PowerCenter 10.4.0 XML 가이드

2019년12월

목차

XML 가이드-저작권

이 제품에는 OpenSSL Toolkit(저작권 The OpenSSL Project. 모든 권리 보유)에서 사용할 수 있도록 OpenSSL Project에서 개발한 소프트웨어가 포함되어 있으며 이 소프트웨어의 재배포는 http:// www.openssl.org 및 http://www.openssl.org/source/license.html의 조항에 따라 변경될 수 있습니다.

이 제품에는 Curl 소프트웨어(저작권 1996-2013, Daniel Stenberg, <daniel@haxx.se>. 모든 권리 보유.)가 포함되어 있습니다. 이 소프트웨어와 관련된 사용 권한 및 제한 사항은 http://curl.haxx.se/docs/copyright.html에 명시된 조항에 따라 변경될 수 있습니다. 위와 같은 저작권 고지 및 이러한 허가 고지가 모든 제품에 표시되어 있는 경우 목적 및 사용료 유무에 관계없이 이 소프트웨어를 사용, 복사, 수정 및 배포할 수 있는 사용 권한이 부여됩니다.

이 제품에는 MetaStuff, Ltd(저작권 2001-2005 ((C)) MetaStuff, Ltd. 모든 권리 보유.)가 저작권을 소유한 소프트웨어가 포함되어 있습니다. 이 소프트웨어와 관련된 사용 권한 및 제한은 http://www.dom4j.org/license.html의 조항에 따라 변경될 수 있습니다.

이 제품에는 Per Bothner(저작권 (c) 1996-2006 Per Bothner. 모든 권리 보유.)가 저작권을 소유한 소프트웨어가 포함되어 있습니다. 이러한 정보를 사용할 수 있는 권리는 http:// www.gnu.org/software/kawa/Software-License.html의 라이선스에 설명되어 있습니다.

이 제품에는 OSSP UUID 소프트웨어(저작권 (c) 2002 Ralf S. Engelschall, 저작권 (c) 2002 The OSSP Project 저작권 (c) 2002 Cable & Wireless Deutschland)가 포함되어 있습니다. 이 소프트웨어와 관련된 사용 권한 및 제한은 http://www.opensource.org/licenses/mit-license.php의 조항에 따라 변경될 수 있습니다.

이 제품에는 Boost(http://www.boost.org/)에서 개발하거나 Boost 소프트웨어 라이선스에 따라 개발된 소프트웨어가 포함되어 있습니다. 이 소프트웨어와 관련된 사용 권한 및 제한은 http:/ /www.boost.org/LICENSE_1_0.txt의 조항에 따라 변경될 수 있습니다.

이 제품에는 University of Cambridge(저작권 (c) 1997-2007 University of Cambridge)가 저작권을 소유한 소프트웨어가 포함되어 있습니다. 이 소프트웨어와 관련된 사용 권한 및 제한은 http:// www.pcre.org/license.txt의 조항에 따라 변경될 수 있습니다.

이 제품에는 Eclipse Foundation(저작권 (c) 2007 The Eclipse Foundation. 모든 권리 보유.)이 저작권을 소유한 소프트웨어가 포함되어 있습니다. 이 소프트웨어와 관련된 사용 권한 및 제한은 http:// www.eclipse.org/org/documents/epl-v10.php 및 http://www.eclipse.org/org/documents/edl-v10.php 의 조항에 따라 변경될 수 있습니다.

이 제품에는 http://www.tcl.tk/software/tcltk/license.html, http://www.bosrup.com/web/overlib/?License, http://www.stlport.org/doc/ license.html, http://asm.ow2.org/license.html, http:// www.cryptix.org/LICENSE.TXT, http://hsqldb.org/web/hsqlLicense.html, http:// httpunit.sourceforge.net/doc/ license.html, http://jung.sourceforge.net/license.txt, http:// www.gzip.org/zlib/zlib_license.html, http://www.openldap.org/software/release/license.html, http://www.libssh2.org, http://slf4j.org/license.html, http://www.sente.ch/software/OpenSourceLicense.html, http://fusesource.com/downloads/license-agreements/fuse-message-broker-v-5-3- license-agreement; http://antlr.org/license.html; http://aopalliance.sourceforge.net/; http://www.bouncycastle.org/licence.html; http://www.jgraph.com/jgraphdownload.html; http://www.jcraft.com/jsch/LICENSE.txt; http://jotm.objectweb.org/bsd_license.html; http:// www.w3.org/Consortium/Legal/2002/copyright-software-20021231, http://www.slf4j.org/license.html, http://nanoxml.sourceforge.net/orig/copyright.html, http://www.json.org/license.html, http://forge.ow2.org/projects/javaservice/, http://www.postgresql.org/about/licence.html, http://www.sqlite.org/copyright.html, http://www.tcl.tk/software/tcltk/license.html, http://www.jaxen.org/faq.html, http://www.jdom.org/docs/faq.html, http://www.slf4j.org/license.html, http://www.iodbc.org/dataspace/iodbc/wiki/iODBC/License, http://

www.keplerproject.org/md5/license.html, http://www.toedter.com/en/jcalendar/license.html, http://www.edankert.com/bounce/index.html, http://www.net-snmp.org/about/license.html, http://www.openmdx.org/#FAQ, http://www.php.net/license/3_01.txt, http://srp.stanford.edu/license.txt, http://www.schneier.com/blowfish.html, http://www.jmock.org/license.html, http://xsom.java.net, http://benalman.com/about/license/, https://github.com/CreateJS/EaselJS/blob/master/src/easeljs/display/Bitmap.js, http://www.h2database.com/html/license.html#summary, http://jsoncpp.sourceforge.net/LICENSE, http://jdbc.postgresql.org/license.html, http://protobuf.googlecode.com/svn/trunk/src/google/protobuf/descriptor.proto, https://github.com/rantav/hector/blob/master/LICENSE, http://web.mit.edu/Kerberos/krb5-current/doc/mitK5license.html, http://jibx.sourceforge.net/jibx-license.html, https://github.com/lyokato/libgeohash/blob/master/LICENSE, https://github.com/hjiang/jsonxx/blob/master/LICENSE, https://code.google.com/p/lz4/, https://github.com/jedisct1/libsodium/blob/master/LICENSE, http://one-jar.sourceforge.net/index.php?page=documents&file=license, https://github.com/EsotericSoftware/kryo/blob/master/license.txt, http://www.scala-lang.org/license.html, https://github.com/tinkerpop/blueprints/blob/master/LICENSE.txt, http://gee.cs.oswego.edu/dl/classes/EDU/oswego/cs/dl/util/concurrent/intro.html, https://aws.amazon.com/asl/, https://github.com/twbs/bootstrap/blob/master/LICENSE 및 https://sourceforge.net/p/xmlunit/code/HEAD/tree/trunk/LICENSE.txt.

이 제품에는 Academic Free License(http://www.opensource.org/licenses/afl-3.0.php), Common Development and Distribution License(http://www.opensource.org/licenses/cddl1.php), Common Public License (http://www.opensource.org/licenses/cpl1.0.php), Sun Binary Code License Agreement Supplemental License Terms, BSD License(http:// www.opensource.org/licenses/bsd-license.php), 새 BSD License(http://opensource.org/licenses/BSD-3-Clause), MIT License(http://www.opensource.org/licenses/mit-license.php), Artistic License(http://www.opensource.org/licenses/artistic-license-1.0) 및 Initial Developer's Public License 버전 1.0(http://www.firebirdsql.org/en/initial-developer-s-public-license-version-1-0/)에 따라 라이선스가 부여된 소프트웨어가 포함되어 있습니다.

이 제품에는 Joe Walnes와 XStream Committers(저작권 (c) 2003-2006 Joe Walnes, 2006-2007 XStream Committers. 모든 권리 보유.)가 저작권을 소유한 소프트웨어가 포함되어 있습니다. 이 소프트웨어와 관련된 사용 권한 및 제한은 http://xstream.codehaus.org/license.html의 조항에 따라 변경될 수 있습니다. 이 제품에는 Indiana University Extreme! Lab에서 개발한 소프트웨어가 포함되어 있습니다. 자세한 내용을 확인하려면 http://www.extreme.indiana.edu/를 방문하십시오.

이 제품에는 Frank Balluffi 및 Markus Moeller(저작권 (c) 2013 Frank Balluffi and Markus Moeller. 모든 권리 보유.)가 저작권을 소유한 소프트웨어가 포함되어 있습니다. 이 소프트웨어와 관련된 사용 권한 및 제한 사항은 MIT license에 명시된 조항에 따라 변경될 수 있습니다.

https://www.informatica.com/legal/patents.html에서 특허를 참조하십시오.

고지 사항: Informatica LLC는 비침해, 상품성 또는 특정 목적에 따른 사용에 대한 묵시적 보증을 포함하여 이에 국한되지 않는 어떠한 종류의 명시적이거나 묵시적인 보증 없이 이 문서를 "있는 그대로" 제공합니다. Informatica LLC는 이 소프트웨어나 문서에 오류가 없음을 보장하지 않습니다. 이 소프트웨어나 설명서에 제공된 정보에는 기술적 오류나 인쇄 오류가 있을 수 있습니다. 이 소프트웨어 및 설명서의 정보는 언제든지 예고 없이 변경될 수 있습니다.

고지 사항

이 Informatica 제품(이하 "소프트웨어")에는 Progress Software Corporation(이하 "DataDirect")의 운영 회사인 DataDirect Technologies의 특정 드라이버(이하 "DataDirect Drivers")가 포함되어 있습니다. 이러한 드라이버에는 다음 조건이 적용됩니다.

1. DataDirect Drivers는 상품성, 특정 목적에의 적합성 및 비침해에 대한 묵시적 보증을 포함하여 이에 국한되지 않는 어떠한 종류의 명시적이거나 묵시적인 보증 없이 "있는 그대로" 제공됩니다.
2. DataDirect 또는 그 타사 공급자는 손해의 발생 가능성을 사전에 알고 있었는지 여부에 관계없이 ODBC 드라이버의 사용으로 발생하는 직접, 간접, 부수적, 특별, 결과적 또는 기타 손해에 대해 어떠한 경우에도 최종 사용자에게 책임을 지지 않습니다. 이러한 제한 사항은 계약 위반, 보증 불이행, 과실, 무과실 책임, 허위 진술 및 기타 불법 행위를 포함하여 이에 국한되지 않는 모든 소송 사유에 적용됩니다.

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.

서문

*XML 가이드*는 데이터 웨어하우스 환경에서 XML 작업을 담당하는 개발자 및 소프트웨어 엔지니어를 위해 작성되었습니다. *XML 가이드*를 사용하기 전에 XML 개념, 해당 운영 체제, 플랫 파일 또는 해당 환경의 메인프레임 시스템에 대해 확실히 이해했는지 확인하십시오. 또한 지원하려는 응용 프로그램의 인터페이스 요구 사항을 숙지했는지 확인하십시오.

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.com을 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.com을 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.com을 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-matrices에서 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.com에서 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.com에서 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.com을 방문하고 eSupport 옵션을 선택하십시오.

XML 개념

XML 개념 개요

XML(Extensible Markup Language)은 일반적인 정보 형식을 작성하고 응용 프로그램과 인터넷 간에 형식 및 데이터를 공유하는 유연한 방법입니다.

XML 정의를 다음 파일 유형에서 PowerCenter®로 가져올 수 있습니다.

- **XML 파일**. XML 파일에는 데이터 및 메타데이터가 포함됩니다. XML 파일은 DTD(문서 유형 정의) 파일 또는 XSD(XML 스키마 정의)를 참조하여 유효성을 검사할 수 있습니다.
- **DTD 파일**. DTD 파일은 XML 파일의 요소 유형, 특성 및 항목을 정의합니다. DTD 파일은 XML 파일 구조에 대한 몇 가지 제약 조건을 제공하지만 DTD 파일은 데이터를 포함하지 않습니다.
- **XML 스키마.** XML 스키마는 요소, 특성 및 유형 정의를 정의합니다. 스키마에는 단순 및 복합 유형이 있습니다. 단순 유형은 텍스트를 포함하는 XML 요소 또는 특성입니다. 복합 유형은 다른 요소 및 특성을 포함하는 XML 요소입니다.

  스키마는 스키마를 통해 참조할 수 있는 요소, 특성 및 대체 그룹을 지원합니다. 대체 그룹을 사용하여 XML 인스턴스 문서에서 한 요소를 다른 요소로 대체합니다. 또한 스키마는 요소, 복합 유형, 요소 및 특성 그룹에 대한 상속을 지원합니다.

XML 파일

XML 파일은 XML 파일의 데이터를 식별하는 태그는 포함하지만 데이터의 형식은 포함하지 않습니다. XML 파일의 기본 구성 요소는 요소입니다. XML 요소에는 요소 시작 태그, 요소 콘텐츠 및 요소 종료 태그가 있습니다. 모든 XML 파일에는 파일의 상단 및 하단에 단일 태그로 정의된 루트 요소가 있어야 합니다. 루트 요소는 파일의 다른 모든 요소를 묶습니다.

XML 파일은 계층 데이터베이스를 모델링합니다. XML 계층에서 요소 위치는 다른 요소에 대한 관계를 나타냅니다. 요소는 하위 요소를 포함할 수 있으며 다른 요소에서 특성을 상속받을 수 있습니다.

예를 들어 다음 XML 파일은 book을 설명합니다.

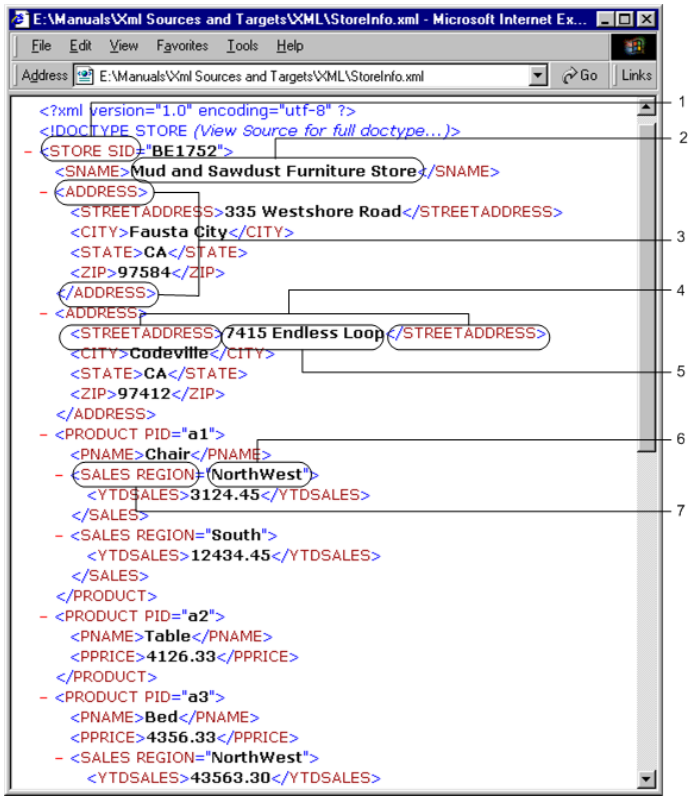
```
<book>
   <title>Fun with XML</title>
   <chapter>
      <heading>Understanding XML</heading>
      <heading>Using XML</heading>
   </chapter>
   <chapter>
      <heading>Using DTD Files</heading>
      <heading>Fun with Schemas</heading>
   </chapter>
</book>
```

book은 루트 요소이며 title 및 chapter 요소가 포함되어 있습니다. book은 title 및 chapter의 상위 요소이며, chapter는 heading의 상위 요소입니다. title 및 chapter는 상위가 동일하기 때문에 형제 요소입니다.

요소는 요소에 대한 추가 정보를 제공하는 특성을 가질 수 있습니다. 다음 예에서 특성 graphic_type은 그림의 콘텐츠를 설명합니다.

```
<picture graphic_type="gif">computer.gif</picture>
```

다음 그림은 XML 파일의 구조, 요소 및 특성을 보여 줍니다.



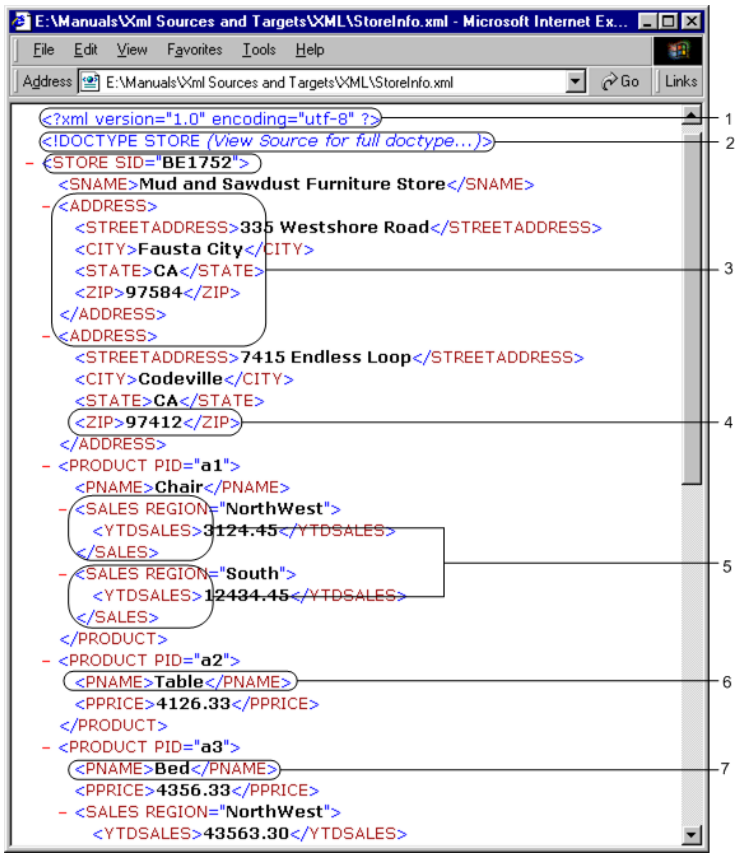
1. 루트 요소
2. 요소 데이터
3. 삽입 요소
4. 요소 태그
5. 요소 데이터
6. 특성 값
7. 특성 태그

XML 파일에는 계층 구조가 있습니다. XML 계층은 다음 요소를 포함합니다.

- **하위 요소.** 다른 요소 안에 포함된 요소입니다.
- **삽입 요소.** 다른 요소를 포함하지만 데이터를 포함하지 않는 요소입니다. 삽입 요소는 다른 삽입 요소를 포함할 수 있습니다.
- **글로벌 요소.** 루트 요소의 직접 하위인 요소입니다. XML 스키마를 통해 글로벌 요소를 참조할 수 있습니다.
- **리프 요소.** 다른 요소를 포함하지 않는 요소입니다. 리프 요소는 XML 계층의 가장 낮은 수준의 요소입니다.
- **로컬 요소.** 다른 요소에 중첩된 요소입니다. 상위 요소의 컨텍스트 내에서만 로컬 요소를 참조할 수 있습니다.
- **다중 발생 요소.** 해당 상위 요소 내에서 두 번 이상 발생하는 요소입니다. 삽입 요소는 다중 발생 요소일 수 있습니다.

- **상위 체인.** 요소에서 루트까지의 경로를 추적하는 연속된 하위-상위 요소입니다.
- **상위 요소.** 다른 요소를 포함하는 요소입니다.
- **단일 발생 요소.** 해당 상위 내에서 한 번 발생하는 요소입니다.

다음 그림은 XML 계층의 일부 요소를 보여 줍니다.



1. 인코딩 특성은 코드 페이지를 식별합니다.
2. DOCTYPE은 연결된 DTD 파일을 식별합니다.
3. 삽입 요소: 요소 Address는 요소 StreetAddress, City, State 및 Zip을 묶습니다. 또한 요소 Address는 상위 요소입니다.
4. 리프 요소: 요소 Zip은 모든 해당 형제 요소와 함께 요소 Address 내에서 가장 낮은 수준의 요소입니다.
5. 다중 발생 요소: 요소 Sales Region은 요소 Product 내에서 두 번 이상 발생합니다.
6. 단일 발생 요소: 요소 PName은 요소 Product 내에서 한 번 발생합니다.
7. 하위 요소: 요소 PName은 Store의 하위인 Product의 하위입니다.

DTD 또는 스키마로 XML 파일의 유효성 검사

유효한 XML 파일은 연결된 DTD 또는 스키마 파일의 구조를 준수합니다.

DTD 파일의 위치 및 이름을 참조하려면 XML 파일에서 DOCTYPE 선언을 사용합니다. 또한 DOCTYPE 선언은 XML 파일의 루트 요소를 명명합니다.

예를 들어 다음 XML 파일은 note.dtd 파일의 위치를 참조합니다.

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM
"http://www.w3schools.com/dtd/note.dtd">
<note>
    <body>XML Data</body>
</note>
```

스키마를 참조하려면 schemaLocation 선언을 사용합니다. schemaLocation은 스키마의 위치 및 이름을 포함합니다.

다음 XML 파일은 외부 위치에서 note.xsd 스키마를 참조합니다.

```
<?xml version="1.0"?>
<note xsi:SchemaLocation="http://www.w3schools.com note.xsd">
   <body>XML Data</body>
</note>
```

유니코드 인코딩

XML 파일은 파일의 코드 페이지를 나타내는 인코딩 특성을 포함합니다. 가장 일반적인 인코딩은 UTF-8 및 UTF-16입니다. UTF-8은 유니코드 기호에 따라 1-4바이트를 사용하는 문자입니다. UTF-16은 문자를 16비트 단어로 나타냅니다.

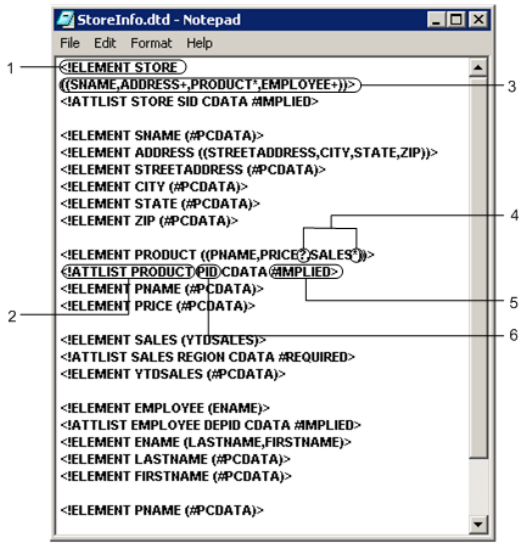
다음 예는 XML 파일에서 UTF-8 특성을 보여 줍니다.

```
<?xml version="1.0"encoding="UTF-16LE"?>
<note xsi:SchemaLocation="http://www.w3schools.com note.xsd">
   <body>XML Data</body>
</note>
```

DTD 파일

DTD(문서 유형 정의) 파일은 XML 파일의 요소 유형 및 특성을 정의합니다. 또한 DTD 파일은 XML 파일 구조에 대한 일부 제약 조건을 제공합니다. DTD 파일은 데이터 또는 요소 데이터 유형을 포함하지 않습니다.

다음 그림은 DTD 파일의 요소 및 특성을 보여 줍니다.



1. 요소
2. 특성
3. 요소 목록
4. 요소 발생
5. 특성 값 옵션
6. 특성 이름

DTD 요소

DTD 파일에서 요소 선언은 XML 요소를 정의합니다. 요소 선언 구문은 다음과 같습니다.

```
<!ELEMENT product (#PCDATA)>
```

DTD 설명은 XML 태그 <product>를 정의합니다. 설명 (#PCDATA)는 구문 분석된 문자 데이터를 지정합니다. 구문 분석된 데이터는 XML 요소의 시작 태그와 종료 태그 사이에 있는 텍스트입니다. 구문 분석된 문자 데이터는 하위 요소가 없는 텍스트입니다.

다음 예는 두 개의 하위 요소를 가진 요소의 DTD 설명을 보여 줍니다.

```
<!ELEMENT boat (brand, type) >
<!ELEMENT brand (#PCDATA) >
<!ELEMENT type (#PCDATA) >
```

brand 및 type은 boat의 하위 요소입니다. 각 하위 요소는 문자를 포함할 수 있습니다. 이 예에서 brand 및 type은 요소 boat 내에서 한 번 발생할 수 있습니다. 다음 DTD 설명은 boat에 대해 brand가 한 번 이상 발생해야 한다고 지정합니다.

```
<!ELEMENT boat (brand+) >
```

DTD 특성

특성은 요소에 대한 추가 정보를 제공합니다. DTD 파일에서 특성은 요소의 시작 태그 내에서 발생합니다.

다음 구문은 DTD 파일의 특성을 설명합니다.

```
<!ATTLIST element_name attribute_name attribute_type “default_value”>
```

다음 매개 변수는 DTD 파일의 특성을 식별합니다.

- **Element_name**. 특성을 가진 요소의 이름입니다.
- **Attribute_name.** 특성의 이름입니다.
- **Attribute_type.** 특성의 종류입니다. 가장 일반적인 특성 유형은 CDATA입니다. CDATA 특성은 문자 데이터입니다.
- **Default_value.** 특성 값이 XML 파일에서 발생하지 않는 경우의 특성 값입니다.

  기본값으로 다음 옵션을 사용합니다.

  - **#REQUIRED.** XML 파일이 특성 값을 포함해야 합니다.
  - **#IMPLIED.** 특성 값은 선택 사항입니다.
  - **#FIXED.** XML 파일이 DTD 파일의 기본값을 포함해야 합니다. 유효한 XML 파일은 DTD와 동일한 특성 값을 포함할 수 있습니다. 그렇지 않은 경우 XML 파일이 특성 값을 가질 수 없습니다. 이 옵션을 사용하여 기본값을 지정해야 합니다.

  다음 예는 고정값을 사용하는 특성을 보여 줍니다.

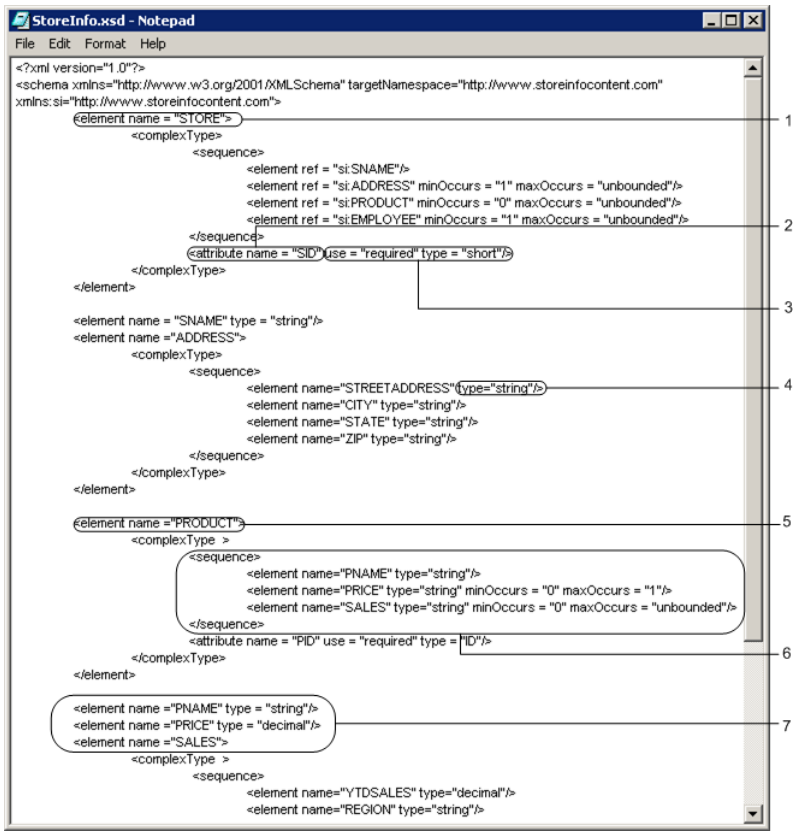
  ```
  <!ATTLIST product product_name CDATA #FIXED “vacuum”>
  ```

  요소 이름은 product입니다. 특성은 product_name입니다. 특성에 기본값인 vacuum이 있습니다.

XML 스키마 파일

XML 스키마는 XML 파일의 유효한 콘텐츠를 정의하는 문서입니다. DTD 파일과 같은 XML 스키마 파일은 메타데이터만 포함합니다. XML 스키마는 연결된 XML 파일에 대한 요소와 특성의 구조 및 유형을 정의합니다. 스키마를 사용하여 XML 파일을 정의하는 경우 데이터 제한, 데이터 형식 정의 및 데이터 유형 간 데이터 변환을 수행할 수 있습니다. XML 스키마는 복합 유형 및 유형 간 상속을 지원합니다. 스키마는 요소 및 특성 그룹, ANY 콘텐츠 및 원형 참조를 지정하는 방법을 제공합니다.

다음 그림은 XML 스키마 구성 요소를 보여 줍니다.



1. 요소 이름
2. 특성
3. 특성 유형 및 Null 구성
4. 요소 데이터 유형
5. 요소 데이터
6. 요소 목록 및 발생
7. 요소 목록 및 데이터 유형

관련 항목:

- "단순 및 복합 XML 유형" 페이지 19
- "구성 요소 그룹" 페이지 26

XML 메타데이터 유형

XML, DTD 또는 XML 스키마 파일에서 PowerCenter XML 정의를 작성할 수 있습니다. XML 파일은 데이터 및 메타데이터를 제공합니다. DTD 파일 및 XML 스키마 파일은 메타데이터를 제공합니다.

PowerCenter는 XML, DTD 및 XML 스키마 파일에서 다음 유형의 메타데이터를 추출합니다.

- **네임스페이스.** XML 파일의 URI(Uniform Resource Identifier) 참조에서 식별한 요소 및 특성 이름의 컬렉션입니다. 네임스페이스는 다른 소스에서 가져온 요소를 구분합니다.

- **이름.** 요소 또는 특성의 이름을 포함하는 태그입니다.
- **계층.** XML 파일의 다른 요소와 관계를 맺고 있는 요소의 위치입니다.
- **카디널리티.** XML 파일에서 요소가 발생하는 횟수입니다.
- **데이터 유형.** 숫자, 문자열, 부울 또는 시간과 같은 데이터 요소의 분류입니다. XML은 사용자 지정 데이터 유형 및 상속을 지원합니다.

네임스페이스

네임스페이스는 스키마 위치를 식별하기 위해 URI를 포함합니다. URI는 인터넷 리소스를 식별하는 문자열입니다. URI는 URL을 추상화한 것입니다. URL은 리소스를 찾지만 URI는 리소스를 식별합니다. DTD 또는 스키마 파일이 URI 위치에 있을 필요는 없습니다.

XML 네임스페이스는 요소의 그룹을 식별합니다. 네임스페이스는 서로 다른 XML 파일의 요소 및 특성을 식별하거나 요소 간 의미를 구별할 수 있습니다. 예를 들어 *math:table* 및 *furniture:table*과 같이 서로 다른 네임스페이스를 선언하여 요소 "table"에 대한 의미를 구별할 수 있습니다. XML은 대/소문자를 구분합니다. 네임스페이스 *Math:table*은 네임스페이스 *math:table*과 다릅니다.

XML 파일의 루트 수준에서 네임스페이스를 선언하거나 XML 구조의 요소 내에서 네임스페이스를 선언할 수 있습니다. 동일한 XML 파일에서 여러 네임스페이스를 선언하는 경우 네임스페이스 접두사를 사용하여 요소를 네임스페이스와 연결합니다. 네임스페이스 선언은 XML 파일에 xmlns로 시작하는 특성으로 나타납니다. xmlns 특성과 함께 네임스페이스 접두사를 선언합니다. 길이 제한 없이 접두사 이름을 작성할 수 있습니다.

다음 예는 XML 인스턴스 문서의 두 네임스페이스를 보여 줍니다.

