatica_dataencryption.jar
```

연산 참조 저장소 속성

연산 참조 저장소 데이터베이스의 속성에는 데이터베이스 공급업체, 버전 및 C_REPOS_DB_RELEASE 테이블의 정보가 포함됩니다.

다음 목록에서는 엔터프라이즈 관리자 도구에 표시되는 연산 참조 저장소 속성을 설명합니다.

데이터베이스 제품 이름

데이터베이스 시스템의 이름입니다.

데이터베이스 제품 버전

사용된 데이터베이스 버전입니다.

환경 ID

환경 ID입니다.

TNS 이름

연산 참조 저장소 데이터베이스의 TNS 이름입니다.

연결 URL

연산 참조 저장소 데이터베이스에 연결하기 위한 URL입니다.

데이터베이스 ID

데이터베이스 ID입니다.

시스템 간 시간 차이

수신 데이터가 미래인지 결정하는 델타 검색 값(초)입니다.

열 길이(바이트)

Oracle 환경용입니다.

SQL*Loader에서 로드 중인 데이터베이스가 UTF-8 데이터베이스인지 확인하는 데 사용됩니다.

기본값은 1로, 데이터베이스가 UTF-8임을 나타냅니다.

로드 템플릿

IBM DB2 환경용입니다. 로깅 모드에 있을 때 db2 로드 명령이 생성하는 로그 파일에 대한 데이터베이스 서버상의 경로입니다.

MTIP 재생성 필요

True로 설정되어 있다면 MTIP 보기가 일치/병합 프로세스 전에 다시 생성되는 것입니다.

기본값이 False라면 보기가 다시 생성되지 않는 것입니다.

엔터프라이즈 관리자 도구의 MTIP 재생성 단추를 통해 MTIP 보기를 다시 생성할 수 있습니다.

참고: ORS가 프로덕션 모드에 있을 경우에는 MTIP 보기를 다시 생성하면 안 됩니다. 엔터프라이즈 관리자 도구에서 MTIP를 다시 생성해 보기 전에 프로덕션 모드 설정을 해제합니다.

글로벌 로깅 안 함

Oracle 및 IBM DB2 환경용입니다. 데이터베이스 복구를 위해 로깅을 활성화하도록 테이블이 생성된 경우 사용됩니다. 기본값은 True로, 로깅하지 않음을 나타냅니다.

모델 리포지토리 서비스 URL

MDM Hub에서 모델 리포지토리 서비스에 연결하는 데 필요한 연결 매개 변수입니다. 엔터프라이즈 관리자 도구에서 연결 매개 변수를 지정합니다.

부록 B

구성 세부 정보 보기

이 부록에 포함된 항목:

- [구성 세부 정보 개요 보기, 637](#)
- [엔터프라이즈 관리자 시작, 637](#)
- [엔터프라이즈 관리자의 속성, 638](#)
- [환경 보고서, 640](#)
- [엔터프라이즈 관리자에서 버전 기록 보기, 640](#)
- [응용 프로그램 서버 로그 사용, 640](#)
- [클라이언트용 Hub 콘솔 로그 사용, 642](#)

구성 세부 정보 개요 보기

Hub 콘솔에서 엔터프라이즈 관리자 도구를 사용하여 Informatica MDM Hub 구현의 구성 세부 정보를 구성하고 볼 수 있습니다.

엔터프라이즈 관리자 도구를 사용하여 Hub 서버, 처리 서버, ORS 데이터베이스 및 MDM Hub 마스터 데이터베이스의 속성, 버전 기록 및 환경 보고서를 볼 수 있습니다. 또한 엔터프라이즈 관리자를 사용하여 ORS 데이터베이스에 대한 데이터베이스 로그를 구성하고 볼 수 있습니다.

엔터프라이즈 관리자 시작

Hub 콘솔에서 엔터프라이즈 관리자 도구를 시작하려면 구성 작업 영역을 확장한 다음 **엔터프라이즈 관리자**를 클릭합니다. Hub 콘솔 도구 모음에서 엔터프라이즈 관리자 도구 빠른 실행 단추를 클릭할 수도 있습니다.

그러면 엔터프라이즈 관리자 도구가 Hub 콘솔에 표시됩니다.

엔터프라이즈 관리자의 속성

엔터프라이즈 관리자 도구를 사용하여 **Hub** 서버, 처리 서버, **MDM Hub** 마스터 데이터베이스 또는 **ORS** 데이터베이스에 대한 속성을 표시합니다. 보려는 서버나 데이터베이스를 선택하려면 먼저 엔터프라이즈 관리자를 시작해야 합니다.

엔터프라이즈 관리자 도구를 사용하여 다음 속성을 봅니다.

Hub 서버

Hub 서버 탭을 선택하면 엔터프라이즈 관리자에 **Hub** 서버 속성이 표시됩니다. 이러한 속성에 대한 자세한 내용은 **cmxserver.properties** 파일을 참조하십시오.

처리 서버

처리 서버 탭을 선택하면 엔터프라이즈 관리자에 처리 서버의 목록이 표시됩니다. 특정 처리 서버를 선택하면 엔터프라이즈 관리자에 해당 속성이 표시됩니다. 이러한 속성에 대한 자세한 내용은 **cmxcleanse.properties** 파일을 참조하십시오.

마스터 데이터베이스

마스터 데이터베이스 탭을 선택하면 엔터프라이즈 관리자에 **MDM Hub** 마스터 데이터베이스 속성이 표시됩니다. 표시되는 유일한 데이터베이스 속성은 데이터베이스 공급업체와 버전입니다.

ORS 데이터베이스

연산 참조 저장소 데이터베이스 탭을 선택하면 엔터프라이즈 관리자에 연산 참조 저장소 데이터베이스의 목록이 표시됩니다. 연산 참조 저장소 데이터베이스를 선택하면 엔터프라이즈 관리자에 해당 연산 참조 저장소에 대한 속성이 표시됩니다.

상위 패널에는 연산 참조 저장소 데이터베이스의 목록이 포함되어 있으며 이러한 목록은 **MDM Hub** 마스터 데이터베이스에 등록되어 있습니다. 하위 패널에는 상위 패널에서 선택한 연산 참조 저장소 데이터베이스의 버전 기록과 속성이 표시됩니다. 연산 참조 저장소 속성에는 데이터베이스 공급업체, 버전 및 **C_REPOS_DB_RELEASE** 테이블의 정보가 포함됩니다. 버전 기록은 **C_REPOS_DB_VERSION** 테이블에서도 유지됩니다.

참고: 엔터프라이즈 관리자에는 **MDM Hub** 현재 버전에서 유효한 연산 참조 저장소 데이터베이스만 표시됩니다. 엔터프라이즈 관리자가 연산 참조 저장소 데이터베이스에 대한 데이터베이스 정보를 얻을 수 없는 경우 엔터프라이즈 관리자에 연산 참조 저장소 데이터베이스가 목록에 포함되지 않은 이유를 설명하는 메시지가 표시됩니다.

C_REPOS_DB_RELEASE 테이블

C_REPOS_DB_RELEASE 테이블에는 연산 참조 저장소 속성이 포함되어 있습니다.

다음 목록에는 사용자의 기본 설정에 따라 엔터프라이즈 관리자가 연산 참조 저장소 데이터베이스에 대해 표시하는 **C_REPOS_DB_RELEASE** 속성이 설명되어 있습니다.

DEBUG_LEVEL

ORS 데이터베이스에 대한 디버그 수준입니다. 디버그 수준은 다음 정수 값 중 하나일 수 있습니다.

100 = 오류 수준 디버깅

200 = 경고 수준 디버깅

300 = 정보 수준 디버깅

400 = 디버그 수준 디버깅

500 = 모든 수준 디버깅

ENVIRONMENT_ID

환경 ID입니다.

DEBUG_FILE_PATH

ORS 데이터베이스 디버그 로그의 위치 경로입니다.

DEBUG_FILE_NAME

ORS 데이터베이스 디버그 로그의 이름입니다.

DEBUG_IND

디버그 활성화 여부를 나타내는 플래그입니다.

0 = 디버그가 활성화되지 않음

1 = 디버그가 활성화됨

DEBUG_LOG_FILE_SIZE

Oracle 환경용입니다. 데이터베이스 로그 파일의 크기(MB 단위)로, 기본값은 5입니다.

DEBUG_LOG_FILE_NUMBER

Oracle 환경용입니다. 로그 롤링에 사용되는 로그 파일 수로, 기본값은 5입니다.

TNSNAME

ORS 데이터베이스의 TNS 이름입니다.

CONNECTION_PORT

ORS 데이터베이스가 수신하는 포트입니다.

ORACLE_SID

Oracle 데이터베이스 ID입니다.

DATABASE_HOST

데이터베이스가 설치되어 있는 호스트입니다.

INTER_SYSTEM_TIME_DELTA_SEC

데이터베이스 서버 시스템 시간과 다른 시스템의 시스템 시간 차이를 구성하는 데 필요한 시간(초)입니다.

COLUMN_LENGTH_IN_BYTES_IND

Oracle 환경용입니다. 로드하는 데이터베이스가 UTF-8 데이터베이스인지 확인하기 위해 SQLLoader에서 사용하는 플래그입니다. 기본값인 1은 데이터베이스가 UTF-8이라는 의미입니다.

LOAD_TEMPLATE

IBM DB2 환경용입니다. 로깅 모드에 있을 때 db2 로드 명령이 생성하는 로그 파일에 대한 데이터베이스 서버상의 경로입니다.

MTIP_REGENERATION_REQUIRED_IND

일치/병합 프로세스 전에 MTIP 보기를 다시 생성할지 나타내는 플래그입니다. 기본값인 0(영)은 보기가 다시 생성되지 않는다는 의미입니다.

GLOBAL_NOLOGGING_IND

Oracle 및 IBM DB2 환경용입니다. DB 복구에 로깅을 활성화하기 위해 테이블을 생성할 때 사용합니다. 기본값인 1은 로깅을 사용하지 않는다는 의미입니다.

환경 보고서

환경 보고서 탭을 선택하면 엔터프라이즈 관리자에서 모든 Hub 구성 요소 탭의 속성 요약과 함께 연관된 오류 메시지가 표시됩니다. 보고서는 다음 순서로 속성을 나열합니다.

- Hub 서버
- 처리 서버
- 마스터 데이터베이스
- ORS 데이터베이스

MDM Hub 환경 보고서 저장

MDM Hub 환경 보고서를 저장하려면 Hub 콘솔에서 엔터프라이즈 관리자 도구를 사용합니다.

1. Hub 콘솔의 구성 작업 영역에서 **엔터프라이즈 관리자** 도구를 선택합니다.
2. **엔터프라이즈 관리자** 도구에서 **환경 보고서** 탭을 선택합니다.
3. **저장**을 클릭합니다.
4. **Hub 환경 보고서 저장** 대화 상자에서 환경 보고서를 저장할 디렉터리로 이동합니다.
5. **저장**을 클릭합니다.

엔터프라이즈 관리자에서 버전 기록 보기

엔터프라이즈 관리자 도구를 사용하여 Hub 서버, 처리 서버, MDM Hub 마스터 데이터베이스 또는 ORS 데이터베이스에 대한 버전 기록을 표시할 수 있습니다. 보려는 서버나 데이터베이스를 선택하려면 먼저 엔터프라이즈 관리자를 시작해야 합니다.

1. 엔터프라이즈 관리자 화면에서 보려는 정보 유형에 대한 탭을 선택합니다.
 - Hub 서버
 - 처리 서버
 - 마스터 데이터베이스
 - ORS 데이터베이스

엔터프라이즈 관리자에 선택에 따른 버전 기록이 표시됩니다.

2. **버전 기록** 탭을 선택합니다.

엔터프라이즈 관리자에 해당 특정 구성 요소에 대한 버전 정보가 표시됩니다. 버전 기록은 설치 시작 시간의 내림차순으로 정렬됩니다.

응용 프로그램 서버 로그 사용

log4j.xml 파일을 사용하여 Hub 서버 또는 처리 서버에 대한 응용 프로그램 서버 로그 파일을 구성합니다. 엔터프라이즈 관리자를 사용하여 응용 프로그램 서버 로그를 구성할 수는 없습니다.

응용 프로그램 서버 로그 수준

응용 프로그램 서버 로그에 대한 log4j.xml 구성 파일에는 정보를 디버깅하고 검색하기 위한 일련의 로그 수준이 포함되어 있습니다.

다음 테이블에서는 응용 프로그램 서버 로그 수준에 대해 설명합니다.

이름	ORS 메타데이터 테이블 값	설명
ALL	500	관련된 모든 정보를 로깅합니다.
DEBUG	400	디버깅하는 데 사용됩니다. 기본값입니다.
INFO	300	응용 프로그램 서버 로그 정보입니다.
WARNING	200	경고 메시지입니다.
ERROR	100	오류 메시지입니다.

로그 파일 롤링

응용 프로그램 서버 로그 파일이 log4j.xml 구성의 MaxFileSize 매개 변수에 정의된 최대 크기에 도달하면 엔터프라이즈 관리자가 로그 롤링 절차를 수행하여 기존 로그 정보를 보관하고 로그 파일이 새 응용 프로그램 서버 정보로 재정의되지 않도록 할 수 있습니다.

- 각각 <infadm_install_dir>\hub\server\conf 및 <infadm_install_dir>\cleanse\server\conf에 있는 Hub 서버 및 처리 서버의 log4j.xml 파일에서 다음과 같은 응용 프로그램 서버 로그 구성 설정을 정의합니다.
 - File. 로그 파일의 이름(예: C:\infadm\hub\server\logs\cmxserver.log)입니다.
WebLogic 및 WebSphere에서는 동일한 위치에서 로그를 수집하도록 다음과 같이 파일 이름을 서로 구분하여 사용하는 것이 가장 좋습니다.
Hub 서버: C:\<infadm_install_dir>\hub\server\logs\cmxserver.log
처리 서버: C:\<infadm_install_dir>\hub\server\logs\cmxcleanse.log
 - MaxFileSize. 응용 프로그램 서버 로그 파일의 최대 파일 크기입니다.
 - MaxBackupIndex. 파일의 최대 개수입니다.
 - Threshold. 임계값 로깅 수준입니다. 이 임계값 수준보다 높은 로깅 수준은 재정의됩니다.
- Hub 서버 및 처리 서버에 사용되는 로그 파일은 다양한 응용 프로그램 서버 메시지를 응용 프로그램 서버 로그 파일에 추가합니다.
- 실제 로그 파일 크기가 MaxFileSize를 초과하면 Hub에서 로그 롤링 절차를 활성화합니다.
 - 활성 로그 파일의 이름이 <filename>.hold로 바꿉니다.
 - 이름이 <filename>.(n)인 파일은 모두 <filename>.(n+1)로 이름이 바뀝니다.
 - n+1이 MaxBackupIndex보다 크면 파일이 삭제됩니다.
 - 롤오버가 응용 프로그램 서버 파일의 최대 개수를 넘어가면 MDM Hub가 원래 로그 파일의 이름을 바꿉니다. 예를 들어, MDM Hub는 cmxserver.log를 cmxserver.log.1로, cmxserver.log.1을 cmxserver.log.2로 이름을 바꿉니다. 그런 다음 Hub는 cmxserver.log를 새 로그 정보로 덮어씁니다.
 - <filename>.hold 파일 이름이 <filename>.1로 바뀝니다.

참고: Hub는 또한 롤오버를 실행하기 전에 log.rolling 파일을 생성합니다. 로그 파일이 예상대로 롤링되지 않으면 로그 파일 디렉터리를 확인하고 로그 롤링이 계속될 수 있도록 log.rolling 파일을 제거합니다.

응용 프로그램 서버 로그 구성

디버깅을 위한 응용 프로그램 서버 설정을 지정해야 합니다.

응용 프로그램 서버 로그는 응용 프로그램 서버 수준에서 유지 관리됩니다. 응용 프로그램 서버 로그 파일의 로그 항목은 ORS가 아니라 응용 프로그램 서버와 관련이 있습니다. Hub 서버 로그 파일은 응용 프로그램 서버와 관련된 cmxserver.log 파일에 기록됩니다. 여러 개의 응용 프로그램 서버를 사용하는 경우 각 응용 프로그램 서버에 고유한 구성과 로그 파일이 포함됩니다. 마찬가지로 WebSphere 및 WebLogic에 설치된 처리 서버는 해당 응용 프로그램 서버의 로그 파일에 기록됩니다. JBoss에 MDM Hub을 설치한 경우에는 Hub 서버 및 처리 서버가 동일한 JBoss 인스턴스에 있고 동일한 공유 응용 프로그램 서버 로그 파일에 기록합니다.

조회할 서버나 데이터베이스를 선택하려면 먼저 엔터프라이즈 관리자를 시작해야 합니다. 응용 프로그램 서버 로그를 구성하려면 다음 단계를 수행하십시오.

1. log4j.xml 파일을 편집하기 위해 엽니다.
log4j.xml 파일은 <infamdm_install_dir>\hub\server\conf에 있습니다.
2. log4j.xml 파일을 편집합니다.
 - 기본값을 "DEBUG"로 변경합니다.
 - log4j.xml 파일의 로깅 범주 이름을 com.siperian, com.informatica 및 com.delos로 설정합니다.
 - 로깅 범주 siperian.performance를 "OFF"로 설정합니다. Informatica에서 요구하는 경우에만 이 로깅 범주를 "ON"으로 설정해야 합니다.

다음 샘플에서는 log4j.xml 파일의 설정을 보여 줍니다.

```
<category name="com.delos">
  <priority value="DEBUG"/>
</category>

<category name="com.siperian">
  <priority value="DEBUG"/>
</category>

<category name="com.informatica">
  <priority value="DEBUG"/>
</category>

<category name="siperian.performance" additivity="false">
  <priority value="OFF"/>
  <appender-ref ref="FILE"/>
</category>
```

클라이언트용 Hub 콘솔 로그 사용

콘솔 로그에는 콘솔 이벤트로 인해 발생한 메시지 및 오류와 콘솔과 서버 간의 통신이 포함됩니다. 콘솔 로그는 클라이언트 시스템의 다음 디렉터리에 있습니다.

```
<Windows_users_home_dir>\Siperian\console.log
```

동일한 폴더에 있는 log4j.properties 파일을 사용하여 로깅 수준, 로그 파일 크기 및 이전 로그의 보관 수를 관리할 수 있습니다. log4j.properties 파일의 마지막 행은 다음과 같이 설정해야 합니다.

```
log4j.rootCategory=DEBUG, FileLog, ConsoleLog
```

부록 C

행 수준 잠금

이 부록에 포함된 항목:

- [행 수준 잠금 개요, 643](#)
- [행 수준 잠금 정보, 643](#)
- [행 수준 잠금 구성, 644](#)
- [SIF 요청과 일괄 처리 사이의 상호 작용 잠금, 645](#)

행 수준 잠금 개요

이 부록에서는 일괄 처리 및 API 프로세스(SIF 요청 포함)나 API/API 프로세스를 비동기식으로 실행할 경우 데이터 무결성을 보호하기 위해 행 수준 잠금을 활성화하고 사용하는 방법에 대해 설명합니다.

참고: 일괄 작업 및 API 호출이 Informatica MDM Hub 구현에서 동시에 실행되고 있지 않은 경우에는 이 섹션을 건너뛸 수 있습니다.

행 수준 잠금 정보

Informatica MDM Hub에서는 기본 개체의 레코드에 대해 일괄 처리 및 SIF 작업이 동시에 실행될 때 행 수준 잠금을 사용하여 데이터 무결성을 관리합니다. 행 수준 잠금을 사용하면 다음과 같은 이점이 있습니다.

- 일괄 처리와 SIF 프로세스 모두에서 기본 개체는 동일하지만 레코드가 다른 기본 개체 데이터를 동시에 업데이트할 수 있습니다.
- 온라인 처리와 일괄 처리 간의 충돌을 방지할 수 있습니다.
- 데이터에 대한 동시 액세스 수준을 높일 수 있습니다.
- 미러된 환경에 필요한 하드웨어 중복 방지 기능을 사용할 수 있습니다.

일괄/API 또는 API/API 프로세스가 Informatica MDM Hub 구현에서 비동기적으로 실행되는 경우에 행 수준 잠금을 활성화해야 합니다. 행 수준 잠금은 비동기 SIF 및 일괄 처리에 적용됩니다. 기존의 응용 프로그램 수준 잠금으로 인해 동기 일괄 처리에는 이 기능이 적용되지 않습니다.

기본 동작

행 수준 잠금은 기본적으로 비활성화됩니다. 행 수준 잠금이 비활성화된 경우에는 동일한 기본 개체에서 API 프로세스(SIF 요청 포함)와 일괄 처리를 동시에 비동기적으로 실행할 수 없습니다. 연산 참조 저장소에 대해 행 수

준 잠금을 명시적으로 활성화한 경우 Informatica MDM Hub에서는 행 수준 잠금 메커니즘을 사용하여 토큰화, 일치 및 병합 프로세스에 대한 동시 업데이트를 관리합니다.

잠금 유형

Informatica MDM Hub에서 데이터를 관리하는 데는 다음과 같은 유형의 잠금이 관련됩니다.

배타적 잠금

잠긴 기본 개체에 대해 다른 모든 작업(API 또는 일괄 처리)이 처리되지 않도록 합니다.

공유 잠금

특정 작업만 실행되지 않도록 합니다. 예를 들어 일괄 작업이 기본 개체에서 비배타적으로 실행되는 경우, 상호 운용성이 활성화되어 있으면 이 공유 잠금으로 인해 다른 일괄 작업은 금지되지만 API 작업은 기본 개체에서 처리될 수 있습니다.

행 수준 잠금

영향을 받는 기본 개체 행을 잠그는 공유 잠금입니다.

행 수준 잠금을 사용하기 위한 고려 사항

Informatica MDM Hub 구현에서 행 수준 잠금을 사용하려면 다음 문제를 고려해야 합니다.

- 일괄 처리에서는 상당 기간 동안 개별 레코드를 잠가 웹을 통한 SIF 액세스를 차단할 수 있습니다.
- 잠깐 동안 SIF 요청을 잠글 수도 있습니다. 그러나 이러한 잠금은 필요할 때만, 10분 미만으로 제한해서 적용해야 합니다.
- Hub 저장소는 일괄 처리와 SIF 요청을 동시에 실행할 때의 결합된 리소스 수요 부하를 지원할 수 있을 만한 크기여야 합니다.

참고: 여러 일괄 작업이 함께 실행될 경우 상호 운용성이 활성화되어 있어야 합니다. 서로 다른 여러 일괄 작업을 실행할 때 동일한 하위(또는 상위) 레코드에 액세스를 시도하는 상위 레코드가 여러 개 있는 경우, 한 작업에서 다른 일괄 작업에 의해 처리되고 있는 레코드를 잠그려고 하거나 해당 레코드를 일괄 처리 대기 시간보다 오랫동안 대기 상태로 두면 해당 작업이 실패합니다. 최대 대기 시간은 C_REPOS_TABLE에서 정의됩니다.

행 수준 잠금 구성

이 섹션에서는 행 수준 잠금을 구성하는 방법에 대해 설명합니다.

ORS에 대해 행 수준 잠금 활성화

기본적으로 행 수준 잠금은 활성화되어 있지 않습니다. 연산 참조 저장소에 대해 행 수준 잠금을 활성화하려면

1. Hub 콘솔에서 데이터베이스 도구를 시작합니다.
2. 구성할 연산 참조 저장소를 선택합니다.
3. 연산 참조 저장소 속성을 편집합니다.
4. **일괄 API 잠금 상호 운용성** 확인란을 선택합니다.
5. 변경 내용을 저장합니다.

참고: 행 수준 잠금을 활성화하면 Put 작업 시 토큰화 속성을 활성화할 수 없습니다.

잠금 대기 시간 구성

잠금 대기 시간이 활성화된 경우 스키마 관리자를 사용하여 연산 참조 저장소의 기본 개체에 대한 SIF 및 일괄 잠금 대기 시간을 구성할 수 있습니다. 대기 시간이 초과되면 Informatica MDM Hub에서 오류 메시지가 표시됩니다.

SIF 요청과 일괄 처리 사이의 상호 작용 잠금

이 섹션에서는 SIF 요청과 일괄 처리 사이의 상호 작용을 잠그는 방법에 대해 설명합니다.

API 일괄 처리 상호 운용성이 활성화된 경우의 상호 작용

다음 테이블에서는 행 수준 잠금이 활성화된 경우 API와 일괄 처리 사이의 상호 작용을 보여 줍니다.

기존 잠금 / 새 수신 호출	일괄 처리 - 배타적 잠금	일괄 처리 - 공유 잠금	API 행 수준 잠금
일괄 처리 - 배타적 잠금	오류 메시지 즉시 표시	오류 메시지 즉시 표시	Batch_Lock_Wait_Seconds가 잠금 여부를 확인할 때까지 기다립니다. 대기 시간 동안 잠금이 해제되지 않으면 오류 메시지를 표시합니다. 각 테이블을 잠그도록 호출됩니다.
일괄 처리 - 공유 잠금	오류 메시지 즉시 표시	오류 메시지 즉시 표시	Batch_Lock_Wait_Seconds가 FOR UPDATE SELECT를 통해 행 수준 잠금을 적용할 때까지 기다립니다. 테이블에서 잠금을 관리하지 않을 경우 오류 메시지를 표시합니다. 각 테이블을 잠그도록 호출됩니다.
API - 행 수준 잠금	오류 메시지 즉시 표시	API_Lock_Wait_Seconds가 FOR UPDATE SELECT를 통해 행 수준 잠금을 적용할 때까지 기다립니다. 테이블에서 잠금을 관리하지 않을 경우 오류 메시지를 표시합니다.	API_Lock_Wait_Seconds가 FOR UPDATE SELECT를 통해 행 수준 잠금을 적용할 때까지 기다립니다. 테이블에서 잠금을 관리하지 않을 경우 오류 메시지를 표시합니다. 각 테이블을 잠그도록 호출됩니다.

참고: 승인 태스크를 실행하면 XREF 테이블의 INTERACTION_ID 열에 대한 사용자 지정 인덱스를 통해 서비스 측 성능이 향상되지만 대규모 일괄 처리, 특히 로드 및 자동 병합에서 성능을 저하시키는 단점이 있습니다.

API 일괄 처리 상호 운용성이 비활성화된 경우의 상호 작용

다음 테이블에서는 API 일괄 처리 상호 운용성이 비활성화된 경우 API 프로세스와 일괄 처리 간의 상호 작용을 보여 줍니다. 이 시나리오에서 일괄 처리는 배타적 잠금을 실행하고, SIF 요청은 배타적 잠금을 확인하지만 아무 잠금도 실행하지 않습니다.

기존 잠금 / 새 수신 호출	일괄 처리	API
일괄 처리 - 배타적 잠금	오류 메시지 즉시 표시	“기본 동작” 페이지 643 참조
API	오류 메시지 즉시 표시	“기본 동작” 페이지 643 를 참조하십시오.

부록 D

MDM Hub 로깅

이 부록에 포함된 항목:

- [MDM Hub 로깅 개요, 646](#)
- [로깅 설정 구성, 647](#)
- [Hub 콘솔 로그, 647](#)
- [Hub 서버 로그, 647](#)
- [처리 서버 로그, 648](#)
- [Informatica Platform 로그, 648](#)
- [Entity 360 로그, 648](#)
- [프로비저닝 도구 로그, 648](#)

MDM Hub 로깅 개요

MDM Hub에서 발생하는 모든 메시지 및 오류는 로그 파일에 저장됩니다. MDM Hub는 MDM Hub를 설치하거나 업그레이드하는 동안 로그 파일을 생성합니다.

다음 테이블에는 MDM Hub 구성 요소에 대한 로그 파일이 설명되어 있습니다.

MDM Hub 구성 요소	로그 파일
Hub 콘솔	console.log
Hub 서버	cmxserver.log
처리 서버	cmxserver.log
Informatica Platform	informatica-mdm-platform.log
Entity 360	entity360view.log
프로비저닝 도구	provisioning.log

로깅 설정 구성

로깅에 대해 Hub 서버를 구성할 수 있습니다. `log4j.xml` 파일에서 로깅에 대한 구성 설정을 지정합니다.

1. 다음 디렉터리에서 `log4j.xml`을 엽니다.
<MDM Hub 설치 디렉터리>/hub/server/conf
2. 다음 범주 이름에 대한 <우선 순위 이름> 요소에서 로깅 수준을 설정합니다.
 - com.siperian
 - com.delos
 - com.informatica로깅 수준은 다음 값 중 하나로 설정할 수 있습니다.
 - DEBUG. 가장 자세한 로깅입니다.
 - INFO. 덜 자세한 로깅
 - ERROR. 가장 덜 자세한 로깅기본값은 INFO입니다.
3. Threshold 매개 변수를 DEBUG로 설정합니다.

Hub 콘솔 로그

MDM Hub는 Hub 콘솔을 실행하는 운영 체제 환경에서 `console.log` 파일을 생성합니다. Windows 환경에서 Hub 콘솔을 실행하면 Hub 콘솔 로그가 C:\Documents and Settings\<user_home>\siperian 디렉터리에 생성됩니다. UNIX 환경에서 Hub 콘솔을 실행하면 Hub 콘솔 로그가 /<user_home>/siperian 디렉터리에 생성됩니다.

Hub 콘솔 로그에는 Hub 콘솔의 로그 메시지가 포함됩니다. MDM Hub가 응용 프로그램 서버와 통신하는 중에 발생한 모든 오류, 응용 프로그램 서버의 오류 메시지 또는 콘솔 오류 메시지가 이 로그 파일에 기록됩니다. 기본적으로 MDM Hub에서는 `console.log`를 생성합니다.

`console.log` 파일은 롤링 로그 파일입니다. 이 로그 파일의 크기가 5MB에 도달하면 로그가 `console.log.1`에 복사되고 다시 시작됩니다. Hub 서버는 잠재적으로 무기한으로 많은 로그 파일을 생성할 수 있습니다. 정기적으로 이전 파일을 삭제하거나 다른 저장소 영역으로 전송해야 합니다.

Hub 서버 로그

Hub 서버는 응용 프로그램 서버에 대한 로그 파일인 `cmxserver.log`를 생성합니다.

Hub 서버 로그는 다음 디렉터리에 나타납니다.

<MDM Hub 설치 디렉터리>/hub/server/logs

Hub 서버 로그에는 응용 프로그램 서버의 로깅 및 디버깅 정보가 포함됩니다. 기본적으로 Hub 서버는 `cmxserver.log` 파일을 생성합니다. `cmxserver.log` 파일은 롤링 로그 파일로서, 파일 크기가 5MB에 도달하면 Hub 서버가 로그 파일을 `cmxserver.log.1`에 복사하고 다시 시작합니다. Hub 서버는 잠재적으로 무기한으로 많은 파일을 생성할 수 있습니다. 정기적으로 이전 파일을 삭제하거나 다른 저장소 영역으로 전송해야 합니다.

처리 서버 로그

처리 서버는 CLEANSE, TOKENIZATION 및 SIMULATION 함수에 대해 `cmxserver.log`를 생성합니다.

처리 서버 로그는 응용 프로그램 서버에 대한 로그 파일입니다. `cmxserver.log` 파일은 다음 디렉터리에 있습니다.

<MDM Hub 설치 디렉터리>/hub/cleanse/logs

`cmxserver.log` 파일에는 정리 프로세스에 대한 디버깅 및 오류 메시지가 포함됩니다. 기본적으로 처리 서버는 `cmxserver.log` 파일을 생성합니다. `cmxserver.log` 파일은 롤링 로그 파일이며, 최대 파일 크기에 도달하면 처리 서버가 해당 파일을 `cmxserver.log.1`에 복사하고 다시 시작합니다.

기본적으로 처리 서버 로그 파일의 최대 크기는 5MB입니다. 그러나 다음 디렉터리에서 최대 크기를 구성할 수 있습니다.

<MDM Hub 설치 디렉터리>/hub/cleanse/conf/log4j.xml

Informatica Platform 로그

MDM Hub는 Informatica Platform 프로세스에 대한 구성 로그 메시지를 저장하는 `informatica-mdm-platform.log`를 생성합니다.

`informatica-mdm-platform.log` 파일이 다음 디렉터리에 나타납니다.

<MDM Hub 설치 디렉터리>/hub/server/logs

Entity 360 로그

MDM Hub에서는 모든 메시지, 오류 및 Entity 360 프레임워크에 대한 전체 스택 추적을 저장하는 `entity360view.log`를 생성합니다.

`entity360view.log` 파일은 다음 디렉터리에 있습니다.

<MDM Hub 설치 디렉터리>/hub/server/logs

`log4j-entity360view.xml` 파일은 Entity 360 프레임워크에 대한 구성 정보를 저장하지만 Hub 서버에 대한 구성 정보는 포함하지 않습니다.

`log4j-entity360view.xml` 파일은 다음 디렉터리에 있습니다.

<MDM Hub 설치 디렉터리>/hub/server/conf

프로비저닝 도구 로그

MDM Hub는 프로비저닝 도구에 대한 구성 로그 메시지를 저장하는 `provisioning.log`를 생성합니다.

`provisioning.log` 파일은 다음 디렉터리에 나타납니다.

<MDM Hub 설치 디렉터리>/hub/server/logs

log4j-provisioning.xml 파일은 프로비저닝 도구에 대한 구성 정보를 저장하지만 Hub 서버에 대한 구성 정보는 포함하지 않습니다.

log4j-provisioning.xml 파일은 다음 디렉터리에 있습니다.

<MDM Hub 설치 디렉터리>/hub/server/conf

부록 E

테이블 분할

이 부록에 포함된 항목:

- [테이블 분할 지원, 650](#)

테이블 분할 지원

Multidomain MDM은 분할된 테이블을 지원하며, MDM 엔지니어링 팀의 테스트를 거치지지는 않았지만 분할된 테이블에서도 정상적으로 작동할 것으로 예상됩니다. 테이블을 분할하면 일부 실시간 MDM 액세스 성능이 개선되겠지만 파티션 경계를 통과하는 레코드의 일치 및 병합 성능에도 영향이 있을 것으로 예상합니다. 따라서 전체 성능에 미치는 영향을 테스트 및 측정하는 과정을 일반 개발 프로세스에 포함할 것을 권장합니다.

부록 F

제품 사용 툴킷을 사용하여 MDM 환경 정보 수집

이 부록에 포함된 항목:

- [제품 사용 툴킷을 사용하여 MDM 환경 정보 수집 개요, 651](#)
- [Hub 서버에서 MDM Hub 데이터 수집 활성화, 653](#)
- [처리 서버에서 MDM Hub 데이터 수집 활성화, 653](#)
- [Hub 서버에서 MDM Hub 데이터 수집 비활성화, 653](#)
- [처리 서버에서 MDM Hub 데이터 수집 비활성화, 654](#)

제품 사용 툴킷을 사용하여 MDM 환경 정보 수집 개요

MDM Hub를 활성화 또는 비활성화하여 MDM Hub 환경에 대한 정보를 Informatica로 전송하거나 전송하지 않도록 할 수 있습니다. 제품 사용 툴킷은 Hub 서버 및 처리 서버의 시스템 구성 요소에 대한 정보와 MDM Hub 환경에 대한 정보를 전송할 수 있습니다.

제품 사용 툴킷을 통해 전송되는 정보는 MDM Hub 구현에 대한 유용한 정보를 제공하는 고급 상태 검사 역할을 합니다. Informatica 전문가는 이러한 정보를 바탕으로 고객의 환경에 가장 적합한 모범 사례를 권장하고 프로젝트 구현을 가속화하는 데 필요한 사항을 제안할 수 있습니다.

데이터 수집을 활성화한 경우 응용 프로그램 서버를 시작하면 10분 후에 제품 사용 툴킷이 Informatica로 정보를 전송합니다. 그 후에는 제품 사용 툴킷이 30분마다 정보를 전송합니다.

중요: MDM Hub를 설치 또는 업그레이드할 경우 데이터 수집은 기본적으로 활성화됩니다. 설치 또는 업그레이드 후 언제든지 데이터 수집을 비활성화할 수 있습니다.

시스템 구성 정보

다음 테이블에는 데이터 수집을 활성화한 경우 수집되는 Hub 서버 구성 요소 및 처리 서버 구성 요소에 대한 정보가 설명되어 있습니다.

수집되는 정보	설명
메모리	총 실제 메모리, 사용 가능한 실제 메모리, 최대 가상 메모리, 사용 가능한 가상 메모리 및 사용 중인 가상 메모리
CPU	CPU 이름, 최대 클럭 속도 및 로드 백분율
환경 변수	환경 변수의 이름 및 값
디스크 드라이브	드라이브 문자 ID, 드라이브 크기 및 사용 가능한 공간
운영 체제 정보	운영 체제 이름, 버전, 제조업체, 구성 및 빌드 유형
패치 관련 운영 체제 정보	운영 체제 패치 목록
서비스	각 서비스의 이름 및 상태
네트워크 구성	네트워크 구성에 대한 세부 정보
가상화	노드 ID, 버전, 일련 번호

MDM Hub 환경 정보

다음 테이블에는 데이터 수집을 활성화한 경우 수집되는 MDM Hub 환경에 대한 정보가 설명되어 있습니다.

수집되는 정보	설명
호스트 유형	호스트의 유형
빌드 번호	MDM Hub 빌드 번호
응용 프로그램 서버	응용 프로그램 서버 속성
Java	Java 속성
Javaopts	Java 옵션
버전 기록	제품 버전, 구성 요소 버전, 서버 이름, 버전 번호, 설치 시작 시간, 설치 종료 시간 및 설치 상태
라이선스	각 MDM Hub 구성 요소의 라이선스 상태, 라이선스 유형, 라이선스 제한, 발급 날짜, 만료 날짜 및 라이선스 키
Log4j 서버	Log4j 서버 속성
MDM Hub 마스터 데이터베이스	MDM Hub 마스터 데이터베이스 속성

수집되는 정보	설명
데이터베이스	데이터베이스 URL, 데이터베이스 버전, 데이터베이스 유형 및 데이터베이스 사용자
연산 참조 저장소	데이터베이스 버전, 워크플로우 엔진 이름, 패키지 정보, 제목 영역 이름, SSA 채우기 정보, 리포지토리 테이블 정보, Informatica Data Director 사용자 수 및 시스템 이름

Hub 서버에서 MDM Hub 데이터 수집 활성화

Hub 서버에서 MDM Hub 데이터 수집을 활성화하려면 `mdmsupport.properties` 파일을 편집합니다.

- 다음 위치로 이동합니다.
 - UNIX의 경우. <MDM Hub 설치 디렉터리>/hub/server/support
 - Windows의 경우. <MDM Hub 설치 디렉터리>\hub\server\support
- 텍스트 편집기에서 `mdmsupport.properties` 파일을 엽니다.
- `phonehome.send_csm_files_to_informatica`를 1로 설정합니다.
- 파일을 저장하고 닫습니다.
- 응용 프로그램 서버를 다시 시작합니다.

처리 서버에서 MDM Hub 데이터 수집 활성화

처리 서버에서 MDM Hub 데이터 수집을 활성화하려면 `mdmsupport.properties` 파일을 편집합니다.

- 다음 위치로 이동합니다.
 - UNIX의 경우. <MDM Hub 설치 디렉터리>/hub/cleanse/support
 - Windows의 경우. <MDM Hub 설치 디렉터리>\hub\cleanse\support
- 텍스트 편집기에서 `mdmsupport.properties` 파일을 엽니다.
- `phonehome.send_csm_files_to_informatica`를 1로 설정합니다.
- 파일을 저장하고 닫습니다.
- 응용 프로그램 서버를 다시 시작합니다.

Hub 서버에서 MDM Hub 데이터 수집 비활성화

Hub 서버에서 MDM Hub 데이터 수집을 비활성화하려면 `mdmsupport.properties` 파일을 편집합니다.

- 다음 위치로 이동합니다.

- UNIX의 경우. <MDM Hub 설치 디렉터리>/hub/server/support
 - Windows의 경우. <MDM Hub 설치 디렉터리>\hub\server\support
2. 텍스트 편집기에서 mdmsupport.properties 파일을 엽니다.
 3. phonehome.send_csm_files_to_informatica를 0으로 설정합니다.
 4. 파일을 저장하고 닫습니다.
 5. 응용 프로그램 서버를 다시 시작합니다.

처리 서버에서 MDM Hub 데이터 수집 비활성화

처리 서버에서 MDM Hub 데이터 수집을 비활성화하려면 mdmsupport.properties 파일을 편집합니다.

1. 다음 위치로 이동합니다.
 - UNIX의 경우. <MDM Hub 설치 디렉터리>/hub/cleanse/support
 - Windows의 경우. <MDM Hub 설치 디렉터리>\hub\cleanse\support
2. 텍스트 편집기에서 mdmsupport.properties 파일을 엽니다.
3. phonehome.send_csm_files_to_informatica를 0으로 설정합니다.
4. 파일을 저장하고 닫습니다.
5. 응용 프로그램 서버를 다시 시작합니다.

부록 G

Informatica 플랫폼 준비

이 부록에 포함된 항목:

- [Informatica Platform 준비 개요, 655](#)
- [Informatica 플랫폼과 MDM Hub 통합, 656](#)
- [Informatica 플랫폼 준비 프로세스, 665](#)
- [통합 선행 조건 완료, 665](#)
- [준비에 사용할 MDM Hub 준비, 666](#)
- [동기화에 사용할 개발자 도구 준비, 669](#)
- [Hub 저장소와 모델 리포지토리 동기화, 671](#)
- [개발자 도구에서 준비 설정 완료, 675](#)
- [매핑 구성 및 실행, 696](#)
- [준비 테이블 관리, 700](#)
- [추가 설명서, 702](#)

Informatica Platform 준비 개요

참고: Informatica Platform 준비는 버전 10.4에서 더 이상 사용되지 않습니다. Informatica 데이터 통합 및 Informatica Data Quality를 계속 사용하여 데이터를 MDM Hub 준비 테이블에 직접 로드할 수 있습니다. 이렇게 하려면 Informatica 데이터 통합 및 Informatica Data Quality 리포지토리를 사용하여 변경 내용을 준비 테이블에 수동으로 동기화해야 합니다. 또는 MDM Hub 준비를 사용합니다.

Informatica Platform 준비는 소스 데이터를 MDM Hub 준비 테이블에 바로 로드하는 프로세스입니다. Informatica Platform 준비를 수행하려면 MDM Hub을 Informatica Platform과 통합해야 합니다. MDM Hub을 Informatica Platform과 통합하면 데이터 통합 서비스가 소스 데이터를 MDM Hub 준비 테이블에 바로 로드합니다.

MDM Hub을 Informatica 플랫폼과 통합하려면 MDM Hub과 Informatica 응용 프로그램 서비스 및 Informatica Developer(Developer tool)와 같은 Informatica 플랫폼 구성 요소를 설치합니다. 서비스를 설치한 후 데이터 통합 서비스 및 모델 리포지토리 서비스를 생성합니다. Hub 콘솔을 사용하여 MDM Hub과 Informatica 플랫폼을 통합합니다.

준비 테이블에 대한 변경 사항을 모델 리포지토리와 동기화합니다. 동기화 프로세스는 데이터 통합 서비스에서 준비에 사용할 데이터 개체를 생성합니다.

동기화 후 데이터 소스의 데이터를 준비 테이블에 로드하는 매핑을 생성합니다. 매핑을 실행하면 데이터 통합 서비스에서 소스 데이터를 MDM Hub 준비 테이블에 바로 로드합니다. 데이터는 랜딩 테이블을 거치지 않습니다. 준비 프로세스 동안 정리 작업을 수행하려면 매핑에서 변환을 구성합니다.

Informatica Platform 준비를 수행할 때는 델타 검색, 영구 삭제 검색 및 감사 내역을 설정할 수 없습니다.