```
<example>
  xmlns:math = “http://www.mathtables.com”
  xmlns:furniture = “http://www.home.com”>
  <math:table>4X6</math:table>
  <furniture:table>Brueners </furniture:table>
</example>
```

한 네임스페이스에는 math 요소가 있고 다른 네임스페이스에는 furniture 요소가 있습니다. 각 네임스페이스에 “table”이라는 요소가 있지만 그 요소는 서로 다른 유형의 데이터를 포함합니다. 네임스페이스 접두사가 math table과 furniture table을 구별합니다.

다음 텍스트는 일반적인 스키마 선언을 보여 줍니다.

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://www.w3XML.com"
   xmlns="http://www.w3XML.com"
   elementFormDefault="qualified">...
...</xs:schema>
```

다음 테이블에는 네임스페이스 선언의 각 부분이 설명되어 있습니다.

스키마 선언	설명
xmlns:xs="http://www.w3.org/2001/XMLSchema"	원시 XML 스키마 및 데이터 유형을 포함하는 네임스페이스입니다. 이 예에서 각 스키마 구성 요소에는 접두사 “xs”가 있습니다.
targetNamespace="http://www.w3XML.com"	스키마를 포함하는 네임스페이스입니다.

스키마 선언	설명
xmlns="http://www.w3XML.com"	기본 네임스페이스 선언입니다. 접두사가 없는 스키마의 모든 요소가 기본 네임스페이스에 속합니다. 접두사 없이 xmlns 특성을 사용하여 기본 네임스페이스를 선언합니다.
elementFormDefault="qualified"	스키마의 모든 요소가 XML 파일에서 네임스페이스를 가져야 한다는 것을 지정합니다.

이름

XML 파일에서 각 태그는 요소 또는 특성의 이름입니다. DTD 파일에서 태그 <!ELEMENT>는 요소의 이름을 지정하고 태그 <!ATTLIST>는 요소에 대한 특성 집합을 나타냅니다. 스키마 파일에서 <element name>은 요소의 이름을 지정하고 <attribute name>은 특성의 이름을 지정합니다.

XML 정의를 가져오는 경우 요소 태그는 기본적으로 PowerCenter 정의의 열 이름이 됩니다.

계층

XML 파일은 계층 데이터베이스를 모델링합니다. XML 계층에서 요소의 위치는 다른 요소에 대한 관계를 나타냅니다. 예를 들어 요소는 하위 요소를 포함할 수 있으며 다른 요소에서 특성을 상속받을 수 있습니다.

카디널리티

DTD 또는 스키마 파일의 요소 카디널리티는 XML 파일에서 요소가 발생하는 횟수입니다. 요소 카디널리티는 XML 정의에서 그룹을 구조화하는 방식에 영향을 미칩니다. 요소의 절대 카디널리티 및 상대 카디널리티는 XML 정의의 구조에 영향을 미칩니다.

절대 카디널리티

요소의 절대 카디널리티는 XML 계층의 상위 요소 내에서 요소가 발생하는 횟수입니다. DTD 및 XML 스키마 파일은 계층 내에서 요소의 절대 카디널리티를 설명합니다. DTD 파일은 기호를 사용하고 XML 스키마 파일은 <minOccurs> 및 <maxOccurs> 특성을 사용하여 요소의 절대 카디널리티를 설명합니다.

예를 들어 요소가 상위 요소 내에서 한 번 발생하는 경우 요소의 절대 카디널리티는 한 번(1)입니다. 그러나 상위 요소의 카디널리티가 1 이상(+)인 경우 XML 계층 내에서 요소가 여러 번 발생할 수 있습니다.

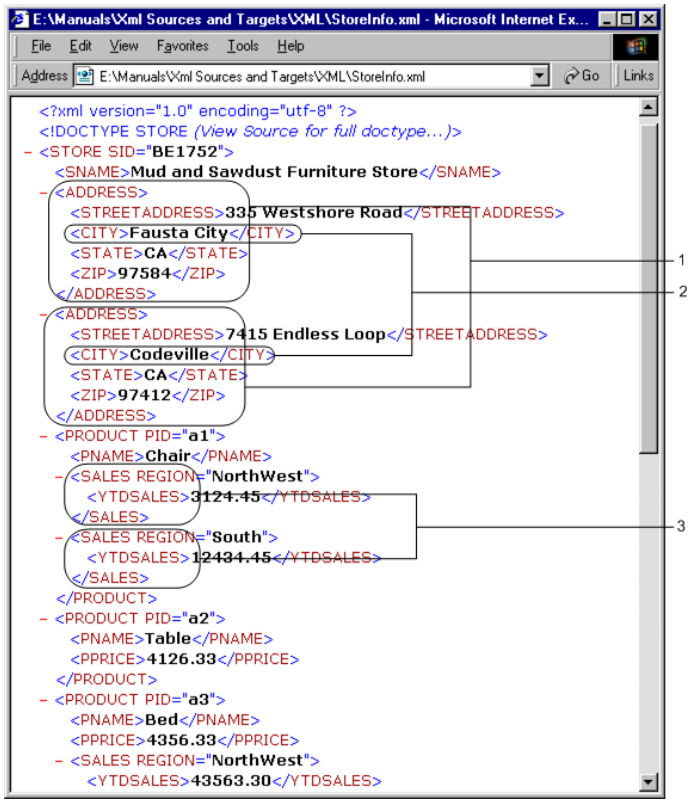
요소의 절대 카디널리티에 따라 해당 Null 제약 조건이 결정됩니다. 절대 카디널리티가 1 이상(+)인 요소는 Null 값을 가질 수 없지만 카디널리티가 0 이상(*)인 요소는 Null 값을 가질 수 있습니다. XML 스키마 또는 DTD 파일에서 고정 또는 필수로 표시된 특성은 Null 값을 가질 수 없지만 암시된 특성은 Null 값을 가질 수 있습니다.

다음 테이블에는 DTD 및 XML 스키마 파일이 카디널리티를 나타내는 방법이 설명되어 있습니다.

절대 카디널리티	DTD	스키마
0 또는 한 번	?	minOccurs=0 maxOccurs=1
0 또는 한 번 이상	*	minOccurs=0 maxOccurs=unbounded minOccurs=0 maxOccurs=n

절대 카디널리티	DTD	스키마
한 번	-	minOccurs=1 maxOccurs=1
한 번 이상	+	minOccurs=1 maxOccurs=unbounded minOccurs=1 maxOccurs=n
참고: 스키마에서 최대 발생 횟수 또는 무제한 발생을 선언할 수 있습니다.		

다음 그림은 샘플 XML 파일에서 요소의 절대 카디널리티를 보여 줍니다.



1. 요소 *Address*는 *Store* 내에서 두 번 이상 발생합니다. 해당 절대 카디널리티가 한 번 이상(+)입니다.
2. 요소 City는 상위 요소 Address 내에서 한 번 발생합니다. 해당 절대 카디널리티가 한 번(1)입니다.
3. 요소 Sales는 상위 요소 Product 내에서 0번 이상 발생합니다. 해당 절대 카디널리티가 0번 이상(*)입니다.

상대 카디널리티

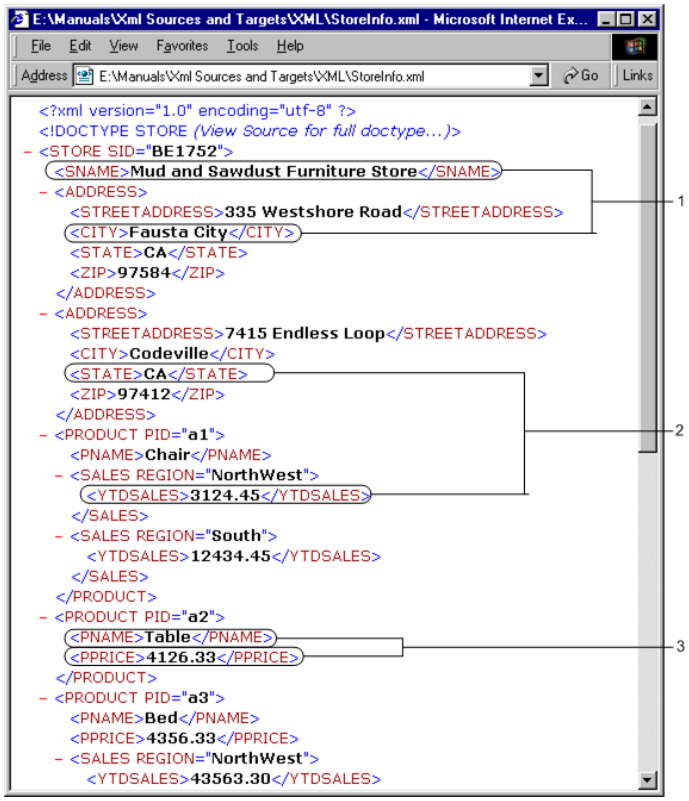
상대 카디널리티는 XML 계층의 다른 요소에 대한 요소의 관계입니다. 요소는 계층의 다른 요소에 대해 일대일, 일대다, 다대다 관계를 가질 수 있습니다.

어떤 요소가 발생할 때마다 다른 요소가 한 번 발생할 수 있는 경우 어떤 요소는 다른 요소와 일대일 관계를 가집니다. 예를 들어 한 직원 요소가 한 개의 사회 보장 번호 요소를 가질 수 있습니다. 직원과 사회 보장 번호는 일대일 관계입니다.

어떤 요소가 발생할 때마다 다른 요소가 여러 번 발생할 수 있는 경우 어떤 요소는 다른 요소와 일대다 관계를 가집니다. 예를 들어 한 직원 요소가 여러 이메일 주소를 가질 수 있습니다. 직원과 이메일 주소는 일대다 관계입니다.

XML 파일이 여러 번 발생하는 두 요소를 가질 수 있는 경우 어떤 요소와 다른 요소의 관계는 다대다입니다. 예를 들어 한 직원이 여러 개의 이메일 주소와 여러 개의 거리 주소를 가질 수 있습니다. 이메일 주소와 거리 주소는 다대다 관계입니다.

다음 그림은 샘플 XML 파일에서 요소 간 상대 카디널리티를 보여 줍니다.



1. 일대다 관계입니다. SNAME이 발생할 때마다 ADDRESS가 여러 번 발생할 수 있으므로 CITY가 여러 번 발생할 수 있습니다.
2. 다대다 관계입니다. STATE가 발생할 때마다 YTDSALES가 여러 번 발생할 수 있습니다. YTDSALES가 발생할 때마다 STATE가 여러 번 발생할 수 있습니다.
3. 일대일 관계입니다. PNAME이 발생할 때마다 PPRICE가 한 번 발생합니다.

단순 및 복합 XML 유형

XML 스키마 언어에는 숫자, 문자열, 시간, XML 및 이진을 비롯한 40개가 넘는 기본 제공 데이터 유형이 있습니다. 이러한 데이터 유형을 단순 유형이라고 합니다. 이 유형은 텍스트는 포함하지만 다른 요소 및 특성은 포함하지 않습니다. 기본 XML 단순 유형에서 새 단순 유형을 파생시킬 수 있습니다.

복합 XML 데이터 유형을 작성할 수 있습니다. 복합 데이터 유형은 둘 이상의 단순 유형이 포함된 데이터 유형입니다. 또한 복합 데이터 유형은 다른 복합 유형 및 특성을 포함할 수 있습니다.

XML 데이터 유형에 대한 자세한 내용은 http://www.w3.org/TR/xmlschema-2에서 XML 데이터 유형에 대한 W3C 사양을 참조하십시오.

단순 유형

단순 데이터 유형은 텍스트를 포함하는 XML 요소 또는 특성입니다. 단순 유형은 나눌 수 없습니다. 단순 유형은 특성을 가질 수 없지만 특성은 단순 유형입니다.

PowerCenter는 다음 단순 유형을 지원합니다.

- **원자성 유형.** 부울, 문자열 또는 정수와 같은 기본 데이터 유형입니다.
- **목록.** 원자성 유형의 배열 컬렉션입니다.
- **합집합.** XML 파일에서 단순 유형으로 매핑하는 하나 이상의 원자성 또는 목록 유형의 조합입니다.

원자성 유형

원자성 데이터 유형은 부울, 문자열, 정수, 10진수 또는 날짜와 같은 기본 데이터 유형입니다. 사용자 지정 원자성 데이터 유형을 정의하려면 원자성 데이터 유형에 제한을 추가하여 콘텐츠를 제한합니다. 패싯을 사용하여 제한하거나 허용할 값을 정의합니다.

패싯은 최소값 또는 최대값, 특정 값 또는 유효한 값의 데이터 패턴을 정의하는 표현식입니다. 예를 들어 패턴 패싯은 요소를 데이터 값의 표현식으로 제한합니다. 열거 패싯은 요소에 대한 올바른 값을 나열합니다.

다음 예는 요소를 a - z의 소문자로 제한하는 패턴 패싯을 포함합니다.

```
<xs:element name="letter">
   <xs:simpleType>
   <xs:restriction base="xs:string">
      <xs:pattern value="[a-z]"/>
   </xs:restriction>
</xs:simpleType></xs:element>
```

다음 예는 문자열을 a, b 또는 c로 제한하는 열거 패싯을 포함합니다.

```
<xs:element name="letter">
   <xs:simpleType>
   <xs:restriction base="xs:string">
      <xs:enumeration value="a"/>
      <xs:enumeration value=”b”/>
      <xs:enumeration value=”c”/>
   </xs:restriction>
</xs:simpleType></xs:element>
```

목록

목록은 이름을 나타내는 문자열 목록과 같은 원자성 유형의 배열 컬렉션입니다. 목록 항목 유형은 목록 구성 요소의 데이터 유형을 정의합니다.

다음 예는 names라는 목록을 보여 줍니다.

```
<xs:simpleType name="names">
   <xs:list itemType="xs:string" />
</xs:simpleType>
```

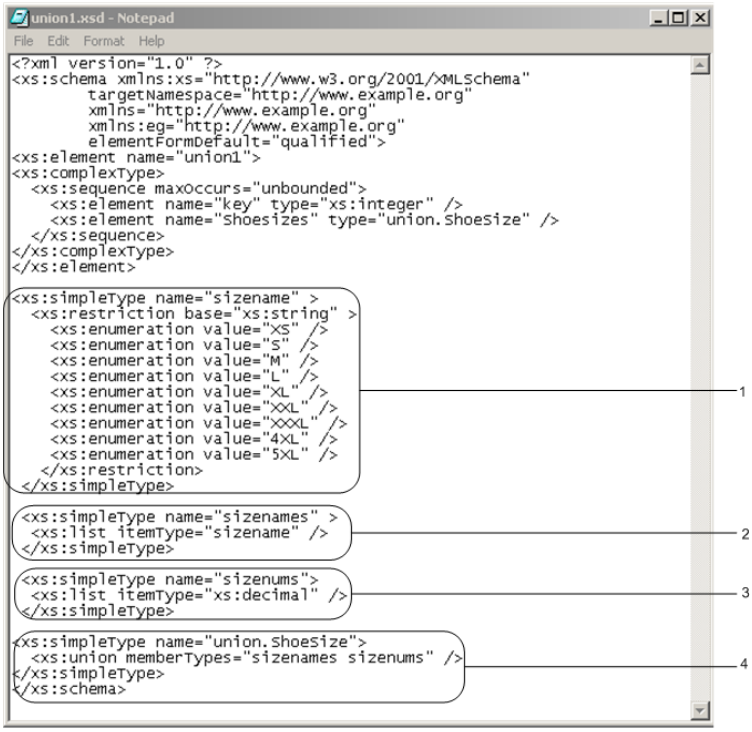
XML 파일은 names 목록에 다음과 같은 데이터를 포함할 수 있습니다.

```
<names>Joe Bob Harry Atlee Will</names>
```

합집합

합집합은 XML 파일에서 하나의 단순 유형으로 매핑하는 하나 이상의 원자성 또는 목록 유형의 조합입니다. 합집합 유형을 정의할 때 결합할 유형을 지정합니다. 예를 들어 크기라는 유형을 작성할 수 있습니다. 크기는 S, M 및 L과 같은 문자열 데이터를 포함할 수 있거나 30, 32 및 34와 같은 10진수 크기를 포함할 수 있습니다. 합집합 유형 요소를 정의하는 경우 XML 파일에 문자열 크기의 경우 sizename 유형을, 숫자 크기의 경우 sizenum 유형을 포함할 수 있습니다.

다음 그림은 sizenames 및 sizenums 목록이 들어 있는 shoesize 합집합이 포함된 스키마 파일을 보여 줍니다.



1. sizename**은 제한된 문자열 유형입니다**.
2. sizenames **유형은 문자열 목록을 허용합니다**.
3. sizenums **유형은** 10**진수 목록을 허용합니다**.
4. shoesize **합집합은** 10**진수와 문자열 목록을 모두 허용합니다**.

합집합은 sizenames 및 sizenums를 합집합 멤버 유형으로 정의합니다. sizenames는 문자열 값 목록을 정의합니다. sizenums는 10진수 값 목록을 정의합니다.

복합 유형

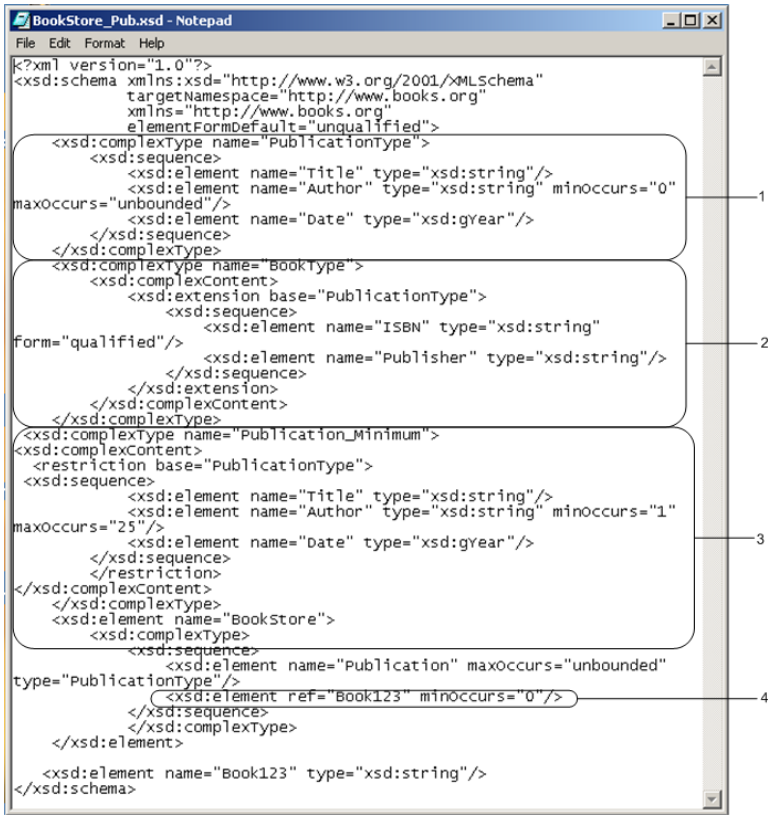
복합 유형은 단순 유형 컬렉션을 논리 단위로 집계합니다. 예를 들어 고객 유형에는 고객 번호, 이름, 거리 주소, 타운, 도시 및 우편 번호가 포함될 수 있습니다. 또한 복합 유형은 다른 복합 유형 또는 요소 및 특성 그룹을 참조할 수 있습니다.

XML은 복합 유형 상속을 지원합니다. 복합 유형을 정의하는 경우 기본 유형의 구성 요소를 상속받는 다른 복합 유형을 작성할 수 있습니다. 유형 관계에서 기본 유형은 다른 유형을 파생하는 복합 유형입니다. 파생된 복합 유형은 기본 유형의 요소를 상속받습니다.

확장된 복합 유형은 기본 유형에서 요소를 상속받는 파생된 유형이며 추가 요소를 포함합니다. 예를 들어 customer_purchases 유형은 customer 복합 유형에서 해당 정의를 상속받을 수 있지만 customer_purchases 유형이 item, cost 및 date_sold 요소를 추가합니다.

제한된 복합 유형은 기본 유형에서 일부 요소를 제한하는 파생된 유형입니다. 예를 들어 mail_list는 customer에서 요소를 상속받을 수 있지만 minoccurs 및 maxoccurs 경계를 0으로 설정하여 phone_number 요소를 제한합니다.

다음 그림은 기본 복합 유형을 제한하고 확장하는 파생된 복합 유형을 보여 줍니다.



1. 기본 복합 유형
2. 확장된 복합 유형
3. 제한된 복합 유형
4. 요소 참조

위의 그림에서 기본 유형은 PublicationType입니다. BookType은 PublicationType을 확장하고 ISBN 및 Publisher 요소를 포함합니다. Publication_Minimum은 PublicationType을 제한합니다. Publication_Minimum은 1에서 25 사이의 Authors가 필요하고 날짜를 년으로 제한합니다.

추상 요소

경우에 따라 스키마가 복합 요소의 기본 구조를 정의하는 기본 유형을 포함하지만 일부 구성 요소는 포함하지 않습니다. 파생된 복합 유형은 더 많은 구성 요소를 사용하여 기본 유형을 확장합니다. 기본 유형은 전체 정의가 아니므로 XML 파일에서 기본 유형을 사용하고 싶지 않을 수 있습니다. 추상화할 기본 유형 요소를 선언할 수 있습니다. 추상 요소는 XML 파일에서 유효하지 않습니다. 파생된 요소만 유효합니다.

추상 요소를 정의하려면 "true" 값을 가진 추상 특성을 추가합니다. 기본값은 false입니다.

예를 들어 PublicationType은 추상 요소입니다. BookType은 PublicationType에서 요소를 상속받지만 ISBN 및 Publisher 요소도 포함합니다. PublicationType은 추상이므로 PublicationType 요소는 XML 파일에서 유효하지 않습니다. XML 파일이 파생된 유형인 BookType을 포함할 수 있습니다.

다음 스키마는 PublicationType 및 BookType을 포함합니다.

```
<xsd:complexType name="PublicationType" abstract="true">
   <xsd:sequence>
     <xsd:element name="Title" type="xsd:string"/>
     <xsd:element name="Author" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
     <xsd:element name="Date" type="xsd:gYear"/>
   </xsd:sequence>
   </xsd:complexType>
<xsd:complexType name="BookType">
   <xsd:complexContent>
     <xsd:extension base="PublicationType" >
       <xsd:sequence>
          <xsd:element name="ISBN" type="xsd:string"/>
           <xsd:element name="Publisher" type="xsd:string"/>
        </xsd:sequence>
     </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
```

모든 유형 요소 및 특성

일부 스키마 요소 및 특성은 XML 파일에서 데이터 유형을 허용합니다. 식별되지 않은 요소 및 특성 유형이 있는 XML 파일의 유효성을 검사해야 하는 경우 이러한 요소 및 특성을 사용합니다.

모든 유형의 데이터를 허용하는 다음 요소 및 특성을 사용합니다.

- **anyType 요소.** 요소가 연결된 XML 파일의 모든 데이터 유형이 될 수 있습니다.
- **anySimpleType 요소.** 요소가 연결된 XML 파일의 모든 simpleType이 될 수 있습니다.
- **ANY 콘텐츠 요소.** 요소가 스키마에서 이미 정의된 모든 요소가 될 수 있습니다.
- **anyAttribute 특성.** 요소가 스키마에서 이미 정의된 모든 특성이 될 수 있습니다.

anyType 요소

anyType 요소가 XML 인스턴스 문서의 모든 데이터 유형이 될 수 있습니다. 요소가 여러 유형의 데이터를 포함하는 경우 요소가 anyType이 되도록 선언합니다.

다음 스키마는 firstname, lastname 및 anyType의 age 요소를 사용하여 person을 설명합니다.

```
<xs:element name="person">
   <xs:complexType>
   <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
      <<xs:element name="age" type="xs:anyType"/>
   </xs:sequence>
   </xs:complexType>
</xs:element>
```

다음 XML 인스턴스 문서는 age 요소에 날짜 유형 및 숫자를 포함합니다.

```
<person>
   <firstname>Danny</firstname>
```

```
   <lastname>Russell</lastname>
   <age>1959-03-03</age>
</person>
<person>
   <firstname>Carla</firstname>
   <lastname>Havers</lastname>
   <age>46</age>
</person>
```

두 유형 모두 스키마에 유효합니다. 스키마에서 요소에 대한 데이터 유형을 선언하지 않은 경우 디자이너에서 스키마를 가져올 때 요소 기본값은 anyType입니다.

anySimpleType 요소

anySimpleType 요소는 모든 원자성 유형을 포함할 수 있습니다. 원자성 유형은 부울, 문자열, 정수, 10진수 또는 날짜와 같은 기본 데이터 유형입니다.

다음 스키마는 firstname, lastname 및 anySimpleType의 other 요소를 사용하여 person을 설명합니다.

```
<xs:element name="person">
   <xs:complexType>
   <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
      <xs:element name="other" type="xs:anySimpleType"/>
   </xs:sequence>
   </xs:complexType>
</xs:element>
```

다음 XML 인스턴스 문서는 anySimpleType 요소를 문자열 데이터 유형으로 대체합니다.

```
<person>
   <firstname>Kathy</firstname>
   <lastname>Russell</lastname>
   <other>Cissy</other>
</person>
```

다음 XML 인스턴스 문서는 anySimpleType 요소를 숫자 데이터 유형으로 대체합니다.

```
<person>
   <firstname>Kathy</firstname>
   <lastname>Russell</lastname>
   <other>34</other>
</person>
```

ANY 콘텐츠 요소

ANY 콘텐츠 요소는 XML 파일에서 모든 콘텐츠를 허용합니다. 스키마에서 ANY 콘텐츠 요소를 선언하는 경우 XML 인스턴스 문서의 모든 이름 및 유형의 요소를 ANY 콘텐츠로 대체할 수 있습니다. 대체 요소가 스키마에 존재해야 합니다.

ANY 콘텐츠를 지정하는 경우 요소 이름 및 요소 유형 대신 키워드 ANY를 사용합니다.

다음 스키마는 firstname, lastname 및 ANY 콘텐츠의 요소를 사용하여 person을 설명합니다.

```
<xs:element name="person">
   <xs:complexType>
   <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
      <xs:any minOccurs="0"/>
```

```
   </xs:sequence>
   </xs:complexType>
</xs:element>
<xs:element name="son" type="xs:string"/>
<xs:element name=”daughter” type="xs:string"/>
```

스키마가 son 요소 및 daughter 요소를 포함합니다. XML 인스턴스 문서의 son 또는 daughter 요소를 ANY 요소로 대체할 수 있습니다.

```
<person>
   <firstname>Danny</firstname>
   <lastname>Russell</lastname>
   <son>Atlee</son>
</person>
<person>
   <firstname>Christine</firstname>
   <lastname>Slade</lastname>
   <daughter>Susie</daughter>
</person>
```

AnyAttribute 특성

anyAttribute 특성은 XML 파일의 모든 특성을 허용합니다. 특성을 anyAttribute로 선언하는 경우 스키마의 모든 특성을 anyAttribute 요소로 대체할 수 있습니다.