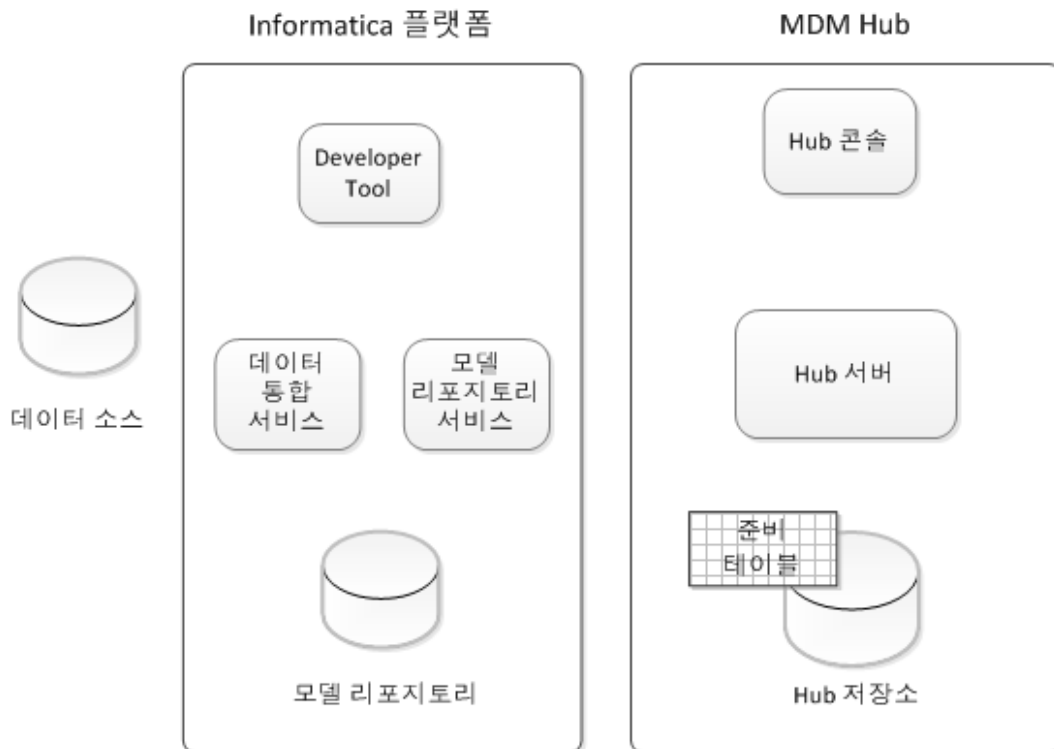
Informatica 플랫폼과 MDM Hub 통합

Informatica 플랫폼 준비를 수행하려면 MDM Hub을 Informatica 플랫폼과 통합합니다. 통합을 하려면 MDM Hub 설정 및 Informatica 플랫폼 구성 요소와 데이터 소스 연결이 필요합니다.

Informatica 플랫폼 준비 구성 요소

Informatica 플랫폼 준비를 위해 MDM Hub을 Informatica 플랫폼과 통합합니다.

다음 이미지는 통합 구성 요소를 보여 줍니다.



Informatica 플랫폼 준비에는 다음 구성 요소가 포함되어 있습니다.

Hub 콘솔

MDM Hub 기능에 액세스하는 클라이언트 응용 프로그램입니다. 모델 리포지토리와 연결하고 동기화하려면 Hub 콘솔을 사용하여 Hub 저장소를 구성합니다. 또한 Hub 콘솔을 사용하여 Informatica 플랫폼 준비에 대한 MDM Hub 준비 테이블을 활성화합니다.

Hub 서버

Hub 저장소 내에서 데이터를 처리하고 MDM Hub을 Informatica 플랫폼과 통합하는 J2EE 응용 프로그램입니다. Hub 서버는 런타임 구성요소로, MDM Hub에 대한 핵심 및 공통 서비스를 관리합니다.

Hub 저장소

MDM Hub에 대한 비즈니스 데이터를 저장하고 통합합니다. Hub 저장소의 특정 연산 참조 저장소에서 Informatica 플랫폼 준비에 대해 준비 테이블을 활성화합니다. 준비 테이블을 모델 리포지토리와 동기화하는 경우 모델 리포지토리 서비스가 모델 리포지토리를 Hub 저장소 메타데이터로 업데이트합니다. 준비 프로세스 중에 소스 시스템의 데이터가 Hub 저장소의 기본 개체와 연결된 준비 테이블로 이동합니다.

Developer tool

동기화 프로세스에서 생성하는 데이터 개체 및 맵셋을 편집하는 데 사용할 수 있는 응용 프로그램 클라이언트입니다. Developer tool을 사용하여 준비 프로세스에 대해 실행해야 하는 매핑을 생성합니다. Developer tool에서 볼 수 있는 개체는 모델 리포지토리에 저장되며 데이터 통합 서비스에서 실행됩니다. 준비를 수행하려면 Developer tool에서 매핑을 실행합니다.

데이터 통합 서비스

Developer tool에서 생성한 매핑을 통해 데이터를 준비 테이블로 데이터를 로드하는 Informatica 도메인의 응용 프로그램 서비스입니다. 데이터 통합 서비스는 Developer tool에서 수신한 요청을 처리하고, 매핑을 실행하고, MDM Hub 준비 테이블로 데이터를 로드합니다.

모델 리포지토리 서비스

모델 리포지토리를 관리하는 응용 프로그램 서비스입니다. 데이터 통합 서비스는 모델 리포지토리 서비스에 종속됩니다. Developer tool 또는 데이터 통합 서비스의 모델 리포지토리 개체에 액세스하면 모델 리포지토리 서비스에 요청이 전송됩니다. 모델 리포지토리 서비스는 모델 리포지토리 데이터베이스 테이블에서 메타데이터를 가져오고, 삽입하고, 업데이트합니다.

모델 리포지토리

관계형 데이터베이스에서 메타데이터를 저장하는 리포지토리입니다. 모델 리포지토리는 MDM Hub의 메타데이터를 저장합니다. MDM Hub 준비 테이블을 모델 리포지토리와 동기화하면 Hub 저장소의 메타데이터가 모델 리포지토리와 동기화됩니다. 동기화 프로세스에서는 준비 테이블을 기준으로 하여 개체를 생성합니다.

모델 리포지토리 개체

모델 리포지토리를 MDM Hub와 동기화하면 동기화 프로세스는 데이터 개체 및 맵셋이 생성되는 범위 내에서 연산 참조 저장소 이름을 기준으로 한 폴더를 생성합니다. 개체를 보려면 개발자 도구를 사용합니다.

참고: 모든 개체 이름은 준비 테이블 이름을 기준으로 합니다.

동기화 프로세스는 다음 개체를 생성합니다.

실제 데이터 개체

준비 테이블 데이터의 실제 표현입니다. 실제 데이터 개체를 생성, 편집 및 삭제할 수 있습니다.

Informatica 플랫폼 준비 프로세스를 구성하면 동기화 프로세스가 이름이 `C_<staging table name>`인 실제 데이터 개체를 생성합니다. 동기화 프로세스가 생성하는 실제 데이터 개체 중 하나는 사용자 지정된 데이터 개체로, 하나 이상의 관계형 리소스와 재사용 가능합니다.

논리적 데이터 개체 모델

준비 테이블의 데이터 구조 및 흐름을 설명하는 모델입니다. 동기화 프로세스는 MDM Hub의 각 준비 테이블에 대해 하나의 논리적 데이터 개체 모델을 생성합니다. 논리적 데이터 개체 모델의 이름은 `C_<staging table name>_Model`입니다. 모델에는 논리적 데이터 개체 및 맵셋이 포함됩니다. 논리적 데이터 개체 모델에서 논리적 데이터 개체 및 맵셋을 편집하고 삭제할 수 있습니다.

논리적 데이터 개체

MDM Hub 준비 테이블을 설명하는 논리적 데이터 개체 모델의 개체입니다. 모델 리포지토리를 MDM Hub 메타데이터와 동기화하면 동기화에서 이름이 `C_<staging table name>_LD0`인 하나의 논리적 데이터 개체가 생

성됩니다. 논리적 데이터 개체에는 논리적 데이터 개체 읽기 매핑 및 논리적 데이터 개체 쓰기 매핑이 포함됩니다.

논리적 데이터 개체 읽기 매핑

입력으로 실제 데이터 개체를, 출력으로 논리적 데이터 개체를 포함합니다. 매핑에서 데이터 통합 서비스는 매핑 소스의 데이터를 읽고 논리적 데이터 개체 읽기 매핑에서 데이터를 볼 수 있도록 해 줍니다.

논리적 데이터 개체 쓰기 매핑

입력으로 논리적 데이터 개체를 포함합니다. 논리적 데이터 개체 쓰기 매핑은 **Hub** 저장소의 대상 준비 테이블에 씁니다. 매핑에서 데이터 통합 서비스는 매핑이 **MDM Hub** 준비 테이블에 쓰기 전에, 맵렛을 통해 데이터를 처리합니다.

맵렛

입력 변환 및 출력 변환이 포함된 재사용 가능한 개체입니다. 입력 변환은 매핑의 업스트림 변환에 연결하여 소스 데이터를 가져올 수 있습니다. 출력 변환은 매핑의 다운스트림 변환에 연결하여 변환된 데이터를 대상 준비 테이블로 전송할 수 있습니다. 모델 리포지토리를 **MDM Hub** 메타데이터와 동기화하면 동기화에서 이름이 `C_<staging table name>_Mapplet`인 맵렛이 생성됩니다. 맵렛을 편집하고 삭제할 수 있습니다.

Informatica 플랫폼 준비를 수행하려면 다음 개체를 생성해야 합니다.

실제 데이터 개체

소스 데이터의 실제 표현입니다. 실제 데이터 개체를 생성, 편집 및 삭제할 수 있습니다. 실제 데이터 개체를 생성하여 소스 데이터에 연결해야 합니다.

매핑

소스 및 대상 준비 테이블 간의 데이터 흐름을 표현하는 입력 및 출력 집합입니다. 매핑은 데이터 변환에 대한 규칙을 정의하는 변환 개체로 연결할 수 있습니다. 데이터 통합 서비스는 매핑에서 구성하는 지침으로 소스의 데이터를 읽고, 데이터를 변환하고, 준비 테이블에 데이터를 씁니다.

준비 테이블 속성

Hub 콘솔을 통해 준비 테이블을 생성하고 관리할 수 있습니다. 준비 테이블을 생성할 때 일부 준비 테이블 속성을 구성합니다.

다음 표에서는 준비 테이블을 생성할 때 구성하는 준비 테이블 속성을 설명합니다.

속성	설명
표시 이름	Hub 콘솔에 표시되는 준비 테이블의 이름입니다.
실제 이름	데이터베이스에 있는 준비 테이블의 이름입니다. MDM Hub에서는 사용자가 입력한 표시 이름을 기준으로 준비 테이블에 대한 실제 이름을 제안합니다.
데이터 테이블스페이스	준비 테이블에 대한 데이터 테이블스페이스의 이름입니다.
인덱스 테이블스페이스	준비 테이블에 대한 인덱스 테이블스페이스의 이름입니다.
설명	준비 테이블에 대한 설명입니다.
테이블 유형	테이블 유형입니다. 기본값은 Staging입니다.
작성 날짜	준비 테이블이 작성된 날짜입니다.
시스템	준비 테이블 데이터의 소스 시스템입니다.

속성	설명
소스 시스템 키 유지	MDM Hub이 소스 시스템의 키 값을 사용해야 하는지 또는 MDM Hub이 생성한 키 값을 사용해야 하는지 지정합니다. 소스 시스템의 키 값을 사용하려면 활성화합니다. MDM Hub이 생성한 키 값을 사용하려면 비활성화합니다. 기본적으로 비활성화되어 있습니다. 참고: 준비 프로세스 동안 여러 레코드에 동일한 PKEY_SRC_OBJECT가 포함된 경우 존속 레코드는 LAST_UPDATE_DATE가 최근인 레코드가 됩니다. 다른 레코드는 거부 테이블로 보내집니다.
공백 채우기	새 레코드 버전을 추가할 경우 레코드 버전의 유효 날짜 간 연속성이 유지되는지 여부를 지정합니다. 활성화할 경우 새 레코드 버전을 기본 개체에 추가할 수 있을 때 MDM Hub에서 레코드 버전의 유효 날짜 간 연속성이 유지됩니다. 활성화하지 않을 경우 MDM Hub에서는 레코드 버전의 유효 기간 간 연속성을 깨뜨리는 레코드 버전의 추가를 거부합니다. 기본적으로 비활성화되어 있습니다.
최상위 예약된 키	첫 번째 로드 후 키를 증가시킬 수입니다. 소스 시스템 키 유지 확인란을 활성화한 경우 속성이 표시됩니다.
Informatica Platform 준비	MDM Hub가 Informatica Platform 준비를 사용하는지 여부를 지정합니다. 기본적으로 비활성화되어 있습니다.
모델 리포지토리 서비스와 동기화	MDM Hub 메타데이터가 모델 리포지토리와 동기화되는지 여부를 지정합니다. 기본적으로 비활성화되어 있습니다.
셀 업데이트	준비 테이블에서 오는 수신 레코드의 값이 같을 경우 MDM Hub에서 대상 테이블의 셀을 업데이트할 수 있도록 지정합니다.
열	준비 테이블의 열입니다.

데이터 소스 연결 속성

Informatica 클라이언트를 통해 데이터 소스 연결을 생성하고 관리할 수 있습니다. 연결을 생성하여 소스 시스템에서 데이터를 가져옵니다. 적절한 연결 속성을 지정하여 Oracle, IBM DB2 및 Microsoft SQL Server 연결을 생성하고 관리합니다.

IBM DB2 연결 속성

IBM DB2 연결을 사용하여 IBM DB2에 액세스합니다. IBM DB2 연결은 관계형 데이터베이스 연결입니다. IBM DB2 연결은 개발자 도구에서 생성하고 관리할 수 있습니다.

다음 테이블에서는 IBM DB2 연결 속성을 설명합니다.

속성	설명
데이터베이스 유형	데이터베이스 유형입니다.
이름	연결 이름입니다. 이름은 대/소문자를 구분하지 않으며 도메인 내에서 고유해야 합니다. 이름은 128자를 초과할 수 없으며 공백 또는 다음 특수 문자를 포함할 수 없습니다. ~ ` ! \$ % ^ & * () - + = { [] } \ : ; " ' < , > . ? /
ID	데이터 통합 서비스에서 연결을 식별하는 데 사용하는 문자열입니다. ID는 대/소문자를 구분하지 않습니다. ID는 255자 이하여야 하며 도메인에서 고유해야 합니다. 이 속성은 연결을 생성한 후에는 변경할 수 없습니다. 기본값은 연결 이름입니다.

속성	설명
설명	연결의 설명입니다. 설명은 765자를 초과할 수 없습니다.
사용자 이름	데이터베이스 사용자 이름입니다.
암호	데이터베이스 사용자 이름의 암호입니다.
통과 보안 활성화됨	연결에 대해 통과 보안을 활성화합니다. 연결에 대해 통과 보안을 활성화하면 도메인 이 연결 개체에서 정의된 자격 증명 대신, 클라이언트 사용자 이름 및 암호를 사용하여 해당하는 데이터베이스에 로그인합니다.
데이터 액세스에 대한 연결 문자열	데이터베이스에서 메타데이터에 액세스하는 데 사용하는 IBM DB2 연결 URL입니다. dbname dbname은 IBM DB2 클라이언트에서 구성된 별칭입니다.
메타데이터 액세스 속성: 연결 문자열	메타데이터에 액세스하는 데 사용하는 연결 문자열입니다. 다음 연결 URL을 사용합니다. jdbc:informatica:db2://<host name>:<port>;DatabaseName=<database name>
AdvancedJDBCSecurityOptions	<p>보안 데이터베이스에 대한 메타데이터 액세스의 데이터베이스 매개 변수입니다. Informatica는 AdvancedJDBCSecurityOptions 필드의 값을 중요한 데이터로 처리하며 암호화된 매개 변수 문자열을 저장합니다.</p> <p>보안 데이터베이스에 연결하려면 다음 매개 변수를 포함합니다.</p> <ul style="list-style-type: none"> - EncryptionMethod. 필수입니다. 네트워크를 통해 전송하는 경우 데이터의 암호화 여부를 나타냅니다. 이 매개 변수는 SSL로 설정되어야 합니다. - ValidateServerCertificate. 선택 사항입니다. 데이터베이스 서버에서 보낸 인증서에 대해 Informatica가 유효성을 검사하는지 나타냅니다. 이 매개 변수를 True로 설정하면 데이터베이스 서버에서 보낸 인증서에 대해 Informatica에서 유효성을 검사합니다. HostNameInCertificate 매개 변수를 지정하면 Informatica에서 인증서의 호스트 이름에 대한 유효성도 검사합니다. 이 매개 변수를 false로 설정하면 데이터베이스 서버에서 보낸 인증서에 대해 Informatica에서 유효성을 검사하지 않습니다. Informatica에서 사용자가 지정한 트러스트 저장소 정보를 모두 무시합니다. - HostNameInCertificate. 선택 사항입니다. 보안 데이터베이스를 호스팅하는 시스템의 호스트 이름입니다. 호스트 이름을 지정하면 Informatica가 SSL 인증서의 호스트 이름에 대해 연결 문자열에 포함된 호스트 이름의 유효성을 검사합니다. - TrustStore. 필수입니다. 데이터베이스에 대한 SSL 인증서를 포함하는 트러스트 저장소 파일의 경로 및 파일 이름입니다. - TrustStorePassword. 필수입니다. 보안 데이터베이스에 대한 트러스트 저장소 파일의 암호입니다. <p>참고: Informatica가 보안 JDBC 매개 변수를 연결 문자열에 추가합니다. 보안 JDBC 매개 변수를 연결 문자열에 바로 포함할 경우 AdvancedJDBCSecurityOptions 필드에 어떤 매개 변수도 입력하지 마십시오.</p>
데이터 액세스 속성: 연결 문자열	데이터베이스의 데이터에 액세스하는 데 사용하는 연결 문자열입니다. IBM DB2의 경우 이는 <데이터베이스 이름>입니다.
코드 페이지	소스 데이터베이스에서 읽거나 대상 데이터베이스 또는 파일에 쓰는 데 사용하는 코드 페이지입니다.
환경 SQL	데이터베이스 연결 시 데이터베이스 환경을 설정하는 SQL 명령입니다. 데이터 통합 서비스는 데이터베이스에 연결할 때마다 연결 환경 SQL을 실행합니다.

속성	설명
트랜잭션 SQL	데이터베이스 연결 시 데이터베이스 환경을 설정하는 SQL 명령입니다. 데이터 통합 서비스는 각 트랜잭션 시작 시 트랜잭션 환경 SQL을 실행합니다.
다시 시도 기간	이 속성은 나중에 사용하기 위해 예약되어 있습니다.
테이블스페이스	데이터베이스의 테이블스페이스 이름입니다.
SQL 식별자 문자	특수 문자 및 예약된 SQL 키워드(예: WHERE)를 식별하는 데 사용하는 문자의 유형입니다. 데이터 통합 서비스는 특수 문자 및 예약된 SQL 키워드 주위에 특정 문자를 배치합니다. 또한 데이터 통합 서비스에서는 대/소문자가 혼합된 식별자 지원 속성에 대해서도 이 문자를 사용합니다. 연결의 데이터베이스를 기준으로 하여 문자를 선택합니다.
대/소문자가 혼합된 식별자 지원	활성화한 경우 연결에서 테이블, 보기, 스키마, 동의어 및 열 이름에 대해 SQL을 생성하고 실행할 때 데이터 통합 서비스는 이러한 개체 주위에 식별자 문자를 배치합니다. 개체 이름에 대/소문자가 혼합되어 있거나 소문자 이름이 있으면 사용합니다. 기본적으로 이 옵션은 선택되어 있지 않습니다.
ODBC 공급자	ODBC. ODBC에서 연결하는 데이터베이스의 유형입니다. 다음 데이터베이스 옵션 중 하나를 선택합니다. - 기타 - Sybase - Microsoft_SQL_Server 기본값은 기타입니다.

Microsoft SQL Server 연결 속성

Microsoft SQL Server 연결을 사용하여 Microsoft SQL Server에 액세스합니다. Microsoft SQL Server 연결은 Microsoft SQL Server 관계형 데이터베이스 연결입니다. 개발자 도구에서 Microsoft SQL Server 연결을 생성하고 관리할 수 있습니다.

다음 테이블에서는 Microsoft SQL Server 연결 속성을 설명합니다.

속성	설명
데이터베이스 유형	데이터베이스 유형입니다.
이름	연결 이름입니다. 이름은 대/소문자를 구분하지 않으며 도메인 내에서 고유해야 합니다. 이름은 128자를 초과할 수 없으며 공백 또는 다음 특수 문자를 포함할 수 없습니다. ~ ` ! \$ % ^ & * () - + = { [] } \ : ; " ' < , > . ? /
ID	데이터 통합 서비스에서 연결을 식별하는 데 사용하는 문자열입니다. ID는 대/소문자를 구분하지 않습니다. ID는 255자 이하여야 하며 도메인에서 고유해야 합니다. 이 속성은 연결을 생성한 후에는 변경할 수 없습니다. 기본값은 연결 이름입니다.
설명	연결의 설명입니다. 설명은 765자를 초과할 수 없습니다.
트러스트된 연결 사용	응용 프로그램 서비스를 활성화하여 Windows 인증을 통해 데이터베이스에 액세스합니다. 응용 프로그램 서비스를 시작하는 사용자 이름은 데이터베이스 액세스 권한이 있는 올바른 Windows 사용자에게야 합니다. 기본적으로 이 옵션은 지워져 있습니다.
사용자 이름	데이터베이스 사용자 이름입니다.

속성	설명
암호	데이터베이스 사용자 이름의 암호입니다.
통과 보안 활성화됨	연결에 대해 통과 보안을 활성화합니다. 연결에 대해 통과 보안을 활성화하면 도메인 이 연결 개체에서 정의된 자격 증명 대신, 클라이언트 사용자 이름 및 암호를 사용하여 해당하는 데이터베이스에 로그인합니다.
메타데이터 액세스 속성: 연결 문자열	메타데이터에 액세스하는 데 사용하는 연결 문자열입니다. 다음 연결 URL을 사용합니다. <code>jdbc:informatica:sqlserver://<host name>:<port>;DatabaseName=<database name></code>
AdvancedJDBCSecurityOptions	보안 데이터베이스에 대한 메타데이터 액세스의 데이터베이스 매개 변수입니다. Informatica는 AdvancedJDBCSecurityOptions 필드의 값을 중요한 데이터로 처리하며 암호화된 매개 변수 문자열을 저장합니다. 보안 데이터베이스에 연결하려면 다음 매개 변수를 포함합니다. <ul style="list-style-type: none"> - EncryptionMethod. 필수입니다. 네트워크를 통해 전송하는 경우 데이터의 암호화 여부를 나타냅니다. 이 매개 변수는 SSL로 설정되어야 합니다. - ValidateServerCertificate. 선택 사항입니다. 데이터베이스 서버에서 보낸 인증서에 대해 Informatica가 유효성을 검사하는지 나타냅니다. 이 매개 변수를 True로 설정하면 데이터베이스 서버에서 보낸 인증서에 대해 Informatica에서 유효성을 검사합니다. HostNameInCertificate 매개 변수를 지정하면 Informatica에서 인증서의 호스트 이름에 대한 유효성도 검사합니다. 이 매개 변수를 false로 설정하면 데이터베이스 서버에서 보낸 인증서에 대해 Informatica에서 유효성을 검사하지 않습니다. Informatica에서 사용자가 지정한 트러스트 저장소 정보를 모두 무시합니다. - HostNameInCertificate. 선택 사항입니다. 보안 데이터베이스를 호스팅하는 시스템의 호스트 이름입니다. 호스트 이름을 지정하면 Informatica가 SSL 인증서의 호스트 이름에 대해 연결 문자열에 포함된 호스트 이름의 유효성을 검사합니다. - TrustStore. 필수입니다. 데이터베이스에 대한 SSL 인증서를 포함하는 트러스트 저장소 파일의 경로 및 파일 이름입니다. - TrustStorePassword. 필수입니다. 보안 데이터베이스에 대한 트러스트 저장소 파일의 암호입니다. ODBC에 적용할 수 없습니다. 참고: Informatica가 보안 JDBC 매개 변수를 연결 문자열에 추가합니다. 보안 JDBC 매개 변수를 연결 문자열에 바로 포함할 경우 AdvancedJDBCSecurityOptions 필드에 어떤 매개 변수도 입력하지 마십시오.
데이터 액세스 속성: 연결 문자열	데이터베이스의 데이터에 액세스하는 데 사용하는 연결 문자열입니다. 다음 연결 문자열을 사용합니다. <code><server name>@<database name></code> 데이터베이스에서 기본 포트를 사용하지 않는 경우 다음 연결 문자열을 사용합니다. <code><server name>:<port>@<dbname></code> <code><servername>/<instancename>:<port>@<dbname></code>
코드 페이지	소스 데이터베이스에서 읽거나 대상 데이터베이스 또는 파일에 쓰는 데 사용하는 코드 페이지입니다.
도메인 이름	도메인의 이름입니다.
패킷 크기	데이터를 전송하는 데 사용하는 패킷 크기입니다. Microsoft SQL Server에 대한 원시 드라이버를 최적화하는 데 사용됩니다.

속성	설명
소유자 이름	스키마의 소유자 이름입니다.
스키마 이름	데이터베이스에 있는 스키마의 이름입니다. 스키마 이름이 데이터베이스 사용자 이름과 다른 경우 프로파일링 웨어하우스에 대해 스키마 이름을 지정해야 합니다. 스키마 이름이 데이터베이스 사용자 이름과 다르고 외부 도구로 캐시를 관리하는 경우 데이터 개체 캐시 데이터베이스에 대해 스키마 이름을 지정해야 합니다.
환경 SQL	데이터베이스 연결 시 데이터베이스 환경을 설정하는 SQL 명령입니다. 데이터 통합 서비스는 데이터베이스에 연결할 때마다 연결 환경 SQL을 실행합니다.
트랜잭션 SQL	데이터베이스 연결 시 데이터베이스 환경을 설정하는 SQL 명령입니다. 데이터 통합 서비스는 각 트랜잭션 시작 시 트랜잭션 환경 SQL을 실행합니다.
다시 시도 기간	이 속성은 나중에 사용하기 위해 예약되어 있습니다.
SQL 식별자 문자	특수 문자 및 예약된 SQL 키워드(예: WHERE)를 식별하는 데 사용하는 문자의 유형입니다. 데이터 통합 서비스는 특수 문자 및 예약된 SQL 키워드 주위에 특정 문자를 배치합니다. 또한 데이터 통합 서비스에서는 대/소문자가 혼합된 식별자 지원 속성에 대해서도 이 문자를 사용합니다. 연결의 데이터베이스를 기준으로 하여 문자를 선택합니다.
대/소문자가 혼합된 식별자 지원	활성화한 경우 연결에서 테이블, 보기, 스키마, 동의어 및 열 이름에 대해 SQL을 생성하고 실행할 때 데이터 통합 서비스는 이러한 개체 주위에 식별자 문자를 배치합니다. 개체 이름에 대/소문자가 혼합되어 있거나 소문자 이름이 있으면 사용합니다. 기본적으로 이 옵션은 선택되어 있지 않습니다.
ODBC 공급자	ODBC. ODBC에서 연결하는 데이터베이스의 유형입니다. 다음 데이터베이스 옵션 중 하나를 선택합니다. - 기타 - Sybase - Microsoft_SQL_Server 기본값은 기타입니다.

Oracle 연결 속성

Oracle 연결을 사용하여 Oracle 데이터베이스에 연결합니다. Oracle 연결은 관계형 연결 유형입니다. 개발자 도구에서 Oracle 연결을 생성하고 관리할 수 있습니다.

다음 표에는 Oracle 연결 속성이 설명되어 있습니다.

속성	설명
데이터베이스 유형	데이터베이스 유형입니다.
이름	연결 이름입니다. 이름은 대/소문자를 구분하지 않으며 도메인 내에서 고유해야 합니다. 이름은 128자를 초과할 수 없으며 공백 또는 다음 특수 문자를 포함할 수 없습니다. ~ ` ! \$ % ^ & * () - + = { [] } \ : ; " ' < , > . ? /
ID	데이터 통합 서비스에서 연결을 식별하는 데 사용하는 문자열입니다. ID는 대/소문자를 구분하지 않습니다. ID는 255자 이하여야 하며 도메인에서 고유해야 합니다. 이 속성은 연결을 생성한 후에는 변경할 수 없습니다. 기본값은 연결 이름입니다.

속성	설명
설명	연결의 설명입니다. 설명은 765자를 초과할 수 없습니다.
사용자 이름	데이터베이스 사용자 이름입니다.
암호	데이터베이스 사용자 이름의 암호입니다.
통과 보안 활성화됨	연결에 대해 통과 보안을 활성화합니다. 연결에 대해 통과 보안을 활성화하면 도메인 이 연결 개체에서 정의된 자격 증명 대신, 클라이언트 사용자 이름 및 암호를 사용하여 해당하는 데이터베이스에 로그인합니다.
메타데이터 액세스 속성: 연결 문자열	메타데이터에 액세스하는 데 사용하는 연결 문자열입니다. 다음 연결 URL을 사용합니다. <code>jdbc:informatica:oracle://<host_name>:<port>;SID=<database name></code>
AdvancedJDBCSecurityOptions	<p>보안 데이터베이스에 대한 메타데이터 액세스의 데이터베이스 매개 변수입니다. Informatica는 AdvancedJDBCSecurityOptions 필드의 값을 중요한 데이터로 처리하며 암호화된 매개 변수 문자열을 저장합니다.</p> <p>보안 데이터베이스에 연결하려면 다음 매개 변수를 포함합니다.</p> <ul style="list-style-type: none"> - EncryptionMethod. 필수입니다. 네트워크를 통해 전송하는 경우 데이터의 암호화 여부를 나타냅니다. 이 매개 변수는 SSL로 설정되어야 합니다. - ValidateServerCertificate. 선택 사항입니다. 데이터베이스 서버에서 보낸 인증서에 대해 Informatica가 유효성을 검사하는지 나타냅니다. 이 매개 변수를 True로 설정하면 데이터베이스 서버에서 보낸 인증서에 대해 Informatica에서 유효성을 검사합니다. HostNameInCertificate 매개 변수를 지정하면 Informatica에서 인증서의 호스트 이름에 대한 유효성도 검사합니다. 이 매개 변수를 false로 설정하면 데이터베이스 서버에서 보낸 인증서에 대해 Informatica에서 유효성을 검사하지 않습니다. Informatica에서 사용자가 지정한 트러스트 저장소 정보를 모두 무시합니다. - HostNameInCertificate. 선택 사항입니다. 보안 데이터베이스를 호스팅하는 시스템의 호스트 이름입니다. 호스트 이름을 지정하면 Informatica가 SSL 인증서의 호스트 이름에 대해 연결 문자열에 포함된 호스트 이름의 유효성을 검사합니다. - TrustStore. 필수입니다. 데이터베이스에 대한 SSL 인증서를 포함하는 트러스트 저장소 파일의 경로 및 파일 이름입니다. - TrustStorePassword. 필수입니다. 보안 데이터베이스에 대한 트러스트 저장소 파일의 암호입니다. <p>참고: Informatica가 보안 JDBC 매개 변수를 연결 문자열에 추가합니다. 보안 JDBC 매개 변수를 연결 문자열에 바로 포함할 경우 AdvancedJDBCSecurityOptions 필드에 어떤 매개 변수도 입력하지 마십시오.</p>
데이터 액세스 속성: 연결 문자열	데이터베이스의 데이터에 액세스하는 데 사용하는 연결 문자열입니다. 다음 연결 문자열을 사용합니다. <code><database name>.world</code>
코드 페이지	소스 데이터베이스에서 읽거나 대상 데이터베이스 또는 파일에 쓰는 데 사용하는 코드 페이지입니다.
환경 SQL	데이터베이스 연결 시 데이터베이스 환경을 설정하는 SQL 명령입니다. 데이터 통합 서비스는 데이터베이스에 연결할 때마다 연결 환경 SQL을 실행합니다.
트랜잭션 SQL	데이터베이스 연결 시 데이터베이스 환경을 설정하는 SQL 명령입니다. 데이터 통합 서비스는 각 트랜잭션 시작 시 트랜잭션 환경 SQL을 실행합니다.

속성	설명
다시 시도 기간	이 속성은 나중에 사용하기 위해 예약되어 있습니다.
병렬 모드 설정	대량 모드의 테이블에 데이터를 로드하는 경우 병렬 처리를 활성화합니다. 기본적으로 이 옵션은 지워져 있습니다.
SQL 식별자 문자	특수 문자 및 예약된 SQL 키워드(예: WHERE)를 식별하는 데 사용하는 문자의 유형입니다. 데이터 통합 서비스는 특수 문자 및 예약된 SQL 키워드 주위에 특정 문자를 배치합니다. 또한 데이터 통합 서비스에서는 대/소문자가 혼합된 식별자 지원 속성에 대해서도 이 문자를 사용합니다. 연결의 데이터베이스를 기준으로 하여 문자를 선택합니다.
대/소문자가 혼합된 식별자 지원	활성화한 경우 연결에서 테이블, 보기, 스키마, 동의어 및 열 이름에 대해 SQL을 생성하고 실행할 때 데이터 통합 서비스는 이러한 개체 주위에 식별자 문자를 배치합니다. 개체 이름에 대/소문자가 혼합되어 있거나 소문자 이름이 있으면 사용합니다. 기본적으로 이 옵션은 선택되어 있지 않습니다.

Informatica 플랫폼 준비 프로세스

MDM Hub을 Informatica 플랫폼과 통합하면 데이터 통합 서비스가 데이터를 MDM Hub 준비 테이블에 로드합니다. 지정한 연결 매개변수가 MDM Hub을 활성화하여 데이터 통합 서비스 및 모델 리포지토리 서비스와 상호 작용합니다.

Informatica 플랫폼 준비를 구성하고 실행하려면 다음 태스크를 수행합니다.

1. 통합 선행 조건을 완료합니다.
2. 준비에 사용할 MDM Hub을 준비합니다.
3. 동기화에 사용할 개발자 도구를 준비합니다.
4. 모델 리포지토리를 Hub 저장소와 동기화합니다.
5. 개발자 도구에서 준비 설정을 완료합니다.
6. 매핑을 구성하고 실행합니다.

통합 선행 조건 완료

Informatica 플랫폼 준비를 수행하기 전에 모든 구성 요소가 설치 및 구성되었는지 확인합니다.

다음 설치 및 구성 태스크를 수행합니다.

1. MDM Hub을 설치 및 구성합니다.
2. Informatica 서비스를 설치하여 도메인을 생성합니다.
3. 다음 응용 프로그램 서비스를 생성하고 구성합니다.
 - 모델 리포지토리 서비스
 - 데이터 통합 서비스
4. Informatica Developer(개발자 도구)를 설치하고 구성하여 모델 리포지토리에 연결합니다.

준비에 사용할 MDM Hub 준비

MDM Hub을 설치하고 구성한 후 준비에 사용할 MDM Hub을 준비합니다. 소스 시스템을 구성하고 Hub 콘솔을 사용하여 연산 참조 저장소에서 준비 테이블을 추가합니다.

참고: MDM Hub에서 INT 데이터 유형의 준비 테이블 열을 생성하면 개발자 도구에서 DECIMAL 데이터 유형으로 표시됩니다.

1단계. 소스 시스템 구성

다양한 소스 시스템에서 데이터 입력을 관리하려면 MDM Hub에 각 소스 시스템에 대해 고유한 내부 이름이 있어야 합니다. MDM Hub에 대한 소스 시스템을 정의하려면 모델 작업 영역에서 시스템 및 트러스트 도구를 사용합니다.

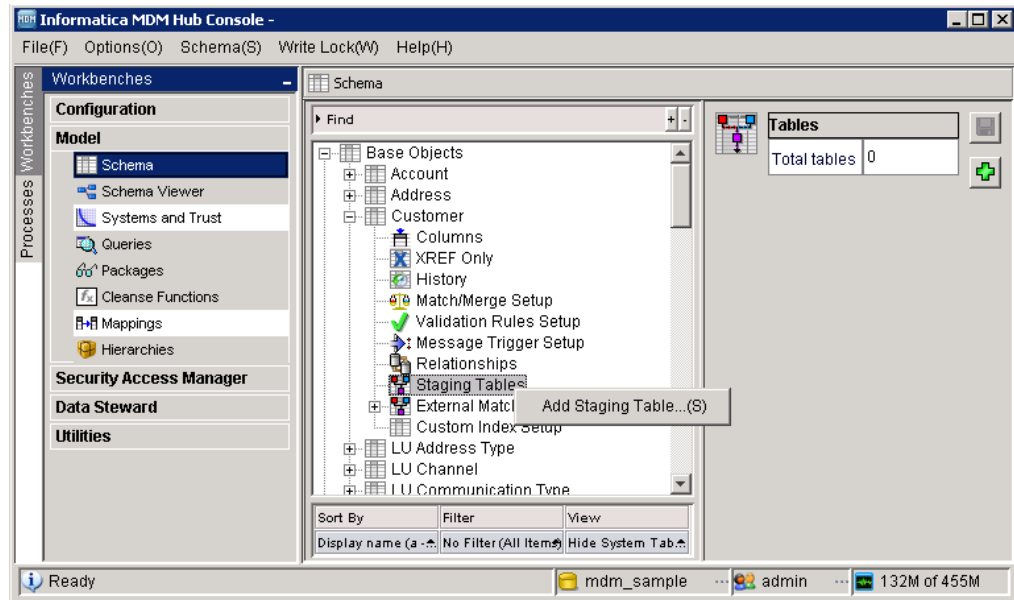
1. Hub 콘솔에서 시스템 및 트러스트 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 시스템 및 트러스트 창에서 마우스 오른쪽 단추를 클릭합니다.
시스템 추가 옵션이 표시됩니다.
4. 시스템 추가를 클릭합니다.
새 시스템 대화 상자가 표시됩니다.
5. 소스 시스템을 식별할 수 있도록 고유한 이름 및 설명을 지정하고 **확인**을 클릭합니다.
시스템 ID 속성 테이블이 가장 오른쪽 창에 표시됩니다.
6. 필요한 경우 시스템 ID 속성을 편집하고 **저장**을 클릭합니다.

2단계. 준비 테이블 추가

준비 데이터를 로드할 준비 테이블을 추가합니다.

1. Hub 콘솔에서 스키마 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서, 준비 테이블을 추가할 기본 개체의 노드를 확장합니다.
4. 준비 테이블 노드를 마우스 오른쪽 단추로 클릭합니다.
준비 테이블 추가 옵션이 표시됩니다.

다음 이미지는 준비 테이블을 고객 기본 개체에 추가하는 **준비 테이블 추가** 옵션을 보여 줍니다.



5. 준비 테이블 추가를 클릭합니다.
기본 개체에 준비 추가 대화 상자가 표시됩니다.
6. 준비 테이블 속성을 지정합니다.

다음 이미지는 기본 개체 고객에 준비 추가 대화 상자(대화 상자의 표시 이름 필드가 S_CRM_CUST로 설정되어 있음)를 보여 줍니다.

Table Identity	
Display name	S_CRM_CUST
Physical name	C_S_CRM_CUST
System	SFA
Preserve Source System Keys	<input type="checkbox"/>
Data Tablespace	CMX_DATA
Index Tablespace	CMX_INDEX
Description	

Columns							
		Column	Lookup Sy...	Lookup Table	Lookup Col...	Allow Null ...	Allow Null ...
<input checked="" type="checkbox"/>	<input type="checkbox"/>	First Name				<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Last Name				<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Gender Cd				<input type="checkbox"/>	<input type="checkbox"/>

☐ Cell update

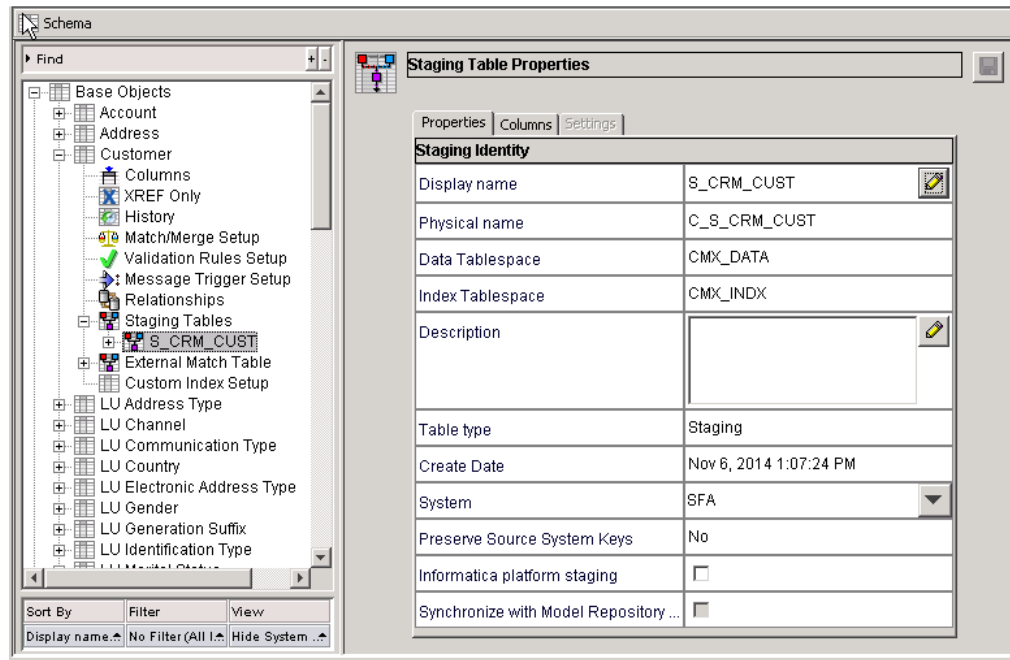
OK Cancel

7. 기본 개체의 열 목록에서 소스 시스템이 제공하는 열을 선택합니다.
모든 열 선택을 클릭하면 모든 기본 개체 열을 선택할 수 있습니다.
8. 열 속성을 지정합니다.

9. **확인**을 클릭합니다.

스키마 도구가 연산 참조 저장소에 지원 테이블과 함께 준비 테이블을 생성한 다음 탐색 트리에서 준비 테이블을 추가합니다.

다음 이미지는 탐색 트리의 S_CRM_CUST 준비 테이블을 보여 줍니다.



동기화에 사용할 개발자 도구 준비

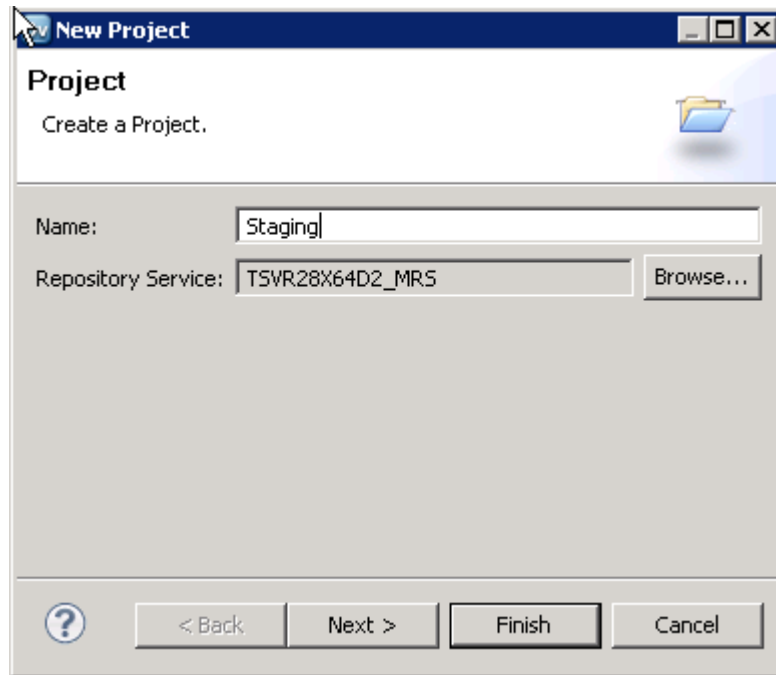
Informatica 서비스 및 개발자 도구를 설치 및 구성한 후에 동기화에 사용할 개발자 도구를 준비합니다. 준비 개체에 액세스할 개발자 도구에서 프로젝트를 생성합니다.

1단계. 프로젝트 생성

동기화 프로세스에서 생성한 개체를 저장하려면 개발자 도구를 사용하여 모델 리포지토리에서 프로젝트를 생성합니다.

1. **개체 탐색기** 보기에서 모델 리포지토리 서비스를 선택합니다.
2. **파일 > 새로 만들기 > 프로젝트**를 클릭합니다.
3. 프로젝트에 대한 이름을 입력합니다.

다음 이미지는 이름 필드에 프로젝트 이름이 준비로 되어 있는 새 프로젝트 대화 상자를 보여 줍니다.

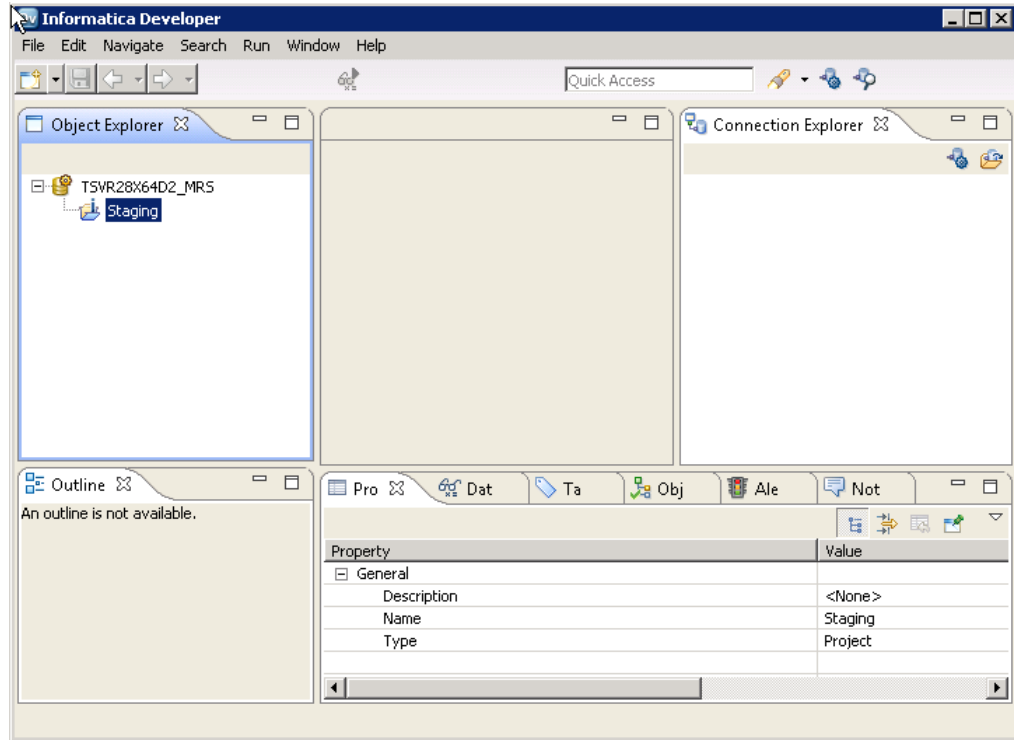


4. 다음을 클릭합니다.
새 프로젝트 대화 상자의 프로젝트 권한 페이지가 표시됩니다.
5. 선택적으로 사용자 또는 그룹을 선택하고 권한을 할당합니다.

6. 마침을 클릭합니다.

개체 탐색기 보기의 모델 리포지토리 서비스 아래에 프로젝트가 표시됩니다.

다음 이미지는 이름이 준비된 프로젝트가 포함되어 있는 TSVR28X64D2_MRS라는 이름의 모델 리포지토리 서비스가 있는 개체 탐색기 보기를 보여 줍니다.



Hub 저장소와 모델 리포지토리 동기화

동기화에 사용할 개발자 도구를 준비한 후에, 모델 리포지토리 서비스 연결을 구성하고 준비를 활성화한 후 모델 리포지토리를 Hub 저장소와 동기화합니다.

모델 리포지토리 서비스 연결 매개 변수

모델 리포지토리를 MDM Hub와 동기화하려면 MDM Hub에서 모델 리포지토리 서비스에 연결할 수 있어야 합니다. Hub 콘솔의 엔터프라이즈 관리자 도구에서 연결 매개 변수를 구성합니다.

모델 리포지토리 서비스에 연결하려면 다음 연결 매개 변수를 구성합니다.

projectName

모델 리포지토리의 프로젝트 이름입니다.

domainPort

MDM Hub가 이 도메인의 서비스와 통신하는 데 사용하는 포트 번호입니다. 기본값은 6005입니다.

domainHost

마스터 게이트웨이 노드를 호스팅하는 도메인의 시스템 이름입니다.

참고: localhost는 사용하지 마십시오. 호스트 이름은 명시적으로 시스템을 식별해야 합니다.

repositoryName

모델 리포지토리의 이름입니다.

securityDomain

보안 도메인의 이름입니다. 값은 **Native**여야 합니다.

사용자 이름

모델 리포지토리에 액세스하는 데 필요한 사용자 이름입니다.

암호

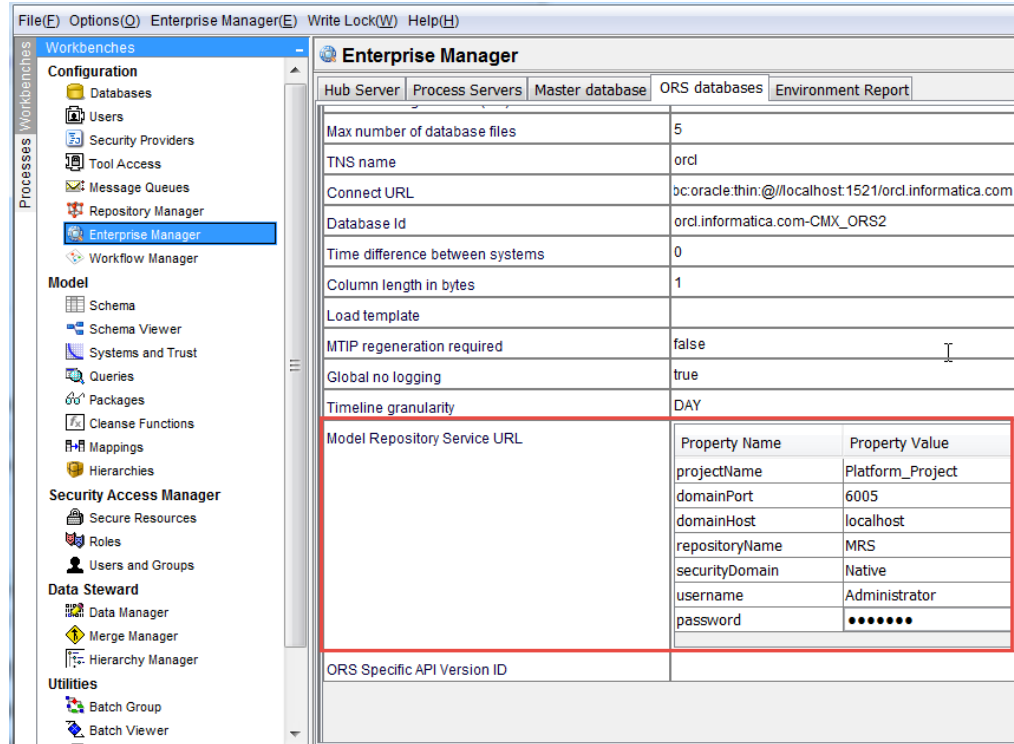
모델 리포지토리에 액세스하는 데 사용하는 암호입니다. 암호는 보안상의 이유로 마스킹됩니다.

1단계. 모델 리포지토리 서비스 연결 구성

MDM Hub과 모델 리포지토리 서비스 간의 연결을 구성합니다. MDM Hub과 모델 리포지토리 간 데이터를 동기화하려면 연결이 필요합니다.

1. Hub 콘솔에서 엔터프라이즈 관리자 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. **ORS 데이터베이스** 탭에서 연산 참조 저장소를 선택합니다.
4. **속성** 탭을 클릭합니다.
연산 참조 저장소 데이터베이스 속성이 표시됩니다.
5. **모델 리포지토리 서비스 URL** 필드에서 모델 리포지토리 서비스 연결 매개 변수를 입력하고 **저장**을 클릭합니다.
매개 변수가 저장되었음을 나타내는 메시지가 표시됩니다.

다음 이미지는 엔터프라이즈 관리자 도구의 **모델 리포지토리 서비스 URL** 필드를 보여 줍니다.



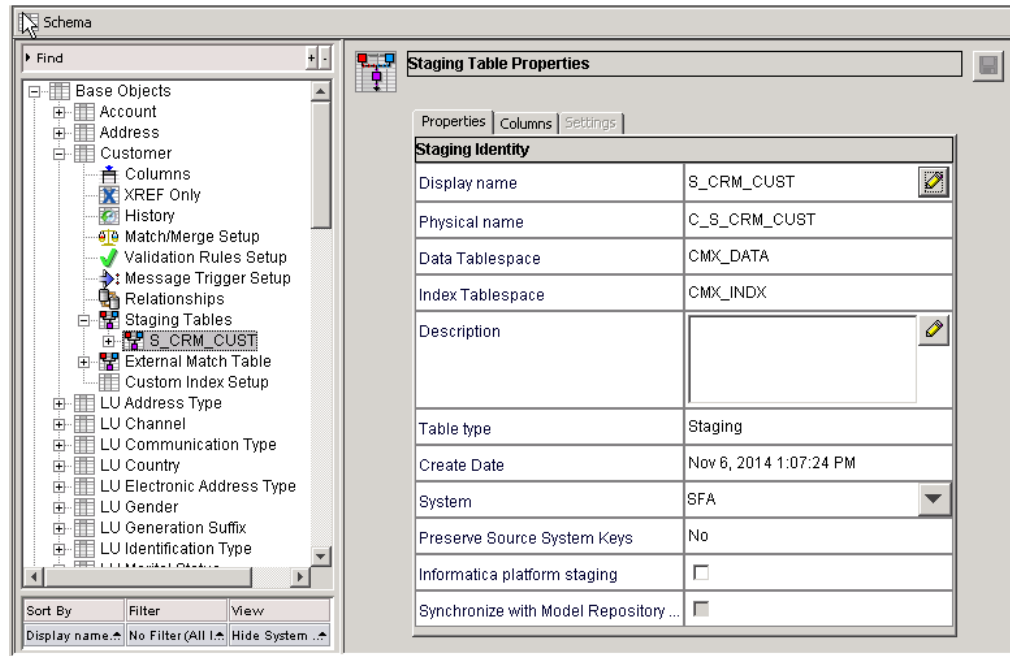
6. **확인**을 클릭합니다.
MDM Hub이 모델 리포지토리 서비스에 연결합니다.
7. 응용 프로그램 서버 및 Hub 콘솔을 다시 시작합니다.

2단계. 준비 활성화

Hub 콘솔을 사용하여 MDM Hub 준비 테이블에 대한 Informatica 플랫폼 준비를 활성화합니다.

1. 스키마 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. **데이터베이스 선택**을 클릭합니다.
데이터베이스 변경 대화 상자가 표시됩니다.
4. 데이터를 저장할 연산 참조 저장소를 선택하고 **연결**을 클릭합니다.
5. 탐색 트리에서 준비에 사용해야 하는 기본 개체의 준비 테이블을 클릭합니다.
준비 테이블 속성 페이지가 나타납니다.

다음 이미지는 Informatica 플랫폼 준비를 활성화할 수 있는 **준비 테이블 속성** 페이지를 보여 줍니다.



- 속성 탭에서 **Informatica 플랫폼 준비**를 활성화하고 **저장**을 클릭합니다.
준비 테이블이 Informatica 플랫폼 준비에 대해 활성화됩니다.

3단계. 모델 리포지토리와 동기화

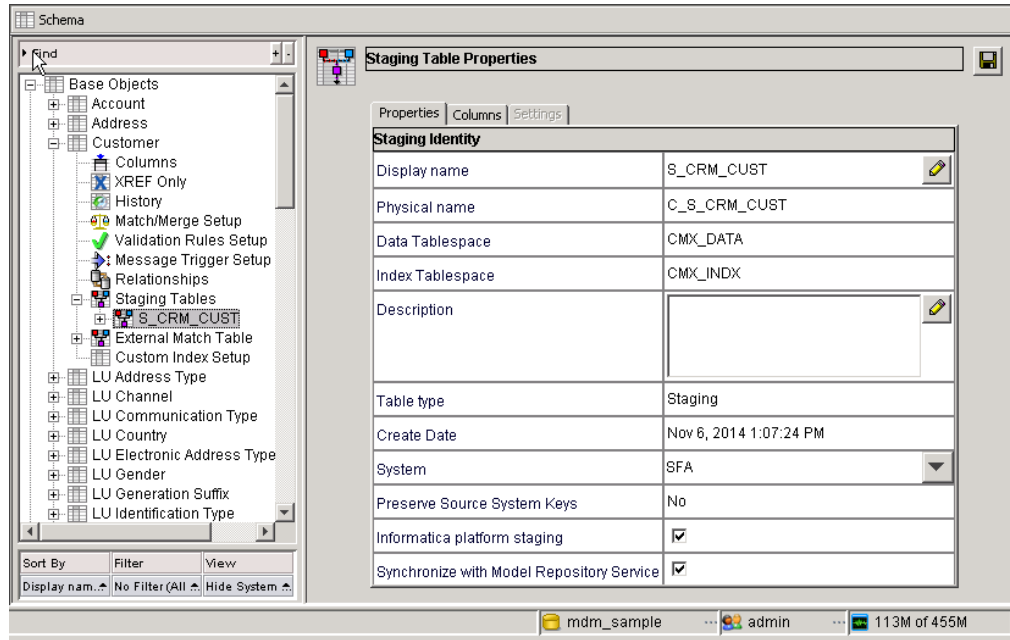
Hub 콘솔을 사용하여 MDM Hub 메타데이터를 모델 리포지토리와 동기화합니다.

- 스키마 도구를 시작합니다.
- 쓰기 잠금을 획득합니다.
- 준비에 사용해야 하는 기본 개체의 준비 테이블을 클릭합니다.
준비 테이블 속성 페이지가 나타납니다.

4. 속성 탭에서 **모델 리포지토리 서비스와 동기화**를 활성화하고 **저장**을 클릭합니다.

Hub 콘솔을 통해 준비 테이블에 변경한 사항이 모델 리포지토리에 표시됩니다. 동기화는 모델 리포지토리에서 실제 및 논리적 데이터 개체와 맵셋을 생성합니다.

다음 이미지는 모델 리포지토리 동기화를 활성화할 수 있는 S_CRM_CUST 준비 테이블에 대한 **준비 테이블 속성** 페이지를 보여 줍니다.



개발자 도구에서 준비 설정 완료

모델 리포지토리를 Hub 저장소와 동기화한 후 개발자 도구에서 개체 구성을 완료합니다. 데이터를 MDM Hub 준비 테이블로 옮기고자 하는 소스 시스템에 대해 연결을 생성해야 합니다.

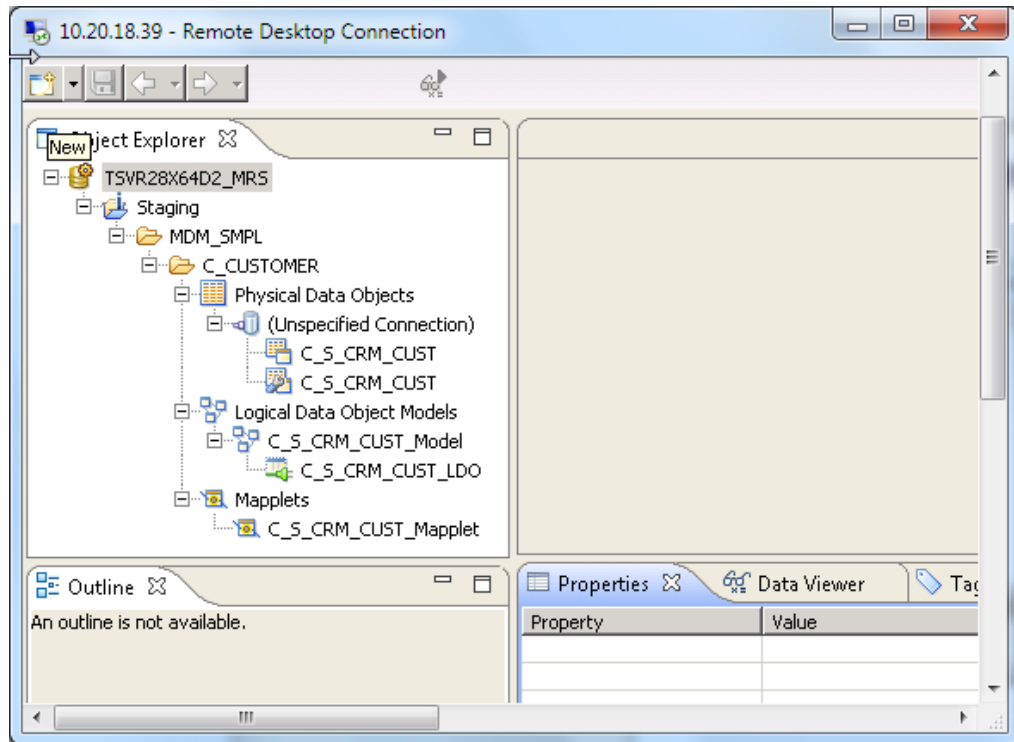
정리 작업을 수행하려는 경우 동기화 프로세스에서 생성하는 맵셋에 변환을 추가합니다. 데이터 통합 서비스가 준비 프로세스 동안 거부된 레코드를 관리합니다.

1단계. 생성된 개체 검토

개발자 도구를 사용하여 동기화 프로세스에서 생성하는 데이터 개체를 검토합니다.

1. 개발자 도구를 시작합니다.
2. 개체 탐색기 보기에서 모델 리포지토리에 연결되어 있는지 확인합니다.
3. 모델 리포지토리 서비스를 마우스 오른쪽 단추로 클릭하고 **새로 고침**을 클릭합니다.
모델 리포지토리 개체가 개체 탐색기 보기에 표시됩니다.
4. 준비에 대해 생성한 프로젝트를 확장합니다.
준비에 대한 실제 및 논리적 데이터 개체와 맵셋을 볼 수 있습니다.
5. 연산 참조 저장소 이름 및 기본 개체 이름으로 된 폴더에서 실제 데이터 개체, 논리적 데이터 개체 모델 및 맵셋을 확장합니다.

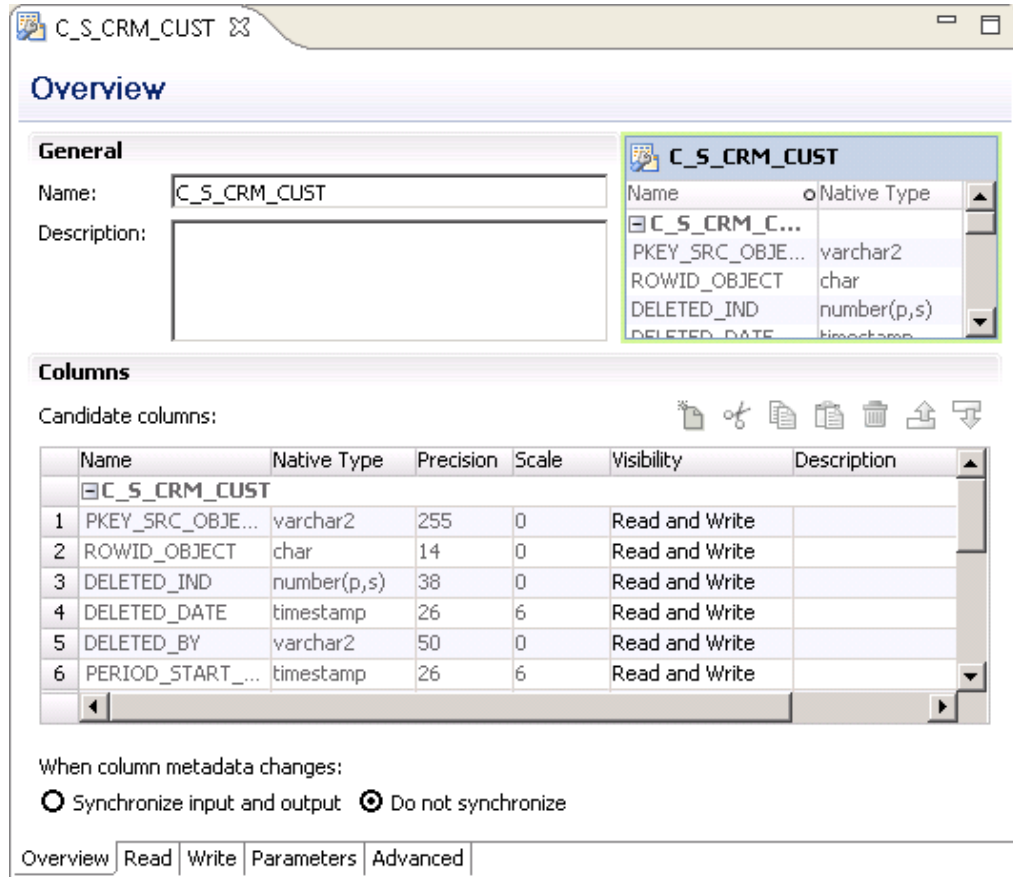
다음 이미지는 개체 탐색기 보기의 모델 리포지토리 개체를 보여 줍니다.



개체 탐색기 보기에서, 준비 프로젝트에는 연산 참조 저장소 이름으로 **MDM_SMPL** 폴더가 포함되어 있습니다. **MDM_SMPL** 폴더에는 기본 개체의 이름에 해당하는 또 다른 폴더인 **C_CUSTOMER**가 포함되어 있습니다. **C_CUSTOMER** 폴더에는 실제 데이터 개체, 논리적 데이터 개체 및 맵렛이 포함되어 있습니다.

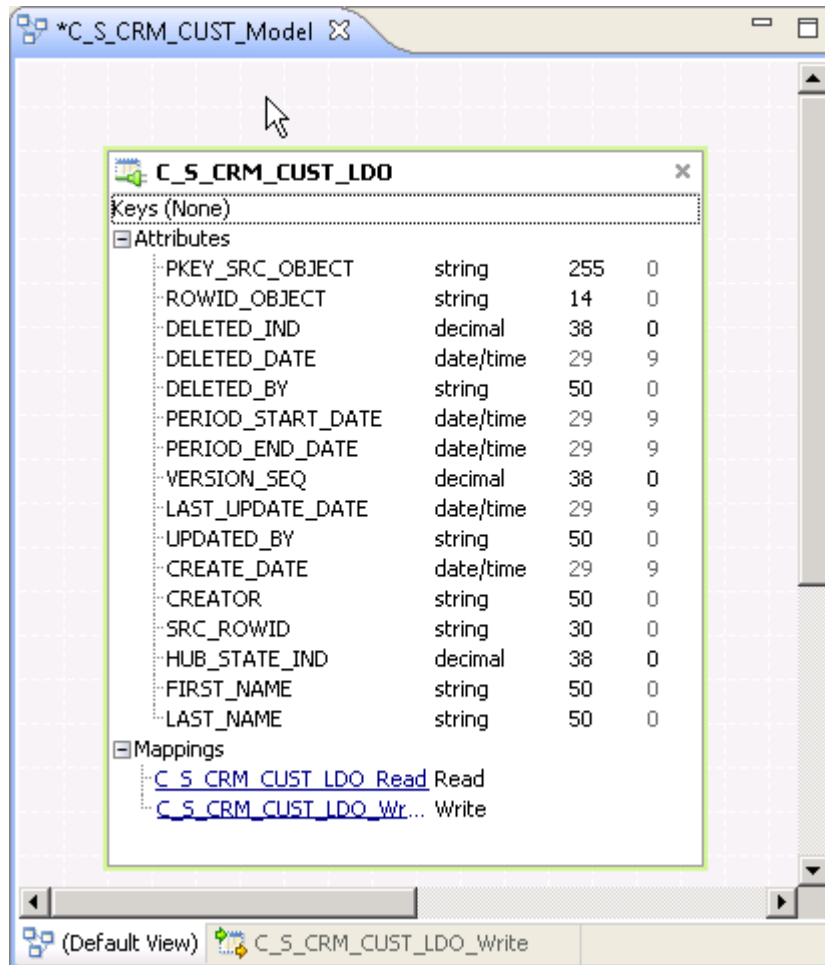
6. 실제 데이터 개체를 마우스 오른쪽 단추로 클릭하고 **열기**를 클릭합니다.
편집기에서 실제 데이터 개체가 열립니다.

다음 이미지는 편집기에서 열려 있는 C_S_CRM_CUST 사용자 지정된 데이터 개체를 보여 줍니다.



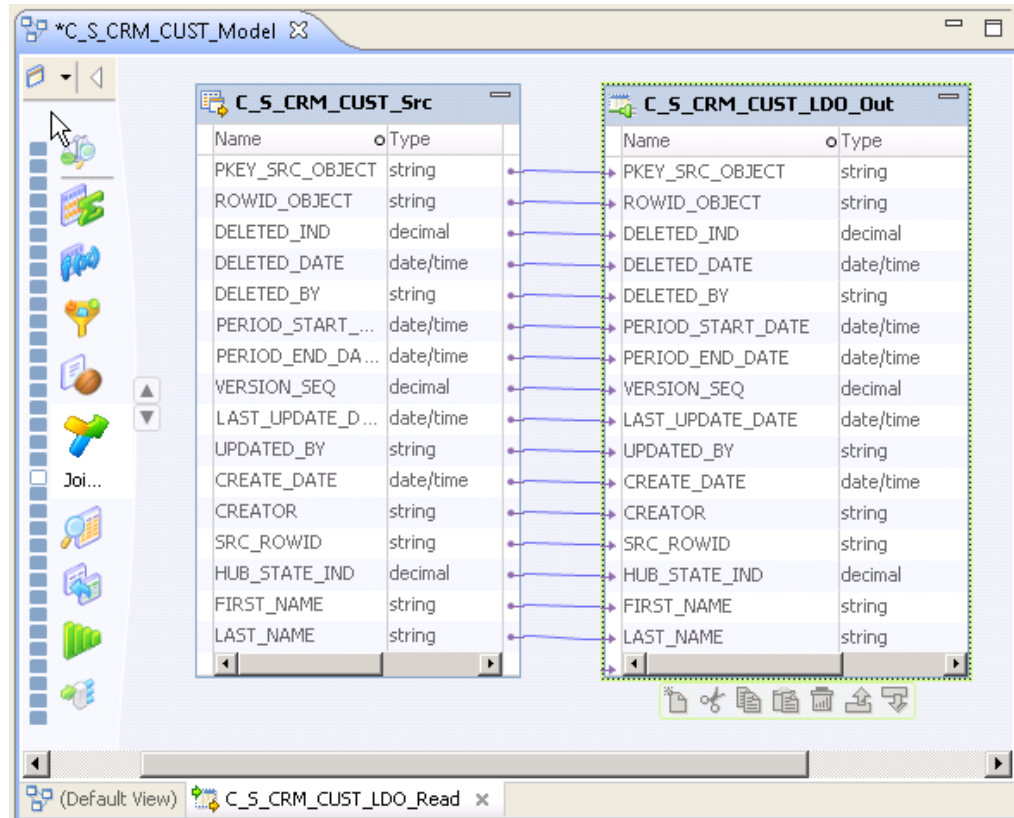
- 논리적 데이터 개체를 마우스 오른쪽 단추로 클릭하고 **열기**를 클릭한 후 **특성** 및 **매핑**을 확장합니다.

다음 이미지는 C_S_CRM_CUST_LDO 논리적 데이터 개체, C_S_CRM_CUST_LDO_Read 논리적 데이터 개체 읽기 매핑 및 C_S_CRM_CUST_LDO_Write 논리적 데이터 개체 쓰기 매핑 링크의 특성을 보여 줍니다.



8. 논리적 데이터 개체 읽기 매핑 링크를 클릭합니다.
논리적 데이터 개체 읽기 매핑이 열립니다.

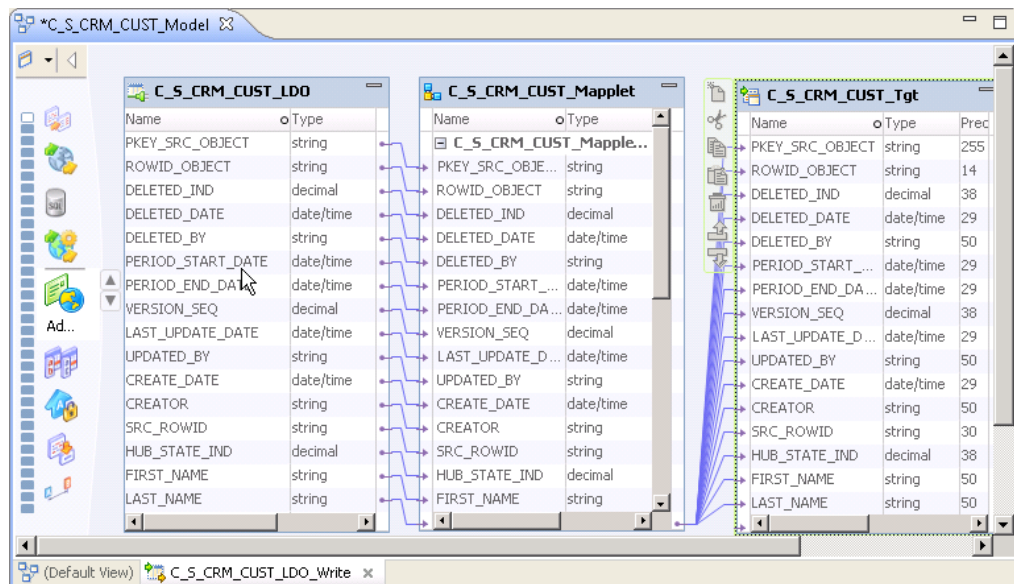
다음 이미지는 C_S_CRM_CUST_LDO_Read 논리적 데이터 개체 읽기 매핑을 보여 줍니다.



9. 논리적 데이터 개체 쓰기 매핑 링크를 클릭합니다.

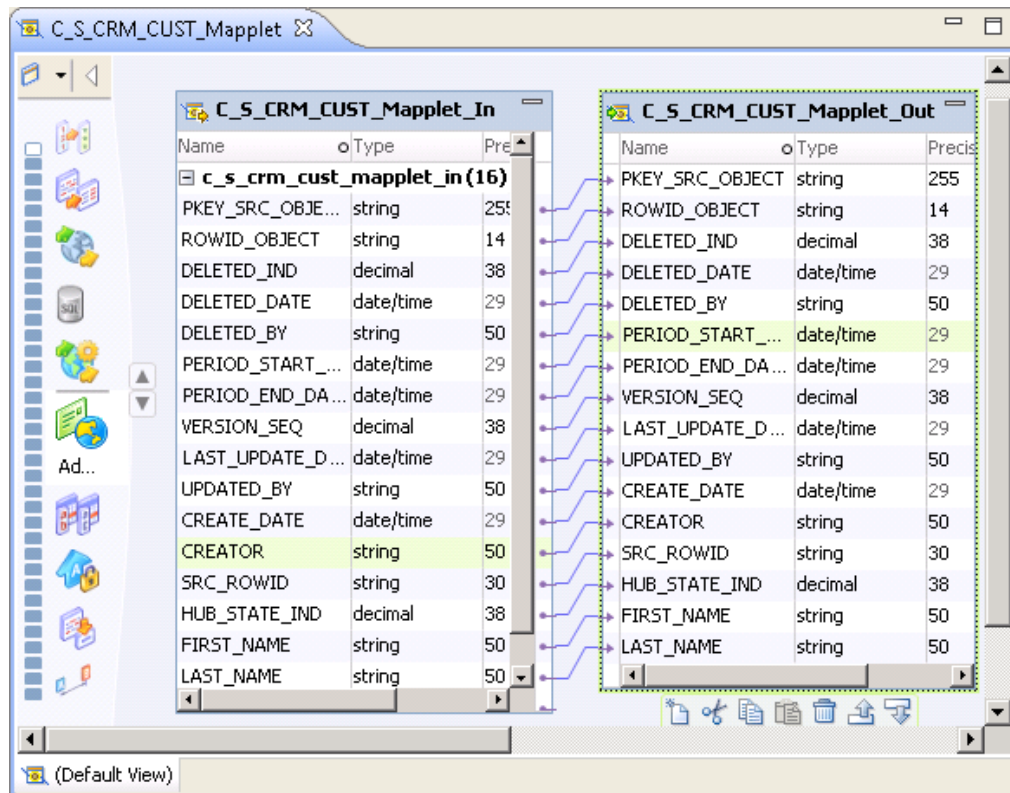
논리적 데이터 개체 쓰기 매핑이 열립니다.

다음 이미지는 C_S_CRM_CUST_LDO_Write 논리적 데이터 개체 쓰기 매핑을 보여 줍니다.



편집기에서 열려 있는 C_S_CRM_CUST_LDO_Write 논리적 데이터 개체입니다.

10. 마우스 오른쪽 단추로 맵렛을 클릭합니다.
다음 이미지는 C_S_CRM_CUST_Mapplet을 보여 줍니다.



C_S_CRM_CUST_Mapplet 맵렛에는 C_S_CRM_CUST_Mapplet_In 입력 변환 및 C_S_CRM_CUST_Mapplet_Out 출력 변환이 포함되어 있습니다.

참고: 숫자 데이터 필드의 전체 자릿수 및 소수 자릿수 설정은 Hub 콘솔에서 볼 수 있는 설정과 동기화되지 않습니다. 필요한 경우 Hub 콘솔의 설정과 일치하도록 맵렛의 전체 자릿수 및 소수 자릿수를 설정합니다.

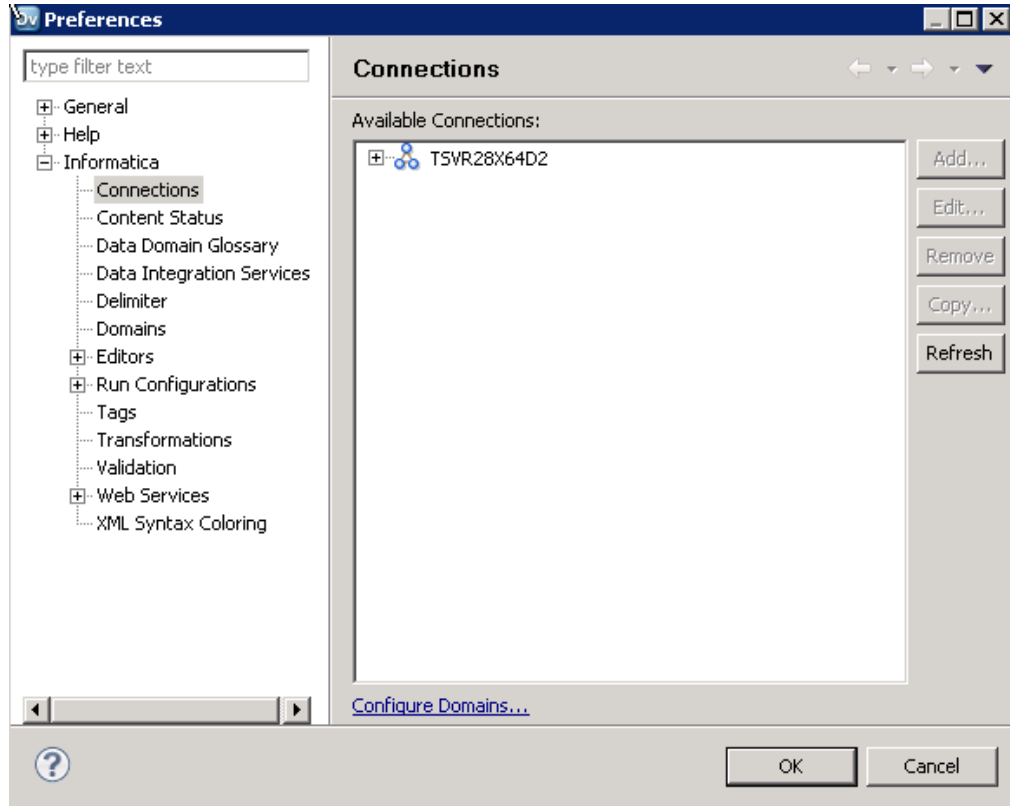
2단계. 소스 시스템 연결 생성

소스 시스템에 대한 연결을 지정하려면 개발자 도구를 사용합니다. 연결을 생성하여 소스 시스템에서 데이터를 가져옵니다.

1. 창 > 기본 설정을 클릭합니다.

기본 설정 대화 상자가 표시됩니다.

다음 이미지는 기본 설정 대화 상자를 보여 줍니다.

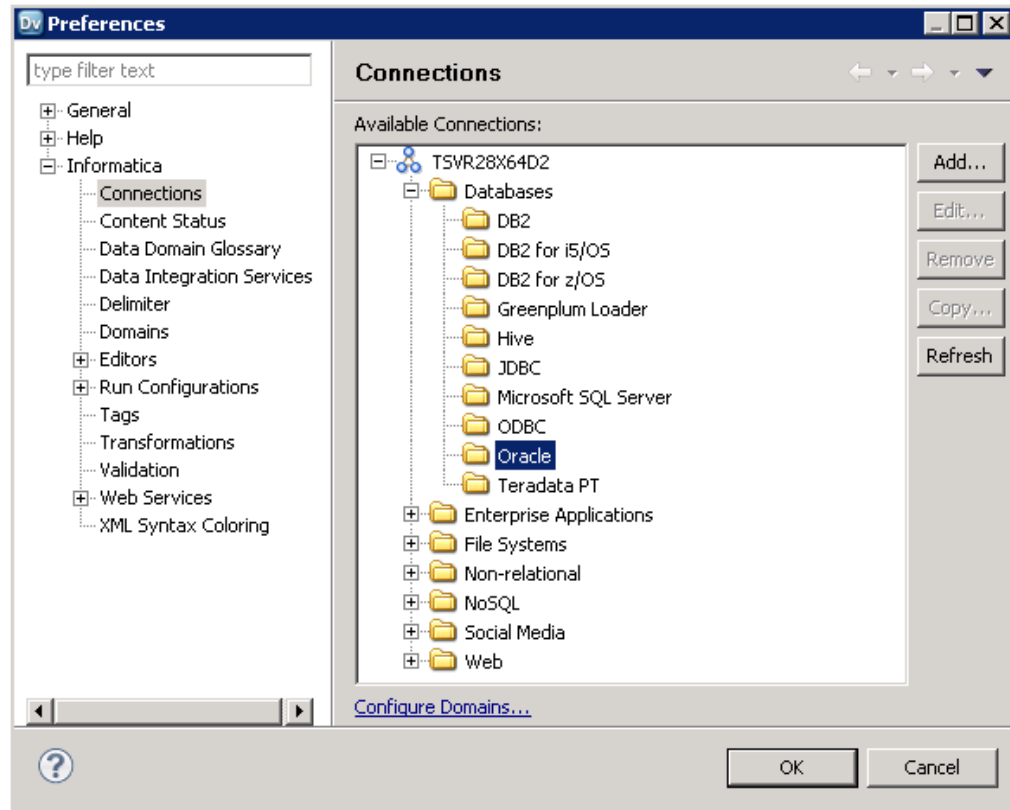


2. Informatica > 연결을 선택합니다.

연결 창이 표시됩니다.

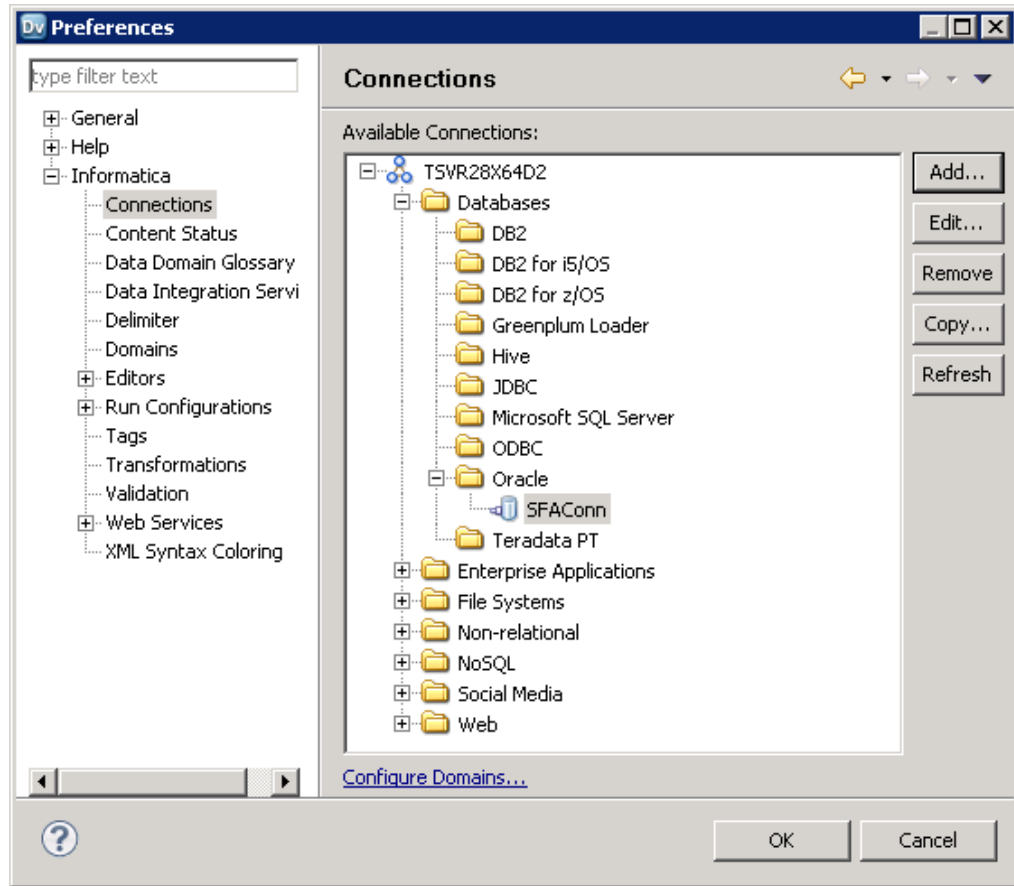
3. 사용 가능한 연결 목록에서 데이터베이스를 확장합니다.

다음 이미지는 **기본 설정** 대화 상자로, **연결** 창의 **사용 가능한 연결** 목록에서 확장된 데이터베이스를 보여 줍니다.



4. **사용 가능한 연결** 목록에서 연결 유형을 선택하고 **추가**를 클릭합니다.
새 데이터베이스 연결 대화 상자가 표시되며 **유형** 필드에 데이터베이스 연결 유형 값이 채워져 있습니다.
5. **이름** 필드에서 데이터베이스 연결 이름을 입력합니다.
6. **찾아보기**를 클릭합니다.
도메인 선택 대화 상자가 표시됩니다.
7. 연결을 저장할 도메인을 선택하고 **확인**을 클릭합니다.
8. **다음**을 클릭합니다.
새 데이터베이스 연결 대화 상자의 연결 세부 정보 페이지가 표시됩니다.
9. 데이터베이스에 대한 연결 세부 정보를 지정하고 **연결 테스트**를 클릭합니다.
연결 테스트 대화 상자가 표시됩니다.
10. 연결이 성공한 경우 **확인**을 클릭한 다음 **마침**을 클릭합니다.
연결 창에 연결이 표시됩니다.

다음 이미지는 기본 설정 대화 상자의 연결 창에서 SFAConn 연결을 보여 줍니다.



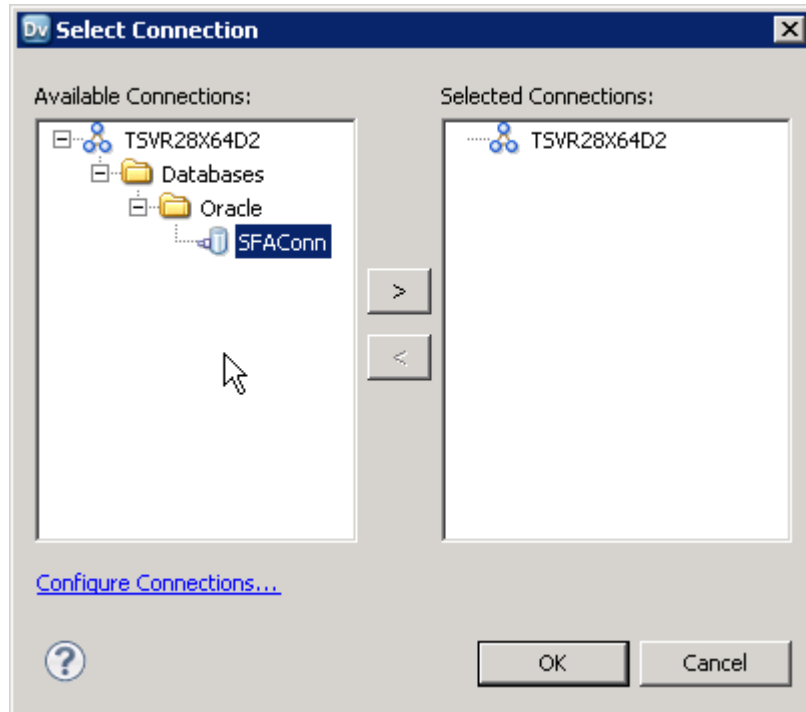
11. **확인**을 클릭합니다.
기본 설정 대화 상자가 닫힙니다.

3단계. 연결 탐색기 보기에 연결 추가

데이터 소스 연결을 생성한 후 **연결 탐색기** 보기에 연결을 추가합니다.

1. **연결 탐색기** 보기를 열려면 **창 > 보기 표시 > 연결 탐색기**를 클릭합니다.
2. **연결 선택** 단추를 클릭합니다.
연결 선택 대화 상자가 표시됩니다.
3. **사용 가능한 연결** 섹션에서 연결을 선택합니다.