다음 스키마는 firstname, lastname 및 anyAttribute의 특성을 사용하여 person을 설명합니다.

```
<xs:element name="person">
<xs:complexType>
   <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
   </xs:sequence>
   <xs:anyAttribute/>
</xs:complexType>
</xs:element>
```

다음 스키마는 성별 특성을 포함합니다.

```
<xs:attribute name="gender">
   <xs:simpleType>
   <xs:restriction base="xs:string">
   <xs:pattern value="male|female"/>
   </xs:restriction>
</xs:simpleType>
</xs:attribute>
```

다음 XML 인스턴스 문서는 anyAttribute를 성별 특성으로 대체합니다.

```
<person gender="female">
   <firstname>Anita</firstname>
   <lastname>Ficks</lastname>
</person>
<person gender="male">
   <firstname>Jim</firstname>
   <lastname>Geimer</lastname>
</person>
```

구성 요소 그룹

XML 스키마에서 다음 그룹의 구성 요소를 작성할 수 있습니다.

- **요소 및 특성 그룹.** 스키마를 통해 참조할 수 있는 요소 또는 특성 그룹입니다.
- **대체 그룹**. 동일한 그룹의 다른 요소로 대체할 수 있는 요소 그룹입니다.

요소 및 특성 그룹

스키마에서 참조할 수 있는 그룹의 요소 및 특성을 설정할 수 있습니다. 그룹을 참조하려면 먼저 요소 또는 특성 그룹을 선언해야 합니다.

다음 예는 요소 그룹의 스키마 구문을 보여 줍니다.

```
<xs:group name="Songs">
      <xs:element name="songTitle" type="xs:string" />
      <xs:element name="artist" type="xs:string" />
      <xs:element name="publisher" type="xs:string" />
  </xs:group>
```

다음 예는 특성 그룹의 스키마 구문을 보여 줍니다.

```
<xs:attributeGroup name="Songs">
   <xs:attribute name="songTitle" type="xs:string" />
   <xs:attribute name="artist" type="xs:string" />
   <xs:attribute name="publisher" type="xs:string" />
</xs:attributeGroup>
```

다음 요소 그룹은 XML 데이터에 대한 제약 조건을 보여 줍니다.

- **Sequence 그룹.** XML 파일의 모든 요소는 스키마가 요소를 나열하는 순서대로 발생해야 합니다. 예를 들어 OrderHeader에는 customerName, orderNumber, orderDate가 차례대로 필요합니다.

  ```
  <xs:group name="OrderHeader">
     <xs:sequence>
        <xs:element name="customerName" type="xs:string" />
        <xs:element name="orderNumber" type="xs:number" />
        <xs:element name="orderDate" type="xs:date" />
     </xs:sequence>
  </xs:group>
  ```

- **선택 그룹**. 그룹의 한 요소가 XML 파일에서 발생할 수 있습니다. 예를 들어 CustomerInfo 그룹은 XML 파일에 대해 선택한 요소를 나열합니다.

  ```
  <xs:group name="CustomerInfo">
     <xs:choice>
        <xs:element name="customerName" type="xs:string" />
        <xs:element name="customerID" type="xs:number" />
        <xs:element name="customerNumber" type="xs:integer" />
     </xs:choice>
  </xs:group>
  ```

- **All 그룹.** 모든 요소가 XML 파일에서 발생하거나 전혀 발생하지 않아야 합니다. 요소는 순서에 상관 없이 발생할 수 있습니다. 예를 들어 CustomerInfo에 다음 세 가지 요소가 모두 필요하거나 전혀 필요하지 않습니다.

  ```
  <xs:group name="CustomerInfo">
     <xs:all>
        <xs:element name="customerName" type="xs:string" />
        <xs:element name="customerAddress" type="xs:string" />
        <xs:element name="customerPhone" type="xs:string" />
  ```

```
      </xs:all>
    </xs:group>
```

대체 그룹

대체 그룹을 사용하여 XML 파일에서 한 요소를 다른 요소로 바꿉니다. 예를 들어 캐나다와 미국 주소가 있는 경우 캐나다에는 주소 유형을 작성하고 미국에는 다른 유형을 작성할 수 있습니다. 어떤 유형의 주소든 허용하는 대체 그룹을 작성할 수 있습니다.

다음 스키마 조각은 Address 기본 유형 및 파생된 유형 CAN_Address 및 USA_Address를 보여 줍니다.

```
<xs:complexType name="Address">
   <xs:sequence>
      <xs:element name="Name" type="xs:string" />
      <xs:element name="Street" type="xs:string"
       minOccurs="1" maxOccurs="3" />
      <xs:element name="City" type="xs:string" />
   </xs:sequence>
</xs:complexType>
<xs:element name="MailAddress" type="Address" />
<xs:complexType name="CAN_Address">
   <xs:complexContent>
   <xs:extension base="Address">
      <xs:sequence>
      <xs:element name="Province" type="xs:string" />
      <xs:element name="PostalCode" type="CAN_PostalCode"/>
      </xs:sequence>
   </xs:extension>
   </xs:complexContent>
</xs:complexType>
<xs:complexType name="USA_Address">
   <xs:complexContent>
   <xs:extension base="Address">
      <xs:sequence>
      <xs:element name="State" type="USPS_StateCode" />
      <xs:element name="ZIP" type="USPS_ZIP"/>
      </xs:sequence>
   </xs:extension>
   </xs:complexContent>
</xs:complexType>
<xs:element name="AddrCAN" type="CAN_Address"
substitutionGroup="MailAddress"/>
<xs:element name="AddrUSA" type="USA_Address"
substitutionGroup="MailAddress"/>
```

CAN_Address는 Province 및 PostalCode를 포함하고 USA_Address는 State 및 Zip을 포함합니다. MailAddress 대체 그룹은 두 주소 유형을 모두 포함합니다.

관련 항목:

- "XML 정의에서 대체 그룹 사용" 페이지 45

XML 경로

XMLPath(XPath)는 XML 파일에서 항목을 찾기 위한 방법을 설명하는 언어입니다. XPath는 루트에서 요소 또는 특성까지의 계층 라우트를 기반으로 하는 주소 지정 구문을 사용합니다. XML 경로는 긴 스키마 구성 요소 이름을 포함할 수 있습니다.

XPath는 슬래시(/)를 사용하여 계층의 요소를 구분합니다. XPath에서 XML 특성 앞에는 “@”이 있습니다.

요소 또는 특성 XPath에 대한 쿼리를 작성하여 XML 데이터를 필터링할 수 있습니다.

관련 항목:

- “XPath 쿼리 조건자 사용” 페이지 47

코드 페이지

XML 파일은 파일에 사용되는 코드 페이지를 나타내는 인코딩 선언을 포함합니다. XML에서 가장 일반적인 코드 페이지는 UTF-8 및 UTF-16입니다. 모든 XML 파서는 이러한 코드 페이지를 지원합니다. XML 문자 인코딩 사양에 대한 자세한 내용은 W3C 웹 사이트(http://www.w3c.org)를 참조하십시오.

PowerCenter는 관계형 데이터베이스 및 다른 플랫 파일에 대해 지원하는 XML 파일에 대한 동일한 코드 페이지 집합을 지원합니다. PowerCenter는 사용자 정의 코드 페이지를 지원하지 않습니다.

XML 소스 및 대상 정의를 작성하는 경우 디자이너는 PowerCenter 클라이언트 코드 페이지를 정의에 할당합니다. 코드 페이지 할당이 포함된 XML 스키마를 가져오는 경우 XML 마법사가 스키마에서 가져온 코드 페이지를 표시합니다. 하지만 XML 마법사는 해당 코드 페이지를 리포지토리에서 작성하는 XML 정의에 적용하지 않습니다.

XML 소스 정의를 위한 코드 페이지를 구성할 수 없습니다. 구문 분석할 때 통합 서비스가 XML 소스 파일을 유니코드로 변환합니다.

디자이너에서 대상 XML 정의를 위한 코드 페이지를 구성할 수 있습니다. 또한 세션 속성에서 XML 대상 인스턴스의 코드 페이지를 변경할 수도 있습니다.

PowerCenter에서 XML 사용

PowerCenter에서 XML 사용 개요

XML 파일, DTD 파일, XML 스키마, 플랫 파일 정의 또는 관계형 테이블 정의에서 PowerCenter의 XML 정의를 작성할 수 있습니다. XML 정의를 작성하는 경우 디자이너가 XML 메타데이터를 추출하고 리포지토리에 스키마를 작성합니다. 스키마는 XML 정의를 편집하고 유효성을 검사하는 구조를 제공합니다.

XML 정의는 여러 그룹을 포함할 수 있습니다. XML 정의에서는 그룹을 보기라고 합니다. XML 계층의 요소 간 관계는 보기 간 관계를 정의합니다.

XML 정의를 작성하는 경우 디자이너가 기본적으로 다중 발생 요소 및 복합 유형에 대한 보기를 스키마에 작성합니다. XML 계층에서 요소의 상대 카디널리티는 PowerCenter가 XML 정의에서 보기를 작성하는 방식에 영향을 미칩니다. 상대 카디널리티는 요소가 동일한 보기에 포함될 수 있는지 결정합니다.

디자이너는 키에 따라 XML 정의에서 보기 간 관계를 정의합니다. 소스 정의는 키를 요구하지 않지만 대상 보기는 키가 있어야 합니다. 각 보기에는 XML 요소 또는 생성된 키인 기본 키가 있습니다.

XML 정의를 작성하는 경우 XML 데이터의 계층 모델 또는 항목 관계 모델을 작성할 수 있습니다. 계층 모델을 작성하는 경우 정규화된 계층 또는 비정규화된 계층을 작성합니다. 정규화된 계층에는 다중 발생 요소에 대한 별도의 보기가 있습니다. 비정규화된 계층에는 다중 발생 요소에 대한 중복 데이터를 가진 하나의 보기가 있습니다.

항목 모델을 작성하는 경우 디자이너가 복합 유형 및 다중 발생 요소에 대한 보기를 작성합니다. 디자이너는 스키마가 제공하는 원형 관계 및 상속을 모델링하는 XML 정의를 작성합니다.

PowerCenter는 요소가 400개 미만인 XML 스키마 작업을 수행할 수 있습니다. PowerCenter 프로필은 최대 3개의 계층 수준을 포함할 수 있으며 다음과 같은 복합 유형 요소를 포함할 수 있습니다.

- 시퀀스
- 임의
- 선택

PowerCenter XML 가져오기 마법사는 최대 400개의 보기를 작성할 수 있습니다.

제한

다음 제한은 PowerCenter에서 XML을 처리할 때 적용됩니다.

- XML 스키마는 요소가 400개 미만이어야 합니다.
- XML 스키마 파일은 100KB보다 작아야 합니다.
- XML 파일 크기는 10MB 이하여야 합니다.
- 복잡성 프로필은 계층 수준이 3개로 제한됩니다.
- XML 가져오기 마법사는 최대 400개의 보기를 작성합니다.

PowerCenter는 다음 함수를 지원하지 않습니다.

- **연결된 열.** 열은 두 요소의 연결이 될 수 없습니다. 예를 들어 두 요소 FIRSTNAME 및 LASTNAME의 연결을 참조하는 열 FULLNAME을 작성할 수 없습니다.
- **복합 키.** 키는 두 요소의 연결이 될 수 없습니다. 예를 들어 두 요소 LASTNAME 및 PHONENUMBER의 연결을 참조하는 키 CUSTOMERID를 작성할 수 없습니다.
- **구문 분석된 목록.** PowerCenter는 목록 유형을 모든 배열 요소를 포함하는 한 문자열로 저장합니다. PowerCenter는 문자열에서 개별 단순 유형을 구문 분석하지 않습니다.

다른 요소 유형을 가진 변환을 작성하고 더 큰 XML 입력 파일을 변환하려면 데이터 프로세서 변환을 사용합니다. 데이터 프로세서 변환을 작성하는 방법에 대한 자세한 내용은 *Informatica Data Transformation 사용자 가이드* 및 *Informatica Data Transformation 시작하기 가이드*를 참조하십시오.

XML 메타데이터 가져오기

XML 정의를 가져오는 경우 디자이너가 정의를 위해 리포지토리에 스키마를 작성합니다. 리포지토리 스키마는 XML 정의를 편집하고 유효성을 검사하는 구조를 제공합니다.

다음 파일 유형에서 메타데이터를 작성할 수 있습니다.

- XML 파일
- DTD 파일
- XML 스키마 파일
- 관계형 테이블
- 플랫 파일

XML 파일에서 메타데이터 가져오기

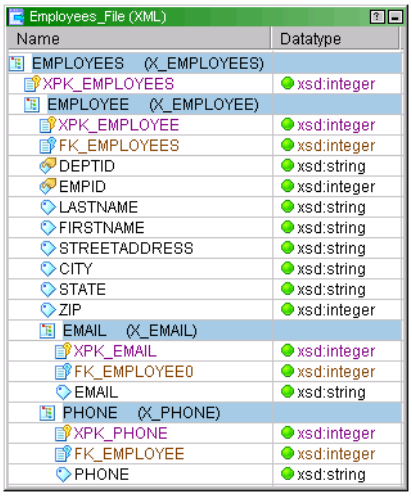
XML 파일에서 태그 쌍은 각 데이터 요소의 시작과 끝을 표시합니다. 이러한 태그는 PowerCenter가 XML 파일에서 추출하는 메타데이터의 기준입니다. 연결된 DTD 또는 XML 스키마가 없는 XML 파일을 가져오는 경우 디자이너가 XML 태그를 읽어 요소, 가능한 발생, 계층에서의 위치를 확인합니다. 디자이너는 요소 태그 내의 데이터를 확인하여 데이터 표현에 따라 데이터 유형을 할당합니다. XML 정의에서 이러한 요소에 대한 데이터 유형을 변경할 수 있습니다.

다음 그림은 샘플 XML 파일을 보여 줍니다.

루트 요소는 Employees입니다. Employee는 다중 발생 요소입니다. Employee 요소는 LastName, FirstName 및 Address를 포함합니다. 또한 Employee 요소는 다중 발생 요소 Phone 및 Email을 포함합니다.

디자이너는 XML 데이터에서 스키마 구조를 결정합니다.

다음 그림은 루트 요소 및 다중 발생 요소에 대해 별도의 보기가 포함된 기본 XML 소스 정의를 보여 줍니다.



XML 파일을 가져오는 경우 XML 정의를 작성하기 위해 모든 XML 데이터가 필요하지는 않지만 XML 파일 계층을 정확하게 표시하려면 충분한 데이터가 있어야 합니다.

디자이너는 DTD 파일 또는 XML 스키마를 참조하는 XML 파일에서 XML 정의를 작성할 수 있습니다. XML 파일에 다른 노드의 DTD 또는 XML 스키마에 대한 참조가 있는 경우 PowerCenter 클라이언트를 호스트하는 노드는 디자이너가 스키마를 읽을 수 있도록 스키마가 있는 노드에 대한 액세스 권한이 있어야 합니다. XML 파일은 DTD 또는 XML 스키마의 주소인 URI(Universal Resource Identifier)를 포함합니다.

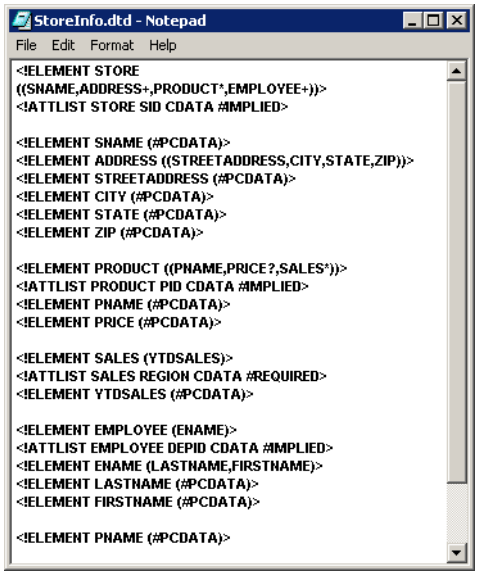
DTD 파일에서 메타데이터 가져오기

DTD 파일은 XML 문서 구조에 대한 제약 조건을 제공합니다. DTD 파일은 XML 문서에 대한 요소, 특성, 항목 및 주석을 나열합니다. DTD 파일은 구성 요소 간 관계를 지정합니다. DTD는 카디널리티 및 Null 제약 조건을 지정합니다. 그러나 DTD 파일은 데이터 또는 데이터 유형을 포함하지 않습니다.

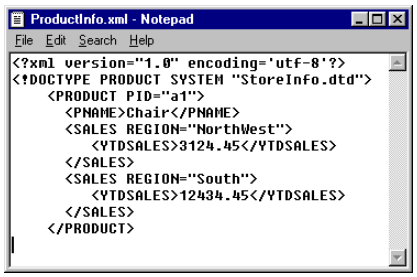
DTD 파일을 가져오는 경우 XML 정의의 요소에 대한 데이터 유형을 변경할 수 있습니다. Null 제약 조건을 변경할 수 있지만 요소 카디널리티는 변경할 수 없습니다.

연결된 DTD가 포함된 XML 파일을 가져오는 경우 디자이너가 DTD 구조에 따라 정의를 작성합니다.

다음 그림은 StoreInfo.dtd가 Store 요소를 포함하고 Product가 Store의 하위 요소 중 하나인 XML 파일의 예를 보여 줍니다.

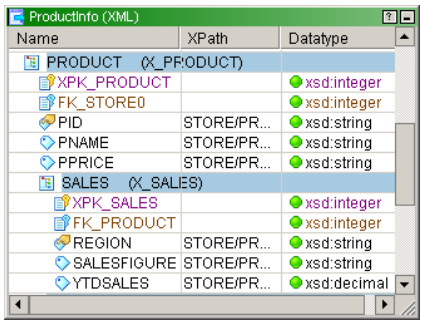


다음 그림은 연결된 DTD를 보여 줍니다.



연결된 DTD에서 ProductInfo.xml은 StoreInfo.dtd의 Product 요소를 사용합니다. Product는 다중 발생 Sales 요소를 포함합니다.

다음 그림은 디자이너가 작성하는 소스 정의를 보여 줍니다.



ProductInfo 정의는 Product 및 Sales 그룹을 포함합니다. XML 파일은 정의에 포함할 요소를 결정합니다. DTD 파일은 XML 정의의 구조를 결정합니다.

XML 스키마에서 메타데이터 가져오기

스키마 파일은 XML 파일의 요소 및 특성에 대한 구조를 정의합니다. 스키마 파일은 파일의 요소 및 특성 유형에 대한 설명을 포함합니다. XML 스키마를 가져오는 경우 디자이너가 요소의 데이터 유형, 전체 자릿수 및 카디널리티를 결정합니다. 요소 정의를 스키마에서 가져오는 경우 PowerCenter에서 요소 정의를 변경할 수 없습니다.

XML 스키마에서 메타데이터를 가져오는 경우 .xsd 파일은 다른 .xsd 파일을 참조하는 import 또는 include 문을 포함할 수 있습니다. 다른 스키마를 포함하는 스키마를 가져오는 경우 다른 스키마는 동일한 네임스페이스를 참조해서는 안 됩니다.

예:

```
<IMPORT
schemaLocation="../../../administration/process/bo/LocationTextBO.xsd"
namespace="http://EnterpriseLibrary/com/acs/enterprise/common/program/administration/process/bo">
<IMPORT
schemaLocation="../../../administration/process/bo/LineOfBusinessBO.xsd"
namespace="http://EnterpriseLibrary/com/acs/enterprise/common/program/administration/process/bo">
<IMPORT
schemaLocation="../../../administration/process/bo/ClaimExceptionBO.xsd"
namespace="http://EnterpriseLibrary/com/acs/enterprise/common/program/administration/process/bo">
```

여러 "import schemalocation" 문을 다음과 같은 하나의 문으로 바꿀 수 있습니다.

<xsd:import schemalocation="imported.xsd" namespace="http://EnterpriseLibrary/com/acs/enterprise/common/program/administration/process/bo"/>

imported.xsd 파일은 다음 구문을 사용하여 다른 XSD 파일을 포함합니다.

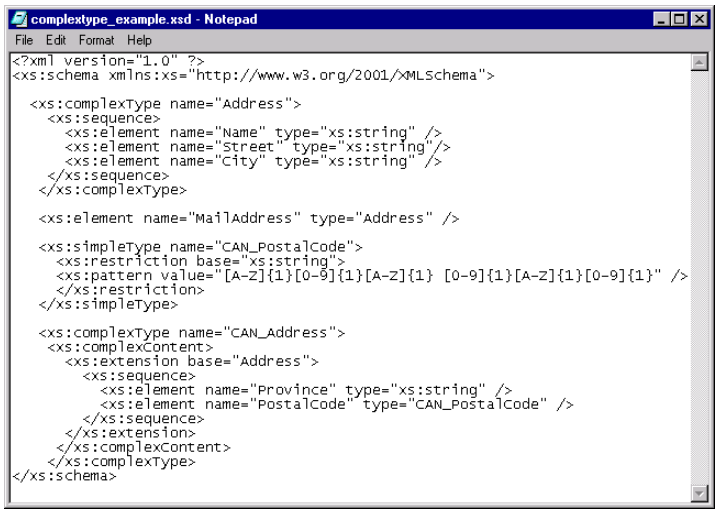
```
   <xsd:schema targetNamespace="http://EnterpriseLibrary/com/acs/enterprise/common/program/
administration/process/bo" elementFormDefault="qualified" >
      <xsd:include schemaLocation=" LocationTextBO.xsd" />
      <xsd:include schemaLocation=" LineOfBusinessBO.xsd" />
      <xsd:include schemaLocation=" ClaimExceptionBO.xsd" />
   </xsd:schema>
```

자세한 내용은 기술 자료 문서 158334를 참조하십시오.

XML 스키마에서 각 단순 유형 정의는 해당 스키마의 다른 단순 유형 정의를 제한합니다. 부울, 문자열 또는 정수와 같은 원자성 데이터 유형은 anySimpleType 데이터 유형을 제한합니다. XML 스키마에서 단순 데이터 유형을 정의하는 경우 기존 데이터 유형에서 새 데이터 유형을 파생합니다. 예를 들어 1에서 20 사이의 숫자만 보유하는 제한된 정수 유형을 파생할 수 있습니다. 기준 유형은 정수입니다.

다른 데이터 유형에서 복합 데이터 유형을 파생하는 경우 기준 유형의 요소를 포함하는 새 데이터 유형을 작성합니다. 새 요소를 파생된 유형에 추가하거나 상속된 요소에 대한 제한을 작성할 수 있습니다. 디자이너는 상속된 구성 요소를 나타내는 열을 복제하지 않고 파생된 유형에 대한 보기를 작성합니다. 이를 통해 메타데이터가 감소하고 리포지토리에서 XML 정의의 크기가 줄어듭니다.

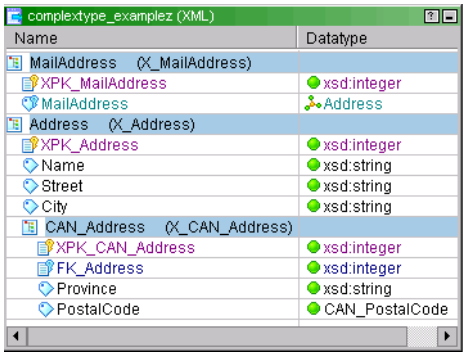
다음 이미지는 단순 파생된 유형 및 복합 파생된 유형을 가진 스키마를 보여 줍니다.



MailAddress 요소는 복합 유형인 Address 유형입니다. 파생된 유형인 CAN_Address는 Address 유형에서 Name, City 및 Street를 상속받고 Province 및 PostalCode를 추가하여 Address를 확장합니다. PostalCode는 CAN_PostalCode라는 단순 유형입니다.

XML 스키마를 가져오는 경우 복합 유형의 모든 단순 유형 또는 특성은 XML 정의에서 열이 될 수 있습니다. 복합 유형은 보기가 됩니다.

다음 그림은 기본 옵션을 사용하는 스키마를 가져오는 경우 스키마의 XML 정의를 보여 줍니다.



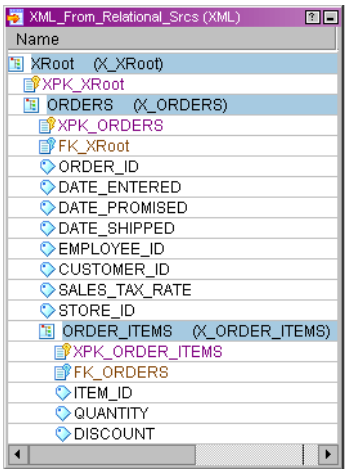
CAN_Address 보기는 해당 유형에 고유한 요소를 포함합니다. 루트 요소는 MailAddress입니다. Address 유형은 Name, Street 및 City를 포함합니다. CAN_Address에는 Address에 대한 외래 키가 있습니다. CAN_Address는 Province 및 PostalCode를 포함합니다.

보기에는 MailAddress에서 상속받는 Name, Street 및 City가 포함되어 있지 않습니다.

관계형 정의에서 메타데이터 작성

여러 관계형 정의를 선택하고 이들 간의 관계를 작성하여 XML 정의를 작성할 수 있습니다. 디자이너는 가져오는 각 관계형 정의에 대한 XML 보기를 작성합니다. 디자이너는 관계형 정의에서 모든 열을 변환하고 기본 키-외래 키 관계를 생성합니다. 루트 보기를 작성하도록 선택할 수 있습니다.

다음 그림은 관계형 정의, Orders 및 Order_Items의 샘플 XML 대상 정의를 보여 줍니다.

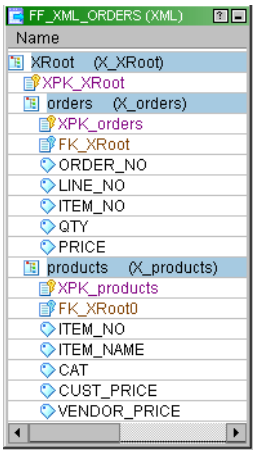


루트는 XRoot입니다. XRoot가 Orders 및 Order 항목을 묶습니다. Order_Items에는 Orders를 가리키는 외래 키가 있습니다.

플랫 파일에서 메타데이터 작성

리포지토리에서 플랫 파일 정의를 가져와 XML 정의를 작성할 수 있습니다. 둘 이상의 플랫 파일 정의를 가져오는 경우 디자이너가 각 플랫 파일에 대한 보기가 포함된 XML 정의를 작성합니다. XML 정의에서 보기는 서로 관계가 없습니다. 루트 보기를 작성하도록 선택하는 경우 디자이너가 루트에 대한 외래 키가 있는 보기를 작성합니다.

다음 그림은 플랫 파일 주문 및 제품에서 샘플 XML 소스 정의를 보여 줍니다.



Products 및 Orders에 루트 보기에 대한 외래 키가 있으며 강조 표시되어 있습니다.

XML 보기 이해

XML 계층에서 요소 간 관계는 PowerCenter 정의에서 XML 보기 간 관계를 정의합니다. 소스 정의에서는 보기가 다른 보기에 연결되지 않아도 됩니다. 그러므로 소스 정의의 보기에는 기본 또는 외래 키가 필요하지 않습니다. 비정규화된 보기는 다른 보기와 독립적일 수 있습니다. 그러나 보기가 다른 보기에 연결된 경우 키 열을 지정하지 않으면 디자이너가 키를 생성합니다.

대상 정의에서 각 보기는 최소 한 개의 다른 그룹에 연결되어야 합니다. 그러므로 다른 보기와 관계를 설정하려면 각 보기에 적어도 한 개의 키가 있어야 합니다. 키를 지정하지 않은 경우 디자이너가 대상 보기에 기본 및 외래 키를 생성합니다. 디자이너가 작성하는 것을 허용하는 대신 XML 편집기에서 보기 및 관계를 작성하면 기본 및 외래 키를 정의할 수 있습니다.

디자이너가 기본 또는 외래 키 열을 작성하는 경우 열 이름에 접두사를 할당합니다. XML 정의에서 생성된 기본 키 열에 대한 접두사는 XPK_이고 생성된 외래 키 열에 대한 접두사는 XFK_입니다. 디자이너는 기본 키를 가리키는 외래 키에 대해 접두사 FK_를 사용합니다.

예를 들어 디자이너가 Sales 그룹에 대해 기본 키 열을 작성하는 경우 디자이너가 열 이름을 XPK_Sales로 명명합니다. 디자이너가 Sales 그룹을 다른 그룹에 연결하는 외래 키 열을 작성하는 경우 열 이름을 XFK_Sales로 명명합니다. 디자이너가 작성하는 열 이름을 바꿀 수 있습니다.