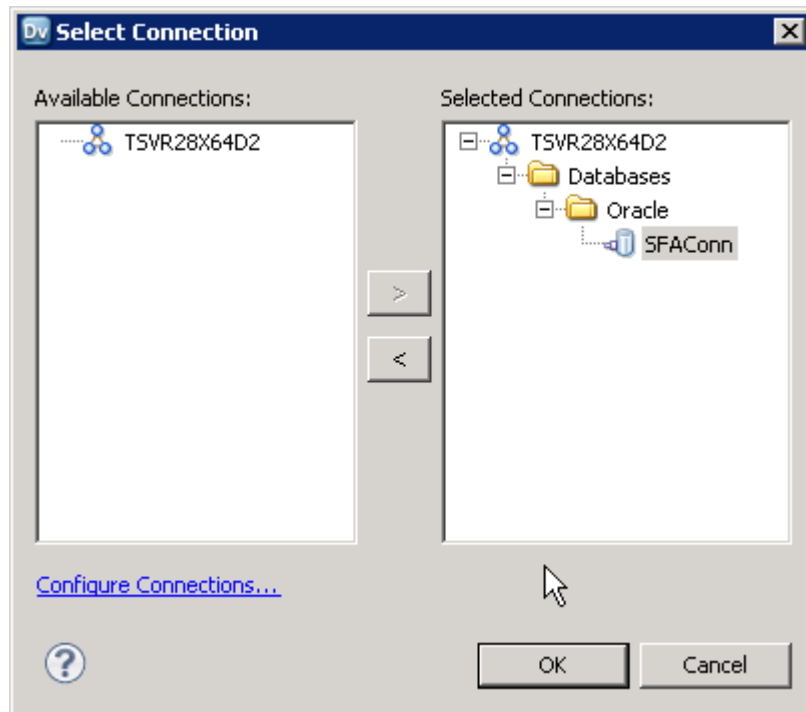
다음 이미지는 **연결 선택** 대화 상자의 **사용 가능한 연결** 섹션에서 선택한 SFACConn 연결을 보여 줍니다.



4. 오른쪽 화살표를 클릭합니다.

선택한 연결이 **연결 선택** 대화 상자의 **선택한 연결** 섹션으로 이동합니다.

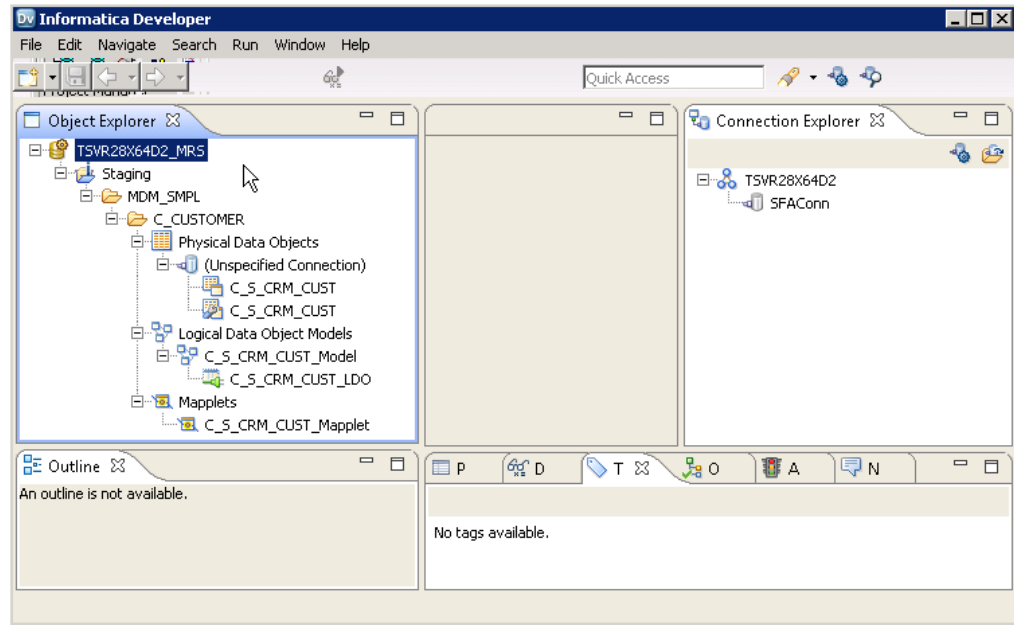
다음 이미지는 **선택한 연결** 대화 상자의 **선택한 연결** 섹션에 있는 SFACConn 연결을 보여 줍니다.



5. **확인**을 클릭합니다.

연결 탐색기 보기에 연결이 표시됩니다.

다음 이미지는 **연결 탐색기** 보기에서의 SFACnn 연결을 보여 줍니다.



4단계. 소스 연결에 대한 실제 데이터 개체 생성

연결을 생성하고 연결 탐색기 보기에 추가한 후 실제 데이터 개체에 연결을 추가합니다.

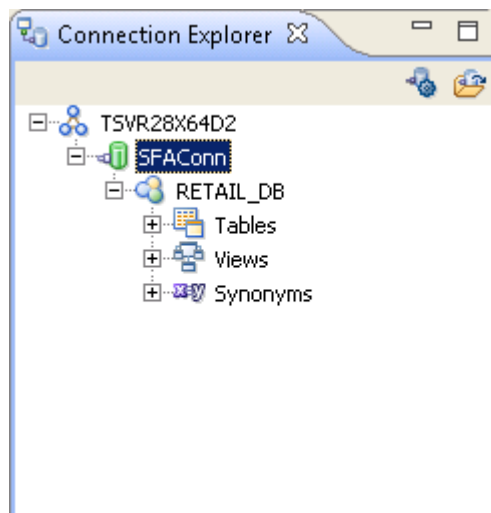
1. **연결 탐색기** 보기에서 마우스 오른쪽 단추로 데이터베이스 연결을 클릭합니다.

연결 메뉴가 표시됩니다.

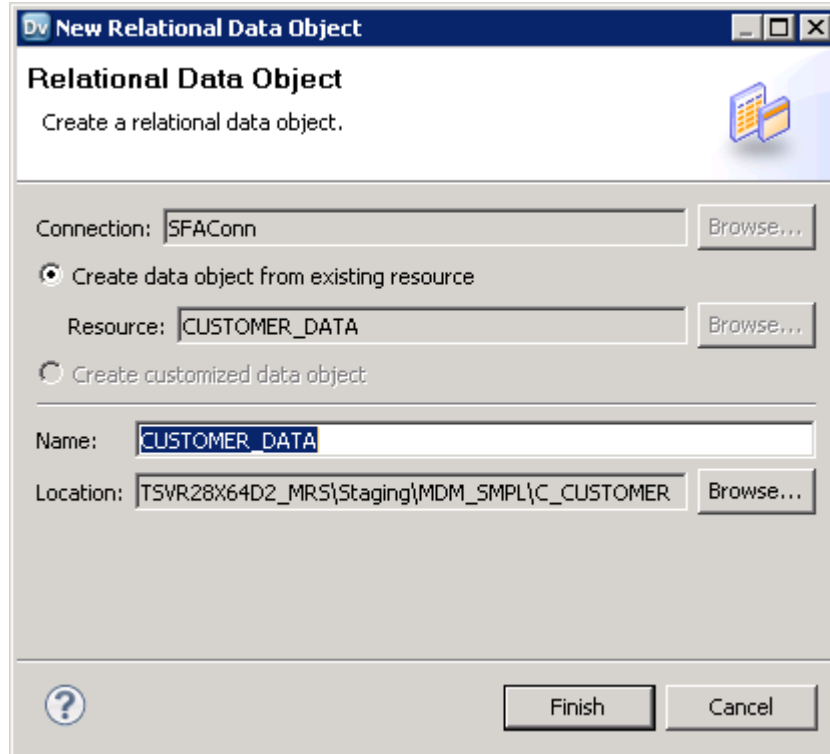
2. **연결**을 클릭합니다.

연결 탐색기 보기에서 연결이 활성화됩니다.

다음 이미지는 SFACnn 활성 Oracle 데이터베이스 연결이 있는 **연결 탐색기** 보기를 보여 줍니다.



3. 데이터베이스를 확장하여 테이블을 보고 연결할 테이블을 마우스 오른쪽 단추로 클릭합니다.
연결 메뉴가 표시됩니다.
4. 프로젝트에 추가를 클릭합니다.
프로젝트에 추가 대화 상자가 나타납니다.
5. 각 리소스에 대한 데이터 개체를 생성하십시오 옵션을 선택합니다.
새 관계형 데이터 개체 대화 상자가 나타납니다.
다음 이미지는 새 관계형 데이터 개체 대화 상자를 보여 줍니다.

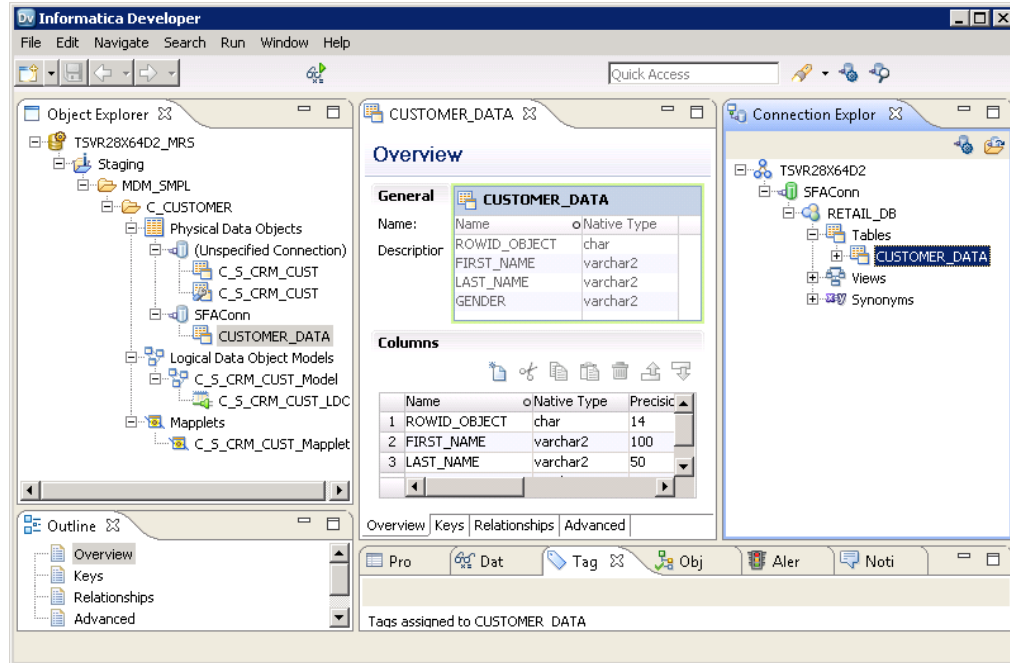


6. 기존 리소스에서 데이터 개체 생성 옵션을 선택합니다.

7. 이름 필드에 소스 데이터 개체에 대한 이름을 입력하고 **마침**을 클릭합니다.

데이터 개체가 개체 탐색기 보기에서 연결과 함께 표시되고 편집기에서 열립니다.

다음 이미지는 개체 탐색기 보기에서 SFAConn 연결로 CUSTOMER_DATA 개체가 있는 개발자 도구가 편집기에서 열려 있는 모습을 보여 줍니다.



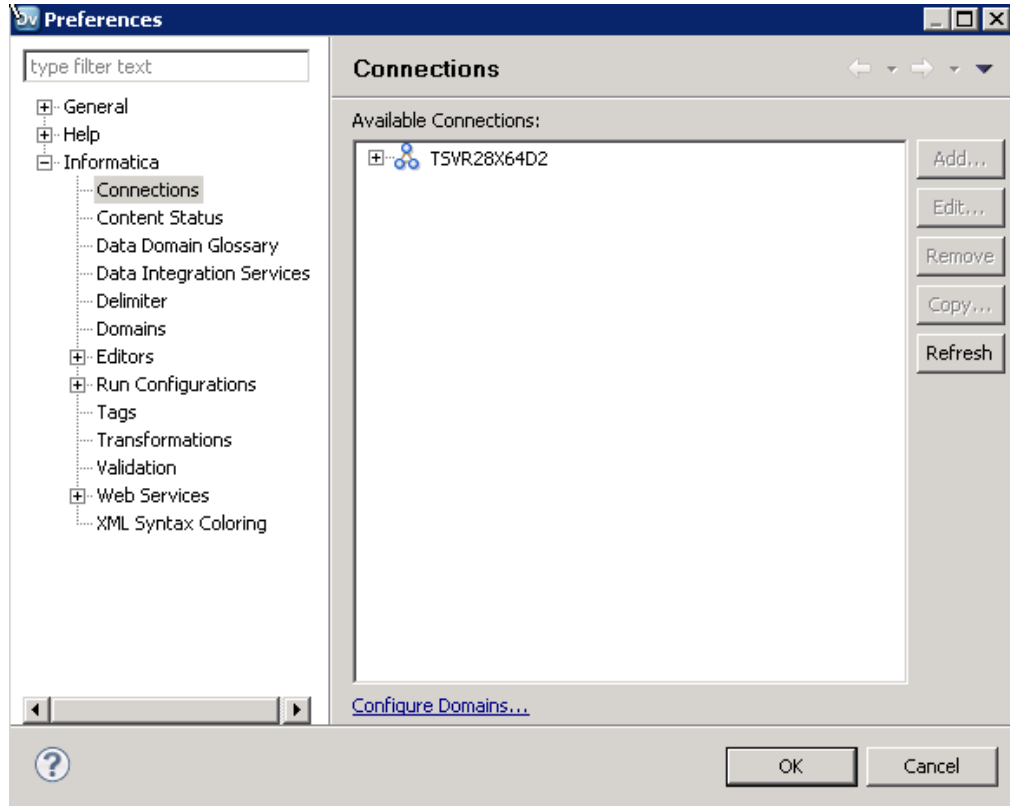
5단계. 대상에 대한 연결 생성

대상 준비 테이블에 대한 연결을 지정하려면 개발자 도구를 사용합니다. 연결을 생성하여 데이터 출력을 준비 테이블에 전송합니다.

1. 창 > 기본 설정을 클릭합니다.

기본 설정 대화 상자가 표시됩니다.

다음 이미지는 기본 설정 대화 상자를 보여 줍니다.

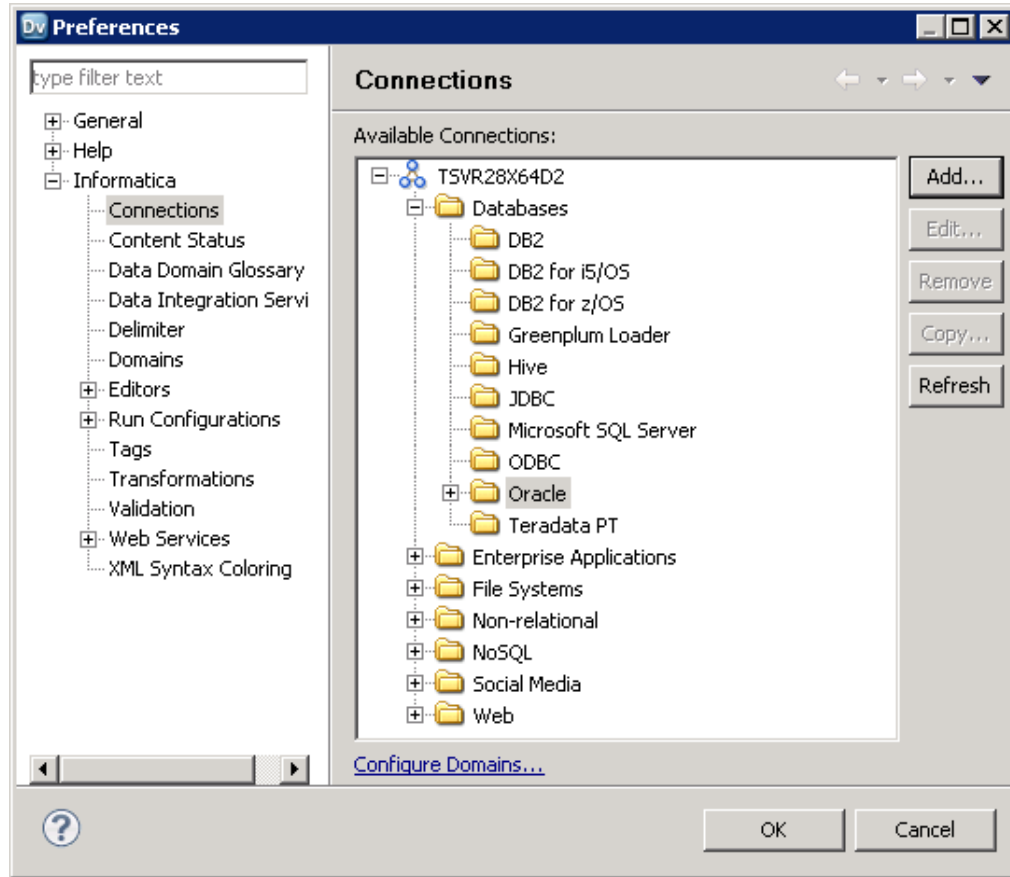


2. Informatica > 연결을 선택합니다.

연결 창이 표시됩니다.

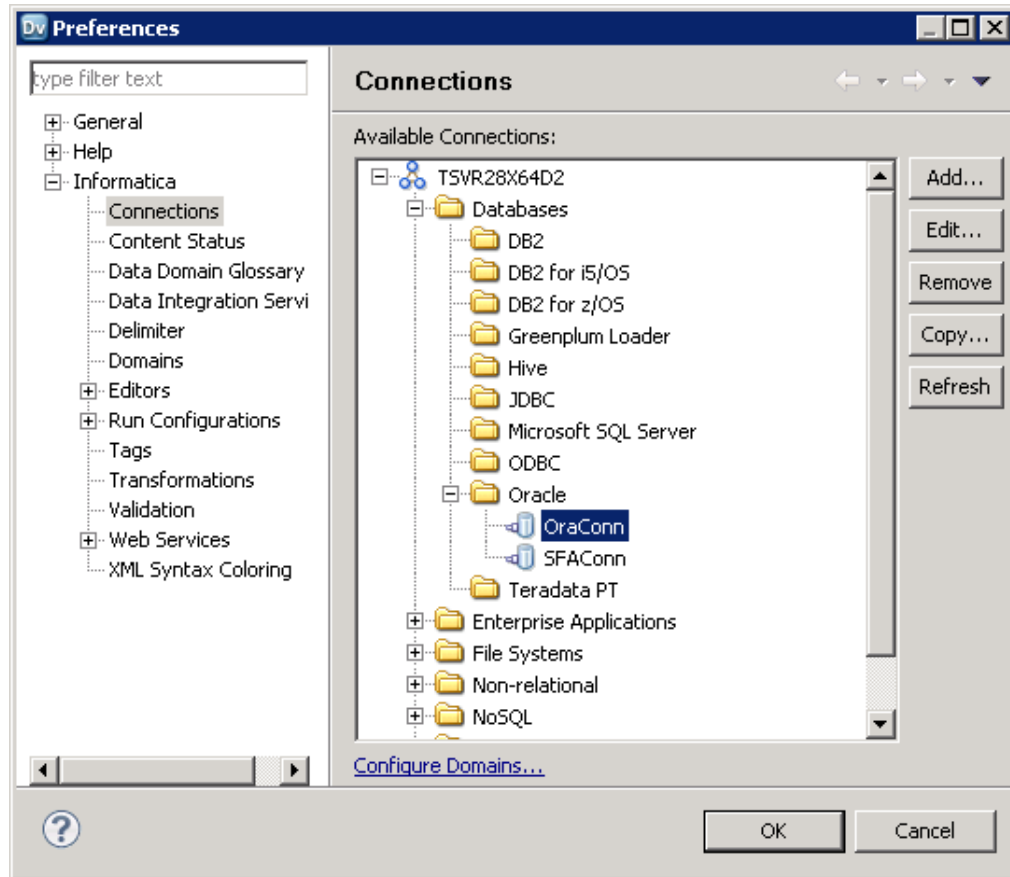
3. 사용 가능한 연결 목록에서 데이터베이스를 확장합니다.

다음 이미지는 기본 설정 대화 상자로, 연결 창의 사용 가능한 연결 목록에서 확장된 데이터베이스를 보여줍니다.



4. 사용 가능한 연결 목록에서 연결 유형을 선택하고 **추가**를 클릭합니다.
새 데이터베이스 연결 대화 상자가 표시되며 **유형** 필드에 데이터베이스 연결 유형 값이 채워져 있습니다.
5. **이름** 필드에서 데이터베이스 연결 이름을 입력합니다.
6. **찾아보기**를 클릭합니다.
도메인 선택 대화 상자가 표시됩니다.
7. 연결을 저장할 도메인을 선택하고 **확인**을 클릭합니다.
8. **다음**을 클릭합니다.
새 데이터베이스 연결 대화 상자의 연결 세부 정보 페이지가 표시됩니다.
9. 데이터베이스에 대한 연결 세부 정보를 지정하고 **연결 테스트**를 클릭합니다.
연결 테스트 대화 상자가 표시됩니다.
10. 연결이 성공한 경우 **확인**을 클릭한 다음 **마침**을 클릭합니다.
연결 창에 연결이 표시됩니다.

다음 이미지는 기본 설정 대화 상자의 연결 창에 있는 OraConn 연결을 보여 줍니다.



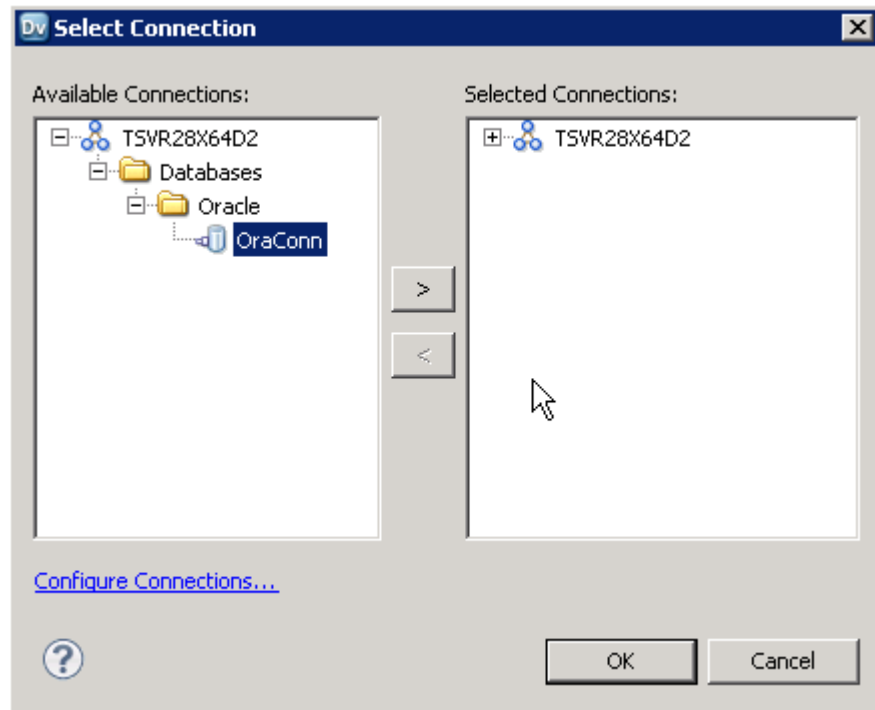
11. 확인을 클릭합니다.

6단계. 연결 탐색기 보기에 연결 추가

대상 준비 테이블에 대한 연결을 생성한 후 연결 탐색기 보기에 연결을 추가합니다.

1. 연결 탐색기 보기를 열려면 창 > 보기 표시 > 연결 탐색기를 클릭합니다.
2. 연결 선택 단추를 클릭합니다.
연결 선택 대화 상자가 표시됩니다.
3. 사용 가능한 연결 섹션에서 연결을 선택합니다.

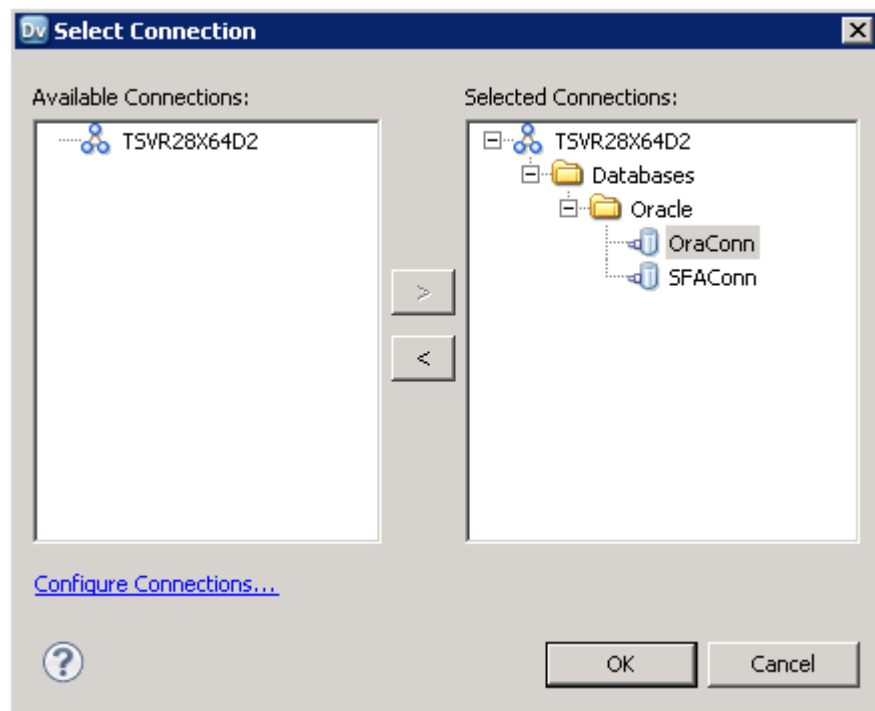
다음 이미지는 **연결 선택** 대화 상자의 **사용 가능한 연결** 섹션에서 선택한 OraConn 연결을 보여 줍니다.



4. 오른쪽 화살표를 클릭합니다.

선택한 연결이 **연결 선택** 대화 상자의 **선택한 연결** 섹션으로 이동합니다.

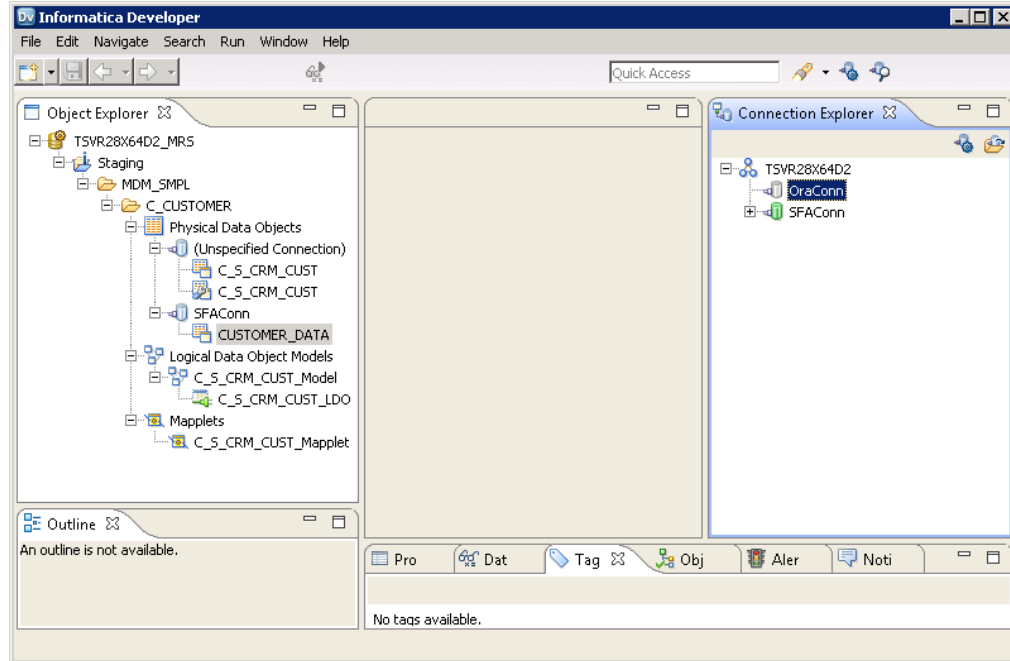
다음 이미지는 **연결 선택** 대화 상자의 **선택한 연결** 섹션에 있는 OraConn 연결을 보여 줍니다.



5. **확인**을 클릭합니다.

연결 탐색기 보기에 연결이 표시됩니다.

다음 이미지는 **연결 탐색기** 보기의 **OraConn** 연결을 보여 줍니다.

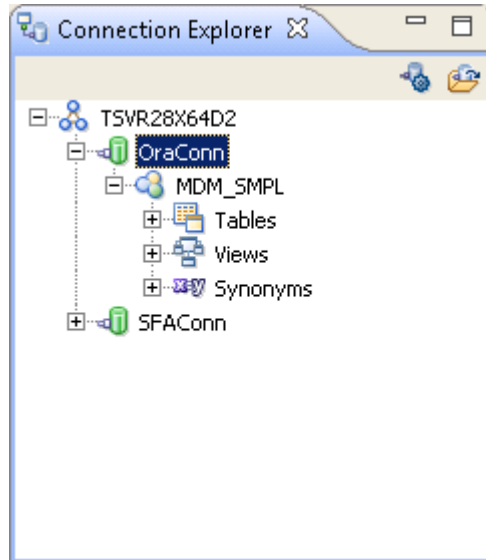


7단계. 실제 데이터 개체에 연결 추가

연결을 생성하여 **연결 탐색기** 보기에 연결을 추가한 후 동기화 프로세스 동안 생성된 실제 데이터 개체에 연결을 추가합니다.

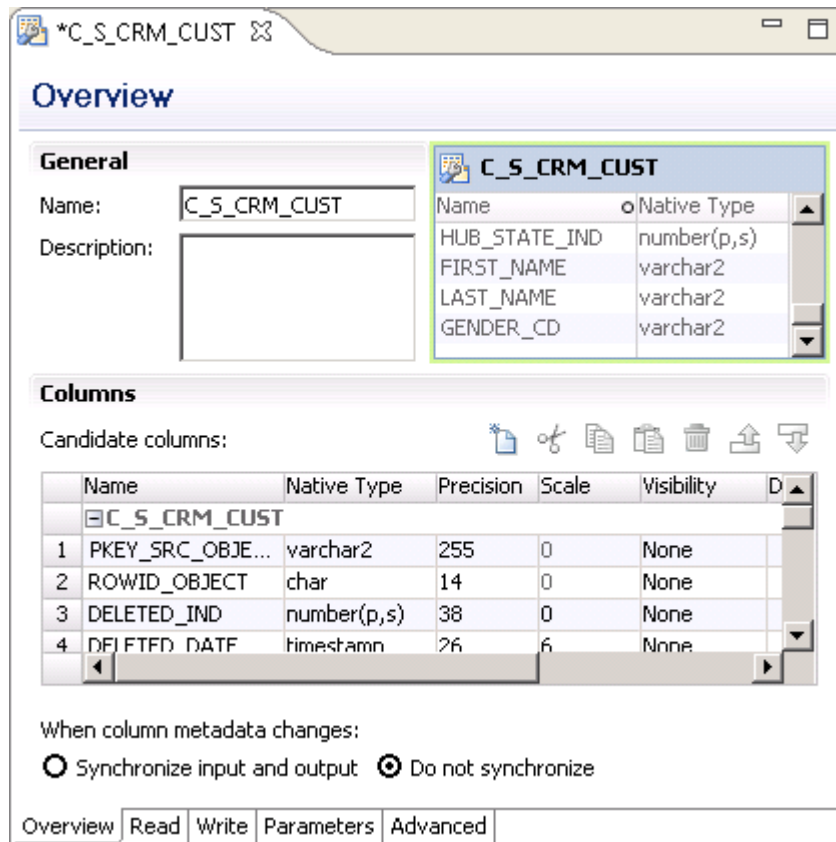
1. **연결 탐색기** 보기에서 마우스 오른쪽 단추로 데이터베이스 연결을 클릭합니다.
연결 메뉴가 표시됩니다.
2. **연결**을 클릭합니다.
연결 탐색기 보기에서 연결이 활성화됩니다.

다음 이미지는 활성 OraConn 데이터베이스 연결이 있는 **연결 탐색기** 보기를 보여 줍니다.



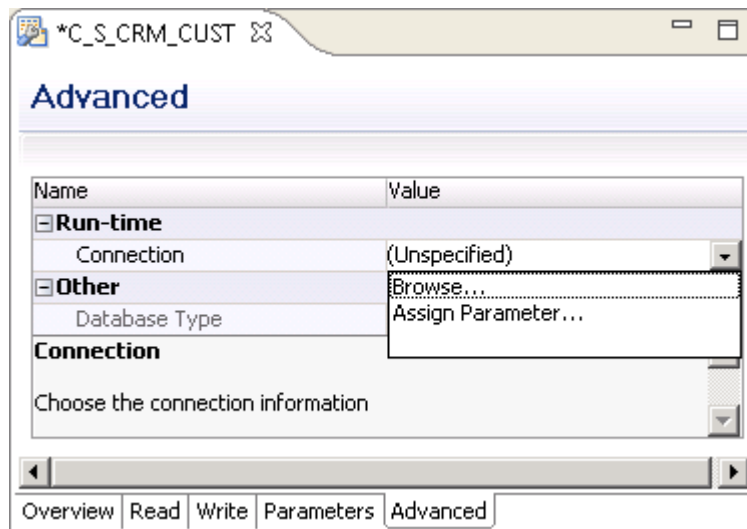
- 데이터 개체를 마우스 오른쪽 단추로 클릭한 다음 **열기**를 클릭합니다.
편집기에서 데이터 개체가 열립니다.

다음 이미지는 편집기에서 열려 있는 C_S_CRM_CUST 데이터 개체를 보여 줍니다.

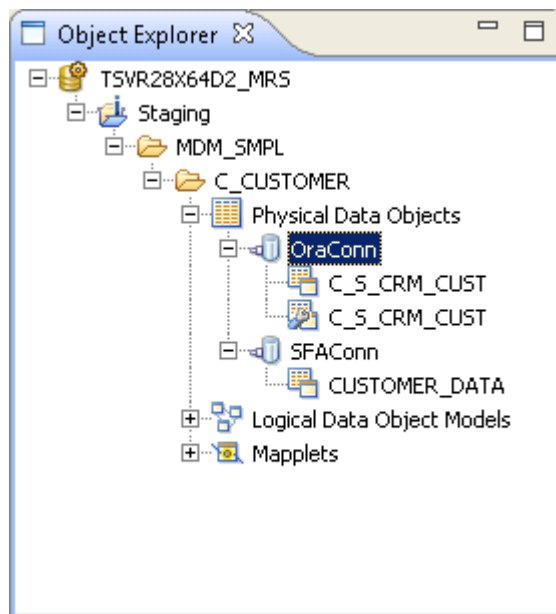


- 고급 탭을 클릭합니다.

데이터 개체의 고급 속성 페이지가 열립니다.
 다음 이미지는 데이터 개체의 고급 속성 페이지를 보여 줍니다.



5. 준비 테이블에 대한 연결을 찾아 선택하고 **저장** 단추를 클릭합니다.
 지정한 연결로 데이터 개체가 나타납니다.
 다음 이미지는 개체 탐색기 보기에서 OraConn 연결이 있는 C_S_CRM_CUST 사용자 지정된 데이터 개체 및 C_S_CRM_CUST 관계형 데이터 개체가 있는 개발자 도구를 보여 줍니다.



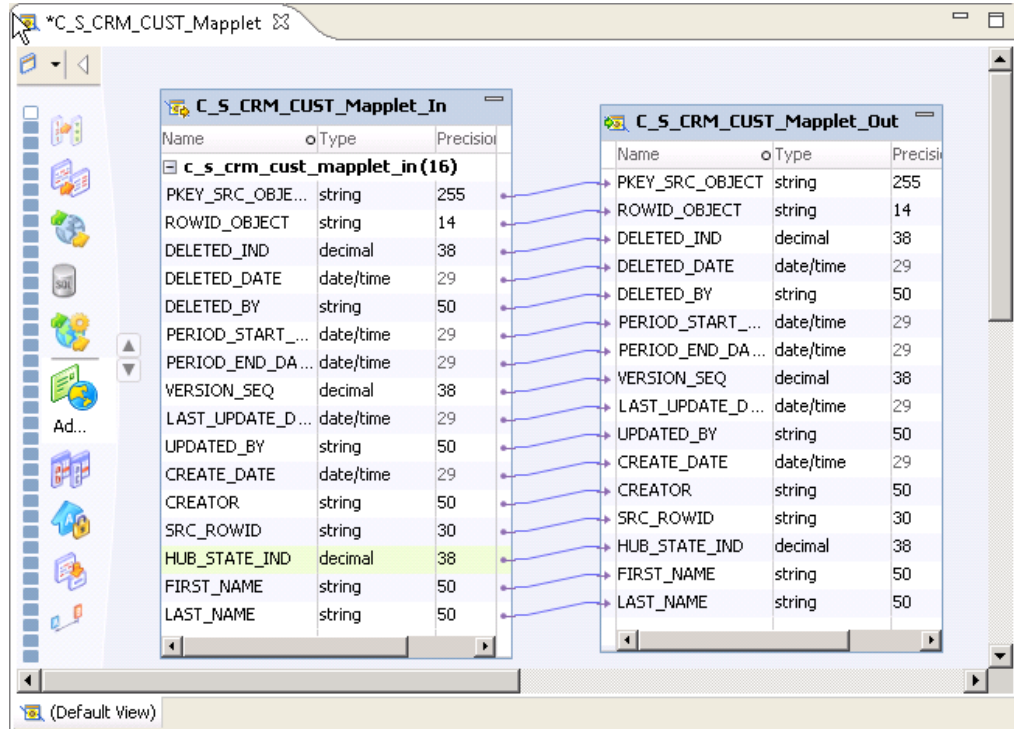
8단계. 맵릿에 변환 추가

데이터 정리 작업을 수행하기 위해 변환을 맵릿에 추가할 수 있습니다.

1. 개체 탐색기 보기에서 마우스 오른쪽 단추로 맵릿을 클릭하고 **열기**를 클릭합니다.

맵릿 편집기에서 맵릿이 열립니다.

다음 이미지는 C_S_CRM_CUST_Maplet 맵릿을 보여 줍니다.

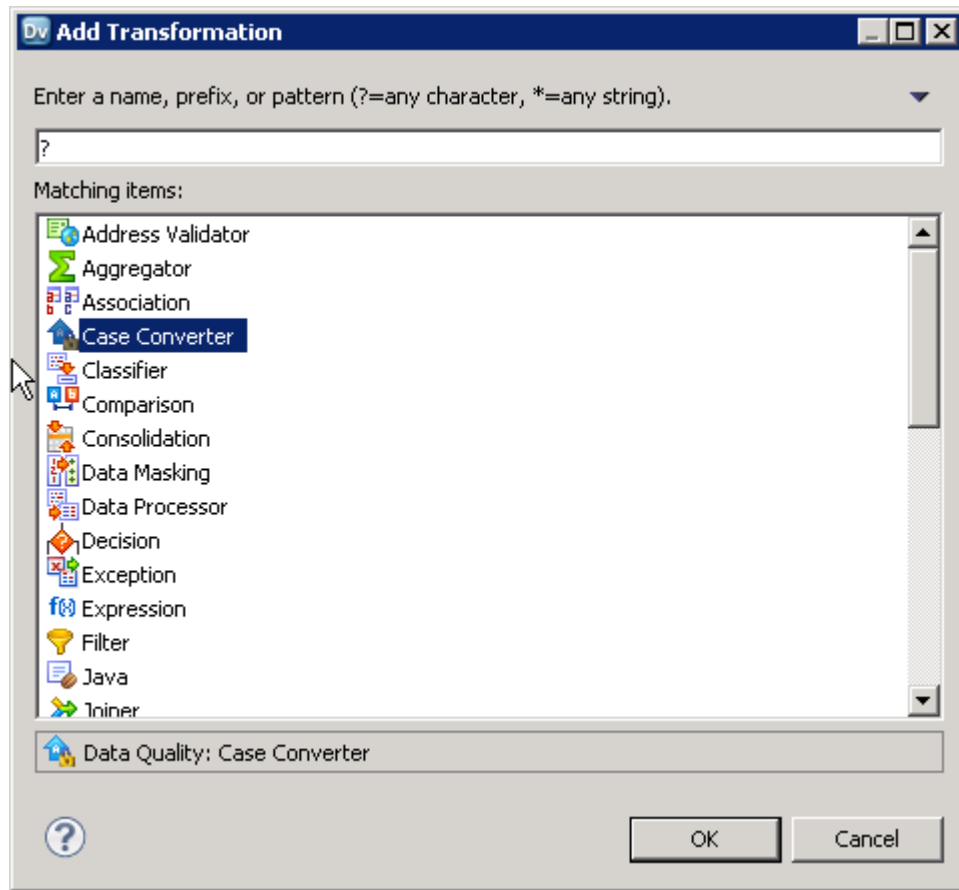


C_S_CRM_CUST_Maplet 맵릿에는 C_S_CRM_CUST_Maplet_In 입력 변환 및 C_S_CRM_CUST_Maplet_Out 출력 변환이 포함되어 있습니다.

2. 맵릿 편집기를 마우스 오른쪽 단추로 클릭하고 **변환 추가**를 클릭합니다.

변환 추가 대화 상자가 나타납니다.

다음 이미지는 **변환 추가** 대화 상자를 보여 줍니다.



3. 원하는 변환을 선택하고 **확인**을 클릭합니다.
빈 변환이 맵렛 편집기에 표시됩니다.
4. 편집기에서 변환을 선택하고 변환을 구성합니다.

매핑 구성 및 실행

데이터를 변환하고 준비 테이블에 로드하려면 매핑을 구성해야 합니다. 준비에 사용하는 매핑은 입력으로 실제 데이터 개체를, 출력으로 논리적 데이터 개체를 포함하고 데이터를 변환하는 맵렛을 포함해야 합니다.

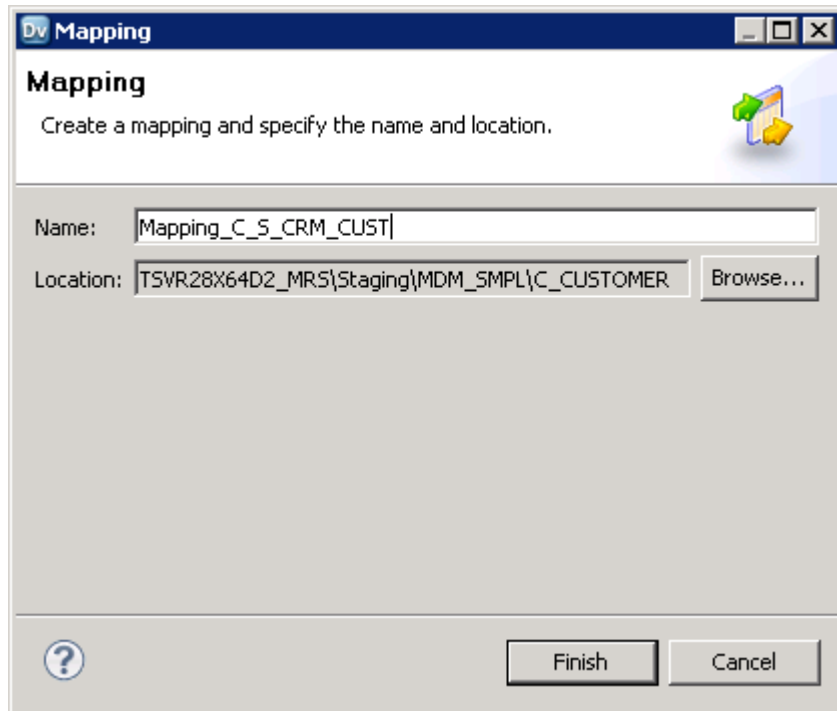
준비를 수행하려면 구성된 매핑을 실행해야 합니다. 데이터 통합 서비스가 매핑을 실행하고 출력을 대상에 씁니다.

1단계. 매핑 구성

소스, 대상 및 변환 개체로 매핑을 생성해야 합니다. 매핑 개체를 추가한 후에는 매핑 개체 간의 포트를 연결해야 합니다. 마지막으로 매핑의 유효성을 검사합니다.

1. 매핑을 생성하여 데이터를 변환하고 준비 테이블에 로드합니다.
 - a. **개체 탐색기** 보기에서 프로젝트 또는 폴더를 선택합니다.
 - b. **파일 > 새로 만들기 > 매핑**을 클릭합니다.

다음 이미지는 이름 및 위치 필드가 있는 매핑 대화 상자를 보여 줍니다.

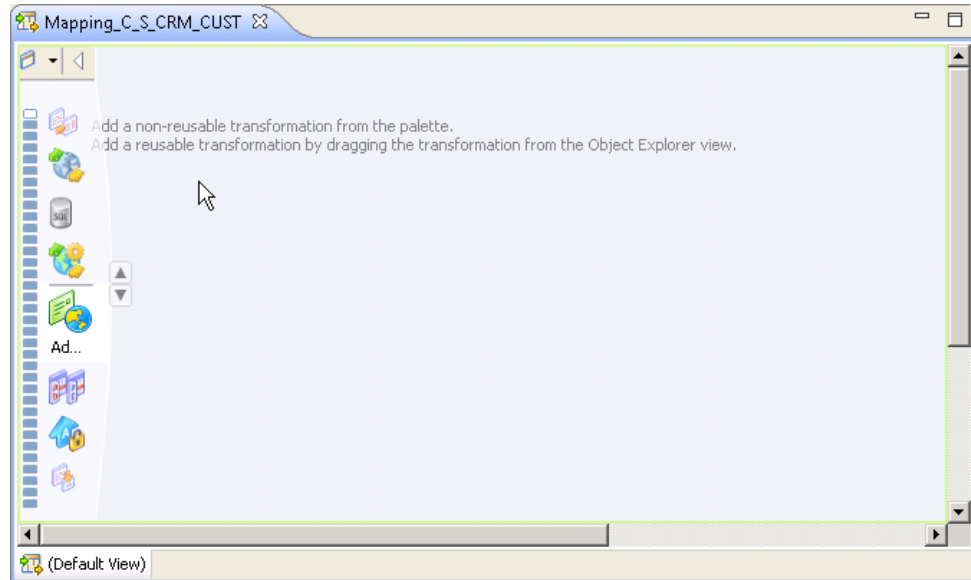


- c. 매핑 이름을 입력합니다.