매핑에 XML 소스가 포함된 경우 세션을 실행할 때 통합 서비스가 소스 정의에서 생성된 기본 키 열에 대한 값을 작성합니다. 생성된 키에 대한 시작 값을 구성할 수 있습니다.

사용자 지정 XML 보기 작성

사용자 지정 보기는 XML 마법사 또는 XML 편집기에서 작성하는 그룹입니다. XML 마법사를 사용하여 사용자 지정 보기를 작성하는 경우 마법사가 스키마의 모든 구성 요소가 포함된 보기를 작성합니다. XML 편집기를 사용하는 경우 각 보기를 정의하고 구성 요소를 선택할 수 있습니다.

보기의 요소 및 보기 간 관계는 정의를 가져올 때 디자이너가 리포지토리에서 작성하는 스키마에 종속됩니다. XML 편집기는 유효한 보기에 대한 규칙을 사용하여 XML 정의의 유효성을 검사합니다.

XML 보기에 대한 규칙 및 지침

보기 키 및 관계 작업을 수행할 때 다음 규칙과 지침을 고려해야 합니다.

- PowerCenter XML 정의는 최대 400개의 보기를 가질 수 있습니다.
- 보기는 한 개의 기본 키를 가질 수 있습니다.
- 보기는 여러 다른 보기에 연결 될 수 있으며 보기는 여러 외래 키를 가질 수 있습니다.
- 열은 기본 키와 외래 키 둘 다일 수는 없습니다.
- 소스 정의의 보기에는 키가 필요하지 않습니다.
- 대상 정의의 보기에는 최소 한 개의 키가 필요합니다.
  - 대상 루트 보기에는 기본 키가 필요하지만 대상 루트에는 외래 키가 필요하지 않습니다.
  - 대상 리프 보기에는 외래 키가 필요하지만 대상 리프 보기에는 기본 키가 필요하지 않습니다.
- 삽입 요소는 키일 수 없습니다.
- 외래 키는 항상 다른 그룹의 기본 키를 참조합니다. 자체 참조 키를 사용할 수 없습니다.
- 생성된 외래 키 열은 항상 생성된 기본 키 열을 참조합니다.
- XML 계층에서 요소의 상대 카디널리티는 PowerCenter가 XML 정의에서 보기를 작성하는 방식에 영향을 미칩니다. 다음 규칙은 요소가 동일한 보기의 일부가 될 수 있는 때를 결정합니다.
  - 일대일 관계를 가진 요소는 동일한 보기의 일부가 될 수 있습니다.
  - 일대다 관계를 가진 요소는 동일한 정규화된 보기 또는 비정규화된 보기의 일부가 될 수 있습니다.
  - 다대다 관계를 가진 요소는 동일한 보기의 일부가 될 수 없습니다.

계층 관계 이해

계층 보기 관계를 포함하는 XML 정의는 계층의 각 요소가 보기의 상위 요소 아래에 나타납니다. 다중 발생 요소는 보기가 될 수 있습니다. 복합 유형은 보기가 될 수 없지만 파생된 복합 유형에 고유한 요소는 보기에서 발생하지 않습니다.

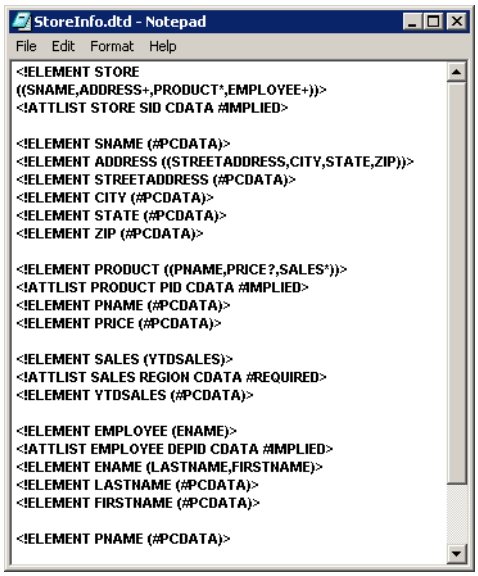
다음과 같은 유형의 계층 보기를 생성할 수 있습니다.

- **정규화된 보기.** 정규화된 보기를 가진 XML 정의는 다중 발생 데이터를 별도의 보기로 분리하여 중복성을 줄입니다. 보기는 기본 및 외래 키로 연결됩니다.
- **비정규화된 보기.** 비정규화된 보기를 포함하는 XML 정의에는 보기의 파생된 복합 유형에 대해 고유하지 않은 모든 요소의 계층이 있습니다. 소스 또는 대상 정의는 하나의 비정규화된 보기를 포함할 수 있습니다.

정규화된 보기

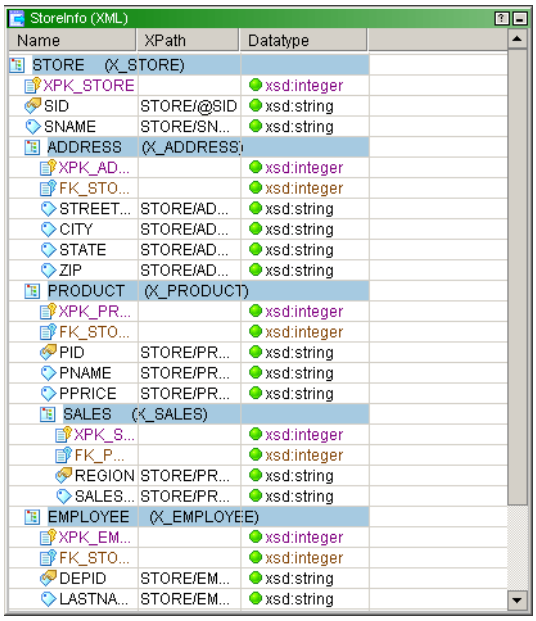
디자이너가 정규화된 보기를 생성하는 경우 XML 정의에서 보기가 되는 다중 발생 요소 및 루트 요소를 설정합니다.

다음 그림은 DTD 파일 및 정규화된 XML 정의에서 보기가 되는 요소를 보여 줍니다.



Store는 루트 요소입니다. Address, product, employee 및 sales는 다중 발생 요소입니다.

다음 그림은 위의 그림의 DTD 파일을 기반으로 소스 정의를 보여 줍니다.



정의에 정규화된 보기가 있습니다. 루트 보기는 Store입니다. Address, Product 및 Sales 보기에는 Store에 대한 외래 키가 있습니다. Sales 보기에는 Product 보기에 대한 외래 키가 있습니다.

다음 테이블은 Store 보기에 대한 데이터 미리보기의 행을 보여 줍니다.

XPK_si_STORE	si_SID	si_NAME
1	BE1752	Mud and Sawdust Furniture Store

다음 테이블은 Address 보기에 대한 데이터 미리보기의 행을 보여 줍니다.

XPK_si_ADDRESS	FK_si_ADDRESS	si_STREETADDRESS	si_CITY	si_STATE	si_ZIP
1	1	335 Westshore Road	Fausta City	CA	95784
2	1	7415 Endless Loop	Codeville	CA	97412

다음 테이블은 Product 보기에 대한 데이터 미리보기의 행을 보여 줍니다.

XPK_si_PRODUCT	FK_si_PRODUCT	si_PNAME	si_PRICE	si_PID
1	1	의자	5690.00	a1
1	1	테이블	1240.00	a2
1	1	침대	1364.99	a3

다음 테이블은 Sales 보기에 대한 데이터 미리보기의 행을 보여 줍니다.

XPK_si_SALES	FK_si_SALES	si_REGION	si_YTDSALES
1	a1	**북서부**	4565.44
2	a2	**남부**	8793.99
3	a3	**동부**	23110.00
4	a4	**남부**	5500.00
5	a5	**북서부**	10095.34
6	a6	**동부**	200.00

다음 테이블은 Employee 보기에 대한 데이터 미리보기의 행을 보여 줍니다.

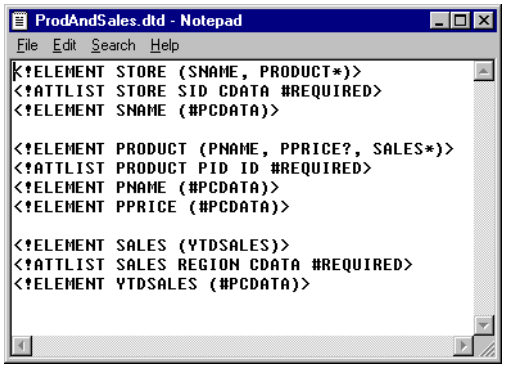
XPK_si_EMPLOYEE	FK_si_EMPLOYEE	si_FIRSTNAME	si_LASTNAME
1	1	James	Bond
2	1	Austin	Powers
3	1	Indiana	Jones
4	1	Foxie	Brown
5	1	Bonnie	Bell
6	1	Laura	Croft

비정규화된 보기

디자이너가 비정규화된 보기를 생성하는 경우 한 보기를 작성하고 계층의 모든 요소를 보기에 넣습니다. 비정규화된 보기의 모든 요소는 동일한 상위 체인에 속합니다. 비정규화된 테이블과 같은 비정규화된 보기는 중복 데이터를 생성합니다.

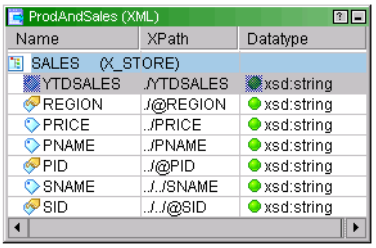
디자이너는 다중 발생 요소가 일대다 관계를 가지고 있고 모두 동일한 상위 체인의 일부인 경우 둘 이상의 다중 발생 요소가 포함된 XML 정의에 대해 비정규화된 보기를 생성할 수 있습니다.

다음 그림은 다중 발생 요소, 여기서는 Product 및 Sales가 포함된 DTD 파일을 보여 줍니다.



Product 및 Sales는 다중 발생 요소입니다. 다중 발생 요소에 일대다 관계가 있기 때문에 디자이너가 모든 요소가 포함된 단일 비정규화된 보기를 작성할 수 있습니다.

다음 그림은 소스 정의에서 ProdAndSales.dtd에 대한 비정규화된 보기를 보여 줍니다.



디자이너가 ProdAndSales 계층의 모든 요소에 대한 단일 보기를 작성합니다. DTD 파일이 데이터 유형을 정의하지 않기 때문에 디자이너가 문자열의 데이터 유형을 모든 열에 할당합니다. 비정규화된 보기에는 기본 또는 외래 키가 필요하지 않습니다.

다음 그림은 비정규화된 보기에 대한 데이터 미리보기를 보여 줍니다.

SID	SNAME	PID	PNAME	PPRICE	REGION	YTDSALES
BE1752	Mud and Sawdust Furniture Store	a1	Chair	NULL	NorthWest	3124.45
BE1752	Mud and Sawdust Furniture Store	a1	Chair	NULL	South	12434.45
BE1752	Mud and Sawdust Furniture Store	a2	Table	4126.33	South	8252.66
BE1752	Mud and Sawdust Furniture Store	a3	Bed	4356.33	NorthWest	43563.30
BE1752	Mud and Sawdust Furniture Store	a3	Bed	4356.33	South	21781.65
BE1752	Mud and Sawdust Furniture Store	a3	Bed	4356.33	East	26137.98
BE1752	Mud and Sawdust Furniture Store	a4	Etagere	5000.00	South	20000.00
BE1752	Mud and Sawdust Furniture Store	a4	Etagere	5000.00	East	5000.00
BE1752	Mud and Sawdust Furniture Store	a4	Etagere	5000.00	NorthWest	15000.00

항목 관계 이해

XML 스키마에서 항목 관계를 작성할 수 있습니다. 항목 관계가 포함된 XML 정의를 작성할 때 디자이너가 다중 발생 요소, 요소 그룹 및 복합 유형에 대한 개별 보기를 생성합니다. 디자이너에는 파생된 모든 복합 유형에 대한 보기가 포함됩니다. 디자이너가 유형과 계층 관계를 기반으로 보기 사이의 링크 및 키를 작성합니다.

XML 스키마를 작성할 때 스키마 구성 요소에서 동일한 정보를 반복하지 않고 스키마의 일부를 참조할 수 있습니다. 구성 요소는 다른 구성 요소의 요소 및 특성을 상속받고 구성 요소에서 요소를 제한하거나 확장할 수 있습니다. 예를 들어 새 복합 유형을 작성하기 위한 기준으로 복합 유형을 사용할 수 있습니다. 더 많은 요소를 새 유형에 추가하여 확장된 복합 유형을 작성할 수 있습니다. 또는 다른 복합 유형의 하위 집합인 제한된 복합 유형을 작성할 수 있습니다.

XML 편집기에서 수동으로 보기를 작성하거나 항목 관계를 다시 작성하는 경우 메타데이터를 구조화하려는 방법을 선택합니다. 상속을 사용하는 XML 스키마를 기반으로 XML 정의를 작성하는 경우 기본 유형 및 파생된 유형에 대해 별도의 보기를 생성할 수 있습니다. XML 데이터를 정규화된 관계형 테이블에 매핑하려는 경우 상속 관계를 작성할 수 있습니다.

XML 유형 I 상속 관계는 두 보기 간의 관계입니다. 각 보기 루트는 글로벌 복합 유형입니다. 한 보기는 다른 보기에서 파생됩니다.

열 및 보기 간의 상속 관계를 작성할 수 있습니다. 이는 XML 유형 II 상속 관계입니다.

디자이너는 대체 그룹에 대해 별도의 보기를 생성합니다.

항목 관계에 대한 규칙 및 지침

디자이너는 다음 지침을 기반으로 항목을 생성합니다.

- 항목은 XML, DTD 또는 XML 스키마 계층의 일부를 나타냅니다. 이 계층은 XML 파일의 루트에서 시작할 필요가 없습니다.
- 디자이너는 DTD 파일에서 정의된 항목을 사용하여 항목 관계를 작성합니다.
- 디자이너는 XML 스키마에서 정의된 유형 구조를 사용하여 항목 관계를 생성합니다.
- 상위 요소 아래에 다중 발생 요소가 발생하는 경우 디자이너가 새 항목을 작성합니다.
- 디자이너는 대체 그룹의 각 멤버에 대해 별도의 보기를 생성합니다.
- 디자이너는 기본 키 및 외래 키를 생성하여 개별 항목을 연결합니다.

유형 1 항목 관계 예

XML 유형 1 항목 관계는 두 보기 간의 관계입니다. 각 보기 루트는 글로벌 복합 유형이어야 합니다. 한 보기는 다른 보기에서 파생되어야 합니다.

다음 스키마는 PublicationType, BookType 및 MagazineType을 포함합니다. PublicationType은 기본 유형입니다. Publication은 Title, Author 및 Date를 포함합니다. BookType 및 MagazineType은 PublicationType을 확장하는 파생된 유형입니다. Book에는 ISBN 및 Publisher가 있고 Magazine에는 Volume 및 Edition이 있습니다.

```
<xsd:complexType name="PublicationType">
   <xsd:sequence>
     <xsd:element name="Title" type="xsd:string"/>
     <xsd:element name="Author" type="xsd:string" maxOccurs="unbounded"/>
     <xsd:element name="Date" type="xsd:string"/>
   </xsd:sequence>
</xsd:complexType>
<xsd:element name="Publication" type="PublicationType"/>
<xsd:complexType name="BookType">
   <xsd:complexContent>
   <xsd:extension base="PublicationType">
   <xsd:sequence>
     <xsd:element name="ISBN" type="xsd:string"/>
     <xsd:element name="Publisher" type="xsd:string
   </xsd:sequence>
   </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="MagazineType">
   <xsd:complexContent>
   <xsd:extension base="PublicationType">
   <xsd:sequence>
     <xsd:element name="Volume" type="xsd:string"/>
     <xsd:element name="Edition" type="xsd:string"/>
   </xsd:sequence>
   </xsd:extension>
   </xsd:complexContent>
</xsd:complexType>
</xsd:schema>
```

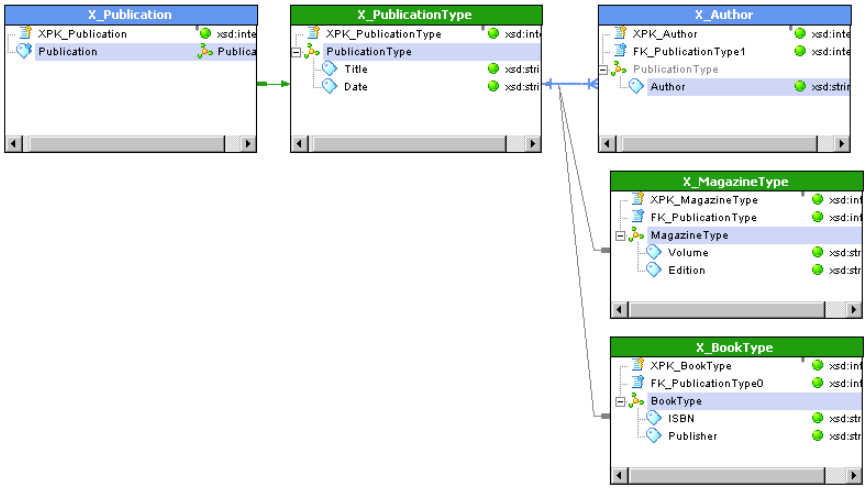
XML 정의에서 XML 보기를 항목으로 작성하는 경우 PublicationType의 Title 및 Date 메타데이터가 BookType 또는 MagazineType에서 반복되지 않습니다. 대신 이러한 보기는 PublicationType과 구별하는

메타데이터를 포함합니다. BookType의 경우 ISBN 및 Publisher, MagazineType의 경우 Volume 및 Edition입니다. 이러한 보기는 PublicationType과 연결하는 외래 키가 있습니다.

기준 유형의 요소는 파생된 유형에서 반복되지 않기 때문에 이 예는 감소된 메타데이터 확장을 사용합니다.

Author는 Publication의 다중 발생 요소입니다. Author는 XML 보기가 됩니다.

다음 그림은 디자이너가 스키마에서 생성하는 기본 보기를 보여 줍니다.



다음 그림은 출판물, 잡지 및 책이 있는 XML 파일을 보여 줍니다.

Bookstore_TYPES.xml - Notepad

File Edit Format Help

```
<?xml version="1.0"?>
<bk:BookStore xmlns:bk="http://www.books.org"
                        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                        xsi:schemaLocation=
                                    "http://www.books.org
                                    BookStore.xsd">
        <Publication>
                <Title>Staying Young Forever</Title>
                <Author>Karin Granstrom Jordan, M.D.</Author>
                <Date>1999</Date>
        </Publication>

        <Publication xsi:type="bk:BookType">
                <Title>Illusions The Adventures of a Reluctant Messiah</Title>
                <Author>Richard Bach</Author>
                <Date>1977</Date>
                <ISBN>0-440-34319-4</ISBN>
                <Publisher>Dell Publishing Co.</Publisher>
        </Publication>

        <Publication xsi:type="bk:MagazineType">
                <Title>The First and Last Freedom</Title>
                <Author>J. Krishnamurti</Author>
                <Date>1954</Date>
                <Volume>IV</Volume>
                <Edition>Spring</Edition>
        </Publication>

        <Publication xsi:type="bk:BookType">
                <Title>Working with XML</Title>
                <Author>Dell Skidmore</Author>
                <Date>1999</Date>
                <ISBN>0-444-84429-4</ISBN>
                <Publisher>Purdam Publishing Co.</Publisher>
        </Publication>

</bk:BookStore>
```

이전 그림의 XML 정의를 사용하여 샘플 XML 파일을 처리하는 경우 다음 보기로 데이터를 작성합니다.

- **PublicationType 보기.** 각 출판물에 대한 제목 및 날짜가 들어 있습니다.

  다음 그림은 PublicationType 보기를 보여 줍니다.

XPK_boo_PublicationType	Title	Date
1	Staying Yo...	1999
2	Illusions Th...	1977
3	The First a...	1954
4	Working wi...	1999

- **BookType 보기.** ISBN 및 출판사가 들어 있습니다. BookType은 PublicationType에 대한 외래 키를 포함합니다.

  다음 그림은 BookType 보기를 보여 줍니다.

XPK_boo_BookType	FK_boo_PublicationType1	ISBN	Publisher
1	2	0-440-34319-4	Dell Publishing Co.
2	4	0-444-84429-4	Purdam Publishing Co.

- **MagazineType 보기.** 볼륨 및 판이 들어 있습니다. 또한 MagazineType은 PublicationType에 대한 외래 키를 포함합니다.

  다음 그림은 MagazineType 보기를 보여 줍니다.

XPK_boo_MagazineType	FK_boo_PublicationType	Volume	Edition
1	3	IV	Spring

- **Author 보기.** 모든 출판물의 작가가 들어 있습니다. Author는 다중 발생 요소이기 때문에 디자이너가 Author에 대한 별도의 보기를 생성합니다. 각 출판물은 여러 작가를 포함할 수 있습니다.

  다음 그림은 Author 보기를 보여 줍니다.

XPK_Author	FK_boo_PublicationType0	Author
1	1	Karin Granstrom Jordan, M.D.
2	2	Richard Bach
3	3	J. Krishnamurti
4	4	Dell Skidmore

유형 II 항목 관계 예

열과 복합 유형 보기 간의 상속 관계를 작성할 수 있습니다. 열은 로컬 복합 유형의 요소가 되어야 합니다. 보기 루트는 글로벌 복합 유형이어야 합니다. 로컬 복합 유형은 글로벌 복합 유형에서 파생되어야 합니다.

예를 들어 다음 스키마는 EmployeeType이라는 복합 유형을 정의합니다. EmployeeType은 EmployeeNumber 및 EmployeeName 요소를 포함합니다.

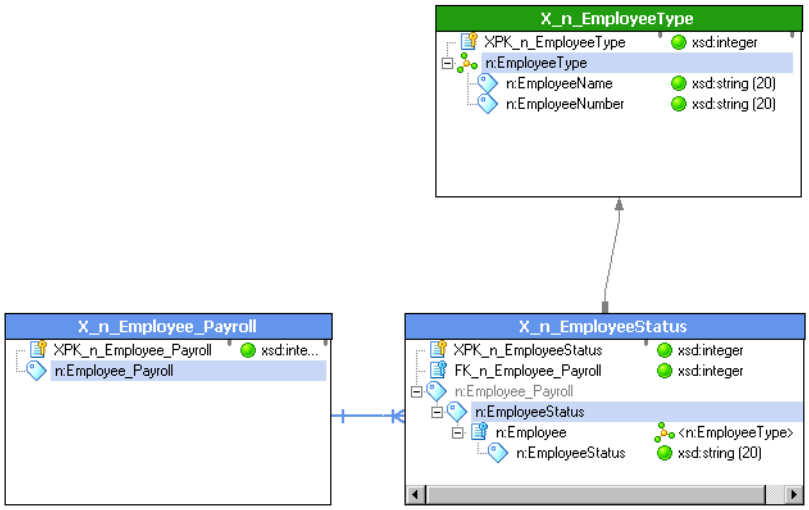
EmployeeStatusType은 EmployeeType을 확장하는 Employee라는 요소를 포함합니다. Employee는 EmployeeStatus 요소를 포함합니다.

```
<xs:element name="Employee_Payroll">
   <xs:complexType>
   <xs:sequence>
      <xs:element name="EmployeeStatus" type="EmpStatusType"          maxOccurs="unbounded"></
xs:element>
   </xs:sequence>
   </xs:complexType>
</xs:element>
<xs:complexType name="EmpStatusType">
   <xs:sequence>
      <xs:element name="Employee" minOccurs="0" maxOccurs="1">
         <xs:complexType>
         <xs:complexContent>
         <xs:extension base="EmployeeType">
```

```
            xs:sequence>
               <xs:element name="EmployeeStatus" type="xs:string">
              </xs:element>
            </xs:sequence>
         </xs:extension>
         </xs:complexContent>
         </xs:complexType>
      </xs:element>
   </xs:sequence>
</xs:complexType>
<xs:complexType name="EmployeeType">
   <xs:sequence>
      <xs:element name="EmployeeName" type="xs:string"></xs:element>
      <xs:element name="EmployeeNumber" type="xs:string"></xs:element>
   </xs:sequence>
</xs:complexType>
</xs:schema>
```

스키마를 가져올 때 디자이너는 Employee_Payroll, EmployeeType 및 EmployeeStatus에 대한 보기를 작성합니다. EmployeeStatus 보기는 Employee라는 열을 포함합니다. Employee는 EmployeeType에서 파생됩니다.

다음 그림은 Employee_Payroll 보기, EmployeeType 보기 및 EmployeeStatus XML 보기를 보여 줍니다.



Employee_Payroll 보기는 Employee_Payroll 요소 및 기본 키, PK_Employee_Payroll을 포함합니다. Employee_Payroll 보기는 보기 간의 일대다 관계를 나타내는 파란색 선으로 EmployeeStatus 보기에 연결됩니다. Employee_Payroll은 EmployeeStatus의 다중 발생을 포함합니다.

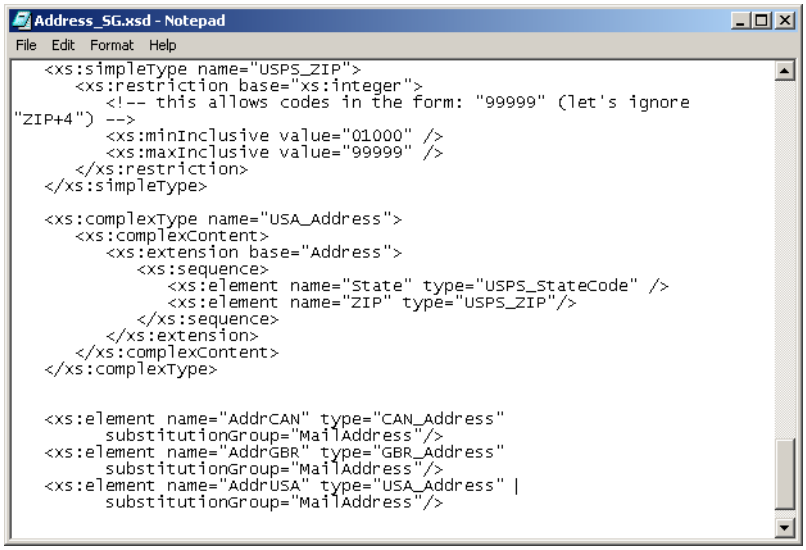
EmployeeStatus 보기는 유형 EmployeeType의 Employee 요소를 포함합니다. Employee 요소는 EmployeeStatus 요소를 포함하여 EmployeeType을 확장합니다. 또한 EmployeeStatus 보기는 Employee_Payroll에 대한 외래 키를 포함합니다. EmployeeStatus 보기는 회색 화살표로 EmployeeType 보기에 연결됩니다. 화살표는 보기 간의 유형 관계를 나타냅니다.

EmployeeType 보기는 EmployeeName 및 EmployeeNumber로 구성된 EmployeeType을 포함합니다.

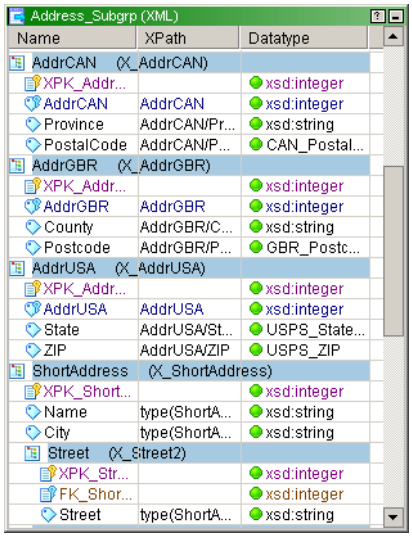
XML 정의에서 대체 그룹 사용

항목 관계가 포함된 XML 정의를 작성하는 경우 디자이너가 요소 그룹 및 복합 유형에 대한 개별 보기를 생성합니다. 대체 그룹을 사용하는 XML 스키마를 가져오는 경우 디자이너가 대체 그룹의 각 멤버를 별도의 항목으로 가져옵니다. 디자이너는 각 그룹에 대해 별도의 보기를 만듭니다.

다음 그림은 대체 그룹 MailAddress가 포함된 XML 스키마의 샘플 부분을 보여 줍니다.



다음 그림은 AddrCAN, AddrGBR, AddrUSA, ShortAddress 및 Street를 포함하여 대체 그룹의 각 멤버에 대한 보기가 포함된 XML 정의를 보여 줍니다.



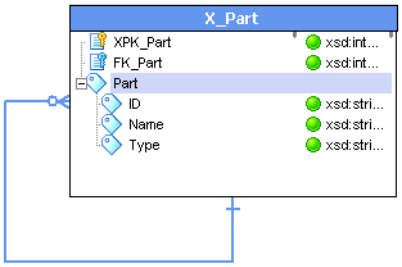
원형 참조 작업

원형 관계는 XML 정의의 단일 보기 내 또는 XML 정의의 두 보기 간의 원형 계층 관계입니다. 예를 들어 Part라는 복합 요소는 ID, 제품 이름 및 다른 제품에 대한 참조를 포함할 수 있습니다.