d. 마침을 클릭합니다.

빈 매핑이 편집기에 표시됩니다.

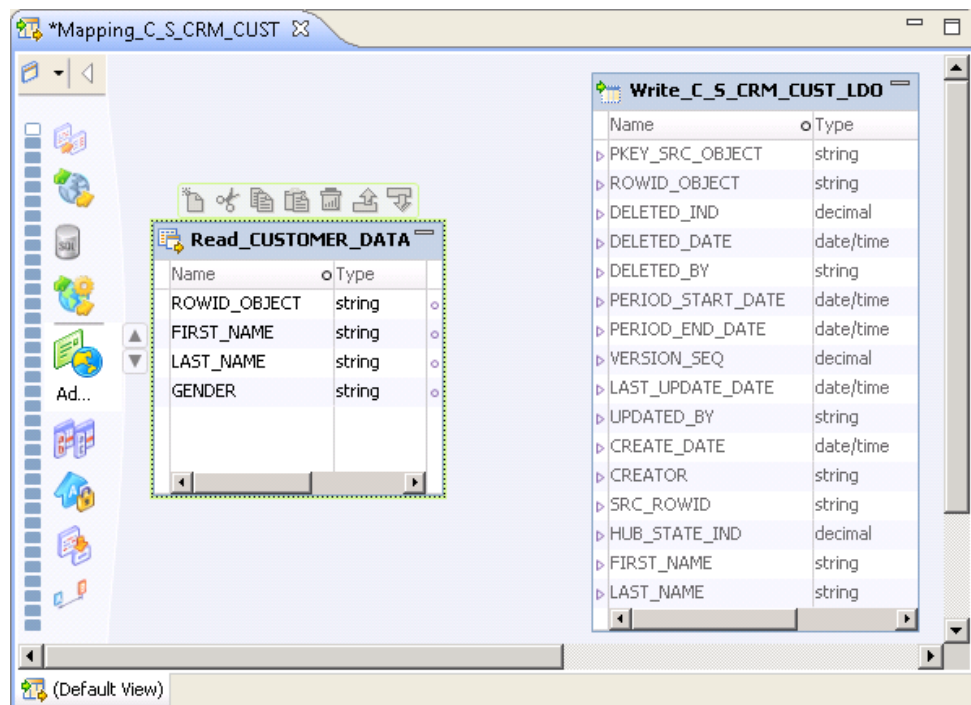
다음 이미지는 빈 매핑 Mapping_C_S_CRM_CUST를 보여 줍니다.



2. 개체를 매핑에 추가하여 소스와 대상 간의 데이터 흐름을 정합니다.

- 소스에 대해 생성한 실제 데이터 개체를 편집기로 끌고 읽기를 선택하여 데이터 개체를 소스로 추가합니다.
- 준비 테이블을 나타내는 논리적 데이터 개체를 편집기로 끌고 쓰기를 선택하여 데이터 개체를 대상으로 추가합니다.

다음 이미지는 실제 데이터 개체 및 논리적 데이터 개체가 있는 Mapping_C_S_CRM_CUST 매핑을 보여 줍니다.

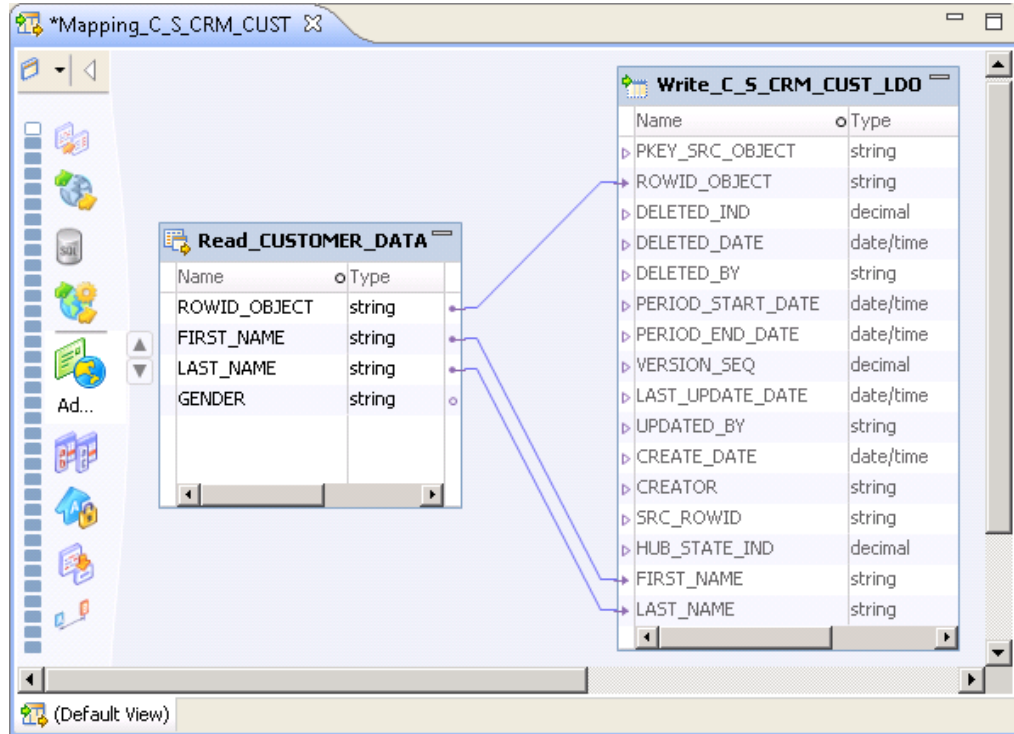


3. 매핑 개체 간의 포트를 연결합니다.

수동으로 또는 자동으로 포트를 연결할 수 있습니다.

참고: 준비 테이블 열을 편집, 추가 또는 삭제하고 모델 리포지토리와 Hub 저장소를 동기화한 후에는 해당하는 맵렛 포트의 연결이 해제됩니다. 그러므로 영향을 받은 포트를 수동으로 연결해야 합니다.

다음 이미지는 실제 데이터 개체 및 논리적 데이터 개체 간의 링크가 있는 Mapping_C_S_CRM_CUST 매핑을 보여 줍니다.



4. 매핑의 유효성을 검사하여 데이터 통합 서비스가 전체 매핑을 읽고 처리할 수 있는지 확인합니다.

a. 편집 > 유효성 검사를 클릭합니다.

유효성 검사 로그 보기에 오류가 표시될 수도 있습니다.

b. 오류를 수정하고 매핑의 유효성을 다시 검사합니다.

2단계. 매핑 실행

매핑을 실행하여 데이터를 변환하고 준비 테이블에 로드합니다.

기본 데이터 통합 서비스를 선택하지 않은 경우 개발자 도구에서 하나를 선택하라는 메시지가 표시됩니다.

- ▶ 매핑 편집기의 빈 영역을 마우스 오른쪽 단추로 클릭하고 **매핑 실행**을 클릭합니다.

데이터 통합 서비스가 매핑을 실행하고 출력을 대상에 씁니다.

준비 테이블 관리

Informatica 플랫폼 준비를 구성할 때 MDM Hub에서 단일 또는 모든 준비 테이블에 대한 준비를 활성화 또는 비활성화할 수 있습니다. 준비를 수행하기 전에 MDM Hub 메타데이터를 모델 리포지토리와 동기화해야 합니다. 단일 또는 모든 준비 테이블에 대한 동기화를 활성화 또는 비활성화할 수 있습니다.

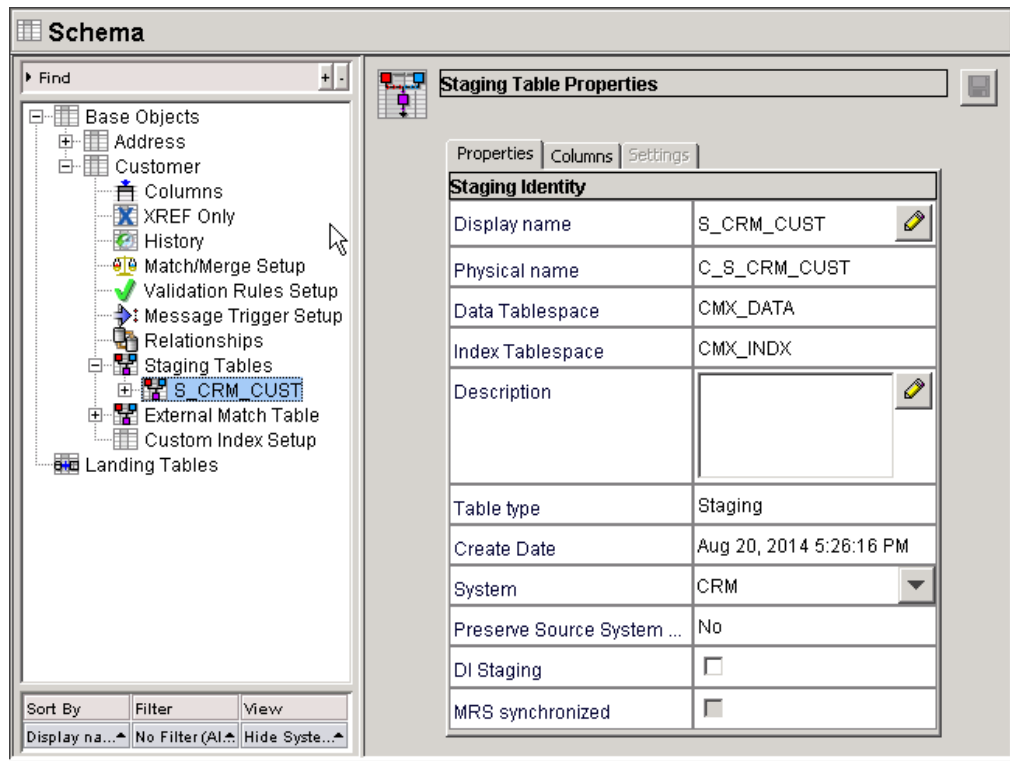
단일 준비 테이블에 대한 준비 비활성화

Hub 콘솔을 사용하여 단일 MDM Hub 준비 테이블에 대한 Informatica 플랫폼 준비를 비활성화할 수 있습니다.

1. 스키마 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 Informatica 플랫폼 준비를 비활성화해야 하는 기본 개체의 준비 테이블을 클릭합니다.

준비 테이블 속성 페이지가 나타납니다.

다음 이미지는 Informatica 플랫폼 준비를 비활성화할 수 있는 S_CRM_CUST 준비 테이블에 대한 준비 테이블 속성 페이지를 보여 줍니다.



4. 속성 탭에서 Informatica 플랫폼 준비를 비활성화하고 **저장**을 클릭합니다.

Informatica 플랫폼 준비에 대한 준비 테이블이 비활성화됩니다.

모든 준비 테이블에 대한 Informatica 플랫폼 준비 비활성화

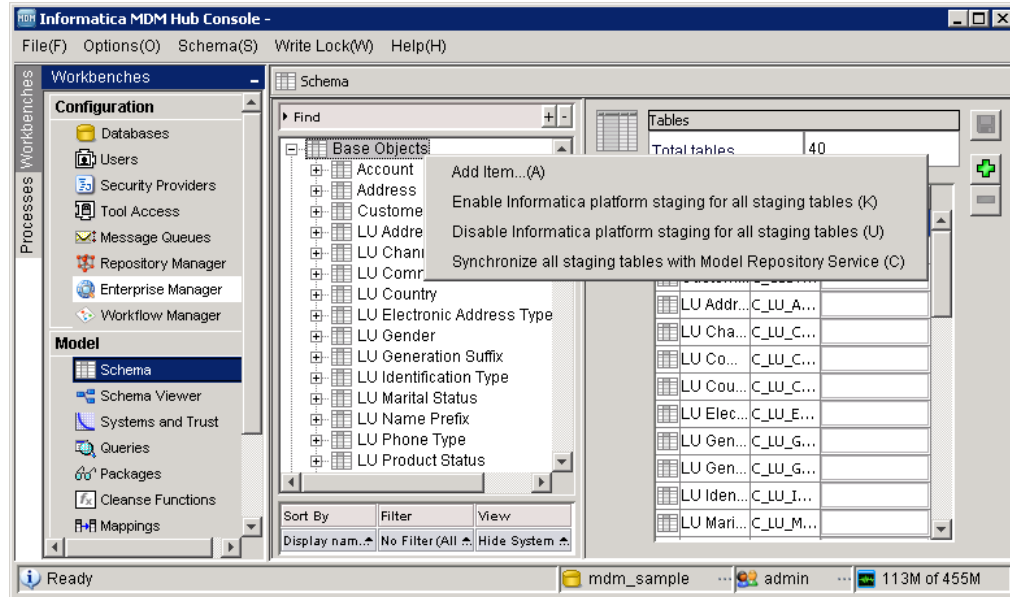
모든 MDM Hub 준비 테이블에 대한 Informatica 플랫폼 준비를 비활성화하면 준비 테이블은 MDM Hub을 통해 준비에 사용할 수 있도록 설정됩니다. Hub 콘솔을 사용하여 모든 MDM Hub 준비 테이블에 대한 Informatica 플랫폼 준비를 비활성화합니다.

1. 스키마 도구를 시작합니다.

2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 기본 개체를 마우스 오른쪽 단추로 클릭한 다음 **모든 준비 테이블에 대한 Informatica 플랫폼 준비 비활성화**를 클릭합니다.

MDM Hub이 모든 MDM Hub 준비 테이블에 대한 Informatica 플랫폼 준비를 비활성화합니다.

다음 이미지는 **모든 준비 테이블에 대한 Informatica 플랫폼 준비 비활성화** 옵션을 보여 줍니다. 이 옵션은 스키마 도구에 표시됩니다.



4. Informatica 플랫폼 준비가 비활성화되었는지 확인하려면 각 기본 개체에 대한 준비 테이블을 클릭하고 **Informatica 플랫폼 준비 및 모델 리포지토리 서비스와 동기화** 옵션이 비활성화되었는지 확인합니다.

모든 준비 테이블에 대한 Informatica 플랫폼 준비 활성화

Hub 콘솔을 사용하여 모든 MDM Hub 준비 테이블에 대한 Informatica 플랫폼 준비를 활성화할 수 있습니다.

1. 스키마 도구를 시작합니다.
2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 기본 개체를 마우스 오른쪽 단추로 클릭한 다음 **모든 준비 테이블에 대한 Informatica 플랫폼 준비 활성화**를 클릭합니다.

MDM Hub이 모든 MDM Hub 준비 테이블에 대한 Informatica 플랫폼 준비를 활성화합니다.

Informatica 플랫폼 준비가 활성화되었는지 확인하려면 각 기본 개체에 대한 준비 테이블을 클릭하고 **Informatica 플랫폼 준비** 옵션이 활성화되었는지 확인합니다.

4. Developer tool에서 Informatica 플랫폼 준비에 대해 생성한 프로젝트를 엽니다.

연산 참조 저장소 이름으로 된 프로젝트가 모델 리포지토리에서 생성된 것을 확인할 수 있습니다. 프로젝트에는 각 준비 테이블에 대한 실제 및 논리적 데이터 개체와 맵셋이 포함되어 있습니다.

모든 준비 테이블에 대한 변경 사항을 모델 리포지토리와 동기화

Hub 콘솔을 사용하여 모든 MDM Hub 준비 테이블에 대한 변경 사항을 모델 리포지토리와 동기화하도록 활성화할 수 있습니다. 모든 MDM Hub 준비 테이블에 대한 변경 사항을 모델 리포지토리와 동기화하기 전에 Informatica 플랫폼 준비에 대해 모든 MDM Hub 준비 테이블을 구성합니다.

1. 스키마 도구를 시작합니다.

2. 쓰기 잠금을 획득합니다.
3. 탐색 트리에서 기본 개체를 마우스 오른쪽 단추로 클릭한 다음 **모든 준비 테이블을 MRS와 동기화**를 클릭합니다.
모든 테이블이 동기화되었습니다. 메시지가 표시됩니다.
4. **확인**을 클릭합니다.
Hub 콘솔을 통해 준비 테이블에 변경한 사항이 모델 리포지토리에 표시됩니다.
5. 개발자 도구를 시작하고 준비에 대해 생성한 프로젝트를 선택합니다.
Informatica 플랫폼 준비에 대한 실제 및 논리적 데이터 개체와 매핑을 볼 수 있습니다.

추가 설명서

Informatica 플랫폼 준비와 관련된 주제에 대한 자세한 내용은 다음 설명서를 참조하십시오.

- *Informatica Developer* 도구 가이드. 데이터 개체 및 연결에 대한 정보를 제공합니다.
- *Informatica* 매핑 가이드. 매핑 및 맵렛에 대한 정보를 제공합니다.
- *Informatica Developer* 변환 가이드. 변환에 대한 정보를 제공합니다.

부록 H

Informatica Platform 매핑 예

이 부록에 포함된 항목:

- [Informatica Platform 매핑 예제 개요, 703](#)
- [기본 키 생성 예제, 703](#)

Informatica Platform 매핑 예제 개요

Informatica Platform 매핑 예제는 소스 데이터를 정리하고 표준화하는 매핑을 보여주고 설명합니다. 매핑 예제는 MDM Hub 리소스 키트에 포함된 MDM Hub 샘플 ORS(연산 참조 저장소)에서 사용 가능한 데이터를 기반으로 합니다. MDM Hub 샘플 ORS의 매핑은 MDM Hub 매핑 도구에서 생성됩니다.

각 매핑 예제는 MDM Hub 매핑 도구에 정의된 매핑과 Informatica Developer(Developer tool)에 정의하는 매핑을 설명합니다. 각 예제의 두 매핑 모두 동일한 기능 결과를 얻습니다. 두 종류의 매핑 모두에 대해 동일한 샘플 입력 데이터를 사용합니다. 둘 중 한 가지 유형의 매핑을 사용하여 소스 데이터를 정리하고 표준화할 수 있습니다.

각 예제는 데이터 정리 및 표준화 작업의 일부인 매핑 개체를 설명합니다. MDM Hub에 정의된 매핑은 MDM Hub 환경의 랜딩 테이블과 준비 테이블 사이에 있습니다. 데이터를 정리하고 표준화하기 전에 ETL 프로세스를 통해 데이터를 랜딩 테이블에 로드해야 합니다. Developer tool 매핑은 소스 데이터와 MDM Hub의 대상 준비 테이블에 쓰는 논리적 데이터 개체 사이에 직접 이루어집니다.

기본 키 생성 예제

당신은 대형 소매 체인의 매핑 개발자입니다. 비즈니스 분석가는 소스 테이블의 데이터를 사용하여 기본 키를 생성할 수 있는 매핑을 정의하는 매핑 사양을 생성합니다. 소스에서 대상 준비 테이블로 로드되는 데이터의 고유성을 유지하려면 기본 키를 생성해야 합니다.

MDM Hub 샘플 ORS(연산 참조 저장소)에는 기본 키를 생성하는 샘플 매핑이 포함되어 있습니다. Developer tool에서 매핑을 디자인하여 기본 키를 생성하고 대상 준비 테이블에 로드합니다.

매핑 생성

기본 키를 생성하는 매핑을 생성합니다.

이 매핑은 다음 태스크를 수행합니다.

- 소스 데이터 읽기

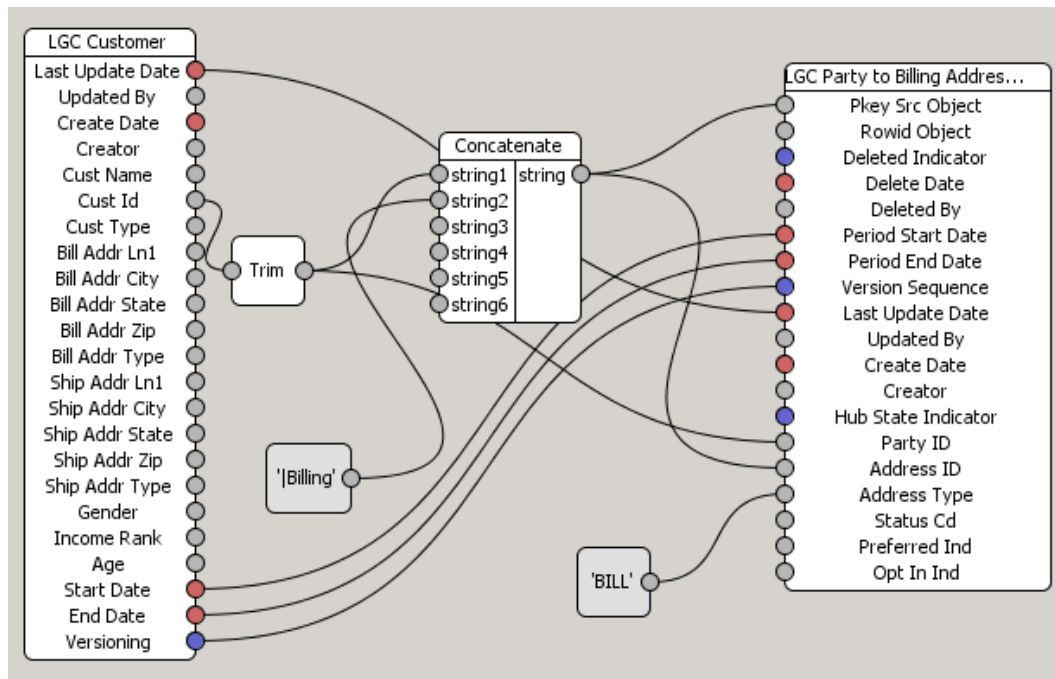
- 고객 ID 주위의 공백 잘라내기
- 공백을 잘라낸 고객 ID를 '| Billing' 상수와 연결
- 주소 유형에 'BILL' 상수를 추가
- 결과를 준비 테이블에 기록

MDM Hub 정리 매핑

MDM Hub 샘플 ORS(연산 참조 저장소)에는 기본 키를 생성하는 매핑이 포함되어 있습니다. 기본 키를 생성하기 위해 매핑은 고객 ID 주위의 공백을 잘라내고 값을 |Billing 상수와 연결합니다. 또한 매핑은 BILL 상수를 주소 유형에 추가합니다.

기본 키를 생성하는 LGC Party to Billing Address 매핑을 보려면 MDM Hub 샘플 ORS의 매핑 도구에서 매핑을 엽니다. 매핑에는 LGC Customer 랜딩 테이블과 LGC Party to Billing Address Stg 준비 테이블이 포함됩니다. LGC Customer 랜딩 테이블은 소스에서 ETL 프로세스를 통해 채워집니다.

다음 이미지는 MDM Hub 매핑 도구에서 LGC Party to Billing Address 샘플 매핑을 보여줍니다.



생성하는 매핑에는 다음 개체가 포함됩니다.

구성 요소 이름	설명
LGC Customer	소스에서 ETL 프로세스를 통해 채워진 소스 데이터를 포함하는 랜딩 테이블.
LGC Party to Billing Address Stg	MDM Hub가 정리되고 표준화된 데이터를 로드하는 준비 테이블.

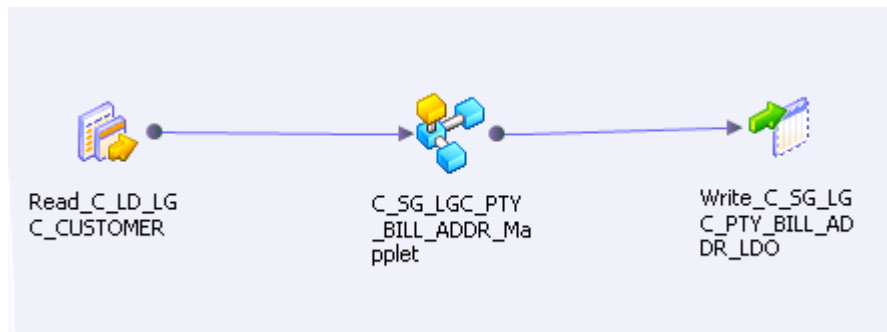
구성 요소 이름	설명
문자열 함수	매핑은 데이터를 정리하고 표준화하는 다음 문자열 함수를 포함합니다. - 연결. 고객 ID를 문자열 값과 연결합니다. - 잘라내기. 고객 ID 주위의 공백을 제거합니다.
상수	매핑에는 다음 상수가 포함됩니다. - Billing. 고객 ID에 연결되는 상수입니다. - BILL. 고객의 주소 유형으로 표시되는 상수입니다.

Informatica Platform 매핑

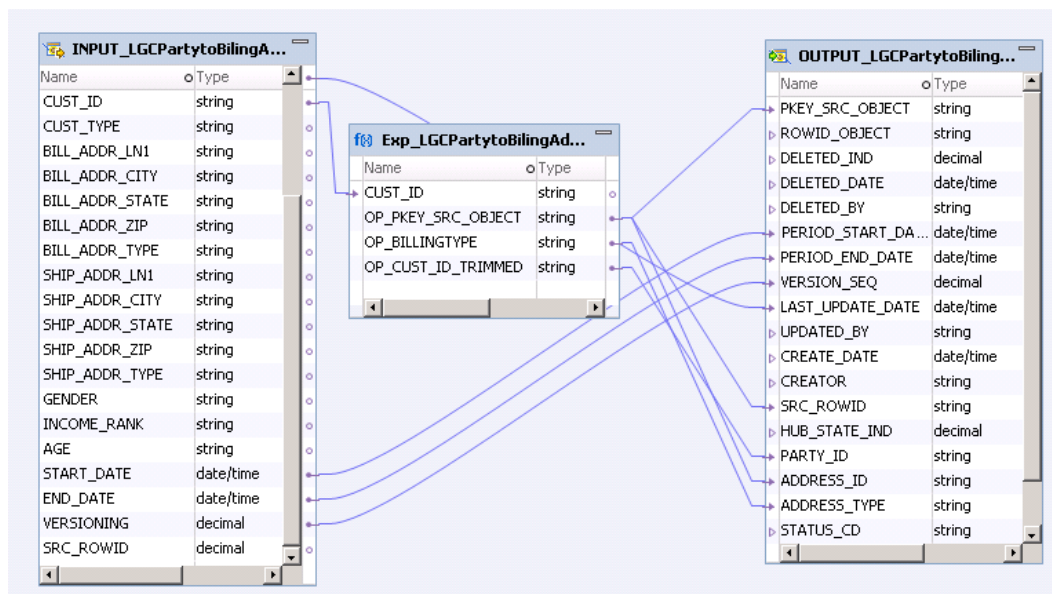
기본 키를 생성하는 Informatica Developer(Developer tool)에서 매핑을 생성합니다. 매핑은 고객 ID 주변의 공백을 잘라내고 값을 |Billing 상수와 연결해야 합니다. 또한 매핑은 BILL 상수를 주소 유형에 추가해야 합니다.

샘플 입력 데이터로 C_LD_LGC_CUSTOMER 소스 데이터를 설정합니다. 실제 데이터 개체, 맵셋 및 논리적 데이터 개체를 사용하여 매핑을 생성합니다. 실제 데이터 개체는 소스 데이터에 연결됩니다. 맵셋에는 기본 키를 생성하기 위한 변환이 포함되어 있습니다. 논리적 데이터 개체는 준비 테이블에 씁니다. ETL 프로세스를 사용하지 않고도 데이터를 정리하고 소스에서 대상 준비 테이블로 직접 이동할 수 있습니다.

다음 이미지는 Developer tool의 매핑을 보여 줍니다.



다음 이미지는 Developer tool의 매핑에서 사용하는 맵셋을 보여줍니다.



생성하는 매핑에는 다음 개체가 포함됩니다.

개체 이름	설명
Read_C_LD_LGC_CUSTOMER	데이터 소스를 나타내기 위해 생성하는 실제 데이터 개체입니다. 데이터 소스에서 데이터를 읽습니다.
C_SG_LGC_PTY_BILL_ADDR_Mapplet	동기화 프로세스 중에 생성되는 맵릿입니다. 식 변환을 맵릿에 추가합니다.
Exp_LGCPartytoBilingAddress	데이터를 정리하고 표준화하기 위한 식 변환 다음 포트를 포함합니다. <ul style="list-style-type: none"> - CUST_ID. 데이터 소스에서 고객 ID를 전달하는 통과 포트. - OP_CUST_ID_TRIMMED. TRIMMEDCUSTID 식을 사용하여 공백을 잘라낸 후 고객 ID를 반환하는 출력 포트. - OP_PKEY_SRC_OBJECT. CONCAT(TRIMMEDCUSTID, ' Billing') 식을 사용하여 Billing 상수를 고객 ID에 연결한 후 기본 키를 반환하는 출력 포트. - OP_BILLINGTYPE. 'BILL' 식을 사용하여 주소 유형 출력에 추가된 BILL 상수를 반환하는 출력 포트. - TRIMMEDCUSTID. LTRIM(RTRIM(CUST_ID)) 식을 사용하여 고객 ID의 오른쪽과 왼쪽 공백을 잘라낸 후 고객 ID를 반환하는 변수 포트.
Write_C_SG_LGC_PTY_BILL_ADDR_LDO	동기화 프로세스 중에 생성되는 논리적 데이터 개체입니다. 대상 준비 테이블인 LGC Party to Billing Address Stg를 나타내며 정리되고 표준화된 데이터를 준비 테이블에 씁니다.

입력 데이터 샘플

입력 데이터 집합에는 각 고객에 대한 고객 ID, 이름 및 주소 세부 사항과 같은 소스 데이터가 포함됩니다.

Informatica Platform 준비의 경우 샘플 입력 데이터를 사용하여 Informatica Developer(Developer tool)에서 연결할 수 있는 데이터 소스를 생성합니다. 데이터 소스를 Developer tool에서 매핑에 실제 데이터 개체로 추가합니다.

MDM Hub 준비의 경우 매핑을 생성하기 전에 데이터 소스의 샘플 입력 데이터를 랜딩 테이블에 로드합니다. ETL 프로세스를 사용하여 데이터를 랜딩 테이블에 로드합니다.

다음 데이터 추출은 소스 고객 데이터의 샘플을 보여줍니다.

CUST_NAME	CUST_ID	BILL_ADDR_LN1	BILL_ADDR_STATE	BILL_ADDR_TYPE
CBS	SLS2051	4217 COLBY AVE SW, WYOMING	MI	B
CITIFINANCIAL	SLS2052	32525 BRIARWOOD DR, RIDGEVILLE	OH	B
RAKE CHARLES	SLS2053	590 N 24TH AVE E, ADOLPH	MN	B
RUBI SAEED	SLS2054	519 S QUEENS RD, ROCHELLE	IL	B
AHMED RAUF	SLS2055	7610 ROSENWALD LN, NOKESVILLE	VA	B
JEFF RIBBON	SLS2056	46 ATLANTA ST, LUMBERTON	NC	B
YIN HUN IAN PUN	SLS2000	2958 RIPLEY RD, EUCLID	OH	B

출력 데이터 샘플

출력 데이터 집합에는 기본 키, 고객 ID, 주소 ID 및 각 고객에 대한 주소 유형이 포함되어 있습니다. 정의한 매핑이 소스 데이터를 정리하고 표준화한 후에 Billing Address Stg 준비 테이블의 LGC Party에 데이터가 로드됩니다.

준비 프로세스가 생성하는 기본 키는 PKEY_SRC_OBJECT 및 ADDRESS_ID 열에 로드됩니다. 기본 키 값은 |Billing 상수와 연결되어 있는 트리밍된 고객 ID로 구성됩니다.

ADDRESS_TYPE 열에는 식 변환의 OP BILLINGTYPE 출력 포트를 통해 채워진 BILL 상수가 포함되어 있습니다.

다음 데이터 추출에서는 Billing Address Stg 준비 테이블의 LGC Party에 로드되는 고객 데이터 샘플이 표시됩니다.

PKEY_SRC_OBJECT		PARTY_ID	ADDRESS_ID		ADDRESS_TYPE
SLS2051	Billing	SLS2051	SLS2051	Billing	BILL
SLS2052	Billing	SLS2052	SLS2052	Billing	BILL
SLS2053	Billing	SLS2053	SLS2053	Billing	BILL
SLS2054	Billing	SLS2054	SLS2054	Billing	BILL
SLS2055	Billing	SLS2055	SLS2055	Billing	BILL
SLS2056	Billing	SLS2056	SLS2056	Billing	BILL
SLS2000	Billing	SLS2000	SLS2000	Billing	BILL

부록 |

용어

BI 공급업체

비즈니스 인텔리전스 소프트웨어 제품을 생산하는 회사입니다.

BPM(Business Process Management)

BPM(Business Process Management)은 조직의 프로세스를 조정하는 데 초점을 맞춥니다. Informatica MDM에는 포함된 BPM 엔진이 함께 제공됩니다. 이 엔진을 사용하면 마스터 데이터의 검토 및 승인 프로세스를 자동화할 수 있습니다.

BVT

[BVT\(최선의 진실\) 페이지 708](#)를 참조하십시오.

BVT(최선의 진실)

소스 레코드에서 최선의 셀 데이터와 통합된 레코드입니다.

병합 스타일 기본 개체의 경우 기본 개체 레코드가 BVT 레코드이며, 이 레코드는 해당하는 소스 레코드에서 가장 신뢰할 수 있는 셀 값과 통합하는 방법으로 만들어집니다.

control table: 제어 테이블

Informatica MDM Hub에서 기본 개체용으로 자동으로 생성하는 연산 참조 저장소의 시스템 테이블 유형 중 하나입니다. 제어 테이블은 로드, 병합 및 병합 해제 프로세스를 지원하는 데 사용됩니다. Informatica MDM Hub는 기본 개체의 트러스트가 활성화된 열 각각에 대한 레코드(소스 시스템의 식별자와 마지막 업데이트 날짜)를 해당되는 제어 테이블에 유지 관리합니다.

creation change list: 생성 변경 목록

리포지토리의 콘텐츠를 내보낸 결과물인 변경 목록입니다. 생성 변경 목록은 리포지토리 관리자에서 디자인 개체를 가져오는 데 사용됩니다.

cross-reference table: 교차 참조 테이블

Informatica MDM Hub에서 기본 개체용으로 자동으로 생성하는 연산 참조 저장소의 시스템 테이블 유형 중 하나입니다. 기본 개체의 각 레코드에 대해 교차 참조 테이블에는 소스 시스템당 0개에서 n개(0-n)의 레코드가 포함됩니다. 이 레코드에는 소스 시스템의 기본 키와 소스 시스템이 기본 개체 테이블의 각 셀에 제공한 최신 값이 들어 있습니다.

Data Access Services: 데이터 액세스 서비스

Informatica MDM Hub에서는 이 응용 프로그램 서버 수준 기능을 사용하여 여러 모드의 데이터 액세스를 지원하고, Informatica SIF(서비스 통합 프레임워크)를 통해 많은 Informatica MDM Hub 데이터 서비스를 제공할 수 있습니다. 이를 이용하면 실시간 동기식 통합과 비동기식 통합을 모두 간편하게 수행할 수 있습니다.

database: 데이터베이스

Hub 저장소에 구성된 데이터 컬렉션입니다. Informatica MDM Hub는 마스터 데이터베이스와 ORS(연산 참조 저장소)의 두 가지 데이터베이스 유형을 지원합니다.

data cleansing: 데이터 정리

데이터 콘텐츠와 레이아웃을 표준화하고, 텍스트 값을 식별 가능한 요소로 분해하여 구문 분석하고, 데이터 라이브러리를 기준으로 식별 가능한 값(예: 우편 번호)을 검증하고, 잘못된 값을 데이터 라이브러리의 올바른 값으로 대체하는 프로세스입니다.

Data Manager: 데이터 관리자

자동 병합을 비롯한 모든 병합 결과를 검토하고 필요한 경우 데이터 콘텐츠를 수정하는 데 사용하는 도구입니다. 이 도구는 각 기본 개체 레코드의 데이터 연계에 대한 보기를 제공합니다. 또한 데이터 관리자를 사용하여 이전에 병합한 레코드의 병합을 해제할 수 있으며, 각 통합된 레코드에 대한 다양한 유형의 기록을 볼 수 있습니다.

데이터 관리자 도구를 사용하여 레코드를 검색하고, 레코드의 교차 참조를 확인하고, 레코드 병합을 해제하고, 레코드 연결을 해제하고, 기록 레코드를 확인하고, 새 레코드를 생성하고, 레코드를 편집하고, 트러스트 설정을 재정의할 수 있습니다. 데이터 관리자는 사용자가 정의한 검색 조건과 일치하는 모든 레코드를 표시합니다.

datasource: 데이터 소스

응용 프로그램 서버 환경에서 데이터 소스는 데이터베이스 서버의 위치, 데이터베이스 이름, 데이터베이스 사용자 ID 및 암호 등 데이터베이스에 대한 정보를 식별하는 JDBC 리소스입니다. Informatica MDM Hub가 연산 참조 저장소와 통신하는 데 이 정보가 필요합니다.

data steward: 데이터 스튜어드

데이터 품질을 일차적으로 책임지는 Informatica MDM Hub 사용자입니다. 데이터 스튜어드는 Hub 콘솔을 통해 Informatica MDM Hub에 액세스하고, Informatica MDM Hub 도구를 사용하여 Hub 저장소의 개체를 구성합니다.

Data Steward workbench: 데이터 스튜어드 작업 영역

Informatica MDM Hub UI의 일부로, 데이터 의미를 파악하고 있으며 조직에서 데이터 신뢰성을 보장해야 하는 데이터 분석가나 스튜어드가 통합된 데이터와 예외 처리를 위해 대기 중인 일치 데이터를 검토하는 데 사용됩니다.

데이터 관리자, 병합 관리자 및 계층 관리자를 사용할 수 있는 도구를 포함합니다.

data type: 데이터 유형

테이블 열에서 허용되는 값의 특성(문자, 숫자, 날짜, 이진 데이터 등)을 정의합니다. Informatica MDM Hub의 열에는 Informatica MDM Hub 구현에 사용된 데이터베이스 플랫폼의 데이터 유형에 직접 매핑되는 공통 데이터 유형 집합이 사용됩니다.

decay curve: 붕괴 곡선

시간이 지남에 따라 트러스트가 붕괴되는 방식을 시각적으로 보여 줍니다. 곡선 형태는 구성된 붕괴 유형과 붕괴 기간에 따라 결정됩니다.

decay period: 붕괴 기간

트러스트 수준이 최대 트러스트 수준에서 최소 트러스트 수준으로 붕괴하는 데 걸리는 기간(일, 주, 월, 분기 및 년)입니다.

decay type: 붕괴 유형

붕괴 기간 동안 트러스트 수준이 감소하는 방식입니다.

deleted state(records): 삭제된 상태(레코드)

삭제된 레코드는 더 이상 Hub 데이터의 일부일 필요가 없는 레코드입니다. 이러한 레코드는 특별히 요청하지 않는 한 프로세스에서 사용되지 않습니다. 레코드는 명시적으로만 삭제할 수 있으며 필요한 경우 삭제한 레코드를 복원할 수 있습니다. 보류 중인 레코드를 삭제하면 영구적으로 삭제되며 복원할 수 없습니다.

delta detection: 델타 검색

이 기능이 활성화되어 있으면 준비 프로세스 중에 MDM Hub 프로세스에서 새로 추가되었거나 변경된 레코드를 처리합니다. 델타 검색은 전체 레코드 비교나 날짜 열을 통해 수행 가능합니다.

design object: 디자인 개체

구현에 대한 스키마 및 기타 구성 설정을 정의하는 데 사용되는 메타데이터 부분입니다. 디자인 개체에 포함되는 Informatica MDM Hub 개체의 유형에는 기본 개체 및 열, 랜딩 및 준비 테이블, 열, 인덱스, 관계, 매핑, 정리 함수, 쿼리 및 패키지, 트러스트 설정, 유효성 검사 및 일치 규칙, 보안 액세스 관리자 정의, 계층 관리자 정의, 기타 설정 등이 있습니다.

distinct mapping: 고유 매핑

랜딩 테이블의 열과 랜딩 테이블에서 고유 레코드만 선택하는 준비 테이블 간의 매핑입니다. 여러 개의 준비 테이블에 공급하는 단일 랜딩 테이블이 있으며 이러한 랜딩 테이블이 비정규화된 상태에서는 고유 매핑을 사용하는 것이 유용합니다. 예를 들어 랜딩 테이블에 고객 데이터 및 주소 데이터가 모두 포함될 수도 있습니다.

distinct source system: 고유 소스 시스템

통합할 필요 없이 기본 개체에 삽입되는 데이터를 제공하는 소스 시스템입니다.

distribution: 분포

조정을 통해 BVT(최선의 진실: Best Version of the Truth)가 설정된 후 마스터 레코드 데이터를 다른 응용 프로그램이나 데이터베이스에 배포하는 프로세스입니다.

downgrade: 다운그레이드

cleansePut 및 Put API를 사용하거나 로드 프로세스를 사용하여 데이터를 삽입 또는 업데이트하는 중 유효성 검사 규칙에 따라 레코드의 트러스트를 백분율 단위로 감소시킬 때 수행되는 작업입니다.

duplicate: 중복

특정 열의 데이터(예: 이름, 주소 또는 조직 데이터)에서 하나 이상의 레코드가 동일하거나 거의 동일합니다. 일치 프로세스 중에 일치 규칙이 실행되어 두 레코드가 통합 시 중복 레코드로 간주할 수 있을 정도로 충분히 유사한지 여부를 확인합니다.

Dynamic Data Masking

중요한 데이터에 무단으로 액세스하지 못하도록 클라이언트와 데이터베이스 간에 작동하는 데이터 보안 제품입니다. Dynamic Data Masking은 데이터베이스에 전송된 요청을 가로채서 이 요청이 다시 클라이언트에게 전송되기 전에 데이터를 마스킹하도록 해당 요청에 데이터 마스킹 규칙을 적용합니다.

entity: 항목

계층 관리자에서 항목은 다른 항목에 연결할 수 있는, 유형이 지정된 개체입니다. 항목의 예로는 개인, 조직, 제품 및 가정이 있습니다.

entity base object: 항목 기본 개체

항목 기본 개체는 계층 관리자 항목에 대한 정보를 저장하는 데 사용되는 기본 개체입니다.

exact match: 정확히 일치

동일한 레코드만 일치하는 것으로 간주하는 일치/검색 전략입니다. 정확히 일치를 지정하는 경우 이 기본 개체에 대해 정확히 일치 열만 정의할 수 있습니다(정확히 일치하는 기본 개체에는 유사 항목 일치 열이 있을 수 없음). 정확히 일치/검색 전략을 사용하는 기본 개체를 정확히 일치하는 기본 개체라고 합니다.

exclusive lock: 배타적 잠금

Hub 콘솔에서 기반 스키마를 배타적으로 변경하기 위해 사용되는 잠금입니다. 배타적 잠금이 설정된 동안에는 다른 모든 Hub 콘솔 사용자가 대상 데이터베이스를 변경할 수 없습니다. 배타적 잠금은 배타적 잠금을 소유한 사용자가 해제해야 합니다. 다른 사용자는 배타적 잠금을 해제할 수 없습니다.

execution path: 실행 경로

Informatica MDM Hub에서 전체 일괄 그룹이 실행될 때 일괄 작업이 실행되는 시퀀스입니다. 실행 경로는 시작 노드에서 시작되어 종료 노드에서 끝납니다. 일괄 그룹 도구는 실행 시퀀스의 유효성을 검사하지 않으므로, 실행 시퀀스가 올바른지 확인하는 것은 사용자의 책임입니다.

export process: 내보내기 프로세스

리포지토리 관리자에서 리포지토리의 메타데이터를 이동식 변경 목록 XML 파일로 내보내는 프로세스로, 다른 리포지토리로 디자인 개체를 가져오거나 보관 용도로 소스 제어 시스템에 디자인 개체를 저장할 때 이 XML 파일을 사용할 수 있습니다. 내보내기 프로세스에서는 모든 지원되는 디자인 개체를 변경 목록 XML 파일에 복사합니다.

external application user: 외부 응용 프로그램 사용자

타사 응용 프로그램을 통해 MDM Hub 데이터에 간접적으로 액세스하는 MDM Hub 사용자입니다.

사용자 구성에 대한 자세한 내용은 *Multidomain MDM 보안 가이드*를 참조하십시오.

external cleanse: 외부 정리

랜덤 테이블을 채우기 전에 데이터를 정리하는 프로세스입니다. 외부 정리는 대개 ETL(추출-변환-로드) 도구나 다른 데이터 정리 유틸리티를 통해 Informatica MDM Hub 외부에서 수행됩니다.

external match: 외부 일치

기본 개체의 데이터를 실제로 변경하거나 기본 개체와 연결된 일치 테이블을 변경하지 않고 유사 항목 일치 기본 개체에서 기존 데이터와 새 데이터(별도의 입력 테이블에 저장되어 있는 데이터)를 일치시키고, 일치 값을 테스트하며, 결과를 검사할 수 있는 프로세스입니다.