다음 예는 Part 요소 구성 요소를 보여 줍니다.

```
<xs:element name="Part">
<xs:complexType>
   <xs:sequence>
   <xs:element name="ID" type="xs:string"/>
   <xs:element name="Name" type="xs:string"/>
   <xs:element name="Type" type="xs:string"/>
   <xs:element ref="Part" minOccurs="0" maxOccurs="unbounded"/>
   /xs:sequence>
</xs:complexType>
</xs:element>
```

다음 그림은 Part라는 복합 요소가 포함된 XML 편집기 작업 공간에서 원형 참조를 보여 줍니다.



Part XML 정의를 사용하여 세션에서 다음 XML 파일을 읽을 수 있습니다.

```
<Part>
<ID>1</ID>
<Name>Big Part</Name>
<Type>L</Type>
   <Part>
   <ID>1.A</ID>
   <Name>Middle Part</Name>
       <Type>M</Type>
      <Part>
      <ID>1.A.B</ID>
      <Name>Small Part</Name>
      <Type>S</Type>
      </Part>
   </Part>
</Part>
```

XML 파일에서 Part 1은 Part 1.A를 포함하고, Part 1.A는 Part 1.A.B를 포함합니다.

다음 그림은 세션이 XML 소스에서 생성할 수 있는 데이터 및 키를 보여 줍니다.

XPK_Part	FK_Part	ID	Name	Type
1	NULL	1	Big Part	L
2	1	1.A	Middle Part	M
3	2	1.A.B	Small Part	S

참고: 세션이 제약 조건 기반 로딩에 대해 활성화된 경우 원형 XML 참조를 포함하는 세션을 실행할 수 없습니다. 세션이 모든 행을 거부합니다.

보기 행 이해

XML 문서에서 데이터를 추출하려면 생성할 행, 포함할 데이터 열, 행을 생성하는 시기를 지정합니다. XML 편집기에서 보기를 정의하는 경우 통합 서비스가 데이터 행을 생성하기 위해 요구하는 보기 행, 요소 또는 글로벌 복합 유형을 작성합니다.

통합 서비스는 보기 행을 사용하여 XML 보기에 대해 데이터를 읽고 기록하는 시기를 결정합니다. 단일 또는 다중 발생 요소에서 보기 행을 설정할 수 있습니다. 보기 행을 설정하면 보기에 추가하는 모든 요소는 보기 행과 일대일 대응이 됩니다.

예를 들어 Employees 보기는 요소 Employee, Name, Firstname 및 Lastname을 포함합니다. 보기 행을 Employee로 설정하는 경우 통합 서비스가 다음 알고리즘을 사용하여 데이터를 추출합니다.

```
For every (Employees/Employee)
extract ./Name/Firstname/Lastname
```

Employees XML 스키마는 다음 요소를 포함할 수 있습니다.

```
EMPLOYEES
    EMPLOYEE+
        ADDRESS+
        NAME
           FIRSTNAME
            LASTNAME
        EMAIL+
```

Employee, Address 및 Email은 다중 발생 요소입니다. 다음 요소를 포함하는 보기를 작성할 수 있습니다.

```
EMPLOYEE
    ADDRESS
    NAME
```

보기 행을 Address로 설정하는 경우 통합 서비스가 XML 데이터에서 모든 Employee/Address에 대해 Name을 추출합니다. Address와 Email 간에 다대다 관계를 작성하기 때문에 Email을 이 보기에 추가할 수 없습니다.

피벗된 다중 발생 열을 보기에 추가할 수 있습니다. 보기 행이 피벗된 열을 포함할 수 있습니다.

예를 들어 Email의 한 인스턴스를 피벗된 열로 Employee 보기에 추가할 수 있습니다. 보기는 다음 요소를 포함합니다.

```
EMPLOYEE
    ADDRESS
    NAME
    EMAIL[1]
```

보기는 보기 행, EMPLOYEE/ADDRESS/EMAIL[1]을 가질 수 있습니다. 통합 서비스가 Employee/Address/Email의 첫 인스턴스를 위해 데이터를 추출합니다.

XPath 쿼리 조건자 사용

XML 보기의 쿼리를 사용하여 XML 소스 데이터를 필터링합니다. 통합 서비스가 쿼리를 기반으로 소스 XML 파일에서 데이터를 추출합니다. 쿼리가 true인 경우 통합 서비스가 보기를 위한 데이터를 추출합니다.

XML 보기에서 쿼리를 작성하려면 XML 편집기에서 XPath 쿼리 조건자를 작성합니다. XPath는 XML 문서에서 항목을 찾는 방법을 설명하는 언어입니다. XPath는 루트 구성 요소에서 XML 계층 경로를 기반으로 하는 주소 지정 구문을 사용합니다. 보기 행이 포함된 XPath가 있는 보기 행 또는 요소 및 특성의 요소에 대해 XPath 쿼리 조건자를 작성할 수 있습니다.

XPath 쿼리 조건자는 추출할 요소 또는 특성 및 조건을 확인하는 쿼리 조건자를 포함합니다. 요소 또는 특성의 값을 확인하거나 요소 또는 특성이 소스 XML 데이터에 있는지 확인할 수 있습니다.

보기 행 사용에 대한 규칙 및 지침

다음 규칙 및 지침을 사용하여 XML 정의의 보기 행을 사용합니다.

- 보기 행은 유형 또는 요소여야 합니다. 보기 행은 특성이 될 수 없습니다.
- 모든 보기에는 요소 또는 복합 유형이 되어야 하는 보기 행이 있어야 합니다.
- 보기 루트는 보기의 최상위 수준 요소입니다. 보기 루트는 보기의 다른 모든 요소에 대해 상위입니다.
- 보기 행은 보기가 비정규화되지 않는 한 보기 루트와 동일할 수 있습니다.
- 두 보기는 XML 소스 또는 XML 파서 변환에서 동일한 보기 행을 가질 수 있습니다.
- 보기 행 요소는 보기에서 가장 낮은 다중 발생 요소여야 합니다. 보기는 다대다 관계를 포함할 수 없습니다.
- 다른 다중 발생 요소가 없는 보기에 다중 발생 요소를 추가하는 경우 기본적으로 보기 행을 새 요소로 변경합니다. 보기에 이미 다중 발생 요소가 있는 경우 다른 다중 발생 요소를 추가할 수 없습니다.
- 빈 보기를 작성할 때 보기 행을 지정할 필요가 없습니다. 그러나 열을 보기에 추가하는 즉시 디자이너가 보기 행을 작성합니다. 기본 키만 추가하는 경우에도 마찬가지입니다.
- 나중에 보기 행을 변경할 수 있지만 보기에 스키마 구성 요소가 없는 경우가 아니라면 보기 루트는 변경할 수 없습니다.
- 다음과 같이 피벗된 요소로 구성된 보기 행을 지정할 수 있습니다.

  ```
  Product/Order[2]/Customer
  ```
- 보기에 대해 유효한 보기 행은 계층 관계의 맨 위에서 보기의 보기 행까지 이르는 보기 행의 경로입니다. 보기는 XML 정의에서 다중 계층 관계를 가질 수 있기 때문에 여러 유효한 보기 행을 가질 수 있습니다.

XML 편집기에서 보기 행 및 유효한 보기 행이 데이터 출력에 영향을 미치는 방식에 영향을 주는 옵션을 지정할 수 있습니다.

열 피벗

여러 번 발생하는 하나의 요소가 다른 값이 들어 있는 동일한 요소 집합인 경우가 있습니다. 예를 들어 12번 발생하는 Sales라는 하나의 요소에 일년 중 각 달에 대한 판매 숫자가 있을 수 있습니다. 또는 두 번 발생하는 Address라는 하나의 요소가 집 주소와 사무실 주소일 수 있습니다.

XML 소스에 이러한 유형의 요소가 있는 경우 피벗을 사용하여 요소가 발생할 때 그룹 안의 별도의 열로 처리합니다. XML 보기에서 요소의 발생을 피벗하려면 정의에 나타내려는 각 발생에 대한 열을 작성합니다. 월별 판매 예에서 모두 12번의 발생을 열로 나타내려면 보기에 12개의 판매 열을 작성합니다. 한 분기의 판매를 나타내려면 3개의 열을 작성합니다. 세션을 실행하면 통합 서비스는 사용자가 정의에 포함시키지 않은 발생에 대한 XML 데이터를 무시합니다.

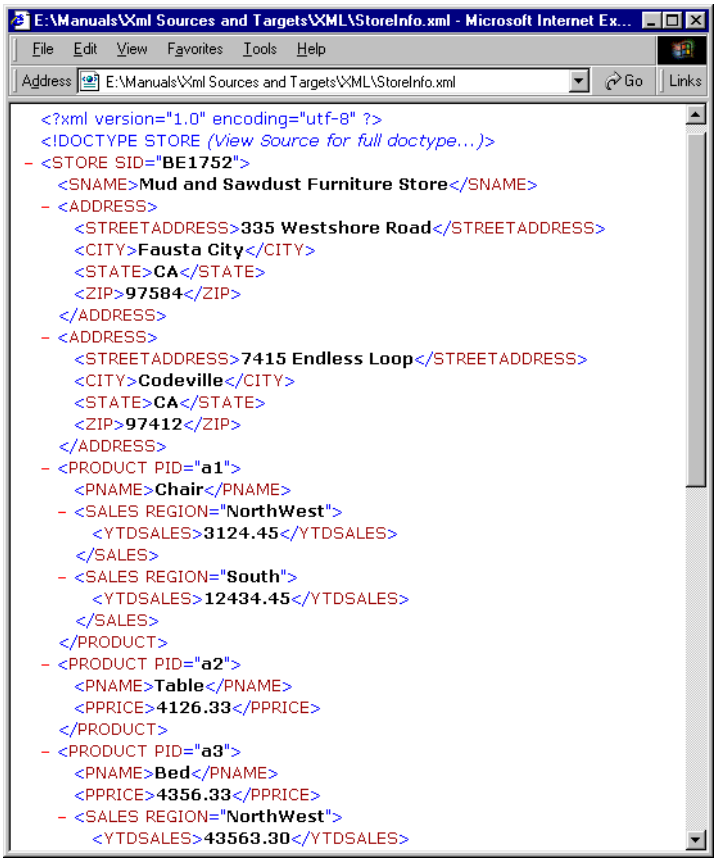
XML 소스 정의에서 보기를 추가하거나 편집할 때 열을 피벗할 수 있습니다.

단순 유형 및 복합 유형을 피벗할 수 있습니다. 기본 키 열은 피벗할 수 없습니다. 보기의 열을 피벗하는 경우 결과 그룹 구조는 유효한 정규화된 보기 또는 비정규화된 보기에 대한 규칙을 따라야 합니다. 피벗된 열이 보기를 무효화하는 경우 디자이너가 경고 및 오류를 표시합니다.

피벗은 요소를 피벗하는 보기의 요소에 영향을 미칩니다. 보기의 요소를 피벗하는 경우 다른 보기에 있는 동일한 요소는 변경되지 않습니다.

참고: XML 대상의 열은 피벗할 수 없습니다.

다음 그림은 StoreInfo XML 파일에서 Address 요소가 두 번 발생하는 것을 보여 줍니다.



처음 발생하는 Address는 접두사 HOM_를 사용하여 집 주소 열로 피벗됩니다. 두 번째 발생하는 Address는 접두사 OFC_를 사용하여 사무실 주소 열로 피벗됩니다. XPath는 동일한 요소에서 가져온 두 열 집합을 보여 줍니다.

다음 그림은 두 주소 열 집합으로 피벗된 StoreInfo XML 파일의 ADDRESS 요소를 보여 줍니다.

PowerCenter Column ...	Type	Not ...	XPath
SID	xsd:short	☐	./@SID
si_SNAME	xsd:string (30)	☐	./si:SNAME
HOM_STREETADDRESS	xsd:string (30)	☐	./si:ADDRESS[1]/STREETADDRESS
HOM_CITY	xsd:string (30)	☐	./si:ADDRESS[1]/CITY
HOM_STATE	xsd:string (30)	☐	./si:ADDRESS[1]/STATE
HOM_ZIP	xsd:string (30)	☐	./si:ADDRESS[1]/ZIP
OFC_STREETADDRESS	xsd:string (30)	☐	./si:ADDRESS[2]/STREETADDRESS
OFC_CITY	xsd:string (30)	☐	./si:ADDRESS[2]/CITY
OFC_STATE	xsd:string (30)	☐	./si:ADDRESS[2]/STATE
OFC_ZIP	xsd:string (30)	☐	./si:ADDRESS[2]/ZIP

다음 그림에서 첫 번째 및 두 번째 주소 발생(접두사 HOM_ 및 OFC_ 포함)이 그룹의 열로 나타납니다.

GPK_ADDRESS	FK_SID	HOM_STREETADDRESS	HOM_CITY	HOM_STATE	HOM_ZIP	OFC_STREETADDRESS	OFC_CITY	OFC_STATE	OFC_ZIP
1	1	335 West Bayshore Road	Fausta city	CA	97584	7415 Endless Loop	Codeville	CA	97412

여러 수준 피벗 사용

한 열에 대한 XPath의 다중 발생 요소에 대해 고정된 오프셋을 지정하여 보기에서 두 수준 이상의 요소를 피벗할 수 있습니다. 예를 들어 보기에 다음 요소가 있을 수 있습니다.

```
STORE
    PRODUCT+
        PNAME
        ORDER+
        ORDERNAME
            CUSTOMER+
                CUSTNAME
```

XPath STORE/PRODUCT[2]/ORDER[1]/ORDERNAME은 매장의 두 번째 제품에 대한 첫 번째 주문의 주문 이름을 참조합니다. XPath STORE/PRODUCT[2]/ORDER/CUSTOMER[1]는 두 번째 제품의 모든 주문에 대한 처음 고객을 참조합니다.

보기 행을 피벗하는 경우 보기 행 아래에 발생하는 XML 보기의 열은 보기 행의 XPath와 일치하는 XPath가 있어야 합니다.

예를 들어 보기에 다음 보기 행이 있을 수 있습니다.

```
Transaction/Trade[1]
```

다음 열은 XPath에서 동일하게 Trade가 발생합니다.

```
Transaction/Trade[1]/Date
Transaction/Trade[1]/Price
Transaction/Trade[1]/Person[1]/Firstname
```

보기에 다음 XPath를 가진 열을 작성할 수 없습니다.

```
Transaction/Trade[2]/Date
```

XML 소스 작업

XML 소스 작업 개요

디자이너는 리포지토리에서 XML 정의를 작성하기 위해 사용할 수 있는 XML 마법사를 제공합니다. URL 또는 로컬 노드에서 파일을 가져와 XML 정의를 작성할 수 있습니다. 또한 PowerCenter 리포지토리에서 관계형 또는 플랫 파일 정의를 가져올 수 있습니다. 다음 파일 유형에서 XML 정의를 작성할 수 있습니다.

- XML 파일
- XML 스키마 파일
- DTD 파일
- 관계형 정의
- 플랫 파일 정의

XML 정의를 작성하는 경우 XML 마법사를 사용하여 파일을 가져와 메타데이터를 XML 보기로 구성합니다. XML 보기는 XML 파일의 요소 및 특성을 포함하는 열 그룹입니다. 마법사가 사용자를 위한 보기를 생성하거나 사용자가 직접 사용자 지정 보기를 작성할 수 있습니다.

XML 마법사에서 보기 간 관계를 작성할 수 있습니다. 계층 관계 또는 항목 관계를 작성할 수 있습니다.

스키마 구조가 변경되는 경우 XML 스키마 파일에 대해 XML 스키마 정의를 동기화할 수 있습니다.

XML 소스 정의 가져오기

XML 스키마 또는 DTD 파일에서 소스 정의를 가져오는 경우 디자이너가 DTD 또는 XML 스키마 파일에서 제공한 설명을 기반으로 데이터의 정확한 정의를 제공할 수 있습니다. 연결된 DTD 또는 XML 스키마가 없는 XML 파일을 기반으로 소스 정의를 가져오는 경우 XML 마법사가 XML 파일에 표시된 데이터를 기반으로 데이터의 유형 및 발생을 결정합니다. XML 정의를 작성할 때 예기치 못한 결과가 발생할 수 있습니다. 예를 들어 디자이너가 문자열 열에 대해 부정확한 소수 자릿수 특성을 정의할 수 있습니다. 부정확한 소수 자릿수 특성이 포함된 XML 소스 정의를 내보내고 정의를 가져오는 경우 오류가 발생합니다.

XML 소스 정의를 작성한 후에는 소스 정의를 다른 소스 유형으로 변경할 수 없습니다. 반대로 다른 유형의 소스 정의를 XML 정의로 변경할 수 없습니다.

XML 마법사는 키를 사용하여 XML 보기를 연결하고 XML 계층을 재구성합니다. 보기 및 기본 키를 생성하도록 선택하거나 보기를 작성하고 키를 지정할 수 있습니다. 사용자 지정 보기를 작성하는 경우 루트를 선택하고 메타데이터 확장을 처리하는 방법을 선택할 수 있습니다.

XML 마법사는 XML 계층 및 보기 정보를 XML 스키마로 리포지토리에 저장합니다. XML 정의를 가져오는 경우 계층에서 요소의 카디널리티 및 데이터 유형을 변경하는 기능은 가져오는 파일 유형에 따라 다릅니다. 예를 들어 DTD 및 XML 파일은 데이터 유형 정보를 저장하지 않습니다. 이러한 파일을 가져와 XML 정의를 작성하는 경우 디자이너에서 데이터 유형, 전체 자릿수 및 소수 자릿수를 구성할 수 있습니다. XML 스키마를 가져오는 경우 전체 자릿수 및 소수 자릿수를 변경할 수 있습니다.

내보낸 리포지토리 개체의 XML 파일에서 XML 소스 정의를 작성할 수 없습니다. 소스 정의를 가져오는 경우 디자이너가 기본 코드 페이지를 리포지토리의 XML 정의에 적용합니다. 코드 페이지는 PowerCenter 클라이언트 코드 페이지를 기반으로 합니다. XML 소스 정의에 대한 코드 페이지는 변경할 수 없지만 XML 대상 정의에 대한 코드 페이지는 작성한 다음 변경할 수 있습니다.

XML 마법사를 사용하여 XML 소스 정의를 가져옵니다.

XML 파일을 가져오려면

1. 소스 > XML 정의 가져오기를 클릭합니다.

   XML 정의 가져오기 대화 상자가 나타납니다.

2. 고급 옵션을 클릭합니다.

   XML 보기 작성 및 명명 변경 옵션 대화 상자가 나타납니다. 옵션을 선택하여 디자이너가 XML 보기를 작성하고 명명하는 방법을 지정합니다.

다음 테이블에는 XML 보기 옵션이 설명되어 있습니다.

옵션	설명
모든 무한 길이 재정의	문자열 같이 정의되지 않은 길이를 가진 구성 요소에 대해 기본 길이를 지정할 수 있습니다. 기본 길이를 설정하지 않은 경우 이러한 구성 요소에 대한 전체 자릿수는 무한으로 설정됩니다. 큰 파일을 가진 세션을 실행할 때 무한 전체 자릿수로 인해 DTM 버퍼 크기 오류가 발생할 수 있습니다.
독립 실행형 XML의 요소/특성을 전역 선언으로 분석	이 옵션을 선택하여 독립 실행형 XML 요소 또는 특성의 전역 선언을 작성합니다. 스키마의 다른 부분에서 글로벌 요소를 참조하여 재사용할 수 있습니다. 이 옵션을 지우는 경우 독립 실행형 XML은 로컬 선언이 됩니다.
삽입 요소에 대해 XML 보기 작성	요소가 두 번 이상 발생하고 하위 요소가 두 번 이상 발생하는 경우 삽입 요소에 대해 별도의 보기를 작성할 수 있습니다. 삽입 요소는 텍스트 콘텐츠 또는 특성이 없지만 하위 요소는 있는 요소입니다.
요소를 열로 피벗	발생 제한이 있는 경우 리프 요소를 피벗할 수 있습니다. 소스 정의에서만 요소를 피벗할 수 있습니다.
고정 요소 및 특성 값 무시	스키마의 고정값을 무시하고 데이터의 다른 요소 값을 허용할 수 있습니다.
금지된 특성 무시	XML 스키마에서 금지된 것으로 특성을 선언할 수 있습니다. 금지된 특성은 복합 유형을 제한합니다. 스키마 또는 파일을 가져오는 경우 금지된 특성을 무시하도록 선택할 수 있습니다.
XML 열에 대한 이름 생성	이 옵션을 선택하여 스키마의 요소 또는 특성 이름이나 시퀀스 번호로 XML 열의 이름을 지정할 수 있습니다. 이름을 사용하는 경우 다음 옵션 중에서 선택합니다. - XMLColumn이 특성을 참조하는 경우 요소 이름을 해당 특성의 접두사로 지정합니다. PowerCenter는 XML 열의 이름에 다음 형식을 사용합니다. `NameOfElement_NameOfAttribute` - 모든 XML 열의 XML 보기 이름에 접두사를 지정합니다. PowerCenter는 XML 열의 이름에 다음 형식을 사용합니다. `NameOfView_NameOfElement` - 모든 외래 키 열의 XML 보기 이름에 접두사를 지정합니다. PowerCenter는 생성된 외래 키 열의 이름에 다음 형식을 사용합니다. `FK_NameOfView_NameOfParentView_NameOfPKColumn` 열 이름의 최대 길이는 80자입니다. PowerCenter에서는 80자를 초과하는 열 이름이 잘립니다. 열 이름이 고유하지 않은 경우 PowerCenter는 숫자 접미사를 추가하여 이름을 고유하게 유지합니다.

3. 확인을 클릭하여 변경 내용을 적용합니다.
4. 가져올 파일 유형을 선택합니다. 다음 옵션을 선택할 수 있습니다.
   - **로컬 XML 파일 또는 URL에서 정의를 가져옵니다.** XML, DTD 또는 XML 스키마 파일에서 소스 정의를 작성합니다. 연결된 DTD 또는 스키마가 포함된 XML 파일을 가져오는 경우 XML 마법사는 DTD 또는 스키마를 사용하여 XML 문서를 생성합니다.
   - **비-XML 소스 또는 대상에서 정의를 가져옵니다.** 이 옵션을 사용하여 플랫 파일 또는 관계형 정의에서 소스 정의를 작성합니다. 새 소스 정의에는 각 입력 정의를 위한 하나의 그룹과 루트 요소 그룹이 포함되어 있습니다.
5. 다음을 클릭하여 XML 마법사를 완료합니다.

다중 행 특성 값

XML 마법사는 새 줄 문자가 포함되고 두 줄 이상에 걸쳐 있는 특성 값을 허용하지 않습니다. 새 줄 문자를 가진 특성이 포함된 XML 파일에서 소스 또는 대상 정의를 가져오는 경우 XML 마법사는 오류를 표시하고 파일을 가져오지 않습니다.

XML 보기 작업

디자이너는 소스 정의에서 그룹으로 보기를 표시합니다.

XML 마법사가 정의의 보기를 작성하기 위한 옵션을 제공하거나 XML 편집기에서 수동으로 보기를 작성할 수 있습니다.

다음 옵션에서 선택하여 XML 보기를 작성할 수 있습니다.

- **항목 관계 생성**. 항목 관계를 작성하는 경우 XML 마법사는 다중 발생 또는 참조된 요소 및 복합 유형에 대한 보기를 생성합니다.
- **계층 관계 생성.** 계층 관계를 작성하는 경우 구성 요소에 대한 각 참조가 해당 상위 요소 아래에 확장됩니다. 계층 관계에서 정규화되거나 비정규화된 XML 보기를 생성할 수 있습니다.
  - **정규화된 XML 보기.** 정규화된 XML 보기를 생성하는 경우 요소 및 특성이 한 번 나타납니다. 다중 발생 요소 또는 일대다 관계의 요소는 키와 관련된 여러 보기에 나타납니다.
  - **비정규화된 XML 보기.** 비정규화된 XML 보기를 생성하는 경우 모든 요소 및 특성이 한 개의 보기에 나타납니다. 디자이너는 XML 정의에서 요소와 특성 간의 다대다 관계를 모델링하지 않습니다.
- **사용자 지정 XML 보기 작성**. 사용자 지정 XML 보기를 작성할 때 글로벌 요소를 루트로 지정할 수 있습니다. 요소, 복합 유형 및 상속된 복합 유형에 대해 메타데이터 확장을 감소하도록 선택할 수 있습니다.
- **XML 정의 동기화**. 기본 스미카가 변경될 때 하나 이상의 XML 정의를 업데이트할 수 있습니다.
- **XML 보기 작성 건너뛰기.** XML 보기 작성을 건너뛰도록 선택하는 경우 나중에 XML 편집기에서 정의할 수 있습니다. XML 편집기에서 보기를 정의하는 경우 대상과 일치시킬 보기를 정의하고 매핑을 간소화할 수 있습니다.
- **XML 파일의 요소 및 특성에 대한 XML 보기 작성.** 연결된 스키마가 포함된 XML 파일을 가져오는 경우 스키마의 모든 구성 요소 대신 XML 파일의 요소 및 특성에 대해서만 XML 보기를 작성할 수 있습니다.

항목 또는 계층 관계를 생성하도록 선택하는 경우 디자이너가 기본 루트를 선택하고 XML 보기를 작성합니다. XML 정의에서 400개를 초과하는 보기가 필요한 경우 정의가 너무 크다는 메시지가 표시됩니다. XML 편집기에서 수동으로 보기를 작성할 수 있습니다. XML 정의를 가져오고 사용자 지정 보기를 작성하도록 선택하거나 XML 보기 생성을 건너뛰도록 선택합니다.

글로벌 요소가 없는 XML 스키마에서 정의를 가져오는 경우 디자이너가 XML 정의에서 루트 보기를 작성할 수 없습니다. 디자이너가 글로벌 요소가 없다는 메시지를 표시합니다.

XML 보기를 작성한 다음 보기에 대해 설정한 구성 옵션을 변경할 수 없습니다. 예를 들어 정규화된 XML 보기를 작성하는 경우 비정규화된 보기로 변경할 수 없습니다. 새 XML 소스 정의를 가져오고 비정규화된 옵션을 선택해야 합니다.

PowerCenter에서 XML 크기 조정에 대한 자세한 내용은 "PowerCenter에서 XML 사용 개요" 페이지 28을 참조하십시오. PowerCenter에서 XML 처리에 적용되는 제한에 대한 자세한 내용은 "제한" 페이지 29을 참조하십시오.

다른 요소 유형을 가진 변환을 작성하고 더 큰 XML 입력 파일을 변환하려면 데이터 프로세서 변환을 사용합니다. 데이터 프로세서 변환을 작성하는 방법에 대한 자세한 내용은 *Informatica Data Transformation 사용자 가이드* 및 *Informatica Data Transformation 시작하기 가이드*를 참조하십시오.

XML 스키마의 일부 가져오기

XML 스키마를 가져오는 경우 디자이너가 전체 스키마를 나타내는 XML 정의를 작성합니다. 연결된 스키마가 포함된 XML 파일을 가져오는 경우 스키마의 모든 구성 요소 대신 XML 파일의 요소 및 특성에 대해서만 XML 보기를 작성할 수 있습니다. 디자이너가 XML 파일의 구성 요소로 제한된 정의를 작성합니다.

스키마의 일부를 가져오려면 스키마를 참조하는 XML 파일을 가져옵니다. 옵션을 선택하여 XML 파일의 요소 및 특성에 대해서만 XML 보기를 작성합니다.

XML 스키마의 일부를 가져오기 위한 규칙 및 지침

다음 규칙 및 지침을 고려하여 스키마의 일부를 가져옵니다.

- 디자이너는 메타데이터를 XML 파일에 있는 내용으로 제한합니다. XML 파일을 변경하고 XML 스키마를 다시 가져오는 경우 XML 정의가 변경됩니다.
- 요소 또는 특성이 XML 파일 계층에서 한 번 발생하지만 두 부분 이상의 스키마 계층에서 발생하는 경우 디자이너가 XML 정의에 있는 요소 또는 특성의 모든 발생을 포함합니다.

  예를 들어 스키마에 두 개의 주소 요소, 즉 하나의 상점/주소 및 하나의 직원/주소가 있을 수 있습니다. 스키마가 포함된 XML 파일을 가져오고 XML 파일에 상점/주소만 있는 경우 디자이너가 상점/주소 및 직원/주소 요소가 포함된 모든 주소 요소를 작성합니다.
- XML 파일에 파생된 복합 유형이 포함된 경우 디자이너가 XML 정의의 모든 기본 유형을 별도의 보기로 포함합니다. 디자이너는 기본 유형을 동일한 보기에 포함하도록 파생된 유형을 확장하지 않습니다.
- XML 파일에 여러 수준의 원형 참조가 있는 경우 디자이너가 XML 정의에서 원형 참조를 확장하지 않습니다.

항목 관계 생성

XML 계층을 항목 관계 모델로 생성할 수 있습니다. XML 보기 항목 관계를 생성하는 경우 디자이너가 다음 작업을 완료합니다.

- 다중 발생, 참조된 요소 및 복합 유형에 대한 보기를 생성합니다.
- 보기 간 관계를 작성합니다.

디자이너가 항목 관계를 생성하는 경우 디자이너가 스키마의 관계에 따라 복합 유형, 글로벌 요소 및 다중 발생 요소에 대해 여러 항목을 생성합니다.

기본 그룹 외의 그룹을 작성하거나 여러 복합 유형의 요소를 결합하려는 경우 사용자 지정 XML 보기를 작성할 수 있습니다.

XML 편집기에서 XML 소스 정의를 볼 때 XML 계층의 각 요소 간 관계를 볼 수 있습니다. 보기 간 각 관계에 대해, XML 편집기는 보기 간 관계 유형을 기반으로 링크를 생성합니다.

항목 관계를 생성하려면

1. 소스 분석기에서 소스 > XML 정의 가져오기를 클릭합니다.

   XML 마법사가 열립니다.

2. 가져오려는 소스로 이동하고 열기를 클릭합니다.
3. 파일의 이름을 입력하고 다음을 클릭합니다.
4. 항목 관계를 선택하고 마침을 클릭합니다.

   XML 마법사는 항목 관계를 사용하는 XML 정의를 생성합니다.

계층 관계 생성

계층 관계를 작성하는 경우 구성 요소에 대한 각 참조가 해당 상위 요소 아래에 확장됩니다. XML 마법사가 기본 루트를 선택하고 기본 설정을 사용하여 XML 그룹을 작성합니다.

계층 관계를 생성하려면

1. 소스 분석기에서 소스 > XML 정의 가져오기를 클릭합니다.

   XML 마법사가 열립니다.
2. 가져오려는 소스로 이동하고 열기를 클릭합니다.
3. 파일의 이름을 입력하고 다음을 클릭합니다.
4. 계층 관계를 선택합니다.
5. 정규화된 XML 보기 또는 비정규화된 XML 보기를 선택하고 마침을 클릭합니다.

   XML 마법사는 계층 관계를 기반으로 XML 보기를 생성합니다.

사용자 지정 XML 보기 작성

XML 마법사를 사용하여 사용자 지정 XML 보기를 작성할 수 있습니다. 사용자 지정 보기를 작성하는 경우 루트를 선택하고 메타데이터를 생성하는 방법을 지정할 수 있습니다. 루트 정보가 처리하려는 데이터에 적용되는지 여부에 따라 글로벌 요소를 포함 또는 제외할지 선택할 수 있습니다. 예를 들어 스키마에 상점 및 고객에 대한 정보가 포함된 경우 고객만 처리하는 XML 정의를 작성할 수 있습니다.

보기와 관련된 메타데이터를 생성하려는 방법을 지정할 수 있습니다. 항목 관계를 생성하여 요소, 복합 유형 및 상속된 복합 유형에 대해 메타데이터 확장을 줄일 수 있습니다. 메타데이터 참조를 줄이지 않는 경우 디자이너가 계층 관계를 생성하고 해당 상위 요소 아래에 모든 하위 요소를 확장합니다.

XML 마법사를 사용하여 사용자 지정 보기를 작성하려면

1. 소스 분석기에서 소스 > XML 정의 가져오기를 클릭합니다.

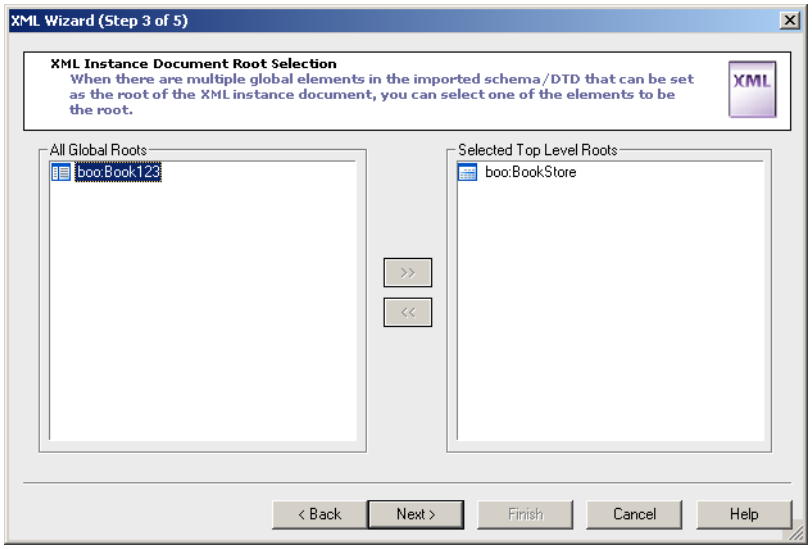
   XML 마법사가 열립니다.
2. 가져오려는 소스로 이동하고 열기를 클릭합니다.
3. 파일의 이름을 입력하고 다음을 클릭합니다.
4. 사용자 지정 XML 보기 작성을 선택하고 다음을 클릭합니다.

   참고: XML 편집기에서 모든 XML 보기를 수동으로 작성하려면 XML 보기 작성 건너뛰기를 선택합니다. XML 마법사가 리포지토리에 스키마는 작성하지만 XML 보기는 작성하지 않습니다.
5. 글로벌 루트 요소 목록에서 루트 요소를 선택하고 다음을 클릭합니다.
6. 요소, 복합 유형 또는 상속된 복합 유형에 대해 메타데이터를 감소하도록 선택하고 마침을 클릭합니다.

루트 요소 선택

사용자 지정 보기를 작성하는 경우 가져온 스키마의 글로벌 요소에서 선택하여 XML 인스턴스 문서에 대한 루트를 설정할 수 있습니다. 글로벌 요소는 상위 루트 요소의 바로 아래에 속하는 XML 스키마 계층의 요소입니다.

다음 그림은 루트 선택 페이지를 보여 줍니다.



이 예에서는 Bookstore 요소가 루트로 선택되었고 Book123 요소가 루트 요소로 선택 해제된 상태입니다.

메타데이터 확장 감소

디자이너가 상속을 사용하는 XML 스키마에 따라 XML 정의를 작성할 때 디자이너는 메타데이터를 참조하는 보기 내의 각 참조된 요소 또는 그룹에 대해 메타데이터를 확장할 수 있습니다. 또는 디자이너가 참조된 개체에 대해 별도의 보기를 작성하고 개체와 다른 보기 간의 관계를 작성할 수 있습니다.

XML 스키마 내의 참조를 사용하는 경우 디자이너가 참조와 관련된 메타데이터를 포함하는 횟수를 줄일 수 있습니다. XML 마법사는 메타데이터 참조를 줄이기 위해 다음 옵션을 제공합니다.

- **요소 확장 감소.** 디자이너는 다중 발생 요소 또는 둘 이상의 다른 요소에서 참조되는 요소에 대해 보기를 작성합니다. 각 보기는 정의의 다른 보기와 여러 계층 관계를 가질 수 있습니다.
- **복합 유형 확장 감소.** 디자이너는 각 참조된 복합 유형 또는 다중 발생 요소에 대해 XML 보기를 작성합니다. XML 보기는 다른 보기와 여러 유형의 관계를 가질 수 있습니다. 스키마가 상속된 복합 유형을 사용하는 경우 상속된 복합 유형의 확장도 감소할 수 있습니다.
- **복합 유형 상속 확장 감소.** 상속된 유형의 경우 XML 마법사가 유형 관계를 작성합니다.

메타데이터 확장을 감소하는 경우 디자이너가 생성하는 XML 보기 간의 항목 관계를 작성합니다.

XML 정의 동기화

XML 정의 작업을 수행하는 경우 XML 정의를 작성하는 데 사용한 파일 또는 소스가 변경될 수 있습니다. 예를 들어 새 요소 또는 복합 유형을 XSD 파일에 추가할 수 있습니다. XML 정의를 작성하는 데 사용한 다음 리포지토리 정의 또는 파일과 XML 정의를 동기화할 수 있습니다.

- 관계형 소스 정의
- 관계형 대상 정의
- 플랫 파일
- URL
- XML 파일
- DTD
- 스키마 파일

XML 정의를 동기화하는 경우 디자이너가 스키마 탐색기에서 XML 스키마를 업데이트합니다. 그러나 XML 정의의 보기는 업데이트하지 않습니다. XML 정의를 동기화한 다음 XML 편집기에서 보기 열을 수동으로 업데이트할 수 있습니다.

팁: 스키마 파일을 사용하여 XML 정의를 동기화합니다.

XML 소스 정의를 동기화하려면

1. 소스 분석기에서 소스 > XML 정의 가져오기를 클릭합니다.

   XML 마법사가 열립니다.
2. XML 정의를 작성하는 데 사용한 리포지토리 정의 또는 파일로 이동하고 열기를 클릭합니다.
3. 마법사의 1단계에서 다음을 클릭합니다. 마법사는 이름에 대한 변경 내용을 무시합니다.
4. XML 마법사의 2단계에서 XML 정의를 동기화하도록 선택하고 다음을 클릭합니다.

   XML 마법사가 5단계로 건너뜁니다.
5. XML 마법사의 5단계에서 동기화하려는 XML 정의를 선택합니다.

   XML 마법사가 선택한 정의와 소스를 동기화합니다.

이 방법을 사용하여 XML 대상 정의를 동기화합니다. XML 소스 정의를 수정하는 경우 대상 정의를 동기화해야 할 수도 있습니다.

참고: 정의를 작성하는 데 사용한 소스와 XML 정의를 동기화하는지 확인합니다. 정의를 작성하는 데 사용하지 않은 소스와 XML 정의를 동기화하는 경우 디자이너가 정의를 동기화할 수 없고 메타데이터를 손실합니다. 편집 > 저장된 상태로 되돌리기를 클릭하여 XML 정의를 복원합니다.

XML 소스 정의 속성 편집

XML 소스 정의를 가져온 다음 정의 이름과 같은 소스 정의 속성을 편집할 수 있습니다. 파일 목록을 읽도록 세션을 구성하는 경우 소스 파일 이름을 각 대상 행에 쓰도록 매핑을 구성할 수 있습니다. 메타데이터 확장을 추가할 수도 있습니다.

XML 소스 정의 속성을 편집하려면

1. 소스 분석기 작업 공간에서 정의의 상단을 마우스 오른쪽 단추로 클릭합니다. 편집을 선택합니다.

2. 테이블 탭에서 다음 설정을 편집합니다.

테이블 설정	설명
테이블 선택	편집하는 소스 정의를 표시합니다.
비즈니스 이름	소스 정의에 대한 설명 이름입니다. 이름 바꾸기 단추를 클릭하여 비즈니스 이름을 편집할 수 있습니다.
소유자 이름	XML 파일에 적용되지 않습니다.
설명	소스에 대한 설명입니다. 문자 제한은 2,000바이트/K이며 여기서 K는 리포지토리 코드 페이지의 각 문자에 대한 최대 바이트 수입니다. 비즈니스 문서에 대한 링크를 입력합니다.
데이터베이스 유형	소스 또는 데이터베이스 유형입니다.
코드 페이지	읽기 전용입니다. XML 소스 파일에 적용되지 않습니다. 통합 서비스가 모든 XML 소스 파일을 유니코드로 변환합니다.

3. 열 탭을 클릭합니다.

   열 탭에서 정의에 있는 열에 대한 정보를 볼 수 있습니다. 열 이름 또는 값을 변경하려면 XML 편집기를 사용합니다.

   다음과 같은 정보를 볼 수 있습니다.

열 설정	설명
테이블 선택	편집 중인 소스 정의입니다.
열 이름	열의 이름입니다.
데이터 유형	PowerCenter 데이터 유형입니다.
전체 자릿수	열의 길이입니다.
배율	숫자 데이터의 소수 자릿수입니다.
Null이 아님	열이 Null을 허용할 수 있는지 여부를 나타냅니다.
키 유형	기본 키, 외래 키 또는 키 없음입니다.
XPath	XML 계층의 현재 열에서 참조된 요소의 경로입니다. XPath는 생성된 기본 또는 외래 키를 표시하지 않습니다.
비즈니스 이름	열에 대한 사용자 정의 설명 이름입니다. 비즈니스 이름이 창에 보이지 않는 경우 오른쪽으로 스크롤하여 열을 보거나 수정합니다.

4. 파일 목록을 읽도록 세션을 구성하고 소스 파일 이름을 각 대상 행에 쓰려는 경우 속성 탭을 클릭하고 현재 처리된 플랫 파일 이름 포트 추가를 선택합니다.

   디자이너가 CurrentlyProcessedFileName 포트를 열 탭에 추가합니다. 처음 그룹의 마지막 열입니다. 통합 서비스가 이 포트를 사용하여 소스 파일 이름을 반환합니다. CurrentlyProcessedFileName 포트는 256자의 기본 전체 자릿수를 가진 문자열 포트입니다.

CurrentlyProcessedFileName 포트를 제거하려면 속성 탭을 클릭하고 현재 처리된 플랫 파일 이름 포트 추가를 지웁니다.

5. 메타데이터 확장 탭을 클릭하여 사용자 정의 메타데이터 확장을 작성, 편집 및 삭제합니다.
6. 확인을 클릭합니다.
7. 리포지토리 > 저장을 클릭하여 변경 내용을 리포지토리에 저장합니다.

리포지토리 정의에서 XML 정의 작성

XML 소스 또는 대상 정의를 리포지토리의 관계형 또는 플랫 파일 정의에서 가져올 수 있습니다. 리포지토리 정의에서 XML 정의를 가져오면 XML 마법사가 선택한 개체 간의 관계에서 XML 계층을 작성합니다. XML 마법사가 계층에 대해 루트 요소를 작성합니다. 작성하는 그룹에서 루트를 선택할 수 있습니다. 또는 별도의 루트를 작성하고 그룹을 여기에 연결할 수 있습니다.

XML 대상 정의를 작성하면 XML 마법사가 키를 생성하여 각 그룹을 루트에 연결합니다.

리포지토리 소스 또는 대상에서 XML 정의를 작성하려면

1. 소스 분석기에서 소스 > XML 정의 가져오기를 클릭합니다.

   -또는-

   웨어하우스 디자이너에서 대상 > XML 정의 가져오기를 클릭합니다.
2. XML 가져오기 대화 상자에서 비-XML 소스 또는 비 XML 대상을 클릭합니다.
3. 소스 또는 대상 목록에서 정의를 선택합니다. 화살표 단추를 클릭하여 정의를 선택한 소스 목록에 추가합니다.

   둘 이상의 입력 및 둘 이상의 입력 유형을 선택할 수 있습니다. 정의가 기본 및 외래 키를 통해 연결된 경우 XML 마법사는 계층을 생성할 때 키를 사용하여 그룹을 연결합니다.
4. 루트 요소에 대해 별도의 그룹을 작성하려면 선택적 XML 루트 이름을 입력합니다.

   루트 이름의 기본값은 XRoot입니다. 루트 그룹은 다른 모든 그룹을 묶습니다. 다른 그룹 중 하나를 루트로 사용하려면 루트 이름을 제거합니다. XML 마법사가 선택하는 각 입력 소스 또는 대상 정의에 대해 그룹을 작성하고 각 그룹에 기본 키를 생성합니다. XML 마법사가 각 그룹에 외래 키를 작성합니다. 외래 키는 루트 그룹 링크 키를 가리킵니다.
5. 열기를 클릭합니다.

   XML 마법사가 나타납니다.
6. XML 마법사를 사용하여 소스 또는 대상 그룹을 생성합니다.

XML 소스 문제 해결

동일한 상위 요소를 가진 두 개의 다중 발생 요소를 어떻게 한 개의 보기에 넣을 수 있습니까? 예를 들어 EMPLOYEE의 모든 요소를 한 개의 보기에 넣어야 합니다.

```
<!ELEMENT EMPLOYEE (EID, EMAIL+, PHONE+)>
```

EMAIL 및 PHONE이 동일한 상위 요소에 속하지만 동일한 상위 체인에 속하지 않습니다. 비정규화된 동일한 보기에 넣을 수 없습니다. 모든 EMPLOYEE 요소를 한 개의 보기에 넣으려면 다중 발생 요소 중 하나를 피벗할 수 있습니다.

다음 단계를 따라 두 개의 다중 발생 요소를 동일한 보기에 추가합니다.

1. EMPLOYEE 보기를 작성합니다.
2. EID 및 EMAIL 요소를 EMPLOYEE 보기에 추가합니다.
3. 보기에 포함하려는 EMAIL의 발생 수를 피벗합니다. 각 EMAIL 발생은 보기에서 단일 발생 요소가 됩니다.
4. PHONE 요소를 추가합니다.

내 DTD에 다음 요소 정의가 있습니다.

```
<!ELEMENT EMPLOYEE (EMPNO, SALARY)+>
```

동일한 보기에서 EMPNO와 SALARY를 어떻게 일치시킬 수 있습니까?

DTD 예는 모호합니다. 정의는 다음과 동일합니다.

```
<!ELEMENT EMPLOYEE (EMPNO+, SALARY+)>
```

DTD 예에서 EMPLOYEE는 다중 발생 요소 EMPNO와 SALARY가 있습니다. 동일한 보기에 두 개의 다중 발생 요소를 가질 수 없습니다.

다음 솔루션 중 하나를 사용합니다.

- **요소 정의를 다시 기록하여 정의를 모호하지 않게 만듭니다.**

  EMPLOYEE 요소를 다음과 같이 정의할 수 있습니다.

  ```
  <!ELEMENT EMPLOYEES (EMPLOYEE+)>
  <!ELEMENT EMPLOYEE (EMPNO, SALARY)>
  ```

  이 구문을 사용하는 경우 각 EMPLOYEE에 대해 하나의 EMPNO와 하나의 SALARY를 정의합니다. EMPLOYEE 보기에 두 요소가 포함되어 있습니다. EMPLOYEE를 EMPLOYEES의 다중 발생 요소로 포함시킵니다.

- **요소를 별도의 보기로 유지하고 매핑에서 소스 정의를 두 번 사용합니다.**

  EMPNO와 SALARY가 다른 보기에 있는 경우 매핑에서 여전히 데이터를 결합할 수 있습니다. 동일한 소스 정의의 두 인스턴스를 사용하고 조이너 변환을 사용합니다.

다음 구조를 가진 XML 파일을 가져왔습니다.

```
<Bookstore>
                    <Book>Book Name</Book>
              <Book>Book Name</Book>
                                  <ISBN>051022906630</ISBN>
</Bookstore>
```

이 XML 파일을 가져올 때 디자이너가 ISBN 요소를 삭제합니다. 이 문제가 발생하는 이유는 무엇입니까? 어떻게 해야 디자이너가 ISBN 요소를 포함할 수 있습니까?

- **스키마를 사용하여 XML 정의 가져오기.** XML 파일을 사용하여 XML 정의를 가져오는 경우 요소에 하위 요소가 없기 때문에 디자이너가 첫 번째 요소를 단순 콘텐츠로 읽습니다. 디자이너가 두 번째 Book 인스턴스의 ISBN 하위 요소를 무시합니다. 스키마를 사용하여 정의를 가져오면 디자이너가 스키마 정의를 사용하여 XML 데이터를 읽는 방법을 결정합니다.
- **XML 파일이 연결된 스키마를 정확하게 표현하는지 확인.** XML 파일을 사용하여 소스 정의를 가져오는 경우 XML 파일이 해당 XML 스키마의 구조를 정확히 표현하는지 확인합니다.

PowerCenter에서 XML 크기 조정에 대한 자세한 내용은 "PowerCenter에서 XML 사용 개요" 페이지 28을 참조하십시오. PowerCenter에서 XML 처리에 적용되는 제한에 대한 자세한 내용은 "제한" 페이지 29을 참조하십시오.

다른 요소 유형을 가진 변환을 작성하고 더 큰 XML 입력 파일을 변환하려면 데이터 프로세서 변환을 사용합니다. 데이터 프로세서 변환을 작성하는 방법에 대한 자세한 내용은 *Informatica Data Transformation 사용자 가이드* 및 *Informatica Data Transformation 시작하기 가이드*를 참조하십시오.

XML 편집기 사용

XML 편집기 사용 개요

디자이너에서 XML 정의를 가져올 때 기본 보기나 사용자 지정 보기를 사용하거나 보기 없이 XML 정의를 작성합니다. XML 정의를 작성한 후 XML 편집기를 사용하여 정의를 변경합니다.

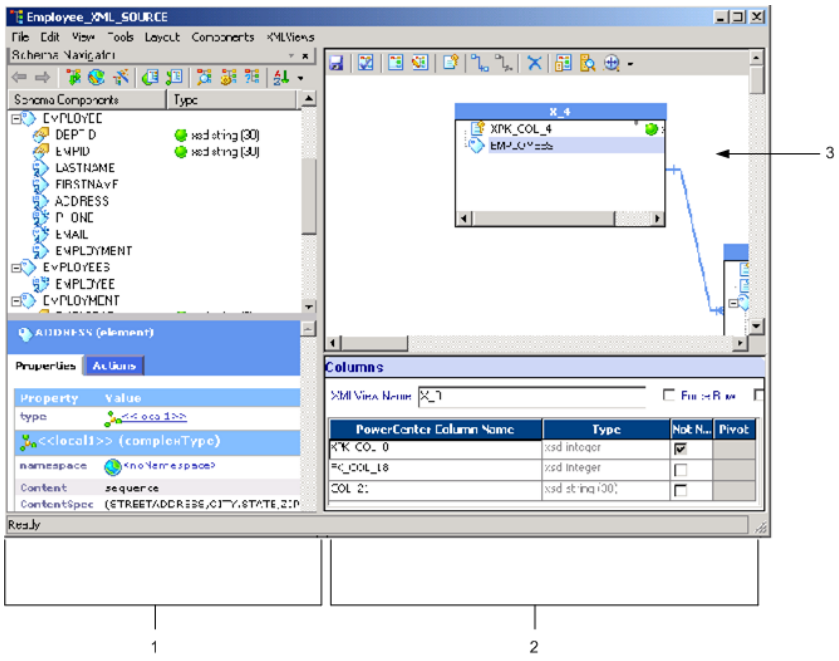
XML 편집기를 사용하여 보기를 작성하고, 구성 요소를 수정하고, 열을 추가하고, 작업 공간에서 보기 관계를 유지 관리하십시오. XML 정의를 업데이트하면 디자이너가 소스가 포함된 모든 매핑에 대한 변경 사항을 전달합니다. XML 정의에 대한 일부 변경 사항은 매핑을 무효화할 수 있습니다.

참고: XML 정의를 작성하는 데 사용한 소스를 상당히 변경하는 경우 수동으로 정의를 편집하지 않고 새 소스에 정의를 동기화할 수 있습니다.

XML 편집기에는 다음 창이 포함되어 있습니다.

- 탐색기
- XML 작업 공간
- 열 창

다음 그림에는 XML 편집기가 나와 있습니다.



1. 탐색기
2. 열 창
3. XML 작업 공간

XML 편집기는 XML 구성 요소 유형을 나타내기 위한 아이콘을 사용합니다. XML 편집기에서 아이콘에 대해 설명하는 범례를 보려면 보기 > 범례를 클릭하십시오.

XML 탐색기

탐색기는 계층 형식으로 스키마를 표시하고 선택된 구성 요소에 대한 정보를 제공합니다. 탐색기에서 유형, 계층 또는 네임스페이스 기준으로 구성 요소를 정렬할 수 있습니다. 또한 구성 요소를 확장하여 계층에서 구성 요소 아래의 구성 요소를 볼 수 있습니다.

탐색기 도구 모음은 대부분의 탐색기 기능에 대한 바로 가기를 제공합니다. 또한 이 도구 모음은 빠르게 계층에서 이전에 표시된 구성 요소를 찾기 위해 클릭할 수 있는 탐색 화살표를 제공합니다.

탐색기에 있는 탭은 다음과 같습니다.

- **속성 탭.** 구성 요소 유형, 길이 및 발생 등 선택된 구성 요소에 대한 정보를 표시합니다.
- **작업 탭.** 선택된 구성 요소에 대한 추가 정보를 보기 위한 옵션 목록을 제공합니다.

속성 탭

속성 탭은 사용자가 탐색기에서 선택하는 구성 요소에 대한 정보를 표시합니다. 구성 요소가 복합 요소인 경우 네임스페이스, 유형 및 콘텐츠 등 스키마의 요소 속성을 볼 수 있습니다. 단순 요소 또는 특성을 보면 속성 탭에 요소의 유형과 길이가 표시됩니다. 속성 탭은 주석도 표시합니다.

XML 파일에서 정의를 가져오는 경우 속성 탭에서 데이터 유형 및 카디널리티를 편집할 수 있습니다. DTD 파일에서 정의를 작성하는 경우 구성 요소 유형을 편집할 수 있습니다.

스키마가 네임스페이스를 사용하는 경우 네임스페이스, 접두사 및 네임스페이스에 대한 스키마 위치를 변경할 수 있습니다. 접두사는 네임스페이스에 속하는 요소나 특성을 식별합니다. 기본 네임스페이스의 요소와 특성에는 접두사가 없습니다. 네임스페이스를 XML 대상에 대한 기본 네임스페이스로 선택할 수 있습니다.

작업 탭

작업 탭에는 선택된 구성 요소에 대한 추가 정보를 보는 데 사용하는 옵션이 나열됩니다. 또한 작업 탭에서 구성 요소에 대해 변경한 사항을 되돌릴 수 있습니다.

선택하는 구성 요소 속성에 따라 다음 옵션이 작업 탭에 나타납니다.

- **ComplexType 참조.** 이 유형의 스키마 구성 요소를 표시합니다.
- **ComplexType 계층.** 선택된 구성 요소에서 파생된 복합 유형을 표시합니다.
- **SimpleType 참조.** 이 유형의 모든 구성 요소를 표시합니다.
- **SimpleType 값 전달.** 이 SimpleType의 모든 구성 요소에 길이 및 배율 값을 전달합니다.
- **요소 참조.** 선택된 요소를 참조하는 구성 요소를 표시합니다.
- **하위 구성 요소.** 선택된 구성 요소가 사용하는 글로벌 스키마 구성 요소를 표시합니다.
- **simpleType 되돌리기.** 사용자가 변경한 경우 유형, 길이 및 전체 자릿수 값을 원래 값으로 변경합니다.
- **XML 보기 참조.** 선택된 구성 요소를 참조하는 모든 XML 보기 및 열을 표시합니다.

XML 작업 공간

XML 작업 공간에는 XML 보기 및 해당 보기 간 관계가 표시됩니다. 작업 공간에서 XML 보기를 작성하고 보기 간 관계를 정의할 수 있습니다.

XML 작업 공간 도구 모음은 이 작업 공간에서 수행할 수 있는 대부분의 기능에 대한 바로 가기를 제공합니다.

다음과 같은 방법으로 XML 작업 공간 크기를 수정할 수 있습니다.

- **열 창 숨기기.** 보기 > 열 속성을 클릭합니다.
- **탐색기 숨기기.** 보기 > 탐색기를 클릭합니다.
- **작업 공간 줄이기.** 작업 공간 도구 모음에서 Zoom(확대/축소) 단추를 클릭합니다.

열 창

열 창에는 작업 공간의 보기에 대한 열이 표시됩니다. 추가하는 열의 이름을 지정하려면 열 창을 사용합니다. 피벗된 열을 사용하는 경우 열 창을 사용하여 다중 발생 요소의 발생을 선택하고 이름을 바꿉니다. 또한 Not Null(Null이 아님), Force Row(강제로 행 적용), Hierarchy(계층) 또는 Type Relationship Row(유형 관계 행) 및 Non-Recursive Row(비반복 행) 등의 옵션을 지정할 수 있습니다. 해당 옵션은 통합 서비스가 XML 대상에 데이터를 쓰는 방식에 영향을 미칩니다.

보기 작성 및 편집

XML 편집기를 사용하여 사용자 지정 XML 보기를 작성하거나, XML 마법사로 작성한 XML 보기를 편집하십시오. 보기를 작성하려면 보기를 정의하고 보기의 열을 지정합니다. 스키마에 다중 발생 요소가 있는 경우

보기에 포함시킬 요소 발생을 지정할 수 있습니다. 또한 XML 대상 파일 이름에 대한 특수 포트 및 XML 파서와 XML 생성기 변환에 대한 통과 포트를 작성할 수 있습니다.

XML 보기를 작성하고 편집하려면 다음 태스크를 완료하십시오.

- **XML 보기 작성.** 작업 공간에 보기를 추가합니다.
- **보기에 열 추가.** 보기에서 새 열을 작성합니다.
- **보기에서 열 삭제.** 보기에서 열을 삭제합니다.
- **복합 유형 확장.** 보기에 추가할 파생된 복합 유형을 선택합니다.
- **anyType 요소 가져오기.** anyType 요소를 가져옵니다.
- **anyAttribute 요소에 콘텐츠 적용.** anyAttribute 요소에 대한 콘텐츠를 정의합니다.
- **anySimpleType 요소 사용.** XML 정의에서 anySimpleType 요소를 사용합니다.
- **통과 포트 추가.** XML 변환을 통해 XML이 아닌 데이터를 전달할 포트를 추가합니다.
- **FileName 열 추가.** 각 XML 대상 파일에 대한 새 파일 이름을 생성하기 위한 열을 추가합니다.

XML 보기 작성

XML 작업 공간에서 보기를 작성할 수 있습니다. 보기 없이 XML 정의를 작성하는 경우 XML 편집기에 빈 작업 공간이 표시됩니다. 보기를 작성하고 열과 보기 행을 이 보기에 추가할 수 있습니다.