extract-transform-load (ETL) tool: ETL(추출-변환-로드) 도구

소스 시스템에서 데이터를 추출하고, 데이터를 원하는 상태로 변환하기 위해 규칙, 조회 테이블 및 기타 기능을 사용하여 데이터를 변환하고, 대상 데이터베이스에 데이터를 로드(쓰기)하는 소프트웨어 도구(Informatica MDM Hub 외부에 있는 도구)입니다. Informatica MDM Hub 구현에서는 소스 시스템에서 데이터를 추출하고 랜덤 테이블에 채우는 데 ETL 도구가 사용됩니다.

foreign key: 외래 키

관계형 데이터베이스에서 다른 테이블(경우에 따라 같은 테이블)의 기본 키 값에 해당하는 값을 포함하는 열(또는 열 집합)을 나타냅니다. 외래 키는 다른 테이블을 가리키는 포인터 역할을 합니다. 예를 들어 Employee 테이블의 Department_Number 열이 Department 테이블의 기본 키를 가리키는 외래 키일 수 있습니다.

function: 함수

[정리 함수 페이지 731](#)를 참조하십시오.

fuzzy match: 유사 항목 일치

철자 변형, 가능한 철자 오류 등과 같이 일치하는 레코드를 동일하지 않은 레코드로 만들 수 있는 차이점을 고려하는 확률 일치 방식을 사용하는 일치/검색 전략입니다. 이 전략을 선택한 경우 Informatica MDM Hub에서 기본 개체에 특수한 열(유사 항목 일치 키)을 추가합니다. 유사 항목 일치/검색 전략을 사용하는 기본 개체를 유사 항목 일치 기본 개체라고 합니다. 유사 항목 일치를 사용하려면 선택된 인구집단이 필요합니다.

fuzzy match key: 유사 항목 일치 키

일치 열에서 유사 항목 일치/검색 전략을 사용하는 경우 스키마 관리자에서 추가하는 기본 개체의 특수 열입니다. 이 열은 검색 및 일치 도중에 해당 기본 개체에 대한 일치 후보를 생성하기 위해 사용되는 기본 필드입니다. 모든 유사 항목 일치 기본 개체에는 유사 항목 일치 키가 하나만 있습니다.

geocode: 좌표 부여

경도, 위도, 고도(선택 사항) 등의 지리적 좌표를 기준으로 한 위치 주소. 근접 검색을 수행하려면 좌표 부여가 필요합니다.

global business identifier (GBID): GBID(글로벌 비즈니스 식별자)

비즈니스 요구 사항을 기준으로 전역에서 고유하게 레코드를 식별할 수 있는 공통 식별자(키 값)를 포함하는 열입니다. 예를 들어 다음과 같은 항목이 여기에 포함됩니다.

- Informatica MDM Hub의 외부 응용 프로그램(예: ERP 또는 CRM 시스템)에 의해 정의된 식별자
- 업종별 코드(AMA 번호, DEA 번호 등) 또는 정부 발행 식별자(주민등록번호, 납세자 번호, 운전면허번호 등)와 같이 외부 기관에 의해 정의된 식별자

hard delete: 영구 삭제

기본 개체 또는 교차 참조 레코드를 데이터베이스에서 물리적으로 제거하는 방법입니다.

Hierarchies Tool: 계층 도구

Informatica MDM Hub 관리자는 디자인 타임 계층 도구(이전의 "계층 관리자 구성 도구")를 사용하여 계층 관리자에서 데이터 관계를 확인하고 조작하는 데 필요한 구조를 설정합니다. 계층 도구를 사용하여 Informatica MDM Hub 구현에 필요한 계층 관리자 구성 요소(항목 유형, 계층, 관계 유형, 패키지, 프로필 등)를 정의할 수 있습니다. 계층 도구는 모델 작업 영역에서 액세스할 수 있습니다.

hierarchy: 계층

계층 관리자에서 관계 유형의 집합입니다. 이러한 관계 유형은 계층 내 항목의 위치를 기준으로 평가되지 않으며 서로 연관될 필요도 없습니다. 쉽게 분류하고 식별할 수 있도록 함께 그룹화된 관계 유형일 뿐입니다.

Hierarchy Manager: 계층 관리자

계층 관리자를 통해 사용자는 MDM Hub에서 관리되는 레코드와 관련된 계층 데이터를 관리할 수 있습니다. 자세한 내용은 *Multidomain MDM 구성 가이드* 및 *Multidomain MDM 데이터 스튜어드 가이드*를 참조하십시오.

hierarchy type: 계층 유형

계층 관리자에서 계층의 논리적 분류를 나타냅니다. 계층 유형은 특정 관계가 속한 계층의 일반 클래스입니다. [hierarchy: 계층 페이지 712](#)를 참조하십시오.

history table: 기록 테이블

연결된 테이블의 변경 내용에 대한 기록 정보를 포함하는 연산 참조 저장소의 테이블 유형입니다. 기록 테이블에서는 병합 및 병합 해제 기록, 사전 정리된 데이터 기록, 기본 개체 기록, 교차 참조 기록 등을 비롯한 세부 변경 내용 추적 옵션을 제공합니다.

HM package: HM 패키지

HM(계층 관리자) 패키지는 MDM 패키지의 하위 집합을 나타내며 계층 관리자에 필요한 메타데이터를 포함합니다.

hotspot: 핫스팟

비즈니스 데이터에서 과도 일치된 데이터(일치 항목의 큰 교집합)를 나타내는 레코드 그룹입니다.

Hub Console: Hub 콘솔

관리자 및 데이터 스튜어드용 도구 집합으로 구성된 Informatica MDM Hub 사용자 인터페이스입니다. 사용자는 각 도구를 사용하여 특정 작업 또는 관련 작업 집합(예: 데이터 모델 작성, 일괄 작업 실행, 데이터 흐름 구성, Informatica MDM Hub 리소스에 대한 외부 응용 프로그램 액세스 권한 구성, 기타 시스템 구성 및 운영 태스크)을 수행할 수 있습니다.

hub object: Hub 개체

Hub에 정의되어 있으며 비즈니스 항목에 대한 정보를 포함하는 다양한 개체 유형에 대한 일반적인 용어입니다. 몇 가지 예로는 기본 개체, 교차 참조 테이블 및 보고 매트릭스와 연결시킬 수 있는 Hub의 모든 개체가 있습니다.

Hub Server: Hub 서버

액세스 권한, 보안 및 세션 관리 등 핵심 및 일반 서비스에 사용되는 중간 계층(응용 프로그램 서버)의 런타임 구성 요소입니다.

Hub Store: Hub 저장소

Informatica MDM Hub 구현에서 마스터 데이터베이스와 하나 이상의 ORS(연산 참조 저장소) 데이터베이스를 포함하는 데이터베이스입니다.

immutable source: 변경할 수 없는 소스

항상 기본 개체에 대한 최종적인 최선의 진실(BVT)을 제공하는 데이터 소스입니다. 변경할 수 없는 소스의 레코드는 고유 레코드로 허용되며 해당 소스의 레코드는 완전히 통합된 후 더 이상 변경되지 않습니다. 이것은 병합의 경우에도 마찬가지입니다. 변경할 수 없는 소스는 고유 시스템이기도 합니다. 변경할 수 없는 소스 시스템의 모든 소스 레코드에 대해 로드 및 PUT의 통합 표시기는 항상 1(통합된 레코드)입니다.

implementer: 구현자

조직의 요구 사항에 따른 Informatica MDM Hub의 디자인, 개발, 테스트 및 배포에 대한 일차적 책임이 있는 Informatica MDM Hub 사용자입니다. 태스크에는 디자인 개체 생성, 스키마 작성, 일치 규칙 정의, 성능 조정 및 기타 활동이 포함되며 여기에만 제한되지 않습니다.

import process: 가져오기 프로세스

리포지토리 관리자에서 라이브러리나 변경 목록의 디자인 개체를 리포지토리에 추가하는 프로세스입니다. 디자인 개체는 대상 리포지토리에 아직 없습니다.

incremental load: **증분 로드**

기본 개체의 초기 데이터 로드가 완료된 후에 실행되는 모든 로드 프로세스를 나타냅니다. 새 데이터나 업데이트된 데이터만 기본 개체에 로드되기 때문에 증분 로드라고 합니다. 중복 데이터는 무시됩니다.

indirect match: **간접 일치**

[전이적 일치 페이지 730](#)를 참조하십시오.

initial data load: **초기 데이터 로드**

빈 기본 개체에 데이터가 처음 로드되는 때를 나타냅니다. 초기 데이터 로드 중에는 준비 테이블의 모든 레코드가 기본 개체에 새 레코드로 삽입됩니다.

internal cleanse: **내부 정리**

준비 프로세스 중에 랜딩 테이블에서 해당 준비 테이블로 복사되는 데이터를 정리하는 프로세스입니다. 내부 정리는 지원되는 정리 엔진과 함께 처리 서버에 의해 실행되는 구성된 정리 함수를 통해 Informatica MDM Hub 내부에서 수행됩니다.

job execution log: **작업 실행 로그**

일괄 처리 뷰어와 일괄 그룹 도구에서 작업 완료 상태와 모든 관련 메시지(성공, 실패, 경고 등)를 보여 주는 로그입니다.

key match job: **키 일치 작업**

둘 이상의 소스에서 동일한 기본 키를 사용하는 경우 이러한 소스의 레코드가 일치하는지 확인하는 Informatica MDM Hub 일괄 작업입니다. 키 일치 작업에서는 새 레코드를 서로 비교하고 기존 레코드와 비교한 다음, 기본 키 일치 규칙에 정의된 대로 소스 레코드 키의 비교 결과에 따라 잠재적 일치 항목을 식별합니다.

key type: **키 유형**

Informatica MDM Hub에서 정확하게 키를 생성하고 더 나은 검색을 수행할 수 있도록 일치 키에 대한 주요 특성을 지정합니다. Informatica MDM Hub에서 제공하는 일치 키 유형에는 Person_Name, Organization_Name 및 Address_Part1이 있습니다.

key width: **키 너비**

일치 프로세스 실행 중의 검색 속도, 반환되는 가능한 일치 후보의 수, 키가 사용할 수 있는 디스크 공간의 양을 결정합니다. 키 너비 옵션에는 표준, 확장됨, 제한됨 및 기본 설정이 있습니다. 키 너비는 유사 항목 일치 개체에 만 적용됩니다.

landing table: **랜딩 테이블**

소스 시스템이 Informatica MDM Hub에서 처리할 데이터를 저장하는 테이블입니다.

land process: **랜드 프로세스**

소스 시스템에서 랜딩 테이블을 채우는 프로세스입니다.

lineage: **연계**

Hub 저장소의 통합 레코드에 제공되는 시스템과 해당 시스템의 레코드입니다.

linear decay: **선형 붕괴**

트러스트 수준은 최대 트러스트에서 최소 트러스트까지 선형으로 감소합니다.

linear unmerge: 선형 병합 해제

기존 병합 트리 구조에서 기본 개체 레코드의 병합을 해제하고 제거하는 작업입니다. 병합 해제된 기본 개체 레코드 자체만 병합 트리 구조에서 제거되며, 병합 트리에서 그 아래에 있는 모든 기본 개체 레코드는 원래 병합 트리에 유지됩니다.

load insert: 삽입형 로드

레코드가 대상 기본 개체에 삽입되는 시점을 나타냅니다. 로드 프로세스 중에 준비 테이블의 레코드가 대상 테이블에 이미 존재하지 않는 경우에는 Informatica MDM Hub에서 대상 테이블에 해당 레코드를 삽입합니다.

load process: 로드 프로세스

준비 테이블의 데이터를 Hub 저장소의 해당 기본 개체에 로드하는 프로세스입니다. 새 데이터가 Hub 저장소의 기존 데이터와 겹치는 경우 Informatica MDM Hub에서는 트러스트 설정과 유효성 검사 규칙을 사용하여 신뢰도가 더 높은 값을 결정합니다. [트러스트 페이지 733](#), [유효성 검사 규칙 페이지 728](#), [load insert: 삽입형 로드 페이지 715](#), [load update: 업데이트형 로드 페이지 715](#)를 참조하십시오.

load update: 업데이트형 로드

레코드가 대상 기본 개체에 삽입되는 시점을 나타냅니다. 로드 프로세스 중에 준비 테이블의 레코드가 대상 테이블에 이미 존재하지 않는 경우에는 Informatica MDM Hub에서 대상 테이블에 해당 레코드를 삽입합니다.

lock: 잠금

자세한 내용은 [쓰기 잠금 페이지 728](#), [exclusive lock: 배타적 잠금 페이지 711](#)을 참조하십시오.

lookup: 조회

로드 작업 중에 상위 테이블에서 데이터 값을 검색하는 프로세스입니다. MDM Hub에서는 기본 개체와 연결된 준비 테이블을 구성할 때 준비 테이블(하위 테이블)의 외래 키 열이 상위 테이블의 기본 키와 관련되어 있으면 해당 상위 테이블에서 데이터를 검색하도록 조회를 구성할 수 있습니다.

manual merge: 수동 병합

레코드를 수동으로 병합하는 프로세스입니다. 일치 규칙으로 인해 자동 병합 또는 수동 병합이 수행될 수 있습니다. Informatica MDM Hub에 데이터 스튜어드가 관심을 가질 만큼 충분한 유사점이 있지만 시스템이 자동으로 레코드를 병합하기에는 유사점이 부족한 레코드를 식별하는 수동 병합을 수행하도록 지시하는 일치 규칙입니다.

manual unmerge: 수동 병합 해제

레코드의 병합을 수동으로 해제하는 프로세스입니다.

mapping: 매핑

소스 데이터에 적용되는 변환 집합을 정의합니다. 매핑은 준비 프로세스에서(또는 SiperianClient CleansePut API 요청을 사용하는 동안) 랜딩 테이블의 데이터를 준비 테이블에 전송하는 데 사용됩니다. 매핑을 통해 랜딩 테이블의 소스 열 및 준비 테이블에서 채울 대상 열과, 데이터 정리에 사용되는 모든 중간 정리 함수가 식별됩니다. 자세한 내용은 [조건부 매핑 페이지 731](#), [distinct mapping: 고유 매핑 페이지 710](#)을 참조하십시오.

master data: 마스터 데이터

회사의 비즈니스에 중요한 것으로 간주되며 둘 이상의 시스템 또는 비즈니스 프로세스에서 사용되어야 하는 공통 핵심 항목의 컬렉션으로, 각 항목의 특성 및 값을 함께 포함합니다. 마스터 데이터의 예로는 고객, 제품, 직원, 공급자 및 위치 데이터가 있습니다.

Master Database: 마스터 데이터베이스

Informatica MDM Hub 환경 구성 설정(사용자 계정, 보안 구성, ORS 레지스트리, 메시지 대기열 설정 등)을 포함하는 데이터베이스입니다. 주어진 Informatica MDM Hub 환경에는 하나의 마스터 데이터베이스만 있을 수 있습니다. 마스터 데이터베이스의 기본 이름은 CMX_SYSTEM입니다. [ORS\(연산 참조 저장소\) 페이지 717](#)를 참조하십시오.

Master Data Management: 마스터 데이터 관리

마스터 데이터를 엔터프라이즈의 레코드 시스템으로 생성하고 유지 관리하는 제어된 프로세스입니다. MDM은 유효성 검사를 통해 마스터 데이터를 올바르게 일관되며 완전하게 유지하기 위해 구현되며, 필요한 경우 내부 또는 외부 비즈니스 프로세스, 응용 프로그램 또는 사용자가 사용할 수 있도록 컨텍스트에서 배포됩니다.

master record: 마스터 레코드

기본 개체에서 지정된 항목(특정 조직이나 개인 등)의 "BVT(최선의 진실, Best Version of the Truth)"를 나타내는 단일 레코드입니다. 마스터 레코드는 항목에 대한 완전히 통합된 데이터를 나타냅니다.

match candidate: 일치 후보

유사 항목 일치 기본 개체에만 해당하며, 일치할 가능성이 있는 기본 개체의 모든 레코드를 나타냅니다.

match column: 일치 열

비교 용도의 일치 규칙에 사용되는 열을 나타냅니다. 각 일치 열은 기본 개체에 포함된 하나 이상의 열을 기반으로 합니다.

match column rule: 일치 열 규칙

일치 열로 정의한 열(예: 성, 이름, 주소1, 주소2)의 값을 기준으로 레코드를 일치시키는 데 사용되는 일치 규칙입니다.

match key: 일치 키

기본 개체의 유사 항목 일치 키 열에 있는 데이터를 나타내는 인코딩된 문자열입니다. 일치 키는 관련 변형이 동일한 일치 키 값을 가지도록 이름이나 주소 내 문자와 숫자의 조합으로 생성되는 압축 및 인코딩된 고정 길이 값으로 구성됩니다. 일치 키는 토큰화 프로세스 중에 생성되는 일치 토큰의 한 부분이며, 일치 키 테이블에 저장된 후 일치 후보를 식별하는 일치 프로세스 중에 사용됩니다.

match key table: 일치 키 테이블

토큰화 프로세스 중에 생성된 일치 토큰(일치 키 + 인코딩되지 않은 Raw 데이터)을 저장하는 시스템 테이블을 나타냅니다. 이 데이터는 일치 프로세스 중에 중복되는 레코드를 확인하기 위해 정의된 일치 규칙에 따라 일치하는 키를 비교하여 일치 후보를 식별하는 데 사용됩니다.

match list: 일치 목록

사용자 지정 표준화 목록을 정의합니다. 주소 확인이나 주소 분해와 같은 특수한 정리 기능에 액세스할 수 있는 미리 정의된 함수가 사용됩니다.

match search strategy: 일치 검색 전략

필요한 성능을 기준으로 일치의 신뢰도를 유사 항목 일치 또는 정확히 일치로 지정합니다. 정확히 일치/검색 전략은 속도가 더 빠르지만 데이터가 불완전한 경우 일부 일치 항목이 누락될 수 있습니다. [fuzzy match: 유사 항목 일치 페이지 712](#), [exact match: 정확히 일치 페이지 711](#), [일치 프로세스 페이지 730](#)를 참조하십시오.

match search strategy: 일치 검색 전략

필요한 성능을 기준으로 일치의 신뢰도를 유사 항목 일치 또는 정확히 일치로 지정합니다. 정확히 일치/검색 전략은 속도가 더 빠르지만 데이터가 불완전한 경우 일부 일치 항목이 누락될 수 있습니다. [fuzzy match: 유사 항목 일치 페이지 712](#), [exact match: 정확히 일치 페이지 711](#), [일치 프로세스 페이지 730](#)를 참조하십시오.

message: 메시지

Informatica MDM Hub에서는 JMS(Java Message Service) 메시지를 나타냅니다. 메시지 대기열 서버는 다음과 같은 두 가지 유형의 JMS 메시지를 처리합니다.

- 인바운드 메시지는 Informatica MDM Hub 서비스 호출의 비동기 처리 시 사용됩니다.
- 아웃바운드 메시지는 JMS를 통해 데이터 변경을 소스 시스템이나 다른 시스템에 배포하는 통신 채널을 제공합니다.

ORS(연산 참조 저장소)

마스터 데이터와 마스터 데이터에 적용되는 규칙을 포함하는 데이터베이스입니다. 규칙에는 마스터 데이터 처리를 위한 규칙, 마스터 데이터 개체 집합 관리를 위한 규칙 및 MDM Hub가 최선의 진실을 정의할 때 사용하는 처리 규칙 및 보조 논리가 포함됩니다. MDM Hub 구성에는 하나 이상의 연산 참조 저장소가 포함될 수 있습니다. ORS의 기본 이름은 CMX_ORS입니다.

PDP(정책 결정 지점)

사용자 ID를 인증하고 사용자에게 MDM Hub 리소스에 대한 액세스 권한을 부여하는 특정 보안 검사점입니다.

PEP(정책 적용 지점)

인증 및 권한 부여 요청에 대해 런타임 시 보안 정책을 적용하는 특정 보안 확인 지점입니다.

proximity search: 근접 검색

지정된 좌표 부여 반지름 이내에 있는 레코드를 일치시키는 프로세스입니다. 근접 검색은 열이 위도, 경도, 고도(선택 사항)로 채워진 기본 개체에 대해 수행됩니다.

RISL 붕괴

RISL(초기에 빠르다 나중에 느려짐 붕괴)은 붕괴 기간 시작 부분에 대부분의 감소를 넣습니다. 트러스트 수준은 오목한 포물 곡선을 따라갑니다. 소스 시스템에 이 붕괴 유형이 포함된 경우 시스템의 새 값이 트러스트되지만 새 값은 곧 재정의될 가능성이 많습니다.

SAM(보안 액세스 관리자)

SAM(보안 액세스 관리자)은 MDM Hub 리소스를 무단 액세스로부터 보호하는 보안 모듈입니다. 런타임 시 SAM은 MDM Hub 구현에 대해 조직의 보안 정책 결정을 적용하며 보안 구성에 따라 사용자 인증 및 액세스 권한 부여를 처리합니다.

SIF(서비스 통합 프레임워크)

클라이언트 프로그램과 상호 작용하는 Informatica MDM Hub의 부분으로, 논리적으로 SIF는 클라이언트/서버 모델에서 중간 계층 역할을 합니다. SIF를 통해 다음 아키텍처 변형 중 하나를 사용하여 요청/응답 상호 작용을 구현할 수 있습니다.

- SOAP 프로토콜을 사용하는 느슨하게 결합된 웹 서비스
- EJB(Enterprise JavaBean) 또는 XML 기반의 강력하게 결합된 Java 원격 프로시저 호출

- 비동기 JMS(Java Message Service) 기반 메시지
- HTTP(Hypertext Transfer Protocol)를 통해 앞뒤로 이동하는 XML 문서

SIRL 붕괴

SIRL(초기에 느리다 나중에 빨라짐) 붕괴는 붕괴 기간 끝 부분에 대부분의 감소를 넣습니다. 트러스트 수준은 불록한 포물 곡선을 따라갑니다. 소스 시스템에 이 붕괴 유형이 포함된 경우 값이 해당 붕괴 기간 끝에 근접할 때까지 소스 시스템에서 설정된 값을 다른 시스템이 재정의할 가능성은 상대적으로 희박합니다.

stripping: 제거

더 이상 사용되지 않는 용어입니다.

strip table: 제거 테이블

더 이상 사용되지 않는 용어입니다.

token table: 토큰 테이블

더 이상 사용되지 않는 용어입니다.

감사 가능 이벤트

내부 감사 메커니즘에서 추적할 수 있는 MDM Hub의 활동입니다. MDM Hub에서는 감사 정보를 감사 로그 테이블 C_REPOS_AUDIT에 저장합니다.

같지 않은 일치

일치 규칙을 구성하면 열의 같은 값이 서로 일치되지 않도록 제한할 수 있습니다. 같지 않은 일치는 정확히 일치 열에만 적용됩니다.

개인 리소스

역할 도구에서 숨겨진 Informatica MDM Hub 리소스는 SIF(서비스 통합 프레임워크) 작업을 통해 액세스할 수 없습니다. Hub 콘솔에서 새 리소스를 추가한 경우(예: 새 기본 개체) 이 리소스는 기본적으로 PRIVATE로 설정됩니다.

거부 테이블

Informatica MDM Hub가 다음과 같은 대상 테이블에 삽입할 수 없는 레코드를 포함한 테이블입니다.

- 지정된 랜딩 테이블의 레코드에 대해 지정된 정리를 수행한 후 준비 테이블(준비 프로세스)
- Hub 저장소 테이블(로드 프로세스)

셀 값이 너무 길거나 레코드 업데이트 날짜가 오늘 날짜보다 이후이기 때문에 레코드가 거부될 수 있습니다.

검색 수준

Informatica MDM Hub가 일치 항목을 엄격하게 검색하는 정도(한정됨, 표준, 포괄적, 최대)를 정의합니다. 목표는 데이터에 대해 최적의 일치 항목 수를 찾는 것입니다. 너무 적거나(부족 일치)(중요한 일치 항목이 누락됨) 너무 많으면(과도 일치)(중요하지 않은 일치 항목을 비롯한 너무 많은 일치 항목이 생성됨) 안 됩니다.

게시

Informatica MDM Hub 메시지를 다른 응용 프로그램, 데이터베이스 등에 분포하기 위해 이 메시지를 메시지 대기열에 제출하는 프로세스입니다.

경로

[일치 경로 페이지 729](#)를 참조하십시오.

계단식 병합 해제

상위 기본 개체의 레코드가 병합 해제되는 경우 수행할 작업을 지정하는 기능입니다. 이 기능이 활성화된 상태에서 상위 개체의 레코드가 병합 해제되면 MDM Hub이 하위 기본 개체에서 영향을 받는 레코드에 대해서도 병합을 해제합니다.

계단식 삭제

상위 개체의 레코드가 삭제되는 경우 하위 개체에서 관련된 레코드가 삭제되는 작업입니다. 계단식 삭제 작업을 활성화하려면 CASCADE_DELETE_IND 매개 변수를 1로 설정합니다.

공급자

[보안 공급자 페이지 724](#)를 참조하십시오.

공급자 속성

보안 공급자가 제공하는 서비스에 액세스하기 위해 보안 공급자가 필요로 할 수 있는 이름-값 쌍입니다.

과도 일치

유사 항목 일치 기본 개체에 한해 관련되지 않은 일치 항목을 비롯한 너무 많은 일치 항목을 반환하는 일치입니다. 일치를 구성할 경우 목적은 데이터에 대해 최적의 일치 항목 수를 찾는 것입니다.

관계

계층 관리자에서 특정한 두 항목 간의 소속을 설명합니다. 계층 관리자 관계는 관계 유형, 계층 유형, 관계 특성 및 관계가 활성 상태로 유지되는 날짜를 지정하는 방식으로 정의됩니다. 자세한 내용은 [관계 유형 페이지 719](#), [hierarchy: 계층 페이지 712](#)을 참조하십시오.

관계 기본 개체

관계 기본 개체는 계층 관리자 관계에 대한 정보를 저장하는 데 사용되는 기본 개체입니다.

관계 유형

일반적인 관계 클래스를 설명합니다. 관계 유형은 다음을 정의합니다.

- 이 유형의 관계에 포함될 수 있는 항목 유형
- 관계 방향(있는 경우)
- Hub 콘솔에 관계가 표시되는 방법

자세한 내용은 [관계 페이지 719](#), [hierarchy: 계층 페이지 712](#)을 참조하십시오.

관리 소스 시스템

기본 소스 시스템입니다. 데이터 관리자 또는 병합 관리자 도구에서 수동 트러스트 재정의 및 데이터 편집에 사용됩니다. [소스 시스템 페이지 726](#)을 참조하십시오.

구성 작업 영역

연산 참조 저장소, 사용자, 보안, 메시지 대기열 및 메타데이터 유효성 검사를 비롯하여 다양한 MDM Hub 개체를 구성하는 데 사용할 수 있는 도구가 포함되어 있습니다.

권한

MDM Hub 리소스에 액세스하는 데 필요한 권한입니다. MDM Hub 내부 권한 부여에서 각 역할에는 다음 권한 중 하나가 할당됩니다.

권한	사용자가 수행 가능한 작업....
READ	데이터를 봅니다.
CREATE	Hub 저장소에서 데이터 레코드를 생성합니다.
UPDATE	Hub 저장소에서 데이터 레코드를 업데이트합니다.
MERGE	데이터를 병합 및 병합 해제할 수 있습니다.
EXECUTE	정리 함수 및 일괄 그룹을 실행합니다.
DELETE	Hub 저장소에서 데이터 레코드를 삭제합니다.

권한은 MDM Hub 리소스에 대해 외부 응용 프로그램 사용자가 갖는 액세스 권한을 결정합니다. 예를 들어 특정 패키지 및 패키지 열에 대해 READ, CREATE, UPDATE 및 MERGE 권한이 포함되도록 역할을 구성할 수 있습니다. 이 권한은 설정이 Hub 콘솔 사용에 어느 정도 영향을 미치더라도 Hub 콘솔을 사용할 때는 적용되지 않습니다.

권한 부여

사용자에게 요청한 Informatica MDM Hub 리소스에 액세스할 수 있는 권한이 있는지 여부를 확인하는 프로세스입니다. Informatica MDM Hub에서는 역할에 리소스 권한이 할당됩니다. 사용자 및 사용자 그룹은 역할에 할당됩니다. 사용자의 리소스 권한은 사용자에게 할당된 역할과 사용자가 속해 있는 사용자 그룹에 할당된 역할에 따라 결정됩니다.

규칙

유효한 동작을 정의하거나 계산 및 비교를 정의하는 문입니다. MDM Hub는 규칙을 관리하고 마스터 데이터에 규칙을 적용합니다. [일치 규칙 페이지 729](#), [메시지 대기열 규칙 페이지 722](#), [상태 변환 규칙 페이지 726](#), [시간 표시 막대 규칙 페이지 727](#) 및 [유효성 검사 규칙 페이지 728](#)을 참조하십시오.

규칙 집합

[일치 규칙 집합 페이지 729](#)를 참조하십시오.

규칙 집합 필터링

일치 규칙 집합에 의한 처리에서 레코드를 제외하는 기능입니다. 예를 들어 여러 유형의 조직(고객, 공급업체, 잠재 고객, 파트너 등)을 포함한 조직 기본 개체가 있는 경우 공급업체만 선택하여 처리하는 일치 규칙 집합을 정의할 수 있습니다.

기간 시작 날짜

레코드 버전의 유효 기간이 시작하는 시간입니다.

기간 종료 날짜

레코드 버전의 유효 기간이 끝나는 시간입니다.

기본 개체

고객 또는 계정과 같은 비즈니스 관련 항목에 대한 정보가 포함된 테이블입니다.

기본 키

관계형 데이터베이스 테이블에서 레코드를 고유하게 식별하는 값을 포함한 열(또는 열 집합)입니다. 예를 들어 Department_Number 열은 부서 테이블의 기본 키가 됩니다.

기본 키 일치 규칙

레코드에 대해 같은 기본 키를 사용하는 두 개의 시스템에서 레코드를 일치시키는 데 사용되는 일치 규칙입니다. [match column rule: 일치 열 규칙 페이지 716](#)을 참조하십시오.

논리적 데이터 개체

조직 내 논리적 항목을 설명하는 개체입니다. 특성과 키로 구성되고 특성 간의 관계를 설명합니다.

논리적 데이터 개체 매핑

논리적 데이터 개체를 하나 이상의 실제 데이터 개체에 연결하는 매핑입니다. 변환 논리가 포함될 수 있습니다.

논리적 데이터 개체 모델

조직 내 데이터와 데이터 간 관계를 설명하는 데이터 모델입니다. 논리적 데이터 개체를 포함하고 있으며 이러한 개체 간 관계를 정의합니다.

논리적 데이터 개체 쓰기 매핑

논리적 데이터 개체를 입력으로 사용하여 대상에 데이터를 쓰는 매핑입니다. 하나 이상의 논리적 데이터 개체를 입력으로, 실제 데이터 개체를 대상으로 포함합니다.

논리적 데이터 개체 읽기 매핑

논리적 데이터 개체를 통해 데이터의 보기를 제공하는 매핑입니다. 하나 이상의 실제 데이터 개체를 소스로, 논리적 데이터 개체를 매핑 출력으로 포함합니다.

대량 병합

[자동 병합 페이지 730](#)을 참조하십시오.

대상 데이터베이스

Hub 콘솔에서 현재 도구의 대상이 되는 마스터 데이터베이스 또는 연산 참조 저장소(연산 참조 저장소)입니다. 사용자 도구와 같이 마스터 데이터베이스에 저장된 데이터를 관리하는 도구를 사용하려면 대상 데이터베이스가 마스터 데이터베이스여야 합니다. 연산 참조 저장소에 저장된 데이터를 관리하는 도구를 사용하려면 대상 연산 참조 저장소를 지정해야 합니다.

데이터 변경 이벤트

기간 동안 유효한 데이터에 대한 변경 사항입니다.

데이터 통합 서비스

Informatica Developer에 대한 데이터 통합 작업을 수행하는 응용 프로그램 서비스입니다. 데이터 통합 작업에는 데이터 미리 보기와 매핑 실행이 포함됩니다.

데이터 통합 준비

데이터 품질 변환을 사용하여 소스 시스템의 데이터를 바로 읽어 데이터를 정리한 후 이러한 데이터를 MDM Hub에서 해당하는 준비 테이블로 옮기는 프로세스입니다.

레코드

개체의 인스턴스를 나타내는, 테이블의 행입니다. 예를 들어 주소 테이블에서 레코드에는 단일 주소가 포함됩니다. [소스 레코드 페이지 726](#), [통합된 레코드 페이지 733](#)를 참조하십시오.

레코드 버전

특정 기간 동안 유효한 레코드의 버전입니다. 레코드에는 여러 버전이 있을 수 있고, 각 버전은 다른 기간 동안 유효할 수 있습니다.

리소스

Informatica MDM Hub 구현에 사용된 모든 Informatica MDM Hub 개체입니다. 특정 리소스는 보안 리소스로 구성할 수 있습니다(기본 개체, 매핑, 패키지, 원격 패키지, 정리 함수, HM 프로필, 감사 테이블 및 사용자 테이블). 또한 콘텐츠 메타데이터, 일치 규칙 집합, 메타데이터, 일괄 그룹, 감사 테이블 및 사용자 테이블 등을 비롯하여 SIF 작업에서 액세스할 수 있는 보안 리소스도 구성할 수 있습니다.

리소스 그룹

권한 할당을 간소화하는 보안 리소스 컬렉션을 통해 한 번에 여러 리소스에 권한을 할당할 수 있습니다(예: 역할에 리소스 그룹 쉽게 할당). 자세한 내용은 [리소스 페이지 722](#), [권한 페이지 720](#)을 참조하십시오.

리소스 키트

Informatica MDM Hub 리소스 키트는 확장 및 구현할 수 있는 Informatica MDM Hub 기능에 대한 예를 제공하는 유틸리티, 예 및 라이브러리 집합입니다.

리포지토리

연산 참조 저장소(연산 참조 저장소)입니다. 연산 참조 저장소는 자체 스키마 및 관련 속성 설정에 대한 메타데이터를 저장합니다. 리포지토리 관리자에서 리포지토리 간 메타데이터를 복사할 경우 항상 복사할 디자인 개체를 포함하는 소스 *리포지토리*와 디자인 개체의 대상인 *대상 리포지토리*가 있습니다.

리포지토리 관리자

Hub 콘솔의 리포지토리 관리자 도구는 리포지토리의 메타데이터 유효성을 검사하고 한 리포지토리에서 다른 리포지토리로 디자인 개체를 승격하며 디자인 개체를 리포지토리로 가져오고 리포지토리를 변경 목록으로 내보내는 데 사용됩니다.

맵렛

Mapplet Designer에서 작성한 변환의 집합입니다. 여러 매핑에서 논리를 재사용하려면 맵렛을 생성합니다.

메시지 대기열

특정 프로세스에서 다른 프로세스로 데이터를 전송하는 메커니즘입니다(예: Informatica MDM Hub에서 외부 응용 프로그램으로 전송).

메시지 대기열 규칙

기본 개체 이벤트를 식별하고 업데이트를 위해 영향을 받는 레코드를 내부 시스템으로 전송하는 메커니즘입니다. 업데이트, 병합 및 고유 레코드로 허용된 레코드에 대해 메시지 대기열 규칙이 지원됩니다.

메시지 대기열 서버

Informatica MDM Hub에서 들어오는 JMS 메시지와 나가는 JMS 메시지를 관리하기 위해 Informatica MDM Hub가 사용하는, 응용 프로그램 서버 환경에서 정의된 JMS(Java Message Service) 서버입니다.

메시지 트리거

Informatica MDM Hub에서 특정 작업이 수행될 경우 발생하는 규칙입니다. 규칙이 정의된 작업이 수행되면 JMS 메시지가 아웃바운드 메시지 대기열에 배치됩니다. 메시지 트리거는 메시지가 생성되는 조건(어떤 개체에 대해 어떤 작업이 수행되는지) 및 메시지가 배치되는 대기열을 식별합니다.

메타데이터

다른 데이터를 설명하는 데 사용되는 데이터입니다. Informatica MDM Hub에서 메타데이터는 관련 구성 설정과 함께 Informatica MDM Hub 구현에 사용된 스키마(데이터 모델)를 설명하는 데 사용됩니다.

메타데이터 유효성 검사

[유효성 검사 프로세스 페이지 729](#)를 참조하십시오.

모델 리포지토리

개발자 도구를 통해 액세스할 수 있는 프로젝트 및 폴더에 대한 메타데이터가 저장되는 관계형 데이터베이스입니다.

모델 리포지토리 서비스

Informatica 도메인의 응용 프로그램 서비스로서, 모델 리포지토리를 실행하고 관리합니다. 모델 리포지토리는 Informatica 제품에서 생성된 메타데이터를 관계형 데이터베이스에 저장하여 제품 간 공동 작업을 가능하게 합니다.

모델 작업 영역

구현자가 배포하는 동안 솔루션을 구성하는 데 사용되거나 변화하는 비즈니스 요구 사항에 맞춰 다양한 유형의 메타데이터 및 규칙의 데이터 아키텍처에 따라 진행 중인 구성에 사용되는 Informatica MDM Hub UI의 일부입니다.

쿼리 그룹을 생성하고 패키지 및 기타 스키마 개체를 정의하며 현재 스키마를 볼 수 있는 도구가 포함됩니다.

변경 목록

대상 리포지토리에 대한 변경 내용의 목록입니다. 변경은 기본 개체 추가 또는 일치 규칙의 속성 업데이트 등과 같이 대상 리포지토리에 대해 실행된, 변경 목록의 작업입니다. 변경 목록은 Hub 리포지토리 간의 차이점 목록을 나타냅니다.

병합 관리자

수동 병합을 위해 대기 중인 레코드를 검토하고 처리하는 데 사용되는 도구입니다.

병합 스타일 기본 개체

Informatica MDM Hub의 일치 및 병합 기능에서 사용되는 기본 개체 유형입니다.

병합 프로세스

레코드가 지정된 일치 열에서 같은 값(또는 매우 유사한 값)을 포함하므로 기본 개체 테이블의 둘 이상의 레코드를 결합하는 프로세스입니다. [통합 프로세스 페이지 733](#), [자동 병합 페이지 730](#), [manual merge: 수동 병합 페이지 715](#)을 참조하십시오.

병합 해제

이전에 병합된 레코드를 병합 해제하는 프로세스입니다. 병합 스타일 기본 개체에만 사용됩니다.

보류 중 상태(레코드)

보류 중인 레코드는 Hub에서 일반적인 사용을 위해 아직 승인되지 않은 레코드입니다. 이러한 레코드에 대해 대부분의 작업을 수행할 수 있지만 단, 작업에서 보류 중인 레코드를 명시적으로 요청해야 합니다. 승인 프로세스를 진행하기 위해 레코드가 필요한 경우 이 레코드는 아직 승인되지 않았으며 승인 프로세스 중에 있습니다.

보안

Informatica MDM Hub 구현에서 무단 액세스, 무단 변경으로부터 데이터 및 기타 리소스를 보호하여 개인 정보, 기밀성 및 데이터 무결성을 보호하는 기능입니다.

보안 공급자

Informatica MDM Hub에 액세스하는 사용자에게 보안 서비스(인증, 권한 부여 및 사용자 프로필 서비스)를 제공하는 타사 응용 프로그램입니다.

보안 리소스

역할 도구에 표시되는 보호된 Informatica MDM Hub 리소스이며 역할 도구를 통해 특정 권한을 포함한 역할에 리소스를 추가할 수 있습니다. 사용자 계정에 특정 역할이 할당된 경우 해당 사용자 계정은 해당 역할에 연결된 권한에 따라 SIF를 사용하여 보안 리소스에 액세스할 수 있습니다. 외부 응용 프로그램이 SIF 작업을 사용하여 Informatica MDM Hub 리소스에 액세스하려면 해당 리소스가 SECURE로 구성되어야 합니다. 모든 Informatica MDM Hub 리소스는 기본적으로 PRIVATE로 설정되므로 리소스를 추가한 후 명시적으로 SECURE로 설정해야 합니다. [개인 리소스 페이지 718](#), [리소스 페이지 722](#)를 참조하십시오.

상태 설정	설명
SECURE	역할 도구에 이 Informatica MDM Hub 리소스를 표시합니다. 역할 도구를 통해 특정 권한을 포함한 역할에 리소스를 추가할 수 있습니다. 사용자 계정에 특정 역할이 할당된 경우 해당 사용자 계정은 해당 역할에 연결된 권한에 따라 SIF 요청을 사용하여 보안 리소스에 액세스할 수 있습니다.
PRIVATE	역할 도구에서 이 Informatica MDM Hub 리소스를 숨깁니다. 기본값입니다. SIF(서비스 통합 프레임워크) 작업을 통한 액세스를 제한합니다. Hub 콘솔에서 새 리소스를 추가한 경우(예: 새 기본 개체) 이 리소스는 기본적으로 PRIVATE로 설정됩니다.

보안 액세스 관리자 작업 영역

사용자, 그룹, 리소스 및 역할을 관리하는 도구를 포함합니다.

보안 페이로드

MDM Hub 작업 요청에 제공된 원시 이진 데이터로, 추가 인증 또는 권한 부여에 필요한 보조 데이터가 포함될 수 있습니다.

부족 일치

유사 항목 일치 기본 개체에만 해당되며, 결과 일치 항목의 수가 너무 적어 관련 일치 항목이 누락되는 일치입니다. 일치를 구성할 경우 목적은 데이터에 대해 최적의 일치 항목 수를 찾는 것입니다.

비교 변경 목록

두 리포지토리의 내용을 비교하고 대상 리포지토리에 대해 변경할 내용의 목록을 생성하는 방법으로 만들어진 변경 목록입니다. 비교 변경 목록은 리포지토리 관리자에서 디자인 개체를 승격하거나 가져올 때 사용됩니다.

비연속 유효 기간

레코드의 유효 기간이 중단되도록 다른 레코드 버전의 유효 기간과 연속되지 않는 레코드 버전의 유효 기간입니다.

비즈니스 프로세스

비즈니스 프로세스는 조직의 목표를 달성하고 비즈니스 업무를 구현하는 워크플로우입니다. 비즈니스 프로세스에는 목표를 달성하는 데 필요한 활동이 포함되어 있고, 이 프로세스에서는 활동을 통해 실행 경로를 정의합니다. Multidomain MDM에는 미리 정의된 Informatica ActiveVOS 비즈니스 프로세스가 포함되어며 이러한 비즈니스 프로세스는 ActiveVOS Server를 통해 관리됩니다. 비즈니스 관리자 또는 데이터 스튜어드 같이 승인된 담당자만이 마스터 데이터에 대한 모든 업데이트를 검토하도록 보장하는 것이 이 프로세스의 조직 관련 목표입니다.

비즈니스 항목

기본 개체의 중첩 구조입니다. 비즈니스 항목의 루트 기본 개체와 관련된 모든 정보를 보려면 Informatica Data Director에서 Entity 360 프레임워크를 사용합니다. 비즈니스 항목 내에서 데이터를 찾으려면 Informatica Data Director에서 검색을 수행합니다.

비즈니스 항목 서비스

비즈니스 항목 서비스는 MDM Hub 코드를 실행하여 비즈니스 항목의 기본 개체 레코드를 작성, 업데이트, 삭제 및 검색하는 작업 집합입니다.

사용자

Informatica MDM Hub 리소스에 액세스할 수 있는 개별 사용자 또는 응용 프로그램입니다. Informatica MDM Hub에서 사용자는 마스터 데이터베이스에 정의된 *사용자 계정*으로 표현됩니다.

사용자 개체

MDM Hub의 기능을 확장하기 위해 MDM Hub에 등록된 사용자 정의 함수입니다.

MDM Hub에는 다음과 같은 유형의 사용자 개체가 있습니다.

사용자 개체	설명
사용자 종료	미리 정의된 고정 매개 변수 집합을 포함하는 Java 코드입니다. 이 사용자 종료는 Informatica MDM Hub 일괄 처리가 실행되는 도중 특정 시점에 실행되도록 기본 개체별로 구성됩니다.
사용자 지정 Java 정리 함수	고객 논리로 표준 정리 라이브러리를 보완하는 Java 정리 함수입니다. 이러한 함수는 기본적으로 Jar 파일이며 데이터베이스에 BLOB로 저장됩니다.
사용자 지정 단추 함수	데이터 관리자, 병합 관리자 및 계층 관리자에 추가 아이콘 및 논리를 제공하는 사용자 지정 UI 함수입니다.

사용자 그룹

사용자 계정의 논리적 컬렉션입니다.

사용자 정의 열

시스템 열이 아닌 테이블의 모든 열. 사용자 정의 열은 스키마 관리자에서 추가되며 일반적으로 비즈니스 데이터를 포함합니다.

사용자 종료

사용자 종료는 MDM Hub의 기능을 확장하기 위한 SIF API 프로세스 또는 일괄 처리의 특정 시점에서 실행되는 Java 코드로 구성됩니다.

개발자는 일괄 작업의 사전 및 사후 처리를 위해 적절한 사용자 종료에 사용자 지정 코드를 추가하여 Informatica MDM Hub 일괄 처리를 확장할 수 있습니다.

상태 관리

MDM 데이터 흐름 전체에서 처리 논리에 영향을 주는 기본 개체 및 교차 참조 레코드의 시스템 상태를 관리하는 프로세스입니다. 레코드를 사용하는 Hub 도구를 통해 데이터 흐름의 여러 단계에서 기본 개체 및 교차 참조 레코드에 시스템 상태를 할당할 수 있습니다. 또한 스키마를 관리하는 여러 Hub 도구를 사용하여 기본 개체의 상태 관리를 활성화하거나 레코드 상태 변경을 제어하는 사용자 사용 권한을 설정할 수 있습니다.

상태 관리는 ACTIVE, PENDING 및 DELETED 상태로 제한됩니다.

상태 변환 규칙

레코드가 특정 상태에서 다른 상태로 변경되었는지 여부 및 변경된 시기를 확인하는 규칙입니다. 기본 개체 레코드와 교차 참조 레코드에 해당하는 상태 변환 규칙은 다릅니다.

상태 사용 기본 개체

상태 관리가 사용하도록 설정된 기본 개체입니다.

세그먼트 일치

일치 규칙을 데이터의 특정 하위 집합으로 제한하는 방법입니다. 예를 들어 서로 다른 국가의 고객에 대해 서로 다른 일치 규칙을 정의하여 세그먼트 일치를 통해 특정 규칙을 특정 국가 코드로 제한할 수 있습니다. 세그먼트 일치 규칙은 규칙별로 구성되며 정확히 일치 기본 개체 및 유사 항목 일치 기본 개체 둘 다에 적용됩니다.

셀

테이블에서 열과 레코드가 교차하는 부분입니다. 셀에는 데이터 값이나 null이 포함됩니다.

소스 레코드

소스 시스템의 원시 레코드입니다.

소스 시스템

Informatica MDM Hub에 데이터를 제공하는 외부 시스템입니다.

스키마

고객의 Informatica MDM Hub 구현에 사용된 데이터 모델입니다. Informatica MDM Hub에서는 특정 스키마를 강요하거나 요구하지 않습니다. 스키마는 소스 시스템과 무관합니다.

스키마 관리자

스키마 관리자는 스키마, 준비 및 랜딩 테이블을 정의하는 데 사용되는 Hub 콘솔의 디자인 타임 구성 요소입니다. 또한 스키마 관리자는 일치 및 병합, 유효성 검사 및 메시지 대기열에 대한 규칙을 정의하는 데 사용됩니다.

스키마 뷰어 도구

스키마 뷰어 도구는 Informatica MDM Hub 구현에 대해 구성된 스키마를 시각화하는 데 사용되는, Hub 콘솔의 디자인 타임 구성 요소입니다. 스키마 뷰어는 특히 복잡한 스키마를 시각화할 때 유용합니다.

승격 프로세스

의미는 컨텍스트에 따라 다릅니다.

- **리포지토리 관리자:** 한 리포지토리의 디자인 개체 변경 내용을 다른 리포지토리에 복사하는 프로세스입니다. 리포지토리 간 증분 변경 내용을 복사하는 데에는 승격이 사용됩니다.
- **상태 관리:** Informatica MDM Hub의 개별 레코드에 대한 시스템 상태를 변경하는 프로세스입니다(예: PENDING 상태에서 ACTIVE 상태로 변경).

시간 표시 막대

기간별 비즈니스 항목의 데이터 변경 이벤트 및 해당 관계입니다. 유효 기간 중의 데이터 변경 이벤트를 정의하십시오.

시간 표시 막대 규칙

데이터 변경 이벤트를 추적하기 위해 MDM Hub에서 적용하는 미리 정의된 규칙입니다. 시간 표시 막대 규칙을 기반으로 레코드 버전 및 해당 유효 기간 내의 데이터 변경 이벤트가 캡처됩니다.

시간 표시 막대 세분성

레코드 버전의 유효 기간을 정의하기 위해 사용할 시간 측정 단위입니다. 예를 들어, 유효 기간을 연, 월, 초 단위로 선택할 수 있습니다.

시간 표시 막대 작업

데이터 변경 이벤트를 추적하는 항목에 수행할 작업입니다. 레코드 추가, 레코드 편집, 유효 기간 편집 등과 같은 작업을 수행할 수 있습니다.

시스템 및 트러스트 도구

시스템 및 트러스트 도구는 Informatica MDM Hub에서 통합할 데이터를 제공할 수 있는 소스 시스템의 이름을 지정하는 데 사용되는 디자인 타임 도구입니다. 이 도구를 사용하여 기본 개체의 트러스트가 활성화된 각 열에 대해 각 소스 시스템과 연결된 트러스트 설정을 정의할 수 있습니다.

시스템 상태

Informatica MDM Hub에서 기본 개체 레코드가 지원되는 방식을 설명합니다. 지원되는 상태: ACTIVE, PENDING 및 DELETED 상태로 제한됩니다.

시스템 열

Informatica MDM Hub에서 자동으로 생성되어 유지 관리되는 테이블 열입니다. 시스템 열에는 메타데이터가 포함됩니다. 기본 개체의 공통 시스템 열로는 ROWID_OBJECT, CONSOLIDATION_IND 및 LAST_UPDATE_DATE가 있습니다.

실시간 모드

타사 응용 프로그램을 사용하여 Informatica MDM Hub와 상호 작용하는 방법으로 SIF(서비스 통합 프레임워크) 인터페이스를 통해 Informatica MDM Hub 작업을 호출합니다. SIF는 레코드 읽기, 정리, 일치, 삽입 및 업데이트 등과 같은 여러 서비스에 대한 작업을 제공합니다. [일괄 모드 페이지 729](#), [SIF\(서비스 통합 프레임워크\) 페이지 717](#)를 참조하십시오.

실제 데이터 개체

읽거나, 조회하거나, 리소스에 쓰는 데 사용되는 데이터의 실제 표현입니다.

쓰기 잠금

Hub 콘솔에서 기본 스키마를 변경하는 데 필요한 잠금입니다. 연산 참조 저장소 보안 도구를 제외하고 데이터 스튜어드 도구가 아닌 모든 도구는 쓰기 잠금을 획득하지 않는 한 읽기 전용 모드에 있습니다. 쓰기 잠금 상태에서는 여러 명의 사용자가 동시에 스키마를 변경할 수 있습니다.

암호 정책

암호 길이, 만료, 로그인 설정, 암호 재사용 및 기타 요구 사항을 비롯하여 Informatica MDM Hub 사용자 계정의 암호 특성을 지정합니다. Informatica MDM Hub 구현의 모든 사용자 계정에 대해 글로벌 암호 정책을 정의할 수 있으며, 개별 사용자의 이 설정을 재정의할 수 있습니다.

역할

보안 Informatica MDM Hub 리소스에 액세스하는 데 사용되는 권한 집합을 정의합니다.

연속 유효 기간

레코드의 유효 기간이 중단되지 않도록 다른 레코드 버전의 유효 기간과 연속된 레코드 버전의 유효 기간입니다.

열

테이블에서 특정 유형의 데이터 값을 포함하는 집합으로, 테이블의 각 행에서 특정 유형마다 하나씩 있습니다. 자세한 내용은 [시스템 열 페이지 727](#), [사용자 정의 열 페이지 725](#)를 참조하십시오.

요청

외부 응용 프로그램이 SIF(서비스 통합 프레임워크) 요청/응답 API 모델을 사용하여 특정 Informatica MDM Hub 기능에 액세스할 수 있도록 해주는 Informatica MDM Hub 요청(API)입니다.

워크플로우

Informatica Multidomain MDM에서 워크플로우는 조직 내의 비즈니스 프로세스를 나타냅니다. [비즈니스 프로세스 페이지 725](#)를 참조하십시오.

원시 테이블

랜딩 테이블의 데이터를 보관하는 테이블입니다.

유틸리티 작업 영역

응용 프로그램 이벤트를 감사하고, 일괄 그룹을 구성 및 실행하고, SIF API를 생성하기 위한 도구가 포함되어 있습니다.

유효 기간

레코드가 유효한 기간입니다. 유효 기간은 시작 날짜와 종료 날짜로 정의됩니다.

유효성 검사 규칙

데이터 값이 유효하지 않은 것으로 간주할 조건을 Informatica MDM Hub에 알려 주는 규칙입니다. 데이터가 유효성 검사 규칙에 지정된 조건을 충족하면 해당 데이터의 트러스트 값이 유효성 검사 규칙에 지정된 백분율만큼 다운그레이드됩니다. 열에 대해 최소 트러스트 보유 플래그가 설정되어 있는 경우에는 트러스트가 열의 최소 트러스트 설정 아래로 다운그레이드될 수 없습니다.

유효성 검사 프로세스

리포지토리를 설명하는 메타데이터의 완전성과 무결성을 확인하는 프로세스입니다. 유효성 검사 프로세스에서는 리포지토리의 논리적 모델을 해당하는 실제 스키마와 비교합니다. 문제가 발생하면 리포지토리 관리자는 주의가 필요한 문제 목록을 생성합니다.

인증

사용자 ID가 본인이 주장하는 것과 일치하는지 확인하는 프로세스입니다. Informatica MDM Hub에서 사용자는 제공된 자격 증명(사용자 이름/암호, 보안 페이로드 또는 둘 모두의 조합)을 기반으로 인증됩니다. Informatica MDM Hub에서는 내부 인증 메커니즘을 제공하며 타사 인증 공급자를 사용한 사용자 인증도 지원합니다.

일괄 그룹

개별 일괄 작업(예: 준비, 로드 및 일치 작업)의 컬렉션으로, 단일 명령으로 실행할 수 있습니다. 그룹의 각 일괄 작업은 순차적으로 실행되거나 다른 작업과 병렬로 실행될 수 있습니다.

일괄 모드

일괄 작업을 통해 MDM Hub와 상호 작용하는 방식으로, Hub 콘솔에서 실행하거나 타사 관리 도구로 실행하여 일괄 작업을 예약하고 실행할 수 있습니다.

일괄 작업

일괄 모드에서 처리되는 명령 시퀀스입니다. 다시 말하면, 명령 시퀀스가 나열된 파일을 배치 파일이라고 하고 이는 단일 프로세스로 실행되도록 전송됩니다.

일시 삭제

기본 개체 레코드나 교차 참조 레코드가 사용자 특성이나 HUB_STATE_IND에서 삭제된 레코드로 표시됩니다.

일치

두 레코드가 지정된 열에서 동일하거나 유사한 값을 가지기 때문에 두 레코드가 자동으로 병합되어야 하는지 아니면 수동 병합을 위해 후보가 되어야 하는지를 결정하는 프로세스입니다.

일치 경로

계층이 기본 개체 사이에 있던지(테이블 간 경로) 단일 기본 개체에 있던지(테이블 내 경로) 상관없이 레코드 간 계층을 이동할 수 있습니다. 일치 경로는 별도의 테이블이나 동일한 테이블에 있는 관련 레코드와 연관된 일치 열 규칙을 구성하는 데 사용됩니다.

일치 규칙

Informatica MDM Hub에서 레코드가 중복인지 여부를 결정하는 기준을 정의합니다. 일치 열은 일치 규칙에 포함되어 두 레코드가 병합하기에 충분히 유사한 것으로 간주되는 조건을 확인합니다. 각 일치 규칙은 Informatica MDM Hub에 유사점을 검사하는 데 필요한 일치 열의 조합을 지정합니다.

일치 규칙 집합

사용자가 일치 프로세스의 다양한 단계에서 다양한 규칙 집합을 실행할 수 있게 해주는 일치 규칙의 논리적 컬렉션입니다. 일치 규칙 집합에는 검색 전략을 지정하는 검색 수준과 원하는 만큼의 자동 및 수동 일치 규칙이 포함되며, 일치 프로세스 동안 레코드를 선택적으로 포함하거나 제외할 수 있는 필터가 선택적으로 포함됩니다. 일치 규칙 집합은 일치 열 규칙에 대해 실행하는 데 사용되지만 기본 키 일치 규칙에 대해 실행하는 데는 사용되지 않습니다.

일치 목적

유사 항목 일치 기본 개체의 경우 일치 규칙에 부속된 주목적을 정의합니다. 예를 들어 두 개의 레코드가 동일인에 대한 것인지 여부를 결정하는 데 있어 주소가 결정적 요소인 사람에 대해 일치 항목을 식별할 경우 "거주자"라고 하는 일치 목적을 사용합니다. 각 일치 목적에는 일치 목적을 달성하기 위해 두 개의 레코드를 최적으로 비교하는 방법에 대한 정보가 포함됩니다. Informatica MDM Hub에서는 일치 규칙을 적용하는 기준으로 선택한 일치 목적을 사용하여 일치하는 레코드를 결정합니다. 규칙의 동작은 선택한 목적에 따라 다릅니다.

일치 유형

각 일치 열에는 비교 일치를 준비하기 위해 일치 열이 토큰화되는 방식을 결정하는 일치 유형이 포함됩니다.

일치 테이블

일치 프로세스를 지원하는, 기본 개체와 연결된 시스템 테이블 유형입니다. 기본 개체의 일치 작업을 실행하는 동안 Informatica MDM Hub는 연결된 해당 일치 테이블을 일치하는 각 레코드 쌍에 대한 ROWID_OBJECT 값 및 일치를 지원하는 일치 규칙 식별자와 자동 병합 표시기로 채웁니다.

일치 토큰

기본 개체의 일치 열에서 인코딩된(일치 키) 값과 인코딩되지 않은(원시) 값을 모두 나타내는 문자열입니다. 일치 토큰은 토큰화 프로세스 동안 생성되어 일치 키 테이블에 저장되며 일치 후보를 식별하기 위해 일치 프로세스 동안 사용됩니다.

일치 프로세스

두 레코드의 유사점을 비교하는 프로세스입니다. 두 레코드가 서로 중복되었음을 나타내기 충분한 수의 유사점이 발견되면 Informatica MDM Hub에서 이러한 레코드를 병합하기 위해 플래그를 지정합니다.

일치 하위 유형

고객, 공급업체 및 파트너 레코드를 포함한 조직 기본 개체와 같은 서로 다른 유형의 데이터가 있는 기본 개체에서 사용됩니다. 일치 하위 유형을 사용하여 같은 기본 개체 내 특정 유형의 데이터에 일치 규칙을 적용할 수 있습니다. 각 일치 규칙에서 "하위 유형" 열로 사용할 **정확한** 일치 열을 지정하여 해당 일치 규칙에 대해 무시할 레코드를 필터링할 수 있습니다.

자동 병합

레코드를 자동으로 병합하는 프로세스입니다. 병합 스타일 기본 개체에만 사용됩니다. 일치 규칙으로 인해 자동 병합 또는 수동 병합이 수행될 수 있습니다. Informatica MDM Hub에서 자동 병합이 수행되도록 지정하는 일치 규칙은 수동 개입 없이 기본 개체 테이블에 있는 둘 이상의 레코드를 자동으로 결합합니다.

작업

더 이상 사용되지 않는 용어입니다. [요청 페이지 728](#)를 참조하십시오.

작업 영역

Hub 콘솔에서 유사한 여러 도구를 그룹화하기 위한 메커니즘입니다. 작업 영역은 관련 도구의 논리적 컬렉션입니다. 예를 들어, 모델 작업 영역에는 스키마, 쿼리, 패키지 및 매핑과 같은 데이터를 모델링하기 위한 도구가 포함되어 있습니다.

전이적 일치

BMG(일치 그룹 빌드) 프로세스 중 다른 일치 동작으로 인해 **간접적**으로 수행되는 일치입니다. 예를 들어 레코드 1이 레코드 2와 일치하고, 레코드 2가 레코드 3과 일치하며, 레코드 3은 레코드 4와 일치하는 경우 BMG 프로세

스를 통해 불필요한 일치 항목이 제거된 후에는 레코드 2, 3 및 4가 레코드 1과 일치할 수 있습니다. 이 예에서 레코드 4를 레코드 1과 일치시키는 명시적인 규칙은 없습니다. 대신 일치가 간접적으로 수행되었습니다.

전체 일치 추적

두 레코드가 중간 레코드를 통해 일치되도록 한 전체 또는 원래 일치 체인을 표시합니다.

정규식

일반적으로 사용되는 구문 규칙 및 기호 패턴에 따라 텍스트 데이터를 일치시키고 조작하는 데 사용되는 계산 식입니다. Informatica MDM Hub에서는 정규식 함수를 사용하여 정리 작업에 정규식을 사용할 수 있습니다. 구문 및 패턴을 포함하여 정규식에 대한 자세한 내용은 Javadoc에서 `java.util.regex.Pattern`을 참조하십시오.

정리

[data cleansing: 데이터 정리 페이지 709](#)를 참조하십시오.

정리 목록

정리 프로세스 중 입력 문자열의 일부를 바꾸기 위한 규칙을 논리적으로 그룹화한 것입니다.

정리 엔진

정리 엔진은 데이터 정리를 수행하기 위해 Informatica MDM Hub와 함께 사용되는 타사 제품입니다.

정리 함수

준비 작업 중 각 입력 문자열을 출력 문자열로 변환하여 수신 데이터를 변경하는 코드입니다. 일반적으로 이러한 함수는 데이터를 표준화하여 일치 프로세스를 최적화하는 데 사용됩니다. 여러 정리 함수를 함께 사용하면 복잡한 필터링과 표준화를 수행할 수 있습니다.

제공되지 않는 교차 참조

기본 개체 레코드의 BVT(최선의 진실, Best Version of the Truth)에 제공되지 않는 교차 참조(XREF) 레코드입니다. 따라서 교차 참조 레코드의 값은 기본 개체 레코드에 표시되지 않습니다. 상태 사용 레코드에만 해당됩니다.

제약 조건 테이블

고유 또는 외래 키 제약 조건이 정의된 데이터베이스 테이블입니다.

조건부 매핑

SQL WHERE 절을 사용하여 랜딩 테이블의 레코드 중 필터 조건을 충족하는 레코드만을 조건부로 선택하는 랜딩 테이블과 준비 테이블 간의 열 매핑입니다.

조정

지정된 항목에 대해 Informatica MDM Hub는 하나 이상의 소스 시스템에서 데이터를 가져온 후 "여러 버전의 진실"을 조정하여 해당 항목에 대한 BVT(최선의 진실), 즉 마스터 레코드를 생성합니다. 조정에는 기본 개체의 레코드 일치 및 통합 프로세스를 최적화하도록 데이터를 미리 정리하는 과정이 포함될 수 있습니다.

[distribution: 분포 페이지 710](#)를 참조하십시오.

존속

Informatica MDM Hub가 두 레코드에서 병합할 셀을 평가하여 결정하는 사항입니다. Informatica MDM Hub에서는 존속해야 하는 셀 데이터와 삭제해야 하는 셀 데이터를 결정합니다. 존속은 트러스트가 활성화된 열과 트러스트가 활성화되지 않은 열 모두에 적용됩니다. 서로 다른 두 레코드의 셀을 비교할 때 Informatica MDM Hub에

서는 데이터의 속성을 기반으로 존속 여부를 결정합니다. 예를 들어 두 열에 트러스트가 활성화된 경우 트러스트 점수가 가장 높은 셀이 우선합니다. 트러스트 점수가 같은 경우에는 `LAST_UPDATE_DATE`가 가장 최신인 셀이 우선합니다. `LAST_UPDATE_DATE`가 같은 경우 Informatica MDM Hub에서는 다른 조건을 사용하여 존속 여부를 결정합니다.

존속 셀 데이터

두 레코드에서 병합할 셀을 평가할 때 Informatica MDM Hub에서는 존속되어야 하는 셀 데이터와 삭제되어야 하는 셀 데이터를 확인합니다. 존속 셀 데이터(또는 우선 셀)는 두 셀 중에서 BVT(최선의 진실, Best Version of the Truth)를 보다 잘 나타내는 것으로 간주됩니다. 최종적으로 통합된 단일 레코드는 최상의 존속 셀 데이터를 포함하며 BVT를 나타냅니다.

준비 테이블

정리된 데이터가 로드 작업을 통해 기본 개체로 로드하기 전에 일시적으로 저장되는 테이블입니다.

준비 프로세스

랜딩 테이블의 데이터를 읽고 구성된 모든 정리를 수행하며 정리된 데이터를 해당 준비 테이블로 옮기는 프로세스입니다. 델타 검색을 활성화한 경우에는 Informatica MDM Hub가 새로운 레코드나 변경된 레코드만 처리합니다.

참조 무결성

구성된 외래 키 관계를 기준으로 테이블 간 적용된 상위-하위 관계 규칙입니다.

채우기

일치 중인 레코드의 데이터에 대한 특정 특성을 정의합니다. 기본적으로 Informatica MDM Hub는 영어로 표기되지만 Informatica는 국가별 표준 인구집단을 제공합니다. 인구집단은 이름, 주소 및 기타 식별 데이터에 존재할 수 있는 불가피한 변형 및 오류를 처리하고 Informatica MDM Hub가 일치 토큰을 작성하는 방법을 지정하며, 검색 전략 및 일치 목적이 일치시킬 데이터 채우기에 대해 작동하는 방법도 지정합니다. 유사 항목 일치/검색 전략에서만 사용됩니다.

처리 서버

정리, 일치 및 일괄 작업을 수행하는 서버입니다. 처리 서버는 응용 프로그램 서버 환경에 배포됩니다. 처리 서버는 Trillium Director와 같은 정리 엔진과 상호 작용하여 데이터를 표준화합니다. 처리 서버는 각 인스턴스에서 여러 요청을 동시에 처리할 수 있는 다중 스레드입니다.

최대 트러스트

값이 변경된 경우 데이터 값에 포함되는 트러스트 수준입니다. 예를 들어 소스 시스템 A에서 전화 번호 필드가 555-1234에서 555-4321로 변경된 경우 새 값에는 전화 번호 필드에 대한 시스템 A의 최대 트러스트 수준이 지정됩니다. 최대 트러스트 수준을 상대적으로 높게 설정하면 일반적으로 소스 시스템의 변경 내용을 기본 개체에 적용할 수 있습니다.

최소 트러스트

붕괴 기간이 경과한 후 데이터 값이 "구식"이 되었을 때 데이터 값에 포함되는 트러스트 수준입니다. 이 값은 최대 트러스트보다 작거나 같아야 합니다. 최대 트러스트와 최소 트러스트가 같은 경우 붕괴 곡선은 일직선이며 붕괴 기간과 붕괴 유형에 아무 영향을 주지 않습니다. [decay period: 붕괴 기간 페이지 709](#)을 참조하십시오.

추적 가능성

통합된 레코드에 데이터를 제공하는 시스템 및 해당 시스템의 레코드를 확인할 수 있도록 데이터를 유지 관리하는 기능입니다.

콘텐츠 메타데이터

Informatica MDM Hub에서 처리된 비즈니스 데이터를 설명하는 데이터입니다. 콘텐츠 메타데이터는 기본 개체의 지원 테이블(교차 참조 테이블, 기록 테이블 등 포함)에 저장됩니다. 콘텐츠 메타데이터를 사용하면 기본 개체의 데이터를 가져온 출처와 시간에 따른 데이터 변경 내역을 쉽게 확인할 수 있습니다.

쿼리

Hub 저장소에서 데이터를 검색하는 요청입니다. Informatica MDM Hub를 통해 관리자는 해당 데이터를 검색하는 데 사용되는 기준을 지정할 수 있습니다. 선택한 열을 반환하고 WHERE 절을 포함한 결과 집합을 필터링하며 복잡한 쿼리 구문(예: GROUP BY, ORDER BY 및 HAVING 절)을 사용하고 집계 함수(예: SUM, COUNT 및 AVG)를 사용하도록 쿼리를 구성할 수 있습니다.

쿼리 그룹

쿼리의 논리 그룹으로, 쿼리 그룹은 쿼리를 구성하는 메커니즘입니다. [쿼리 페이지 733](#)를 참조하십시오.

테이블

데이터베이스에서 행(레코드) 및 열로 구성된 데이터 컬렉션입니다. 테이블은 개체에 해당하는 값의 2차원 집합으로 볼 수 있습니다. 테이블의 열은 개체의 특성을 나타내며 행은 개체의 인스턴스를 나타냅니다. Hub 저장소에서 마스터 데이터베이스와 각 연산 참조 저장소(연산 참조 저장소)는 테이블의 컬렉션을 나타냅니다. 기본 개체는 연산 참조 저장소에 테이블로 저장됩니다.

토큰화 프로세스

일치 비교가 수행되기 전에 수행되는 특수한 형식의 데이터 표준화입니다. 대부분의 기본 일치 유형의 경우에는 토큰화 시 단순히 공백 및 구두점 같은 "노이즈" 문자가 제거됩니다. 가장 복잡한 일치 유형의 경우에는 필요한 유사 정도에 따라 정교한 일치 코드(비교할 데이터의 내용을 나타내는 문자열)가 생성됩니다.

통합된 레코드

[master record: 마스터 레코드 페이지 716](#)를 참조하십시오.

통합 표시기

기본 개체에 있는 레코드의 통합 상태를 나타냅니다. CONSOLIDATION_IND 열에 저장되어 있습니다.

통합 프로세스

중복 레코드를 단일 레코드에 병합 또는 연결하는 프로세스입니다. Informatica MDM Hub의 목적은 모든 중복 데이터를 확인하여 제거하고, 이를 통합된 단일 레코드로 병합하거나 연결하면서 추적 가능성을 완전하게 유지 관리하는 것입니다.

트랜잭션 데이터

응용 프로그램에서 수행되는 작업을 나타내며, 일반적으로 응용 프로그램에서 정상적인 작업의 일부로 캡처되거나 생성됩니다. 일반적으로 트랜잭션 데이터는 한 레코드 시스템에서만 유지 관리되며, 대체로 해당 컨텍스트에서 정확하고 신뢰할 수 있는 것으로 간주됩니다. 예를 들어 은행에는 대개 당좌 예금 계좌에서 이루어지는 출금, 입금 및 이체로 인한 트랜잭션 데이터를 관리하기 위한 단 하나의 응용 프로그램이 있습니다.

트러스트

소스 시스템, 변경 기록 및 기타 비즈니스 규칙에 따라 각 셀과 연관된 신뢰도를 측정하는 메커니즘입니다. 트러스트는 데이터 보존 기간, 시간에 따라 신뢰성이 약화되는 정도 및 데이터의 유효성을 고려합니다.

트러스트 수준

Informatica MDM Hub에 레코드를 제공하는 소스 시스템에 대해 다른 소스 시스템에 상대적인 신뢰도 수준을 할당하는 0에서 100 사이의 수입니다. 트러스트 수준은 다른 소스 시스템의 트러스트 수준과 비교할 때만 의미가 있습니다.

트러스트 점수

특정 레코드의 현재 신뢰도 수준입니다. Informatica MDM Hub는 로드 작업 중에 각 레코드의 트러스트 점수를 계산합니다. 기본 개체에 대해 유효성 검사 규칙이 정의되어 있으면 로드 작업 중 해당 유효성 검사 규칙이 데이터에 적용되어 트러스트 점수가 추가로 다운그레이드될 수 있습니다. 통합 프로세스 중에는 두 레코드가 병합 또는 링크 후보일 경우 트러스트 점수가 더 높은 레코드의 값이 우선합니다. 데이터 스튜어드는 병합 관리자 도구에서 트러스트 점수를 수동으로 재정의할 수 있습니다.

트리 병합 해제

병합된 기본 개체 레코드의 트리를 원래 하위 구조로 병합 해제합니다. 병합 해제된 기본 개체 레코드를 루트로 하는 하위 트리는 원래 병합 트리 구조에서 제거됩니다. 예를 들어 a1과 a2를 a로 병합한 다음 b1과 b2를 b로 병합하고 마지막으로 a와 b를 c로 병합한 경우, a에 대해 트리 병합 해제를 수행하고 a1에서 a를 병합 해제하면 a2는 원래 트리 c에서 제거되는 하위 트리가 됩니다. 따라서 병합 해제 후 a는 트리의 루트가 됩니다.

파밍 배포

응용 프로그램을 JBoss 클러스터에 배포하는 배포 유형입니다. 응용 프로그램 보관 파일을 클러스터 멤버 중 하나의 팜 디렉터리에 배포하면 응용 프로그램이 동일한 클러스터의 모든 노드에 자동으로 복제됩니다.

패키지

Informatica MDM Hub에서 패키지¹는 하나 이상의 기본 테이블에 대한 공용 뷰입니다. 패키지는 이러한 테이블과 이러한 테이블에 조인된 다른 테이블에 있는 열의 하위 집합을 나타냅니다. 패키지는 쿼리를 기반으로 합니다. 기본 쿼리에서는 테이블 또는 다른 패키지의 레코드 하위 집합을 선택할 수 있습니다.

프로세스

[비즈니스 프로세스 페이지 725](#)를 참조하십시오.

프로필

계층 관리자에서 HM 사용자가 표시, 편집 또는 추가할 수 있는 필드와 레코드를 설명합니다. 예를 들어 전체 항목 및 관계에 대한 모든 읽기/쓰기 권한을 허용하는 프로필과 추가 또는 편집 작업은 허용하지 않고 읽기만 허용하는 프로필이 있을 수 있습니다.

항목 유형

계층 관리자에서 항목 유형은 계층 관리자를 사용하여 연결할 수 있는 개체의 종류를 정의합니다. 예를 들어 개인, 조직, 제품, 가정 등이 포함됩니다. 항목 유형이 동일한 모든 항목은 동일한 항목 기본 개체에 저장됩니다. HM 구성 도구에서 항목 유형은 탐색 트리에서 해당 유형이 연결된 항목 개체 아래에 표시됩니다.

행

[레코드 페이지 722](#)를 참조하십시오.

허용 한도

일치 항목의 허용 가능성을 결정하는 수입니다. 허용 한도는 Informatica에 의해 인구집단 내에 일치 목적에 따라 정의됩니다.

활성 상태(레코드)

기본 개체 또는 상호 참조 레코드와 연결된 상태입니다. 교차 참조 레코드 중 하나 이상이 활성 상태인 경우 기본 개체 레코드가 활성 상태입니다. 교차 참조 레코드는 활성 상태인 경우에만 통합된 기본 개체에 데이터를 제공합니다.

기본적으로 활성 레코드는 **MDM Hub** 프로세스에 관여합니다. 이러한 레코드는 모든 작업에 관여할 수 있는 레코드입니다. 레코드가 승인 프로세스를 거쳐야 하는 경우 이러한 레코드는 승인 프로세스를 통해 승인되었음을 나타냅니다.

인덱스

A

ACTION

- 감사 로그 테이블 열 [605](#)
- Address_Part1 키 유형 [392](#)
- AssignTasks 사용자 종료
 - 인터페이스 [579](#)
- AUTOMERGE_IND
 - 외부 일치 출력 테이블 열 [548](#)

B

- BMG(일치 그룹 빌드) [277](#)
- BPM 워크플로우 도구 [170](#)
- build_war 매크로 [589](#)
- BVT 스냅샷 작업 [547](#)
- BVT(최선의 진실, Best Version of the Truth)
 - 정보 [280](#)
- 패키지
 - PUT 사용 [135](#)
 - 개요 [122](#)
 - 날기 가능 [135](#)
 - 병합 [135](#)
 - 삭제 [138](#)
 - 스키마 변경 [124](#)
 - 업데이트 패키지, 정보 [135](#)
 - 정보 [135](#), [211](#)
 - 조인 쿼리 [138](#)
 - 추가 [136](#)
 - 쿼리 변경 후 새로 고침 [137](#)
 - 패키지 도구 [123](#)
 - 편집 [137](#)
 - 표시 패키지, 정보 [135](#)
- 패키지 도구 [123](#)
- 포괄적 검색 수준 [398](#)
- 표시 이름
 - 열 속성 [108](#)
- 표준 검색 수준 [398](#)
- 표준 키 너비 [393](#)
- 프로비저닝 도구
 - provisioning.log [648](#)
- 프로세스 보기
 - 정보 [33](#)
- 프로필
 - 복사 [217](#)
 - 삭제 [217](#)
 - 유효성 검사 [216](#)
 - 추가 [215](#)
 - 프로필 정보 [215](#)
- 필드 일치 목적 [406](#)
- 필터
 - 삭제 [386](#)
 - 속성 [385](#)
 - 추가 [385](#)

필터 (계속)

- 편집 [386](#)
- 필터 정보 [385](#)
- 필터링된 일치 규칙 [274](#), [403](#)
- 한정된 검색 수준 [398](#)
- 항목
 - 정보 [195](#)
 - 표시 옵션 [201](#)
- 항목 기본 개체
 - 기본 개체로 되돌리기 [202](#)
 - 기본 개체에서 변환 [197](#)
 - 작성 [196](#)
 - 정보 [195](#)
- 항목 아이콘
 - 구성 [194](#)
 - 기본값 업로드 [193](#)
 - 삭제 [194](#)
 - 추가 [194](#)
 - 편집 [194](#)
- 항목 유형
 - HM 패키지 할당 [214](#)
 - 삭제 [201](#)
 - 생성 [200](#)
 - 예제 [199](#)
 - 정보 [198](#)
 - 편집 [201](#)
- 행 ID별 로드 [303](#)
- 행 수준 잠금
 - 구성 [644](#)
 - 기본 동작 [643](#)
 - 대기 시간 [645](#)
 - 사용 시 고려 사항 [644](#)
 - 행 수준 잠금 정보 [643](#)
 - 활성화 [644](#)
- 확장된 키 너비 [393](#)
- 환경 보고서
 - 저장 [640](#)
- 활성 상태, 정보 [167](#)

C

- C_REPOS_AUDIT
 - 감사 로그 테이블 [605](#)
- C_REPOS_DB_RELEASE 테이블
 - 열 [638](#)
- C_REPOS_EXT_HARD_DEL_DETECT
 - 기본 키 소스 열 [318](#)
- C_REPOS_SYSTEM 테이블
 - 교차 참조 테이블에서 ROWID_SYSTEM에 의해 참조됨 [99](#)
- CM_DIRTY_IND
 - 기본 개체 열 [95](#)
- cmxcleanse.properties
 - 정보 [627](#)

cmxserver.log
 설명 [647](#), [648](#)
cmxserver.properties
 정보 [608](#)
COMPONENT
 감사 로그 테이블 열 [605](#)
console.log
 설명 [647](#)
CONSOLIDATION_IND
 기본 개체 열 [95](#)
CONTEXT
 감사 로그 테이블 열 [605](#)
Corporate_Entity 일치 목적 [406](#)
CREATE_DATE
 감사 로그 테이블 열 [605](#)
 교차 참조 테이블 열 [99](#)
 기본 개체 열 [95](#)
 외부 일치 출력 테이블 열 [548](#)
 준비 테이블 열 [353](#)
CREATOR
 감사 로그 테이블 열 [605](#)
 교차 참조 테이블 열 [99](#)
 기본 개체 열 [95](#)
 외부 일치 출력 테이블 열 [548](#)
 준비 테이블 열 [353](#)

D

DATA_XML
 감사 로그 테이블 열
 CONTENT_XML
 감사 로그 테이블 열 [605](#)
DataEncryptor
 구현 [181](#)
 인터페이스 [181](#)
DELETED_BY
 교차 참조 테이블 열 [99](#)
 기본 개체 열 [95](#)
 준비 테이블 열 [353](#)
DELETED_DATE
 교차 참조 테이블 열 [99](#)
 기본 개체 열 [95](#)
 준비 테이블 열 [353](#)
DELETED_IND
 교차 참조 테이블 열 [99](#)
 기본 개체 열 [95](#)
 준비 테이블 열 [353](#)

E

Elasticsearch
 고가용성 [456](#)
 인덱스 설정 [465](#)
elasticsearch 보관 파일
 추출 [457](#)
Elasticsearch 설치
 선행 조건 [456](#)
 설치 전 태스크 [456](#)
EMO 테이블 참조 외부 일치 출력 테이블
Entity 360 보기
 entity360view.log [648](#)
entity360view.log
 설명 [648](#)

F

FILE_NAME
 외부 일치 출력 테이블 열 [548](#)
FROM_SYSTEM
 감사 로그 테이블 열 [605](#)

G

GBID
 열 속성 [108](#)
GBID 열 [110](#)
GBID(글로벌 식별자) 열 [110](#)
GetAssignableUsersForTask 사용자 종료
 인터페이스 [580](#)
GOV
 교차 참조 테이블 열 [99](#)

H

HM 패키지
 삭제 [212](#)
 편집 [212](#)
 항목 유형에 할당 [214](#)
HPCT 테이블 참조 보류 중인 제어 테이블
HTTPS
 처리 서버용 [331](#)
Hub 상태 참조 레코드 상태
Hub 서버
 cmxserver.log [647](#)
 데이터 수집 비활성화 [653](#)
 데이터 수집 활성화 [653](#)
 로그 설정 [647](#)
 속성 [608](#)
Hub 저장소
 ORS(연산 레코드 저장소) [60](#)
 마스터 데이터베이스 [60](#)
 속성 [68](#)
 스키마 [75](#)
 테이블 유형 [76](#)
Hub 콘솔
 시작 [32](#)
HUB_STATE_IND
 교차 참조 테이블 열 [99](#)
 준비 테이블 열 [353](#)
HUB_STATE_IND 열
 정보 [168](#)

I

IBM DB2
 데이터 유형 [107](#)
IDL(초기 데이터 로드) [261](#)
Informatica Data Director 기본 검색
 누락된 하위 레코드 허용 [387](#)
Informatica Data Quality
 일괄 작업 및 매핑 [296](#)
Informatica Platform
 informatica-mdm-platform.log [648](#)
Informatica Platform 준비
 개요 [655](#)
Informatica 플랫폼 준비
 개요 [260](#)

informatica-mdm-platform.log

설명 [648](#)

INTERACTION_ID

감사 로그 테이블 열 [605](#)

교차 참조 테이블 열 [99](#)

INTERACTION_ID 열

정보 [169](#)

J

JAR 파일

ORS 관련, 다운로드 [589](#)

ORS 관련, 사용 [590](#)

사용자 종료 JAR 파일 구현 [567](#)

사용자 종료용 [567](#)

JAR(Java 보관) 파일

tools.jar [593](#)

Java 사용자 종료

정보 [565](#)

JMS 이벤트 스키마 관리자

동기화되지 않은 개체 찾기 [593](#)

동기화되지 않은 개체에 대한 자동 검색 [594](#)

시작 [592](#)

정보 [591](#)

L

LAST_ROWID_SYSTEM

기본 개체 열 [95](#)

LAST_UPDATE_DATE

감사 로그 테이블 열 [605](#)

교차 참조 테이블 열 [99](#)

기본 개체 열 [95](#)

준비 테이블 열 [353](#)

log4j-entity360view.xml

설명 [648](#)

M

MDM Hub

Informatica 플랫폼 통합 [656](#)

로그 [646](#)

MDM Hub 준비

개요 [259](#), [289](#)

MDM Hub 콘솔

로그인 [37](#)

정보 [32](#)

MDM Hub 콘솔 인터페이스

도구 모음 [40](#)

마법사 시작 화면 [40](#)

빠른 실행 탭 [40](#)

사용자 지정 [40](#)

일반 탭 [40](#)

창 크기 및 위치 [40](#)

Microsoft SQL Server

데이터 유형 [107](#)

사용자 지정 인덱스 제한 [105](#)

행 너비 제한 [107](#)

Multidomain MDM

설치 세부 정보 [40](#)

N

Null 가능

열 속성 [108](#)

null 값

열에 null 값 허용 [108](#)

Null 값 적용

열 속성 [108](#)

null 업데이트 허용

준비 테이블 열에 대해 [356](#)

null 외래 키 허용

준비 테이블 열에서 [356](#)

NULL 일치 [412](#)

널기 가능

열 속성 [108](#)

널기 가능 패키지 [135](#)

누락된 하위 레코드

정보 [387](#)

다중 병합 작업 [561](#)

대상 데이터베이스

변경 [37](#)

선택 [32](#)

데이터

손상됨, 문제 해결 [47](#)

이해 [373](#)

데이터 로드

IDL(초기 데이터 로드) [261](#)

중분 로드 [261](#)

데이터 소스

JDBC 데이터 소스 [73](#)

데이터 소스 정보 [73](#)

제거 [74](#)

데이터 수집

Hub 서버 [653](#)

MDM Hub 환경 정보 [652](#)

시스템 구성 정보 [652](#)

정보 [651](#)

처리 서버 [653](#), [654](#)

데이터 스튜어드 작업 영역

정보 [42](#)

데이터 암호화

Hub 서버 구성 [182](#)

개념 [179](#)

개요 [179](#)

구성 [181](#)

샘플 속성 파일 [184](#)

속성 파일 [182](#)

아키텍처 [179](#)

유틸리티 [180](#)

제한 [180](#)

지원되는 API 요청 [183](#)

처리 서버 구성 [183](#)

데이터 액세스

계층 관리자 [212](#)

데이터 유형

열 속성 [108](#)

정보 [107](#)

데이터 정리

Informatica 플랫폼에서 [346](#)

MDM Hub의 [327](#)

데이터 정리 정보 [327](#)

설정 태스크 [327](#), [346](#)

유니코드 설정 [47](#)

처리 서버 [328](#)

데이터베이스

대상 데이터베이스 [32](#)

데이터베이스 ID [68](#)

데이터베이스 (계속)

- 유니코드, 구성 [44](#)
- 데이터베이스 개체 이름, 제약 조건 [78](#)
- 데이터베이스 도구
 - 데이터베이스 도구 정보 [62](#)
- 데이터베이스 속성
 - 연산 참조 저장소 [638](#)
- 델타 검색
 - 구성 [305](#)
 - 랜딩 테이블 구성 [287](#)
 - 사용 시 고려 사항 [306](#)
 - 처리 방식 [306](#)
- 도구
 - 데이터베이스 도구 [63](#)
 - 도구 액세스 도구 [52](#)
 - 매핑 도구 [564](#)
 - 병합 관리자 도구 [278](#)
 - 사용자 도구 [51](#)
 - 사용자 액세스 권한 [51](#)
 - 스키마 관리자 [93](#)
 - 일괄 처리 뷰어 도구 [526](#)
 - 정리 함수 도구 [333](#)
- 도구 액세스 도구 [52](#)
- 동기화 작업 [365](#), [564](#)
- 랜드 프로세스
 - C_REPOS_SYSTEM 테이블 [284](#)
 - ETL(추출-변환-로드) 도구 [258](#)
 - 개요 [257](#)
 - 관리 [258](#)
 - 랜딩 테이블 [257](#)
 - 랜딩 테이블을 채우는 방식 [258](#)
 - 소스 시스템 [257](#)
 - 실시간 처리(API 호출) [258](#)
 - 외부 일괄 처리 [258](#)
- 랜딩 테이블
 - 랜딩 테이블 정보 [286](#)
 - 속성 [287](#)
 - 열 [286](#)
 - 정보 [290](#)
 - 정의됨 [76](#)
 - 제거 [288](#)
 - 추가 [287](#)
 - 편집 [288](#)
- 레코드
 - 업데이트 [158](#)
 - 일치 보류 [171](#)
 - 편집 [158](#)
- 레코드 버전
 - 업데이트 [158](#)
 - 추가 [157](#)
 - 편집 [158](#)
- 레코드 버전 추가
 - 예제 [158](#)
- 레코드 상태
 - 정보 [167](#)
- 로그 파일 롤링
 - 정보 [641](#)
- 로그, 응용 프로그램 서버
 - 구성 [642](#)
 - 수준 [641](#)
- 로그, 응용 프로그램 서버 데이터베이스
 - 정보 [640](#)
- 로드 시 일치 토큰 생성 [555](#)
- 로드 일괄
 - 시간 표시 막대 설정 [154](#)
 - 여러 레코드 버전 [154](#)
 - 예제 [154](#)

- 로드 작업
 - 강제된 업데이트, 정보 [554](#)
 - 거부된 레코드 [532](#)
 - 로드 시 일치 토큰 생성 [555](#)
 - 로드 일괄 처리 크기 [102](#)
- 로드 프로세스
 - 개요 [260](#)
 - 데이터 관리 단계 [263](#)
 - 사용자 종료 [572](#)
 - 삼입형 로드 [263](#)
- 마스터 데이터베이스
 - 생성 [61](#)
 - 암호 변경 [71](#)
- 매핑
 - Informatica Data Quality 일괄 작업 [296](#)
 - 고유 매핑 [302](#)
 - 관리 [309](#)
 - 구성 [295](#)
 - 다이어그램 [297](#)
 - 속성 [292](#)
 - 스키마로 이동 [309](#)
 - 실행 [699](#)
 - 열 [301](#)
 - 정리 [296](#)
 - 제거 [310](#)
 - 조건부 매핑 [302](#)
 - 준비 테이블과 랜딩 테이블 간 [76](#)
 - 추가 [298](#)
 - 쿼리 매개 변수 [303](#)
 - 테스트 [310](#)
 - 통과 [296](#)
 - 편집 [309](#)
 - 행 ID별 로드 [303](#)
- 매핑 도구 [297](#), [564](#)
- 메시지
 - 메시지 필드 [506](#)
 - 예
 - AmRule 메시지 [495](#)
 - BoDelete 메시지 [495](#)
 - BoSetToDelete 메시지 [496](#)
 - PendingInsert 메시지 [500](#)
 - PendingUpdate 메시지 [501](#)
 - PendingUpdateXref 메시지 [502](#)
 - XREF 업데이트 메시지 [504](#)
 - XRefDelete 메시지 [504](#)
 - XRefSetToDelete 메시지 [505](#)
 - 고유 항목으로 허용 메시지 [494](#)
 - 병합 메시지 [498](#), [510](#)
 - 병합 업데이트 메시지 [499](#)
 - 병합 해제 메시지 [502](#)
 - 삼입 메시지 [498](#)
 - 업데이트 메시지 [503](#)
 - 작업 없음 메시지 [500](#)