작업 공간에서 새 XML 보기를 작성하려면:

1. XML 편집기에서 XML 정의를 엽니다.
2. XML Views(XML 보기) > Create XML View(XML 보기 작성)를 클릭합니다.

   XML 편집기가 작업 공간에 빈 보기를 작성하고 열 창에 빈 열을 표시합니다.
3. 열 창에서 보기의 이름을 입력합니다.

   작업 공간의 XML 보기에 이름이 나타납니다.
4. 보기를 작성하려는 스키마 탐색기에서 노드를 강조 표시합니다.
5. 구성 요소 > XPath 탐색기를 클릭합니다.

   XPath 탐색기가 탐색기 창에 나타납니다.
6. XPath 탐색기의 열 모드를 보기 행 모드로 설정하여 보기 행을 추가합니다.
7. 탐색기에서 요소를 선택하고 작업 공간의 보기로 요소를 끕니다.

   XML 편집기에서 보기 행이 파란색으로 강조 표시됩니다.

   처음으로 보기에 열을 추가하면 디자이너에서 이 열이 보기 행일 수 있는지 확인합니다. 보기 행을 추가하도록 지정하지 않는 경우에도 마찬가지입니다.
8. 보기 행을 다른 열로 변경하려면 보기에서 해당 행을 마우스 오른쪽 단추로 클릭하고 Set As View Row(보기 행으로 설정)를 선택합니다.

보기에 열 추가

XML 작업 공간 내 XML 보기에 열을 추가할 수 있습니다. 열을 추가하려면 XPath 탐색기에서 열을 선택합니다.

다음 조건이 참인 경우 XML 보기에 열을 추가할 수 있습니다.

- 구성 요소 경로가 해당 보기의 보기 루트인 스키마의 요소에서 시작됩니다.
- 구성 요소가 삽입 요소가 아닙니다.
- 구성 요소가 정규화 또는 피벗 규칙을 위반하지 않습니다. 예를 들어 두 개 이상의 다중 발생 요소를 보기에 추가할 수 없습니다.
- 단순 또는 복합 유형으로 혼합 콘텐츠 요소를 추가할 수 있습니다.
- 두 개의 보기가 같은 열을 공유할 수 있으며, 보기에 동일한 열 여러 개가 포함될 수 있습니다.

보기에 열을 추가하려면:

1. 작업 공간에서 XML 보기를 선택합니다.
2. 추가할 구성 요소가 포함된 탐색기에서 상위 요소를 강조 표시합니다.
3. 구성 요소 > XPath 탐색기를 클릭합니다.

   XPath 탐색기가 탐색기 창에 나타납니다.
4. 모드 단추를 클릭하고 열 또는 보기 행을 선택하여 추가합니다.
5. 고급을 선택하여 피벗된 열을 보기에 추가합니다.

   피벗된 열은 다중 발생 요소의 발생입니다. 고급 모드에서 단일 발생 열을 추가할 수 있습니다.

   참고: XML 대상 정의에서는 피벗된 열을 작성할 수 없습니다.
6. XPath 탐색기에서 XML 작업 공간의 해당 보기로 열을 끕니다. 한 번에 여러 열을 선택할 수 있습니다.

   XML 편집기에서 사용자가 추가하는 열의 유효성을 검사합니다. 열이 보기에 대해 유효하지 않은 경우, 열을 끌어오는 동안 상태 표시줄에 메시지가 나타납니다. 새 열을 보기에 추가하면 이 열이 표시됩니다.

피벗된 열 추가

XML 정의의 피벗된 열은 보기에서 요소 발생에 대한 별도의 열을 형성하는 다중 발생 요소입니다. XML 정의의 다중 발생 요소에서 피벗된 열을 작성할 수 있습니다. 또한 보기 행에서 요소를 피벗할 수 있습니다.

피벗된 열을 보기에 추가하면 기본 발생 번호가 열 창에 나타납니다. 이 번호는 열에서 사용할 요소의 발생을 나타냅니다. 발생 번호를 변경하거나 요소의 추가 발생을 새 열로 추가할 수 있습니다. 열 이름을 바꾸지 않는 경우 XML 편집기가 각각의 피벗된 열 이름에 시퀀스 번호를 추가합니다.

참고: 피벗 값이 보기 행의 일부인 경우 피벗 값을 변경할 수 없습니다.

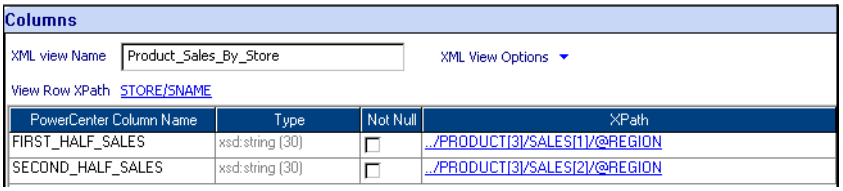
피벗된 열을 보기에 추가하려면:

1. 작업 공간에서 XML 보기를 선택합니다.
2. 탐색기에서 피벗하려는 요소를 강조 표시합니다.

   또는 피벗하려는 요소의 상위 체인에 있는 아무 요소나 강조 표시합니다.
3. 구성 요소 > XPath 탐색기를 클릭합니다.
4. 고급 모드를 선택합니다.
5. 피벗할 열을 XPath 탐색기에서 XML 작업 공간의 보기로 끕니다.

   디자이너가 열 창에서 기본 발생 번호를 추가합니다. 이 번호는 열에서 사용할 요소의 발생을 나타냅니다.

다음 그림에서는 보기에 Product의 세 번째 발생에 대한 Sales 발생 두 개가 포함되어 있습니다.



첫 번째 Sales 발생은 First_Half_Sales라는 열에 있습니다. 두 번째 Sales 발생은 Second_Half_Sales입니다. Region은 특성입니다.

6. XPath 링크를 클릭하여 요소 발생 번호를 변경합니다.

   Specify Query Predicate for XPath(XPath의 쿼리 조건자 지정) 창이 나타납니다.

7. 편집할 다중 발생 요소를 선택합니다.
8. Edit Pivot(피벗 편집) 상자에서 발생 번호를 변경하고 확인을 클릭합니다.

보기에서 열 삭제

보기에서 열을 삭제할 수 있습니다.

보기에서 열을 삭제하려면:

1. 작업 공간에서 보기의 열을 마우스 오른쪽 단추로 클릭합니다.
2. Delete This Column(이 열 삭제)을 선택합니다.

   XML 편집기에 열을 삭제할지 확인하라는 메시지가 표시됩니다.

3. 예를 클릭하여 확인합니다.

   XML 편집기가 보기에서 해당 열을 제거합니다. 하지만 XPath 탐색기 계층에 이 열이 남아 있습니다.

피벗된 열 삭제

피벗된 열의 발생을 삭제하려면 열 창에서 해당 열을 선택하고 삭제합니다.

피벗된 열을 삭제하려면:

1. 열 창에서 삭제할 열을 마우스 오른쪽 단추로 클릭합니다.
2. 삭제 > 피벗을 클릭합니다.
3. 예를 클릭하여 삭제를 확인합니다.

복합 유형 확장

스키마는 두 개 이상의 유형에 대해 기본 유형인 복합 유형을 정의할 수 있습니다. 예를 들어 Publication(게시)은 Magazine(잡지) 또는 Newspaper(신문)일 수 있습니다. XML 보기를 작성할 때 Publication(게시)을 Magazine(잡지) 또는 Newspaper(신문) 유형으로 사용하도록 선택할 수 있습니다.

XPath 탐색기에서 복합 유형을 보는 경우 파생된 유형을 볼 수 있습니다.

복합 유형을 확장하려면:

1. XPath 탐색기에서 복합 유형을 강조 표시합니다.

   Expand Complex Types(복합 유형 확장) 목록에 파생된 유형이 표시됩니다.

2. 사용할 열을 선택합니다.

   XML 정의에 구성 요소를 추가하는 경우, 선택하는 유형이 이 정의에 포함됩니다.

anyType 요소 가져오기

anyType 요소가 포함된 XML 스키마를 가져올 수 있습니다. 유형 anyType의 요소는 XML 문서에서 발생하는 모든 데이터 유형을 포함할 수 있습니다. 예를 들어 XML 스키마의 다음 섹션은 anyType인 요소 Document를 포함합니다.

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <xsd:element name="Publication" type="xsd:string"/>
   <xsd:element name="Date" type="xsd:string"/>
   <xsd:element name="Document" type="xsd:anyType"/>
```

anyType 요소와 함께 스키마를 가져오는 경우 스키마 탐색기에서 anyType 요소가 anyType으로 나타납니다. 디자이너는 유형 anyType의 요소에 대한 포트를 작성하지 않습니다.

디자이너에서 anyType 요소를 글로벌 복합 유형으로 변경하여 PowerCenter에서 anyType 요소를 사용해야 합니다.

anyType 요소를 다른 글로벌 복합 유형으로 변경하려면:

1. 스키마 탐색기에서 anyType 요소를 강조 표시합니다.

   anyType 속성이 스키마 탐색기의 속성 탭에 나타납니다.

2. anyType 속성을 클릭합니다.

   스키마에 글로벌 복합 유형이 포함되지 않은 경우 디자이너에는 선택할 글로벌 복합 유형이 없다는 오류가 표시됩니다.

3. 복합 유형을 선택하고 확인을 클릭합니다.

anyAttribute 또는 ANY 요소에 콘텐츠 적용

anyAttribute 또는 ANY 콘텐츠 유형이 포함된 XML 스키마를 가져올 수 있습니다. 보기에서 요소를 사용하려면 XML 편집기에서 요소에 대한 콘텐츠를 정의해야 합니다. anyAttribute 또는 ANY 콘텐츠 요소와 함께 스키마를 가져오는 경우 스키마 탐색기에서 해당 요소가 속성 없이 나타납니다. 해당 요소는 보기에 나타나지 않습니다.

예를 들어 다음 스키마 요소에는 ANY 콘텐츠가 포함되어 있습니다.

```
<xs:element name="person">
   <xs:complexType>
   <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
      <xs:any minOccurs="0"/>
   </xs:sequence>
   </xs:complexType>
</xs:element>
```

콘텐츠를 anyAttribute 또는 ANY 콘텐츠에 적용하려면:

1. 스키마 탐색기에서 요소 콘텐츠 링크를 클릭합니다.

   Edit Any or anyAttribute Type Content(Any 또는 anyAttribute 유형 콘텐츠 편집) 대화 상자가 나타납니다.

2. 유형 추가를 클릭합니다.

   새 행이 나타납니다.

3. XML 정의에 대한 스키마의 유효한 요소 목록에서 요소를 선택합니다.
4. 카디널리티를 선택하고 확인을 클릭합니다.

   대체된 유형이 스키마 탐색기에 나타납니다.

XML 편집기에서 anySimpleType 사용

anySimpleType 요소는 문자열, 정수, 10진수 또는 날짜 등 모든 원자성 데이터 유형을 포함할 수 있습니다. XML 인스턴스 문서에 있는 요소의 데이터 유형을 모르는 경우 스키마에서 anySimpleType을 사용하십시오.

요소를 anySimpleType으로 정의하는 경우 스키마를 가져오면 디자이너가 이 요소에 대한 anySimpleType 열을 작성합니다. 매핑에서 해당 열을 사용하면 XML 소스 한정자가 이 유형을 문자열에 매핑합니다.

통과 포트 추가

통과 포트를 XML 파서 또는 XML 생성기 변환에 추가하여 변환을 통해 XML이 아닌 데이터를 전달할 수 있습니다. 변환에서 포트를 정의할 때 XML 파서 변환의 DataInput 그룹 또는 XML 생성기 변환의 DataOutput 그룹에 포트를 추가합니다.

통과 포트를 생성한 후 다른 포트를 추가하여 변환을 통해 데이터를 전달합니다. 이 포트는 참조 포트입니다. XML 파서 변환에서는 통과 포트가 변환으로 데이터를 전달하고 참조 포트는 변환에서 나온 데이터를 전달합니다. XML 생성기에서는 통과 포트가 변환에서 나온 데이터를 전달하고 참조 포트는 변환으로 데이터를 전달합니다.

XML 정의에 통과 포트가 있는 경우 해당 참조 포트를 결정할 수 있습니다.

통과 포트에 대한 참조 포트를 결정하려면:

1. 통과 포트를 마우스 오른쪽 단추로 클릭합니다.
2. Navigate to(탐색 대상) > Referenced Column(참조된 열)을 선택합니다.

   XML 편집기가 작업 공간에 있는 참조된 열을 강조 표시합니다.

FileName 열 추가

세션을 실행하면 새 루트 값이 데이터에서 발생할 때마다 통합 서비스가 새 대상 XML 파일을 출력합니다. FileName 열을 XML 보기에 추가하여 각 XML 파일에 대한 고유한 파일 이름을 생성할 수 있습니다. 이 파일 이름이 세션 속성의 기본 출력 파일 이름을 재정의합니다.

FileName 열을 사용하는 경우 매핑에서 식 변환 또는 기타 변환을 설정하여 FileName 열에 전달할 고유한 파일 이름을 생성합니다.

XML 보기에서 FileName 열을 작성하려면:

1. XML 편집기에서 보기를 마우스 오른쪽 단추로 클릭합니다.
2. Create FileName Column(FileName 열 작성)을 선택합니다.

   XML 편집기가 보기에서 새 문자열 열을 작성합니다.

3. 열 창에서 열 이름을 변경합니다.

4. XML 편집기를 종료합니다.

   열이 XML 정의에 나타납니다. XPath는 $Filename입니다.

   참고: XML FileName 포트를 사용하는 경우 파일에 대한 디렉터리 이름을 지정해야 합니다.

XPath 쿼리 조건자 작성

XML 계층의 보기 행 아래에 있는 열 또는 보기 행에서 XPath 쿼리 조건자를 작성하여 세션에서 XML 데이터를 필터링할 수 있습니다. XPath 쿼리 조건자를 작성하면 XML 편집기가 XPath 쿼리 조건자를 보기 행 XPath에 추가합니다.

XML 편집기에서 XPath 쿼리 조건자를 작성하면 XML 편집기가 쿼리를 작성하기 위한 요소, 특성, 연산자 및 함수를 제공합니다. 구성 요소를 선택하거나, 구성 요소를 입력하거나, 구성 요소를 쿼리에 복사할 수 있습니다. XML 편집기는 사용자가 작성하는 각 쿼리의 유효성을 검사합니다.

요소 또는 특성의 값을 쿼리하거나, 요소 또는 특성이 존재하는지 확인할 수 있습니다.

특성 요소의 값 쿼리

XPath 쿼리 조건자를 작성하여 보기에서 요소 또는 특성 값을 필터링할 수 있습니다. 예를 들어 부서 100의 직원을 추출하려면 다음 XPath 쿼리 조건자를 작성합니다.

```
EMPLOYEE [./DEPT = '100']
```

쿼리 식은 괄호 안에 있습니다. Dept의 XPath는 "./"로 축약되어 경로가 Employee에서 이어짐을 나타냅니다.

성이 Smith인 경우 다음 XPath 쿼리 조건자가 직원을 추출합니다.

```
EMPLOYEE[./NAME/LASTNAME='SMITH']
```

Name은 Employee의 하위 요소이며 Lastname의 상위입니다.

XPath 쿼리 조건자에서 부울 또는 숫자 연산자를 사용하십시오. 또한 쿼리에 문자열, 숫자 및 부울 함수를 사용할 수 있습니다.

혼합 콘텐츠 쿼리

XML 요소에 값이 포함되고 하위 요소가 포함된 경우 이 요소에는 혼합 콘텐츠가 있습니다. 하위 요소로 나누는 요소 값을 필터링하기 위한 XPath 쿼리 조건자를 작성할 수 있습니다. 하지만 통합 서비스는 혼합 콘텐츠의 첫 번째 하위 요소 다음에 발생하는 조건자를 평가하지 않습니다.

예를 들어 XML 파일에 혼합 콘텐츠와 함께 NAME 요소가 포함될 수 있습니다.

```
<NAME>
  Kathy
    <MIDDLE> Mary </MIDDLE>
  Russell
</NAME>
```

요소 NAME에는 값 "Kathy", 하위 요소 "MIDDLE" 및 두 번째 값 "Russell"이 있습니다. NAME 열 값은 "KathyRussell"입니다. 하지만 통합 서비스가 NAME "Kathy"를 평가합니다.

다음 쿼리는 false입니다.

```
EMPLOYEE [./NAME = 'KathyRussell']
```

다음 쿼리는 true입니다.

```
EMPLOYEE [./NAME = 'Kathy']
```

부울 연산자

XPath 쿼리 조건자에서 다음 부울 연산자를 사용하십시오.

```
and or < <= > >= = !=
```

다음 XPath 쿼리 조건자를 사용하여 부서 100에서 성이 Jones인 직원을 추출하십시오.

```
EMPLOYEE [./DEPT = '100' and ./ENAME/LASTNAME = 'JONES']
```

숫자 연산자

XPath 쿼리 조건자에서 다음 숫자 연산자를 사용하십시오.

```
+ - * div mod
```

다음 XPath 쿼리 조건자를 사용하여 가격이 세금을 더한 비용보다 큰 제품을 추출하십시오.

```
PRODUCT[./PRICE > ./COST + /.TAX]]
```

함수

XPath 쿼리 조건자에서 다음 유형의 함수를 사용합니다.

- **문자열.** 문자열 함수를 사용하여 하위 문자열 값을 테스트하거나, 문자열을 연결하거나, 문자열을 다른 문자열로 변환합니다. 다음 XPath 쿼리 조건자는 직원의 전체 이름이 성과 이름의 연결과 같은지 결정합니다.

  ```
  EMPLOYEE[./FULLNAME=concat(./ENAME/LASTNAME,./ENAME/FIRSTNAME)]
  ```

- **숫자.** 요소 및 특성 값이 포함된 숫자 함수를 사용합니다. 숫자 함수는 숫자에서 작업을 수행하고 정수를 반환합니다. 예를 들어 다음 XPath 쿼리 조건자는 할인을 반올림하여 결과가 15보다 큰지 테스트합니다.

  ```
  ORDER_ITEMS[round(./DISCOUNT > 15]
  ```

- **부울.** 부울 함수는 true 또는 false를 반환합니다. 이 함수를 사용하여 요소를 테스트하거나, 언어 특성을 확인하거나, true 또는 false 결과를 강제 적용하십시오. 예를 들어 문자열 값이 0보다 큰 경우 문자열이 true입니다.

  ```
  boolean(string)
  ```

요소 또는 특성 테스트

요소나 특성이 XML 데이터에서 발생하는지 결정할 수 있습니다. 다음 XPath 쿼리 조건자는 부서에 부서 이름 특성이 있는지 결정합니다.

```
COMPANY/DEPT[@DEPTNAME]/EMPLOYEE
```

Deptname은 특성입니다. 특성은 XML 쿼리 조건자 식에서 “@” 뒤에 와야 합니다.

세션을 실행하면, 직원의 부서에 부서 이름이 있는 경우 통합 서비스가 XML 소스에서 직원 데이터를 추출합니다. 그렇지 않으면 통합 서비스가 직원 데이터를 추출하지 않습니다.

XPath 쿼리 조건자 규칙 및 지침

XPath 쿼리 조건자를 작성할 때 다음 규칙 및 지침을 사용하십시오.

- 보기 행의 모든 요소에 대한 XPath 쿼리 조건자를 구성할 수 있습니다.

  예를 들어 보기 행이 Company/Dept인 경우 다음 XPath 쿼리 조건자를 작성할 수 있습니다.

  ```
  COMPANY[./DEPT=100]
  ```
- 콘텐츠를 일치시킬 수 있습니다.
- 열이 보기 XML 계층의 보기 행 아래에서 발생하고 열 XPath에 보기 행이 포함된 경우 XPath 쿼리 조건자를 열에 추가할 수 있습니다.

  예를 들어 보기 행이 Product/Toys[1]인 경우 다음 XPath 쿼리 조건자를 작성할 수 있습니다.

  ```
  Product/Toys[1][./Sales > 100]
  ```

  다음 예에서는 Product/Toys[1] 보기 행에 대한 올바르지 않은 XPath 쿼리 조건자를 보여 줍니다.