 - 예(레거시)
 - bo 삭제 메시지 [507](#)
 - XREF 삭제 메시지 [516](#)
 - XREF 업데이트 메시지 [515](#)
 - XREF 업데이트 보류 중 메시지 [513](#)
 - 고유 항목으로 허용 메시지 [507](#)
 - 병합 업데이트 메시지 [511](#)
 - 병합 해제 메시지 [515](#)
 - 삭제 메시지 [497](#), [509](#)
 - 삭제할 bo 집합 메시지 [508](#)
 - 삭제할 XREF 집합 메시지 [517](#)
 - 삼입 메시지 [510](#)
 - 삼입 보류 중 메시지 [512](#)
 - 업데이트 메시지 [514](#)

- 메시지 (계속)
 - 예(레거시) (계속)
 - 업데이트 보류 중 메시지 [512](#)
 - 요소 [492](#)
 - 필터링 [494](#), [506](#)
- 메시지 대기열
 - 감사 [603](#)
 - 메시지 대기열 도구 [482](#)
 - 메시지 대기열 정보 [281](#), [485](#)
 - 메시지 확인 간격 [482](#)
 - 삭제 [486](#)
 - 상태 [482](#)
 - 속성 [485](#)
 - 수신 일괄 처리 크기 [482](#)
 - 수신 제한 시간 [482](#)
 - 추가 [485](#)
 - 편집 [486](#)
- 메시지 대기열 서버
 - 메시지 대기열 서버 정보 [483](#)
 - 삭제 [485](#)
 - 추가 [484](#)
 - 편집 [484](#)
- 메시지 스키마
 - ORS 관련, 생성 및 배포 [593](#)
- 메시지 트리거
 - 메시지 트리거 정보 [487](#)
 - 보류 중인 업데이트에 대해 활성화 [176](#)
 - 삭제 [491](#)
 - 상태 변경 활성화 [176](#)
 - 유형 [488](#)
 - 정보 [280](#)
 - 추가 [490](#)
 - 편집 [491](#)
- 메타데이터
 - 동기화 [113](#)
 - 트러스트 [113](#)
- 메타데이터 동기화 [113](#)
- 명령 단추
 - 개체 속성 편집 [39](#), [40](#)
 - 개체 제거 [39](#), [40](#)
 - 개체 추가 [39](#), [40](#)
- 모델 리포지토리
 - 동기화 [674](#)
- 모델 작업 영역
 - 정보 [41](#)
- 목적, 일치 [404](#)
- 문제 해결
 - cmxserver.log 파일 [647](#)
- 배타적 잠금
 - 정보 [35](#)
 - 획득 [36](#)
- 변경할 수 없는 rowid 개체 [475](#)
- 병렬도 [102](#)
- 병합
 - 수동 병합 작업 [556](#)
 - 수동 병합 해제 작업 [557](#)
 - 일괄 병합 해제 [546](#)
- 병합 관리자 도구 [278](#)
- 병합 패키지 [135](#)
- 병합 프로세스
 - 사용자 종료 [576](#)
 - 정보 [475](#)
- 병합 해제
 - 계단식 병합 해제 [476](#)
 - 상위 항목이 병합 해제된 경우 하위 항목 병합 해제 [476](#)
 - 수동 병합 해제 [557](#)
 - 일괄 병합 해제 [546](#)
- 병합 해제 프로세스
 - 사용자 종료 [577](#)
 - 보류 레코드, 일치 활성화 [171](#)
 - 보류 중 상태, 정보 [167](#)
 - 보류 중인 제어 테이블
 - 정보 [480](#)
 - 보안 액세스 관리자 작업 영역
 - 정보 [42](#)
 - 보안 통신
 - 활성화, 처리 서버 [331](#)
 - 부서 일치 목적 [406](#)
 - 분석기
 - 문자 필터
 - 토큰나이저 [465](#)
 - 토큰 필터 [465](#)
- 붕괴 곡선 [362](#)
- 붕괴 유형
 - RISL [362](#)
 - SIRL [362](#)
 - 선형 [362](#)
- 사용자
 - 도구 액세스 [51](#)
- 사용자 개체
 - 정보 [595](#)
- 사용자 개체 레지스트리
 - 사용자 종료, 보기 [596](#)
 - 사용자 지정 Java 정리 함수, 보기 [597](#)
 - 사용자 지정 단추 함수, 보기 [597](#)
 - 시작 [596](#)
 - 정보 [595](#)
- 사용자 정의 열
 - 준비 테이블 [353](#)
- 사용자 종료
 - AssignTasks 인터페이스 [579](#)
 - GetAssignableUsersForTask 인터페이스 [580](#)
 - JAR 파일 [567](#)
 - JAR 파일 구현 [567](#)
 - SIF API 호출 [580](#)
 - UserExitContext 클래스 [568](#)
 - 로드 프로세스 [572](#)
 - 롤백 [566](#)
 - 병합 프로세스 [576](#)
 - 병합 해제 프로세스 [577](#)
 - 보기 [596](#)
 - 사용자 종료
 - 사후 로드 매개 변수 [573](#)
 - 사전 병합 [575](#)
 - 사전 병합 해제 인터페이스 [578](#)
 - 사전 일치 인터페이스 [575](#)
 - 사전 준비 [571](#)
 - 사전 준비 인터페이스 [571](#)
 - 사후 랜딩 [570](#)
 - 사후 랜딩 사용자 종료 인터페이스 [570](#)
 - 사후 로드 [573](#)
 - 사후 로드 사용자 종료
 - 매개 변수 [573](#)
 - 사후 로드 인터페이스 [573](#)
 - 사후 병합 [576](#)
 - 사후 병합 인터페이스 [576](#)
 - 사후 병합 해제 [578](#)
 - 사후 병합 해제 인터페이스 [578](#)
 - 사후 일치 [575](#)
 - 사후 일치 인터페이스 [575](#)
 - 사후 준비 [571](#)
 - 사후 준비 인터페이스 [572](#)
 - 생성 [580](#)
 - 업로드 [567](#)

사용자 종료 (계속)

- 예 [581](#)
- 일치 프로세스 [574](#)
- 정보 [565](#), [595](#), [596](#)
- 제거 [567](#)
- 준비 프로세스 [569](#)
- 지원되는 SIF API [582](#)
- 지침 [583](#)
- 처리 [566](#)
- 태스크 관리 [579](#)

사용자 지정 Java 정리 함수

- 보기 [596](#), [597](#)
- 정보 [595](#)

사용자 지정 단추

- 나열 [57](#)
- 모양 [54](#)
- 배포 [57](#)
- 사용자 지정 단추 정보 [53](#)
- 사용자 지정 함수, 쓰기 [54](#)
- 속성 파일 [57](#)
- 아이콘 [57](#)
- 업데이트 [57](#)
- 예 [55](#)
- 유형 변경 [57](#)
- 추가 [57](#)
- 클릭 [54](#)
- 텍스트 레이블 [57](#)

사용자 지정 단추 함수

- 등록 [597](#)
- 보기 [597](#)
- 정보 [595](#)

사용자 지정 인덱스

- MDM Hub 외부에서 생성 [106](#)
- 노드로 이동 [105](#)
- 삭제 [105](#)
- 생성 [105](#)
- 정보 [105](#)
- 편집 [105](#)

사용자 지정 쿼리

- SQL 구문 [133](#)
- SQL 유효성 검사 [134](#)
- 삭제 [132](#)
- 정보 [133](#)
- 추가 [134](#)
- 쿼리 도구 [123](#)
- 편집 [135](#)

사용자 지정 함수

- 삭제 [57](#)
- 서버 기반 [55](#)
- 쓰기 [54](#)
- 클라이언트 기반 [55](#)

사전 병합 사용자 종료

- 정보 [575](#)

사전 병합 해제 사용자 종료

- 인터페이스 [578](#)

사전 일치 사용자 종료

- 인터페이스 [575](#)

사전 준비 사용자 종료

- 인터페이스 [571](#)
- 정보 [571](#)

사후 랜딩 사용자 종료

- 인터페이스 [570](#)
- 정보 [570](#)

사후 로드 사용자 종료

- 인터페이스 [573](#)
- 정보 [573](#)

사후 병합 사용자 종료

- 인터페이스 [576](#)
- 정보 [576](#)

사후 병합 해제 사용자 종료

- 인터페이스 [578](#)
- 정보 [578](#)

사후 일치 사용자 종료

- 인터페이스 [575](#)
- 정보 [575](#)

사후 준비 사용자 종료

- 인터페이스 [572](#)
- 정보 [571](#)

삭제됨 상태, 정보

- 상수 [339](#)

상위 항목 병합 시 대기열에 다시 넣기

- [102](#)

상태 관리

- HUB_STATE_IND 열 [168](#)
- INTERACTION_ID 열 [169](#)
- 기본 개체 레코드 존속 [170](#)
- 데이터 로드 규칙 [170](#)
- 레코드 상태
- 상태 전환 [169](#)
- 레코드 승격 [176](#), [177](#)
- 로드 작업 [552](#)
- 메시지 트리거, 활성화 [176](#)
- 상태 변환 규칙, 정보 [169](#)
- 시간 표시 막대 [144](#)
- 일치 작업 [558](#)
- 정보 [166](#), [167](#)
- 활성화 [171](#)

상태 관리 재정의 시스템

- 로드 작업 [554](#)

상태, 레코드

- [167](#)

상태, 설정

- [528](#)

새 쿼리 마법사

- [134](#)

샌드박스

- [216](#)

샘플 응용 프로그램

- 데이터 암호화용 [180](#)

생성

- 로드 시 일치 토큰, 상위 항목 병합 시 대기열에 다시 넣기 [102](#)

선형 붕괴

- [362](#)

세그먼트 일치

- [417](#)

셀 업데이트

- [356](#)

소스 시스템

- 고유 소스 시스템 [476](#)
- 관리 소스 시스템 [284](#)
- 변경할 수 없는 소스 시스템 [475](#)
- 소스 시스템 정보 [283](#)
- 속성 [285](#)
- 시스템 리포지토리 테이블(C_REPOS_SYSTEM) [284](#)
- 시스템 및 트러스트 도구, 시작 [284](#)
- 이름 바꾸기 [285](#)
- 제거 [286](#)
- 최상위 예약된 키 [355](#)
- 추가 [285](#)
- 키 유지 [355](#)

소스 시스템 키 유지

- [355](#)

손상된 데이터

- 문제 해결 [47](#)

수동

- 수동 병합 작업 [556](#)
- 수동 병합 해제 작업 [557](#)
- 일괄 작업 [527](#)

스키마

- 변경 내용은 쿼리 및 패키지에 영향을 미침 [124](#)
- 스키마 정보 [75](#)
- 스키마 개체 [78](#)

- 스키마 관리자
 - 기본 개체 [94](#)
 - 시작 [93](#)
 - 외래 키 관계 [115](#)
 - 테이블에 열 추가 [107](#)
- 스키마 뷰어
 - JPG로 저장 [120](#)
 - 개요 창 [118](#)
 - 계층 보기 [119](#)
 - 다이어그램 창 [118](#)
 - 명령 단추 [118](#)
 - 모두 확대/축소 [119](#)
 - 방향 [120](#)
 - 보기 전환 [119](#)
 - 시작 [118](#)
 - 열 이름 [120](#)
 - 옴션 [120](#)
 - 인쇄 [121](#)
 - 작각 보기 [119](#)
 - 창 [118](#)
 - 축소 [119](#)
 - 컨텍스트 메뉴 [119](#)
 - 확대 [119](#)
- 스키마 일치 열 [563](#)
- 스키마 트러스트 열 [564](#)
- 승격 작업 [561](#), [645](#)
- 시간 표시 막대
 - 개요 [139](#)
 - 구성 [153](#)
 - 규칙 [144](#)
 - 로드 일괄 [154](#)
 - 상태 관리 [144](#)
 - 세분성 [143](#)
 - 예제 [140](#)
 - 지침 [140](#)
 - 활성화 [101](#), [104](#), [153](#)
- 시간 표시 막대 규칙
 - 검치는 유효 기간 [144](#)
 - 연속성 [144](#)
- 시간 표시 막대 레코드 버튼
 - 예제 [142](#)
- 시스템 리포지토리 테이블 [284](#)
- 시스템 및 트러스트 도구 [284](#)
- 시스템 상태 *참조 레코드 상태*
- 시스템 열
 - 기본 개체 [95](#)
 - 설명됨 [115](#)
 - 외부 일치 테이블 [548](#)
 - 준비 테이블 [353](#)
- 실제 이름
 - 열 속성 [108](#)
- 쓰기 잠금
 - 정보 [35](#)
 - 획득 [36](#)
- 암호
 - 변경 [37](#)
 - 암호화 [71](#)
- 암호 암호화 [71](#)
- 언어
 - Oracle 환경용 구성 [48](#)
- 엔터프라이즈 관리자
 - 연산 참조 저장소 속성 [638](#)
- 연락처 일치 목적 [406](#)
- 연산 참조 저장소
 - 데이터베이스 속성 [638](#)
 - 등록 [63-66](#)

- 열
 - 데이터 유형 [107](#)
 - 속성 [108](#)
 - 예약된 이름 [78](#)
 - 테이블에 추가 [107](#)
- 영구 삭제 검색
 - C_REPOS_EXT_HARD_DEL_DETECT 테이블 [313](#)
 - 개요 [311](#)
 - 기본 키 소스 열 [318](#)
 - 사용자 종료 내 [326](#)
 - 삭제 플래그 값 [312](#)
 - 영구 삭제 검색 테이블 [313](#)
 - 유형 [312](#)
 - 종료 날짜 [319](#), [322](#)
 - 준비 프로세스 [311](#)
 - 직접 삭제 [318](#), [319](#), [322](#)
 - 트러스트 및 유효성 검사 규칙 [312](#)
 - 합의 삭제 [321](#)
- 오류, 감사 [604](#)
- 외래 키
 - 조회 [265](#), [359](#)
- 외래 키 관계
 - 가상 관계 [115](#)
- 삭제 [117](#)
- 생성 [115](#)
- 정보 [115](#)
- 정의됨 [115](#)
- 지원됨 [523](#)
- 추가 [115](#)
- 편집 [116](#)
- 외래 키 관계 기본 개체
 - 생성 [207](#)
 - 정보 [206](#)
- 외부 일치 작업
 - 실행 [550](#)
 - 외부 일치 작업 정보 [547](#)
 - 입력 테이블 [548](#)
 - 출력 테이블 [548](#)
- 외부 일치 출력 테이블
 - 열 [548](#)
 - 정보 [548](#)
- 외부 일치 테이블
 - 시스템 열 [548](#)
- 용어 [708](#)
- 워크플로우
 - 레코드 상태 [167](#)
- 워크플로우 도구 [170](#)
- 워크플로우 엔진
 - 추가 [172](#)
- 유사 항목 일치
 - 유사 항목 일치 기본 개체 [275](#), [392](#)
 - 유사 항목 일치 열 [389](#)
 - 유사 항목 일치 전략 [378](#)
 - 유사 항목 일치/검색 전략 [403](#)
- 유사 항목 일치 규칙 [274](#)
- 유틸리티 작업 영역
 - 정보 [43](#)
- 유효 기간 늘리기
 - 예제 [156](#)
- 유효 기간 줄이기
 - 예제 [156](#)
- 유효성 검사
 - 열 속성 [108](#)
- 유효성 검사 규칙
 - 규칙 SQL [369](#)
 - 규칙 열 속성 [368](#)
 - 규칙 유형 [368](#)

유효성 검사 규칙 (계속)

- 규칙 이름 [368](#)
- 다운그레이드 백분율 [368](#)
- 도메인 확인 [368](#)
- 사용자 지정 유효성 검사 규칙 [368](#), [370](#)
- 사용자 지정, 예제 [370](#)
- 상태 사용 기본 개체 [366](#)
- 속성 [368](#)
- 실행 시퀀스 [367](#)
- 예 [369](#)
- 유효성 검사 규칙 정보 [365](#)
- 유효성 검사 열 활성화 [367](#)
- 유효성 검사 확인 [366](#)
- 정의 [365](#)
- 정의됨 [365](#)
- 제거 [372](#)
- 존재 확인 [368](#)
- 참조 무결성 [368](#)
- 최소 트러스트 보유 [368](#)
- 추가 [370](#)
- 패턴 유효성 검사 [368](#)
- 편집 [371](#)
- 필요한 열 [366](#)
- 유효성 검사 확인 [366](#)
- 유효성 다시 검사 작업 [563](#)
- 응용 프로그램 서버 로그
 - 구성 [642](#)
 - 로그 파일 롤링 [641](#)
 - 수준 [641](#)
 - 정보 [640](#)
- 인구집단
 - 구성 [45](#)
 - 미국 외 인구집단 [45](#)
 - 선택 [378](#)
- 인덱스
 - 사용자 지정 인덱스 [105](#)
- 인덱스 설정
 - 분석기 [465](#)
- 일괄 그룹
 - 삭제 [536](#)
 - 수준, 구성 [536](#)
 - 실행 [539](#)
 - 추가 [535](#)
 - 편집 [535](#)
- 일괄 상호 운용성 [645](#)
- 일괄 작업
 - BVT 스냅샷 작업 [547](#)
 - 거부된 레코드 [532](#)
 - 구성 [519](#)
 - 구성 가능한 옵션 [528](#)
 - 기록 지우기 [533](#)
 - 다중 병합 작업 [561](#)
 - 다중 스레드 일괄 작업 [520](#)
 - 동기화 작업 [365](#), [564](#)
 - 디자인 고려 사항 [523](#)
 - 로드 작업 [552](#)
 - 명령 단추 [528](#)
 - 변경이 발생하는 경우 [525](#)
 - 불완전 작업 상태로 설정 [528](#)
 - 상태 새로 고침 [528](#)
 - 상태, 설정 [528](#), [542](#)
 - 선택 [527](#)
 - 속성 [527](#)
 - 수동 병합 작업 [556](#)
 - 수동 병합 해제 작업 [557](#)
 - 수동으로 실행 [527](#)
 - 승격 작업 [561](#)

일괄 작업 (계속)

- 실행 [521](#), [528](#)
- 외래 키 관계 및 [523](#)
- 외부 일치 작업 [547](#)
- 유효성 다시 검사 작업 [563](#)
- 일괄 병합 해제 작업 [546](#)
- 일괄 작업 순서 지정 [522](#)
- 일치 분석 작업 [559](#)
- 일치 작업 [557](#)
- 일치 테이블 재설정 작업 [563](#)
- 일치 토큰 생성 작업 [550](#)
- 일치되지 않은 레코드를 고유 레코드로 허용 [544](#)
- 자동 병합 작업 [545](#)
- 자동 일치 및 병합 작업 [545](#)
- 자동으로 생성된 일괄 작업 [524](#)
- 작업 실행 로그 [529](#)
- 작업 실행 상태 [529](#)
- 정보 [519](#)
- 준비 작업 [564](#)
- 중복 데이터에 대한 일치 작업 [560](#)
- 지원 테이블 [522](#)
- 키 일치 작업 [552](#)
- 일괄 작업 기록 지우기 [533](#)
- 일괄 작업 순서 지정 [522](#)
- 일괄 처리 뷰어 도구
 - 정보 [526](#)
- 일반 캐리
 - SQL 문, 보기 [132](#)
 - 발당 블록
 - 비교 조건, 정의 [130](#)
 - 상수, 정의 [129](#)
 - 열, 선택 [128](#)
 - 정렬 순서, 정의 [131](#)
 - 테이블, 선택 [128](#)
 - 함수, 정의 [129](#)
- 삭제 [132](#)
- 정보 [126](#)
- 추가 [126](#)
- 쿼리 도구 [123](#)
- 쿼리 조건 구체화 [127](#)
- 편집 [127](#)
- 일치
 - NULL 일치 [412](#)
 - 같지 않은 일치 [415](#)
 - 누락된 하위 레코드 허용 [387](#)
 - 동적 일치 분석 임계값 설정 [379](#)
 - 보류 레코드 [171](#)
 - 세그먼트 일치 [417](#)
 - 속성
 - 일치 속성 정보 [376](#)
 - 수동 통합의 최대 일치 항목 수 [377](#)
 - 유사 항목 인구집단 [378](#)
 - 유사 항목 일치에 대한 인구집단 [378](#)
 - 이전 rowid 개체 설정과만 일치 [378](#)
 - 인구집단 [45](#)
 - 일치 문자열, 가져오기 [345](#)
 - 일치 분석 작업 [559](#)
 - 일치 시간(분), 최대 경과 시간 [102](#)
 - 일치 출력 문자열, 가져오기 [346](#)
 - 일치 테이블 [558](#)
 - 일치 테이블, 재설정 [563](#)
 - 일치 토큰, PUT 작업 시 생성 [102](#)
 - 일치 하위 유형 [411](#)
 - 일치/검색 전략 [378](#)
 - 일치되지 않은 모든 행을 고유 행으로 허용 [377](#)
 - 전략
 - 유사 항목 일치 [378](#)

일치 (계속)

전략 (계속)

정확히 일치 [378](#)

정리 목록의 문자열 일치 [345](#)

중복 데이터 [276](#)

중복 데이터에 대한 일치 작업 [560](#)

하위 레코드 [418](#)

한 번만 일치 설정 [379](#)

일치 경로

관계 기본 개체 [380](#)

테이블 간 경로 [380](#)

테이블 내 경로 [382](#)

일치 규칙

기본 키 일치 규칙

추가 [423](#)

편집 [424](#)

유형 [274](#)

일치 규칙 정보 [274](#)

일치 목적

Corporate_Entity 일치 목적 [406](#)

Person_Name 일치 목적 [406](#)

Wide_Contact 일치 목적 [406](#)

Wide_Household 일치 목적 [406](#)

가정 일치 목적 [406](#)

가족 일치 목적 [406](#)

개인 일치 목적 [406](#)

거주자 일치 목적 [406](#)

부서 일치 목적 [406](#)

연락처 일치 목적 [406](#)

정보 [404](#)

조직 일치 목적 [406](#)

주소 일치 목적 [406](#)

필드 일치 목적 [406](#)

일치 수준 [410](#)

일치 테이블 재설정 작업 [563](#)

일치/검색 전략 [403](#)

정의 [374](#), [402](#)

정확히 일치 열 [418](#)

허용 한도 [411](#)

일치 규칙 속성

좌표 부여 반지름 [411](#)

일치 규칙 집합

검색 수준 [398](#)

삭제 [402](#)

속성 [398](#)

이름 편집 [401](#)

일치 규칙 집합 정보 [397](#)

추가 [401](#)

편집 [401](#)

필터 [400](#)

일치 규칙 집합의 검색 수준 [398](#)

일치 목적

필드 유형 [390](#)

필드 이름 [390](#)

일치 열

누락된 하위 레코드 [387](#)

유사 항목 일치 기본 개체 [392](#)

유사 항목 일치 열 [389](#)

일치 열 정보 [389](#)

일치 키 유형 [392](#)

정확히 일치 열 [389](#)

정확히 일치하는 기본 개체 [395](#)

키 너비 [393](#)

일치 열 규칙

삭제 [421](#)

추가 [419](#)

편집 [420](#)

일치 작업

상태 사용 BO [558](#)

일치 키 테이블

정의됨 [76](#)

일치 테이블 재설정 작업 [563](#)

일치 토큰 생성 작업 [550](#)

일치 프로세스

BMG(일치 그룹 빌드) [277](#)

개요 [273](#)

관련 기본 개체 속성 [374](#)

사용자 종료 [574](#)

유사 항목 일치 기본 개체 [275](#)

인구집단 [276](#)

일치 규칙 [274](#)

일치 키 테이블 [276](#)

일치 테이블 [276](#)

전이적 일치 [277](#)

정확히 일치하는 기본 개체 [275](#)

지원 테이블 [276](#)

일치 하위 유형 [411](#)

일치되지 않은 레코드를 고유 레코드로 허용 작업 [544](#)

입력 [339](#)

자동 병합 작업

메트릭스 [546](#)

자동 일치 및 병합 작업 [546](#)

트러스트가 활성화된 열 [546](#)

자동 일치 및 병합 작업

메트릭스 [545](#)

작업 영역 보기

도구 [34](#)

정보 [33](#)

잠금

잠금 만료 [36](#)

배타적 잠금 [35](#), [36](#)

비배타적 잠금 [35](#)

서버 캐싱 [36](#)

쓰기 잠금 [35](#), [36](#)

잠금 유형 [644](#)

지우기 [37](#)

해제 [36](#)

획득 [34](#), [36](#)

전체 토큰화 비율 [102](#)

정규식 함수

추가 [336](#)

정리 목록

defaultValue 속성 [343](#)

Input string 속성 [343](#)

matchFlag 속성 [344](#)

replaceAllOccurrences 속성 [343](#)

searchType 속성 [343](#)

SQL 일치 [345](#)

stopOnHit 속성 [343](#)

Strip 속성 [343](#)

문자열 일치 [345](#)

속성 [343](#)

일치 문자열, 가져오기 [345](#)

일치 출력 문자열, 가져오기 [346](#)

일치된 속성 [344](#)

정규식 일치 [345](#)

정리 목록 정보 [342](#)

정확히 일치 [345](#)

추가 [342](#)

출력 속성 [344](#)

편집 [344](#)

정리 함수

Java 정리 라이브러리 [335](#)

가용성 [332](#)

정리 함수 (계속)

- 구성 개요 [334](#)
- 그래프 함수 [336](#)
- 라이브러리 [332](#)
- 로깅 [338](#)
- 매핑 [296](#)
- 보안 리소스 [332](#)
- 분해 [296](#)
- 사용자 정리 라이브러리 [334](#)
- 상수 [339](#)
- 속성 [333](#)
- 유형 [332](#), [333](#)
- 입력 [339](#)
- 작업 공간 단추 [338](#)
- 정리 목록 [342](#)
- 정리 함수 도구 [333](#)
- 정리 함수 정보 [332](#)
- 조건부 실행 구성 요소 [341](#)
- 집계 [296](#)
- 출력 [340](#)
- 테스트 [340](#)
- 함수 모드 [338](#)

정리 함수 도구

- 시작 [333](#)
- 작업 공간 단추 [338](#)

정의

[283](#)

정확히 일치

- 정확히 일치 열 [389](#), [418](#)
- 정확히 일치 전략 [378](#)
- 정확히 일치/검색 전략 [403](#)
- 정확히 일치하는 기본 개체 [275](#)
- 필터링된 일치 [403](#)

정확히 일치 규칙

[274](#)

제약 조건

- 비활성화 [102](#)

제어 테이블

[361](#)

제품 사용 툴킷

- MDM Hub 환경 정보 [652](#)
- 시스템 구성 정보 [652](#)
- 정보 [651](#)

제한된 키 너비

[393](#)

조건부 매핑

[302](#)

조건부 실행 구성 요소

- 사용 시기 [341](#)
- 조건부 실행 구성 요소 정보 [341](#)
- 추가 [341](#)

조직 일치 목적

[406](#)

조회

- 구성 [359](#)
- 대상:외래 키 [359](#)
- 조회 정보 [359](#)

존속

[256](#), [257](#)

주소 일치 목적

[406](#)

준비 작업

- 거부된 레코드 [532](#)

준비 테이블

- null 외래 키 허용 [356](#)
- 관리 [307](#)
- 구성 [353](#)
- 로드 프로세스 [261](#)
- 사용자 정의 열 [353](#)
- 셀 업데이트 [356](#)
- 소스 시스템 키 유지 [355](#)
- 소스 시스템으로 이동 [308](#)
- 속성 [291](#), [658](#)
- 시스템 열 [353](#)
- 열 [111](#)

준비 테이블 (계속)

- 열 속성 [356](#)
- 열, 생성 [111](#)
- 정보 [290](#), [353](#)
- 정의됨 [76](#)
- 제거 [308](#)
- 최상위 예약된 키 [355](#)
- 편집 [307](#), [358](#)

준비 테이블 열

- null 업데이트 허용 속성 [356](#)
- 조회 열 속성 [356](#)
- 조회 테이블 속성 [356](#)

준비 프로세스

- 사용자 종료 [569](#)
- 준비 테이블 [353](#)
- 테이블 [290](#)

중복 데이터

- 일치 대상 [276](#)
- 중복 일치 임계값 [102](#)

증분 로드

[261](#)

지리적 좌표

- 경도 [389](#)
- 고도 [389](#)
- 위도 [389](#)

채우기

- 여러 채우기 [45](#)

처리 서버

- cmxserver.log [648](#)
- HTTPS, 활성화 [331](#)
- 구성 [328](#)
- 데이터 수집 비활성화 [654](#)
- 데이터 수집 활성화 [653](#)
- 모드 [328](#)
- 배포 [328](#)
- 삭제 [332](#)
- 속성 [329](#), [627](#)
- 온라인 작업 [329](#)
- 일괄 작업 [329](#)
- 정리 요청 [329](#)
- 정보 [328](#)
- 처리 서버 도구 [329](#)
- 추가 [331](#)
- 테스트 [332](#)
- 편집 [331](#)

최대 검색 수준

[398](#)

최대 트러스트

[362](#)

최상위 예약된 키

- 예제 [356](#)

최소 트러스트

[362](#)

추적 가능성

[278](#)

출력

[340](#)

쿼리

- 개요 [122](#)
- 결과, 보기 [132](#)
- 구성 [125](#)
- 스키마 변경 [124](#)
- 영향 분석, 보기 [132](#)
- 조인 쿼리 [138](#)
- 패키지 종속성, 분석 [132](#)

쿼리 그룹

- 삭제 [125](#)
- 정보 [125](#)
- 추가 [125](#)
- 편집 [125](#)

쿼리 도구

[123](#)

키

- 기본 [300](#)

키 (계속)

기본, 단일 열에서 [300](#)
기본, 여러 열에서 [300](#)

키 너비 [393](#)

키 유형 [392](#)

키 일치 작업 [552](#)

탐색 창

정보 [37](#)

탐색 창 하위 노드

숨기기 [37](#)

표시 [37](#)

탐색 창의 하위 노드

숨기기 [37](#)

표시 [37](#)

탐색 트리

명령 실행 [39](#)

상위 노드 [37](#)

표시 이름을 기준으로 정렬 [38](#)

필터링 옵션 [38](#)

하위 노드 [37](#)

항목 검색 [39](#)

항목 보기 변경 [38](#)

항목 필터링 [38](#)

태스크

BPM 워크플로우 도구 [170](#)

태스크 관리

사용자 종료 [579](#)

테이블

Hub 저장소 [76](#)

교차 참조 테이블 [76](#)

기록 테이블 [76](#)

기본 개체 [76](#)

랜딩 테이블 [76](#)

시스템 리포지토리 테이블(C_REPOS_SYSTEM) [284](#)

열 추가 [107](#)

일괄 처리에서 사용되는 자원 테이블 [522](#)

일치 키 테이블 [76](#), [267](#)

제어 테이블 [361](#)

준비 테이블 [76](#)

테이블 간 경로 [380](#)

테이블 간 일치

설명됨 [418](#)

테이블 내 경로 [382](#)

테이블 열

GBID(글로벌 식별자) 열 [110](#)

다른 테이블에서 가져오기 [112](#)

삭제 [114](#)

준비 테이블 [111](#)

추가 [112](#)

테이블 열 정보 [107](#)

편집 [113](#)

테이블 열 정의 가져오기 [112](#)

토큰나이저

기본 제공 [467](#)

토큰 필터

기본 제공 [467](#)

토큰화 프로세스

더티 테이블 [269](#)

일치 키 [267](#)

일치 키 테이블 [267](#)

일치 토큰 [267](#)

주요 개념 [268](#)

토큰화 프로세스 정보 [267](#)

통합

BVT(최선의 진실, Best Version of the Truth) [279](#)

통합 감사 [599](#)

통합 표시기

값 [255](#)

시퀀스 [256](#)

통합 표시기 정보 [255](#)

통합 프로세스

개요 [278](#)

옵션 [279](#)

정보 [475](#)

통합된 레코드 [94](#)

트러스트

SIRL(초기에 느리다 나중에 빨라짐) 붕괴 [362](#)

계산 [361](#)

붕괴 곡선 [362](#)

붕괴 그래프 유형 [362](#)

붕괴 기간 [361](#)

상태 사용 기본 개체 [366](#)

설정 시 고려 사항 [363](#)

속성 [362](#)

수준 [361](#)

시스템 및 트러스트 도구 [284](#)

열 [363](#)

열 속성 [108](#)

열에 대해 활성화 [364](#)

최대 트러스트 [362](#)

최소 트러스트 [362](#)

트러스트 설정 동기화 [564](#)

트러스트 수준, 정의됨 [361](#)

트러스트 정보 [360](#)

O

Oracle

데이터 유형 [107](#)

Organization_Name 키 유형 [392](#)

ORIG_ROWID_OBJECT

교차 참조 테이블 열 [99](#)

ORS 관련 API

API 보관 테이블

유지 관리 [588](#)

build_war 매크로 [590](#)

개요 [586](#)

리포지토리 개체 [587](#)

리포지토리 개체 상태 [588](#)

보관 테이블 [588](#)

사용 [590](#)

성능 [587](#)

속성 [587](#)

클라이언트 JAR 다운로드 [589](#)

ORS 관련 SIF API

생성 및 배포 [589](#)

ORS 관련 메시지 스키마

개요 [591](#)

ORS(연산 참조 저장소)

GETLIST 제한(행) [68](#)

JNDI 데이터 소스 이름 [68](#)

ORS 정보 [60](#)

등록 해제 [73](#)

생성 [61](#)

암호, 변경 [71](#)

연결 테스트 [70](#)

편집 [68](#)

P

PCTL 테이블 참조 보류 중인 제어 테이블

PERIOD_END_DATE
교차 참조 테이블 열 [99](#)
준비 테이블 열 [353](#)
PERIOD_START_DATE
교차 참조 테이블 열 [99](#)
준비 테이블 열 [353](#)
Person_Name 일치 목적 [406](#)
Person_Name 키 유형 [392](#)
PKEY_SRC_OBJECT
교차 참조 테이블 열 [99](#)
준비 테이블 열 [353](#)
PROMOTE_IND
교차 참조 테이블 열 [99](#)
provisioning.log
설명 [648](#)
PUT 사용 패키지 [135](#)
PUT_UPDATE_MERGE_IND
교차 참조 테이블 열 [99](#)

R

Raw 테이블
정보 [290](#)
RBO(리포지토리 기본 개체) 테이블 [193](#)
RISL(초기에 빠르다 나중에 느려짐) 붐과 [362](#)
ROWID_AUDIT
감사 로그 테이블 열 [605](#)
ROWID_AUDIT_PREVIOUS
감사 로그 테이블 열 [605](#)
ROWID_MATCH_RULE
외부 일치 출력 테이블 열 [548](#)
ROWID_OBJECT
감사 로그 테이블 열 [605](#)
교차 참조 테이블 열 [99](#)
기본 개체 열 [95](#)
준비 테이블 열 [353](#)
ROWID_OBJECT_MATCH
외부 일치 출력 테이블 열 [548](#)
ROWID_SYSTEM
교차 참조 테이블 열 [99](#)
ROWID_XREF
교차 참조 테이블 열 [99](#)

S

S_
교차 참조 테이블 열 [99](#)
SIF API
ORS 관련, 이름 바꾸기 [589](#)
ORS 관련, 제거 [590](#)
사용자 종료에 지원됨 [582](#)
SIF API에 대한 사용자 종료
생성 [580](#)
예 [581](#)
SMOS
상태 관리 재정의 시스템 [284](#)
SMOS(상태 관리 재정의 시스템)
정보 [284](#)
SOURCE_KEY
외부 일치 출력 테이블 열 [548](#)
SOURCE_NAME
외부 일치 출력 테이블 열 [548](#)
SQL
사용자 지정 쿼리 [133](#)
SQL*Loader
손상된 데이터 [47](#)

SRC_LUD
교차 참조 테이블 열 [99](#)
SRC_ROWID
준비 테이블 열 [353](#)
STATUS
감사 로그 테이블 열 [605](#)
STG_ROWID_TABLE
교차 참조 테이블 열 [99](#)
stopwords.txt 파일
사용자 지정 [459](#)
synonyms.txt 파일
사용자 지정 [459](#)

T

TABLE_NAME
감사 로그 테이블 열 [605](#)
TO_SYSTEM
감사 로그 테이블 열 [605](#)

U

Unicode
NLS_LANG [48](#)
Unix 및 로컬 권장 사항 [47](#)
정리 설정 [47](#)
UPDATED_BY
감사 로그 테이블 열 [605](#)
교차 참조 테이블의 열 [99](#)
기본 개체 열 [95](#)
준비 테이블 열 [353](#)
UserExitContext
정보 [568](#)
USERNAME
감사 로그 테이블 열 [605](#)

V

VERSION_SEQ
준비 테이블 열 [353](#)

W

Wide_Contact 일치 목적 [406](#)
Wide_Household 일치 목적 [406](#)

X

XSD 파일
다운로드 [593](#)

┐

가정 일치 목적 [406](#)
가족 일치 목적 [406](#)
감사
API 요청 [603](#)
XML [600](#)
감사 관리자 도구 [600](#)
감사 로그 [604](#)
감사 로그 보기 [607](#)
감사 로그 제거 [607](#)

감사 (계속)

- 감사 로그 테이블 [605](#)
- 감사할 시스템 [602](#)
- 구성 가능한 설정 [602](#)
- 메시지 대기열 [603](#)
- 암호 변경 [601](#)
- 오류 [604](#)
- 이벤트 [599](#)
- 인증 및 [600](#)
- 통합 감사 정보 [599](#)
- 활성화 [600](#)

감사 관리자

- 감사 관리자 정보 [600](#)
- 감사할 항목 유형 [601](#)
- 시작 [601](#)

감사 내역, 구성

감사 로그 테이블

- C_REPOS_AUDIT [605](#)

- 같지 않은 일치 [415](#)

- 개인 일치 목적 [406](#)

거부 테이블

- 정보 [290](#)

- 거주자 일치 목적 [406](#)

검색

- Elasticsearch 사용 [454](#)
- Elasticsearch를 사용한 아키텍처 [455](#)
- stopwords.txt 파일 [459](#)
- synonyms.txt 파일 [459](#)
- 동의어 [459](#)
- 무시할 단어 [459](#)
- 중지 단어 [459](#)

게시 프로세스

- ORS 관련 스키마 파일 [281](#)

- XSD 파일 [281](#)
- 런타임 흐름 [282](#)
- 메시지 대기열 [281](#)
- 메시지 트리거 [280](#)
- 분포 흐름 [280](#)
- 정보 [280](#)

경로 구성 요소

- 삭제 [389](#)
- 추가 [388](#)
- 편집 [388](#)

- 계단식 병합 해제 [476](#)

계층

- 계층 노드 [202](#)
- 삭제 [203](#)
- 예 [187](#)
- 정보 [186](#), [202](#)
- 추가 [202](#)

계층 관리자

- 구성 개요 [188](#)
- 데이터 구성 [212](#)
- 리포지토리 기본 개체 테이블 [193](#)
- 샌드박스 [216](#)
- 선행 조건 [188](#)
- 항목 아이콘, 업로드 [193](#)

계층 도구

- 구성 개요 [188](#)

고유

- 열 속성 [108](#)

- 고유 매핑 [302](#)

- 고유 소스 시스템 [476](#)

- 고유 키 [300](#)

관계

- 세부 정보 [116](#)
- 외래 키 관계 [115](#)

관계 개체

- 정보 [203](#)

관계 기본 개체

- 관계 기간 삭제 [162](#)
- 관계 업데이트 [159](#)
- 관계 종료 [161](#)
- 기본 개체로 되돌리기 [206](#)
- 모든 관계 기간 삭제 [163](#)
- 변환 [205](#)
- 삭제 예제 [162](#), [163](#)
- 생성 [204](#)
- 업데이트 예제 [159](#), [160](#)
- 외래 키 관계 기본 개체 [206](#)
- 외래 키 관계 기본 개체 생성 [207](#)
- 정보 [203](#)
- 종료 예제 [161](#)

관계 유형

- 삭제 [211](#)
- 생성 [210](#)
- 예 [209](#)
- 정보 [207](#)
- 편집 [211](#)

관리 소스 시스템

- 관리 소스 시스템 정보 [284](#)

- 이름 바꾸기 [285](#)

- 교차 참조 승격 이력, 활성화 [171](#)

교차 참조 테이블

- ROWID_XREF 열 [99](#)
- 교차 참조 테이블 정보 [97](#)
- 기록 테이블 [100](#)
- 기본 개체와의 관계 [97](#)
- 로드 프로세스 [261](#)
- 설명됨 [76](#)
- 승격 이력 활성화 [171](#)
- 열 [99](#)
- 정의됨 [76](#)

구성 작업 영역

- 정보 [41](#)
- 그래프 함수
- 입력 [336](#)
- 조건부 실행 구성 요소 [341](#)
- 추가 [337](#)
- 출력 [336](#)
- 함수 추가 [337](#)

그룹 실행 로그

- 보기 [540](#)
- 상태 값 [540](#)

근접 검색

- 개요 [427](#)
- 구성 [428](#)

기록

- 활성화 [101](#)

기록 테이블

- 교차 참조 기록 테이블 [100](#)
- 기본 개체 기록 테이블 [100](#)
- 정의됨 [76](#)
- 활성화 [104](#)

기본 개체

- 개요 [95](#)
- 관계 기본 개체 [380](#)
- 관계 기본 개체에서 되돌리기 [206](#)
- 기록 테이블 [100](#)
- 레코드 존속, 상태 관리 [170](#)
- 삭제 [106](#)
- 삼입형 로드 [263](#)
- 생성 [104](#)
- 설명됨 [76](#)

기본 개체 (계속)

스타일 [102](#)

시스템 열 [95](#)

업데이트형 로드 [264](#)

열 추가 [78](#)

영향 분석 [106](#)

예약된 접미사 [78](#)

예약된 특수 문자 [78](#)

유사 항목 일치 기본 개체 [275](#)

정의 [95](#)

정확히 일치하는 기본 개체 [275](#)

편집 [104](#)

항목 기본 개체 [195](#)

항목 기본 개체로 변환 [197](#)

기본 개체 속성

일치 프로세스 동작 [374](#)

기본 개체 스타일 [102](#)

기본 설정 키 너비 [393](#)

기본 키

단일 열에서 [300](#)

여러 열에서 [300](#)

영구 삭제 검색 [318](#)

기본 키 일치 규칙

삭제 [424](#)

정보 [423](#)

추가 [423](#)

편집 [424](#)

기본값

열 속성 [108](#)

기본값 있음

열 속성 [108](#)