  ```
  Product/Toys[2][./Sales > 100]
  ```

  Product/Toys[1]은 보기 행입니다. Product/Toys[2]는 사용할 수 없습니다.
- 단일 발생 요소 또는 특성을 사용하십시오. 다중 발생 요소의 XPath 쿼리 조건자는 작성할 수 없습니다.
- 삽입 요소에 값이 포함되지 않기 때문에 삽입 요소의 XPath 조건자 쿼리를 작성할 수 없습니다.

XPath 쿼리 조건자 작성 단계

XML 소스 정의에서 보기에 대한 XPath 쿼리 조건자를 작성할 수 있습니다.

XPath 쿼리 조건자를 작성하려면:

1. XML 편집기에서 XML 소스 정의를 엽니다.
2. XML 편집기 작업 공간에서 보기를 선택합니다.

   보기 열이 열 창에 나타납니다.
3. 보기 행 XPath를 클릭합니다.

   Specify Query Predicate for XPath(XPath의 쿼리 조건자 지정) 창이 나타납니다. XPath 조건자 창에서 XPath 쿼리 조건자를 입력하거나 대화 상자의 탭에서 요소, 연산자 및 함수를 선택할 수 있습니다.
4. Child Elements and Attributes(하위 요소 및 특성) 탭을 클릭하여 XPath 쿼리 조건자에 추가할 수 있는 요소 및 특성을 표시합니다.
5. 요소 또는 특성을 두 번 클릭하여 XPath 쿼리 조건자에 추가합니다.

   구성 요소가 패널에 나타납니다.
6. 연산자 탭을 클릭합니다.

   연산자를 사용하여 식을 작성합니다. 요소를 값이나 다른 요소와 비교하거나 수학 식을 작성할 수 있습니다.

다음 테이블에는 XPath 쿼리 조건자에 추가할 수 있는 연산자가 설명되어 있습니다.

연산자	설명
+	추가
-	빼기
*	곱하기
div	나누기
mod	모듈러스
and	부울 및
or	부울 또는
<	보다 작음
<=	작거나 같음
>	보다 큼
>=	크거나 같음
=	같음
!=	같지 않음

7. 연산자를 두 번 클릭하여 XPath 쿼리 조건자에 이 연산자를 추가합니다.
8. PowerCenter XPath 쿼리 조건자 함수를 추가하려면 함수 탭을 클릭합니다. 함수는 인수를 허용하고 값을 반환합니다.
9. Child Element and Attributes(하위 요소 및 특성) 탭에서 다른 요소나 특성을 선택하거나 XPath 조건자 창에 값을 입력하여 식을 작성합니다.
10. 유효성 검사를 클릭하여 XPath 쿼리 조건자의 유효성을 검사합니다.

보기 관계 유지 관리

다음 태스크를 완료하여 XML 편집기에서 보기 관계를 유지 관리할 수 있습니다.

- **보기 간 관계 작성.** 작업 공간에서 보기 간 관계를 정의합니다.
- **유형 관계 작성.** 작업 공간의 보기 및 유형 보기에 있는 열 사이 유형 관계를 정의합니다.
- **항목 관계 다시 작성.** XML 마법사와 동일한 옵션을 사용하여 보기 및 관계를 생성합니다.

보기 간 관계 작성

XML 편집기를 사용하여 보기 간 계층 또는 상속 관계를 작성하십시오. XML 소스 및 XML이 아닌 소스 간 관계는 작성할 수 없습니다.

XML 보기 간 관계를 작성하려면:

1. 작업 공간에서 하위 XML 보기의 상단을 마우스 오른쪽 단추로 클릭합니다.
2. 관계 작성을 선택합니다.
3. 상위 보기로 포인터를 이동하여 관계를 설정합니다.

   포인터를 이동하면 보기 간에 링크가 나타나고 XML 편집기에서 관계가 올바른지 확인합니다. 관계가 올바르지 않은 경우 상태 표시줄에 오류 메시지가 나타납니다.
4. 오류 메시지가 나타나지 않으면 상위 보기를 클릭하여 관계를 설정합니다.

   XML 편집기가 관계를 작성하고 해당 외래 키를 추가합니다.
5. 관계에 대한 세부 정보를 보려면 보기 간 링크 위에 커서를 놓습니다.

   편집기에 관계 유형 및 기본 키와 외래 키가 표시됩니다.

유형 관계 작성

보기 및 유형 보기에 있는 열 사이 유형 관계를 작성할 수 있습니다. 열이 피벗된 경우 관계에 포함시킬 발생을 선택할 수 있습니다.

유형 관계를 작성하려면:

1. 사용할 보기의 열을 마우스 오른쪽 단추로 클릭합니다.
2. 관계 작성을 선택합니다.
3. 열이 피벗된 경우, 사용할 발생을 선택합니다.
4. 유형 보기로 포인터를 이동하여 관계를 설정합니다.

   포인터를 이동하면 보기 간에 링크가 나타나고 XML 편집기에서 관계가 올바른지 확인합니다. 관계가 올바르지 않은 경우 상태 표시줄에 오류 메시지가 나타납니다.
5. 오류 메시지가 나타나지 않으면 상위 보기를 클릭하여 관계를 설정합니다.

   XML 편집기에서 관계를 작성합니다.

항목 관계 다시 작성

XML 정의에 대한 항목 관계를 다시 작성할 수 있습니다. Recreate Entity Relationships(항목 관계 다시 작성) 대화 상자에서 XML 마법사와 동일한 옵션을 사용하여 새로운 XML 보기를 생성하십시오. 보기를 다시 생성하는 경우 기존 보기를 유지하도록 선택할 수 있습니다. XML 정의에는 모든 보기가 포함됩니다.

XML 정의에 대한 항목 관계를 다시 작성하려면:

1. XML 편집기에서 XML 정의를 엽니다.
2. 탐색기에서 XML 루트를 강조 표시합니다.

   다른 구성 요소를 강조 표시하는 경우 XML 편집기에서 해당 구성 요소를 루트로 사용합니다.
3. XML Views(XML 보기) > Create Entity Relationship(항목 관계 작성)을 클릭합니다.

4. 다음 옵션 중에서 선택하여 메타데이터 확장을 줄입니다.
   - **요소 확장 감소.** 다중 발생 또는 참조 요소의 경우 XML 마법사가 여러 계층 관계가 포함된 하나의 XML 보기를 작성합니다.
   - **복합 유형 확장 감소.** 다중 발생 또는 참조된 복합 유형의 경우 XML 마법사가 여러 유형 관계가 포함된 하나의 XML 보기를 작성합니다. 스키마가 상속된 복합 유형을 사용하는 경우 상속된 복합 유형의 확장도 감소할 수 있습니다.
   - **복합 유형 상속 확장 감소.** 상속된 유형의 경우 XML 마법사가 여러 유형 관계를 사용하여 하나의 XML 보기를 작성합니다.
   - **기존 XML 보기 공유.** 기존 XML 보기를 제거하지 마십시오.
   - **공유 XML 보기 새로 고침.** 기존 보기를 저장할 뿐만 아니라 업데이트도 합니다.
5. 다음을 클릭합니다.

   (항목 관계 다시 작성) 대화 상자가 나타납니다.
6. 하위 구성 요소를 표시하려면 공유 요소나 복합 유형을 선택하고 이름을 클릭합니다.
7. 하위 구성 요소를 제외하려면 Exclude Child Components(하위 구성 요소 제외) 창에서 해당 요소를 선택 취소합니다.

   새 보기를 생성하려면 요소 또는 복합 유형을 선택합니다. 새 항목 관계를 작성하는 경우 해당 요소가 포함된 보기를 보기 루트로 생성합니다.

스키마 구성 요소 보기

탐색기 및 작업 공간에서 구성 요소를 보려면 다음 태스크를 완료하십시오.

- **네임스페이스 업데이트.** XML 대상에서 스키마 위치 또는 기본 네임스페이스를 변경합니다.
- **구성 요소로 이동.** 하나의 구성 요소에서 다른 구성 요소 또는 XML 편집기 창의 영역으로 이동하여 구성 요소를 찾습니다.
- **작업 공간에서 보기 정렬.** 작업 공간에서 보기를 계층에 따라 정렬합니다. 작업 공간에서 계층 정렬로 보기를 구성할 수 있습니다. 작업 공간에서 보기를 정렬하려면 레이아웃 > Arrange(정렬)를 클릭하거나, 작업 공간을 마우스 오른쪽 단추로 클릭하고 Arrange(정렬)를 선택합니다.
- **구성 요소 검색.** 탐색기나 작업 공간에서 구성 요소를 찾습니다.
- **단순 또는 복합 유형의 계층 표시.** XML 스키마에서 단순 또는 복합 유형의 계층을 봅니다.
- **XML 메타데이터 보기.** XML 편집기가 XML 정의에서 작성하는 XML 파일, 스키마 또는 DTD를 봅니다.
- **XML 데이터 미리보기.** 외부 XML 파일의 샘플 데이터를 사용하여 XML 보기를 표시합니다.
- **XML 정의 유효성 검사.** XML 정의의 유효성을 검사하고 오류를 봅니다.

네임스페이스 업데이트

XML 정의를 작성할 때 XML 편집기에서 네임스페이스 접두사 및 스키마 위치를 변경할 수 있습니다. 네임스페이스에 스키마를 추가할 수도 있습니다.

하나 이상의 네임스페이스가 있는 대상 XML 정의를 작성하는 경우 기본 네임스페이스를 선택할 수 있습니다. 세션을 실행하면 통합 서비스가 네임스페이스 접두사 없이 기본 네임스페이스의 요소 및 특성을 씁니다.

네임스페이스 접두사로 “xml” 또는 “xmlns”를 사용하지 마십시오. “xml” 접두사는 기본적으로 http://www.w3.org/XML 네임스페이스를 가리킵니다. “Xmlns”는 XML 스키마의 네임스페이스에 요소를 연결합니다.

참고: XML 편집기를 사용하여 네임스페이스를 추가할 수 없습니다.

네임스페이스를 업데이트하려면:

1. 탐색기에서 요소를 선택합니다.
2. 속성 탭을 클릭합니다.
3. 네임스페이스 링크를 클릭합니다.

   Edit Namespace Prefix and Schema Location(네임스페이스 접두사 및 스키마 위치 편집) 대화 상자가 나타납니다.
4. 접두사 또는 스키마 위치를 변경하려면 변경할 텍스트를 선택하고 새 값을 입력합니다.
5. 네임스페이스에 두 개 이상의 스키마를 추가하려면 스키마 위치를 선택하고 추가를 클릭합니다.

   빈 스키마가 네임스페이스 스키마 위치에 나타납니다.
6. 스키마를 삭제하려면 스키마 위치를 강조 표시하고 삭제를 클릭합니다.
7. XML 대상에서 기본 네임스페이스를 작성하려면 네임스페이스를 선택합니다.

   선택하는 네임스페이스의 모든 구성 요소는 XML 대상의 기본 네임스페이스에 속합니다. 세션을 실행하면 통합 서비스가 XML 대상 파일에서 기본 네임스페이스 접두사를 쓰지 않습니다.

구성 요소로 이동

빠르게 구성 요소를 찾으려면 대규모 XML 정의에서 탐색할 작업 공간 구성 요소를 선택하고 탐색 옵션을 선택합니다. 예를 들어 보기에서 외래 키를 클릭하면 열 창의 열이나 연결된 기본 키로 이동할 수 있습니다. 작업 공간, 열 창 및 탐색기에서 구성 요소 간에 이동할 수 있습니다.

구성 요소로 이동하려면:

1. 작업 공간 또는 열 창에서 구성 요소를 마우스 오른쪽 단추로 클릭합니다.
2. Navigate to(탐색 대상)를 선택합니다.
3. 사용 가능한 옵션을 선택합니다.

   탐색하기 위해 선택하는 구성 요소에 따라 다음 옵션 중에서 선택할 수 있습니다.
   - **스키마 구성 요소.** 탐색기에서 구성 요소를 강조 표시합니다.
   - **PowerCenter 열.** 열 창에서 열을 강조 표시합니다.
   - **기본 키**. 선택된 외래 키와 연결된 기본 키를 강조 표시합니다.
   - **참조된 열.** XML 파서 또는 생성기 변환에서 통과 포트와 연결되어 있는 참조된 열을 강조 표시합니다.
   - **XPath 탐색기.** 선택된 구성 요소에 대한 경로를 표시합니다.
   - **XML 보기.** 열 창의 선택된 열이 포함된 작업 공간에서 보기를 강조 표시합니다.

구성 요소 검색

XML 정의에서 구성 요소를 검색할 수 있습니다. XML 편집기에 구성 요소의 모든 발생이 표시됩니다. 검색 결과를 클릭할 수 있으며 XML 편집기에는 스키마 또는 보기의 구성 요소가 강조 표시됩니다. 작업 공간에서 XML 스키마 또는 XML 보기를 검색할 수 있습니다.

XML 스키마 검색

이름, 유형 및 네임스페이스 기준으로 구성 요소를 검색할 수 있습니다. 주석, 고정값이나 기본값 또는 길이 등 검색할 속성을 지정할 수 있습니다. 열거 속성을 사용하여 유효한 값을 검색할 수 있습니다.

스키마에서 구성 요소를 검색하려면:

1. 편집 > Search In Schema(스키마에서 검색)를 클릭합니다.

   Search Components(구성 요소 검색) 대화 상자가 나타납니다.
2. 구성 요소 속성 기준으로 검색하려면 고급 옵션 링크를 클릭하여 검색할 수 있는 속성을 봅니다.
3. 검색할 이름, 유형 또는 속성을 입력합니다.
4. 특정 네임스페이스에서 검색하려면 모두를 클릭하고 목록에서 네임스페이스를 선택합니다.
5. 검색을 클릭합니다.

   검색 결과가 대화 상자에 나타납니다.
6. 검색 결과를 클릭하여 속성 창에서 구성 요소를 봅니다.

XML 보기에서 검색

XML 보기에서 구성 요소 및 열을 검색할 수 있습니다. 구성 요소 이름 기준으로 검색하는 경우 정의에서 연결된 열을 모두 찾을 수 있습니다. 예를 들어 구성 요소 "숫자"를 검색하면 결과에는 구성 요소 "숫자"가 들어 있는 보기 및 열이 포함됩니다.

부분 키를 사용하여 검색하려면 열 또는 구성 요소 이름의 처음 몇 개 문자를 입력하십시오.

XML 보기에서 구성 요소를 검색하려면:

1. 편집 > Search XML Views(XML 보기 검색)를 클릭합니다.

   Search XML Views and Columns(XML 보기 및 열 검색) 대화 상자가 나타납니다.
2. 검색 조건을 입력합니다.

   모든 열 유형, 일반 열 유형, 생성된 키 또는 기타 유형을 검색할 수 있습니다. 기타 유형에는 FileName 열, 참조 포트 및 통과 필드가 포함됩니다.
3. 검색을 클릭합니다.

   검색 결과가 대화 상자에 나타납니다.
4. 검색 필드를 지우려면 New Search(새 검색)를 클릭합니다.

단순 또는 복합 유형 계층 보기

스키마 정의에서 XML 단순 유형의 계층을 볼 수 있습니다. 단순 유형의 계층을 보려면 보기 > SimpleType Hierarchy(SimpleType 계층)를 클릭하십시오. 창에 단순 유형의 계층이 표시됩니다.

스키마 정의에서 복합 유형의 계층을 표시할 수 있습니다. 복합 유형의 계층을 보려면 보기 > ComplexType Hierarchy(ComplexType 계층)를 클릭하십시오. 스키마에 있는 복합 유형의 계층이 창에 표시됩니다. ComplexType Hierarchy(ComplexType 계층) 창에서 구성 요소를 선택하여 스키마의 구성 요소로 이동하십시오.

XML 메타데이터 보기

XML, DTD 또는 XML 스키마 파일로 XML 정의를 볼 수 있습니다.

XML 메타데이터를 보려면:

1. 샘플 XML 문서로 메타데이터를 보려면 탐색기에서 글로벌 구성 요소를 선택합니다.
2. 보기 > XML Metadata(XML 메타데이터)를 클릭합니다.

   View XML Metadata(XML 메타데이터 보기) 대화 상자가 나타납니다.
3. XML 정의를 XML 파일, DTD 파일 또는 XML 스키마로 표시하도록 선택합니다.

   여러 네임스페이스를 사용하는 경우 사용할 네임스페이스를 선택합니다.

   기본 응용 프로그램 또는 텍스트 편집기에 메타데이터가 표시됩니다.
4. XML, DTD 또는 XML 스키마 파일의 사본을 저장하려면 다른 이름으로 저장을 클릭합니다.
5. 새 파일 이름을 입력합니다.

   기본 표시 응용 프로그램이 텍스트 편집기인 경우 파일 이름과 함께 해당 파일 접미사를 포함해야 합니다. 접미사는 작업 중인 파일 유형에 따라 .xml, .dtd 또는 .xsd입니다.

XML 정의 유효성 검사

XML 편집기에서 XML 정의의 유효성을 검사할 수 있습니다.

XML 정의의 유효성을 검사하려면:

1. XML 편집기에서 정의를 엽니다.
2. 작업 공간 내부를 클릭합니다.
3. XML Views(XML 보기) > Validate XML Definition(XML 정의 유효성 검사)을 클릭합니다.

   유효성 검사 창에 결과가 표시됩니다.

XML 보기 옵션 설정

기본적으로 통합 서비스는 보기 행에 데이터가 있는 각 보기에 대한 행을 생성합니다. 통합 서비스가 일부 XML 소스 정의 보기에 대한 행을 생성하는 방법을 변경하려면 XML Columns(XML 열) 창에서 XML View Options(XML 보기 옵션)를 선택합니다.

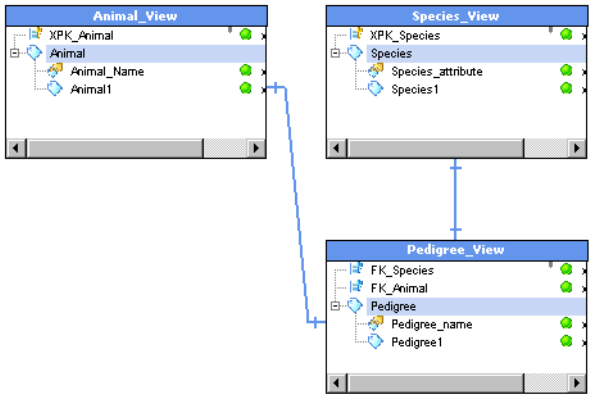
다음 방법을 사용하여 통합 서비스가 언제 XML 소스에서 행을 생성하는지 결정하십시오.

- **보기에서 모든 외래 키 생성.** 기본적으로 통합 서비스는 보기에서 하나의 외래 키에 대한 값을 생성합니다. 보기에 두 개 이상의 외래 키가 있는 경우 다른 외래 키에 null 값이 있습니다. 모든 외래 키에 대한 값을 생성할 수 있습니다.
- **원형 관계에서 반복 읽기 중지.** 기본적으로 통합 서비스는 원형 관계에서 모든 데이터 발생에 대한 행을 생성합니다. 반복 데이터의 첫 번째 발생에 대해서만 행을 생성할 수 있습니다.
- **상위가 있는 경우 하위 보기에 대한 행 생성.** 기본적으로 통합 서비스는 보기 행에서 데이터가 있는 모든 보기에 대한 행을 작성합니다. 상위 보기에 데이터가 있는 경우에만 하위 보기에 대한 행을 생성할 수 있습니다.
- **보기 행 데이터 없이 보기에 대한 행 생성.** 기본적으로 통합 서비스는 보기 행에 데이터가 없는 경우 보기에 대한 데이터를 생성합니다. 보기 행에 데이터가 없는 보기에 대한 행을 생성할 수 있습니다.
- **유형 관계에서 특정 복합 유형에 대한 행 생성.** 기본적으로 통합 서비스는 XML 정의에서 모든 보기에 대한 행을 생성합니다. 유형 관계에서 특정 보기에 대한 행을 생성할 수 있습니다.

모든 계층 외래 키 생성

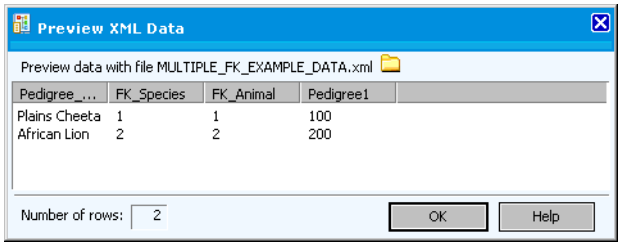
기본적으로 통합 서비스는 보기에서 하나의 외래 키에 대한 값을 생성합니다. 보기에 두 개 이상의 외래 키가 있는 경우 다른 외래 키에 null 값이 있습니다. 보기에서 모든 외래 키에 대한 키 값을 생성하도록 선택할 수 있습니다. All Hierarchy Foreign Keys(모든 계층 외래 키) 옵션을 선택하십시오.

다음 그림에는 두 개의 외래 키 FK_Species 및 FK_Animal이 있는 Pedigree_View가 나와 있습니다.



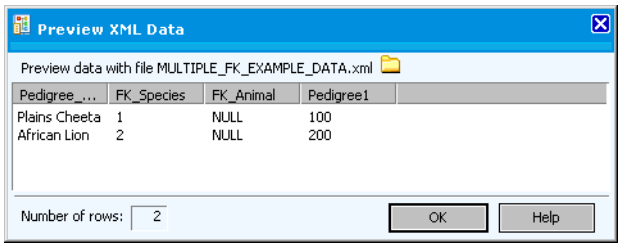
Pedigree_View에 대해 All Hierarchy Foreign Keys(모든 계층 외래 키) 옵션을 선택하는 경우 통합 서비스가 FK_Species 및 FK_Animal에 대한 키 값을 생성합니다.

다음 그림에는 All Hierarchy Foreign Keys(모든 계층 외래 키) 옵션을 사용하는 Pedigree_View에 대한 샘플 데이터가 나와 있습니다.



All Hierarchy Foreign Keys(모든 계층 외래 키) 옵션을 선택 취소하면 통합 서비스가 하나의 외래 키 열에 대한 키 값을 생성합니다. 이 예에서는 Species_View가 XML 계층에서 Pedigree_View의 가장 가까운 상위이기 때문에 통합 서비스가 FK_Species에 대한 값을 생성합니다. FK_Animal 외래 키에는 null 값이 있습니다.

다음 그림에는 All Hierarchy Foreign Key(모든 계층 외래 키) 옵션을 선택 취소하는 경우 Pedigree_View에 대한 샘플 데이터가 나와 있습니다.



통합 서비스가 가장 가까운 상위에 대한 외래 키를 생성합니다.

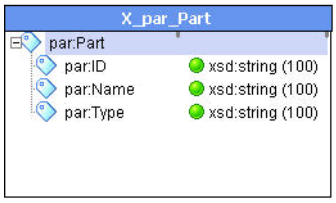
원형 관계에서 행 생성

기본적으로 통합 서비스는 원형 관계에서 모든 데이터 발생에 대한 행을 생성합니다. 첫 번째 발생에 대해서만 행을 작성하도록 선택할 수 있습니다. Non-Recursive row(비반복 행) 옵션을 선택하십시오.

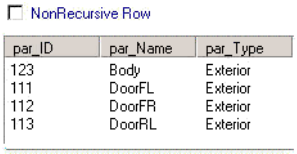
다음 XML 파일에는 순환 참조가 있는 Part 요소가 포함되어 있습니다.

```
<?xml version="1.0" encoding="utf-8"?>
<Vehicle xmlns="http://www.PartInvoice.org"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.PartInvoice.org part.xsd">
<VehInfo>
      <make>Honda</make>
      <model>Civic</model>
      <type>Compact</type>
</VehInfo>
<Part>
  <ID>123</ID>
  <Name>Body</Name>
  <Type>Exterior</Type>
  <Part>
    <ID>111</ID>
    <Name>DoorFL</Name>
        <Type>Exterior</Type>
        <Part>
      <ID>1112</ID>
      <Name>KnobL</Name>
      <Type>Exterior</Type>
    </Part>
    <Part>
      <ID>1113</ID>
      <Name>Window</Name>
      <Type>Exterior</Type>
    </Part>
  </Part>
</Part>
</Vehicle>
```

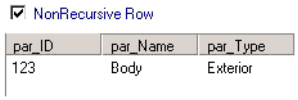
다음 그림에서는 Part 요소가 XML 정의에서 X_par_Part 보기에 대한 보기 행임을 보여 줍니다.



다음 그림에서는 세션을 실행할 때 통합 서비스가 Part 123 및 모든 구성 요소 부분에 대한 행을 생성함을 보여 줍니다.



다음 그림에서는 NonRecursive Row(비반복 행)를 선택하면 통합 서비스가 Part 요소의 첫 번째 발생을 읽고 Part 123에 대한 하나의 데이터 행을 생성함을 보여 줍니다.



계층 관계 행 생성

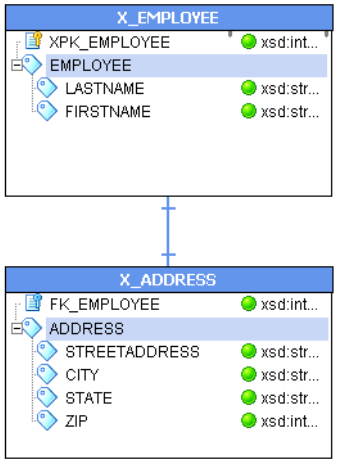
기본적으로 통합 서비스는 보기 행에서 데이터가 있는 모든 보기에 대한 행을 작성합니다. 상위 보기에 계층 관계의 해당 데이터가 있는 경우 Hierarchy Relationship Row(계층 관계 행)를 선택하여 하위 보기에 대한 행을 생성하십시오. 상위 보기에는 하위 보기에 대한 행을 생성하기 위한 데이터가 있어야 합니다.

예를 들어 XML 정의에 Employee 보기 및 Address 보기로 구성된 계층이 있을 수 있습니다. Employee가 상위 보기입니다. 주소 데이터는 Employee\Addresses 또는 Store\Addresses를 포함할 수 있습니다. Employee\Address를 출력하도록 선택할 수 있습니다.

다음 XML 파일에는 Store 요소 내 Address 및 Employee 요소 내 Address가 있습니다

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE STORE >
<STORE SID="BE1752">
  <SNAME>Mud and Sawdust Furniture Store</SNAME>
  <ADDRESS>
    <STREETADDRESS>335 Westshore Road</STREETADDRESS>
    <CITY>Fausta City</CITY>
    <STATE>CA</STATE>
    <ZIP>97584</ZIP>
  </ADDRESS>
  <EMPLOYEE DEPID="34">
    <ENAME>
      <LASTNAME>Bacon</LASTNAME>
      <FIRSTNAME>Allyn</FIRSTNAME>
    </ENAME>
  <ADDRESS>
    <STREETADDRESS>1000 Seaport Blvd</STREETADDRESS>
    <CITY>Redwood City</CITY>
    <STATE>CA</STATE>
    <ZIP>94063</ZIP>
  </ADDRESS>
 <EPHONE>(408)226-7415</EPHONE>
 <EPHONE>(650)687-6831</EPHONE>
</EMPLOYEE>
</STORE>
```

다음 그림에는 Employee 보기 및 Address 보기 간 계층 관계가 나와 있습니다.



Employee 보기는 상위 및 하위 보기 간 일대일 관계를 정의하는 파란색 선이 있는 Address 보기에 연결되어 있습니다. Employee 보기에는 기본 키 XPK_Employee 그리고 LastName 및 FirstName으로 구성된 Employee 요소가 있습니다. Address 보기에는 외래 키 FK_Employee 그리고 StreetAddress, City, State 및 Zip으로 구성된 Address 요소가 있습니다.

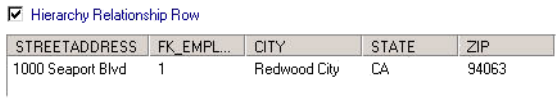
기본적으로 통합 서비스는 Address 요소의 각 발생에 대한 행을 생성합니다. 통합 서비스는 Store\Address에 대한 행 하나와 Employee\Address에 대한 행 하나를 생성합니다.

다음 그림에는 Hierarchy Relationship Row(계층 관계 행) 옵션을 선택 취소하는 경우 Address XML 데이터가 나와 있습니다.

Hierarchy Relationship Row(계층 관계 행) 옵션을 선택하면 통합 서비스가 다음과 같이 세션에서 행을 생성합니다.

- Employee 보기에 세션의 해당 데이터가 있는 경우 통합 서비스가 Address 보기에 대한 행을 생성합니다.
- 통합 서비스가 Employee\Address 계층 관계를 나타내는 행을 생성합니다.
- 통합 서비스가 Store\Address에 대한 행을 생성하지 않습니다.

다음 그림에는 Hierarchy Relationship Row(계층 관계 행) 옵션을 선택하는 경우 Address 데이터가 나와 있습니다.

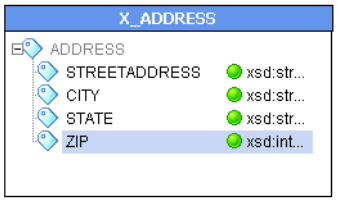


Force Row(강제로 행 적용) 옵션 설정

기본적으로 통합 서비스는 보기 행에 데이터가 없는 경우 보기에 대한 데이터를 생성합니다. 보기 행 요소에 데이터가 포함되어 있는지 관계없이 Force Row(강제로 행 적용)를 선택하여 XML 보기에 대한 행을 생성하

십시오. 예를 들어 보기 행이 address\zip인 경우 이 보기 행에 zip 데이터가 없더라도 주소를 출력하도록 선택할 수 있습니다.

다음 그림에는 보기 행으로 zip 요소가 나와 있습니다.



목록 하단에 zip 요소가 강조 표시되어 있으며, zip 요소 위에 street, city 및 state 요소 행이 나열되어 있습니다.

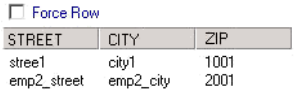
기본적으로 통합 서비스는 address 요소 내 zip 요소의 각 발생에 대한 행을 생성합니다.

예를 들어 세션에서 다음 XML 파일을 처리할 수 있습니다.

```
<?xml version="1.0" ?>
<company xmlns="http://www.example.org"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.example.org forcerow.xsd" >
<name>company1</name>
  <address>
    <street>stree1</street>
    <city>city1</city>
    <zip>1001</zip>
  </address>
<employee>
  <name>emp1</name>
  <address>
    <street>emp1_street</street>
    <city>empl_city</city>
    </address>
</employee>
<employee>
  <name>emp2</name>
  <address>
    <street>emp2_street</street>
    <city>emp2_city</city>
    <zip>2001</zip>
  </address>
</employee>
</company>
```

기본적으로 통합 서비스는 Address 보기에 대한 stree1 및 emp2_street 행을 생성합니다.

다음 그림에는 Address 보기에 대한 stree1 및 emp2_street 행이 나와 있습니다.



보기 행이 zip이고 emp1에 zip 요소의 데이터가 없는 경우 통합 서비스가 emp1에 대한 행을 생성하지 않습니다.

Force Row(강제로 행 적용)를 활성화하면 zip이 있거나 없는 street 및 city 요소를 출력할 수 있습니다. emp1에 zip 요소의 데이터가 없더라도 세션에서 emp1에 대한 행을 생성합니다.

다음 그림에는 Force Row(강제로 행 적용)를 활성화하는 경우 통합 서비스에서 생성하는 행이 나와 있습니다.

유형 관계에서 보기에 대한 행 생성

기본적으로 통합 서비스는 유형 관계에서 모든 보기에 대한 행을 생성합니다. Type Relationship Row(유형 관계 행) 옵션을 선택하여 유형 관계에서 특정 복합 유형에 대한 행을 생성하십시오. 출력하려는 각 보기에 대한 Type Relationship Row(유형 관계 행)를 설정하십시오.

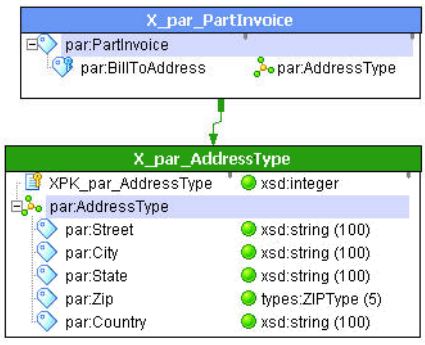
예를 들어 정의에 BillToAddress 및 ShipToAddress가 포함된 계층이 있을 수 있습니다. BillToAddress에 대한 행을 생성하려면 Type Relationship Row(유형 관계 행) 옵션을 사용하십시오.

유형 관계 정의

다음 스키마는 BillToAddress 및 ShipToAddress를 정의합니다. BillToAddress는 AddressType이며 ShipToAddress는 USAddressType입니다. USAddressType은 AddressType을 확장합니다.

```
<xsd:sequence>
  <xsd:element name="Street" type="xsd:string" />
  <xsd:element name= "City" type="xsd:string" />
  <xsd:element name="State" type="xsd:string" />
  <xsd:element name="Zip" type="types:ZIPType" />
  <xsd:element name="Country" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
  <xsd:complexType name="USAddressType">
    <xsd:complexContent>
    <xsd:extension base="AddressType">
      <xsd:sequence>
      <xsd:element name="PostalCode" type="xsd:string" minOccurs="0"/>
      <xsd:sequence>
    </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:element name="BillToAddress" type="AddressType" />
<xsd:element name="ShipToAddress" type="USAddressType" />
```

다음 그림에는 X_par_PartInvoice XML 보기의 BillToAddress 요소 및 아래에 관련 X-par_AddressType XML 보기를 보여 주는 PartInvoice 보기가 있는 XML 정의의 유형 관계가 나와 있습니다



PartInvoice 보기에는 invoice 데이터가 포함되어 있습니다. 해당 보기에는 BillToAddress가 포함되어 있습니다. XML 정의의 유형 관계는 BillToAddress 및 AddressType 사이에 있습니다.

AddressType 데이터를 BillToAddress로 제한하려면 XML 편집기 작업 공간에서 X_par_PartInvoice 보기를 선택하십시오. Type Relationship Row(유형 관계 행) 옵션을 선택하십시오. 세션을 실행하면 통합 서비스가 Address 행을 BillToAddress에 대해 생성하지만 ShipToAddress에 대해서는 생성하지 않습니다. ShipToAddress는 유형 관계에 있지 않습니다.

예제

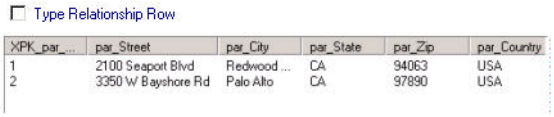
다음 예에서는 유형 관계의 특정 유형으로 데이터를 제한하는 방법을 보여 줍니다. 이 예는 PartInvoice 보기 및 AddressType 보기를 사용합니다.

다음 XML 파일에는 BillToAddress 및 ShipToAddress가 들어 있는 invoice 데이터가 포함되어 있습니다.

```
<xsd:complexType name="AddressType">
<?xml version="1.0" encoding="utf-8"?>
<Invoices xmlns="http://www.PartInvoice.org"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.PartInvoice.org Part.xsd">
<PartInvoice InvoiceNum="185" DateShipped="2005-01-01">
    <PartOrder>
          <PartID>HLG100</PartID>
          <PartName>Halogen Bulb</PartName>
          <Quantity>2</Quantity>
          <UnitPrice>35</UnitPrice>
    </PartOrder>
        <BillToAddress>
              <Street>2100 Seaport Blvd</Street>
              <City>Redwood City</City>
              <State>CA</State>
              <Zip>94063</Zip>
              <Country>USA</Country>
        </BillToAddress>
        <ShipToAddress xsi:type="USAddressType">
              <Street>3350 W Bayshore Rd</Street>
              <City>Palo Alto</City>
              <State>CA</State>
              <Zip>97890</Zip>
              <Country>USA</Country>
              <PostalCode>97890</PostalCode>
        </ShipToAddress>
    </PartInvoice>
</Invoices>
```

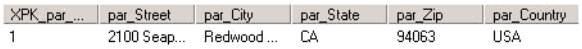
세션을 실행하면 통합 서비스가 각 AddressType에 대한 X_par_AddressType 행을 생성합니다.

다음 그림에는 통합 서비스가 기본적으로 생성하는 BillToAddress 및 ShipToAddress 행이 나와 있습니다.



PartInvoice 보기와 관련된 AddressType 행을 생성하려면 PartInvoice 보기에 대한 Type Relationship Row(유형 관계 행) 옵션을 설정하십시오.

다음 그림에는 Type Relationship Row(유형 관계 행) 옵션이 생성하는 BillToAddress 행이 나와 있습니다.



XPK_par_...	par_Street	par_City	par_State	par_Zip	par_Country
1	2100 Seap...	Redwood ...	CA	94063	USA

ShipToAddress가 유형 관계에 있지 않기 때문에 ShipToAddress에 대한 행이 생성되지 않습니다.

XML 편집기 작업 문제 해결

XML 정의에 대한 유효성을 검사하면 XML 정의가 너무 크다는 오류가 발생합니다. 이 오류가 발생하는 이유는 무엇입니까?

무한 길이로 정의된 구성 요소가 있는 XML 파일을 가져오는 경우 총 열 길이가 500MB 제한을 쉽게 초과할 수 있습니다. XML 편집기에서 열 길이를 변경하거나, 모든 무한 길이를 재정의하는 옵션을 설정하고 파일을 다시 가져올 수 있습니다.

XML 메타데이터를 볼 때 내가 만든 DTD 또는 XML 스키마 파일을 찾을 수 없습니다.

사용자가 볼 수 있는 DTD 또는 XML 스키마 파일은 디자이너가 보기 위해 작성하는 임시 파일입니다. 기타 용도로 해당 파일을 사용하려면 이 파일을 볼 때 다른 이름 및 디렉터리로 저장하십시오.

XML 소스 보기에 열을 추가하면 소스 XML 파일의 계층이 동일하게 남아 있습니다.

XML 소스 보기에 열을 추가할 때 기본 계층에 요소를 추가하는 것이 아닙니다. 보기를 작성하는 방법이나 보기의 열을 계층의 요소에 매핑하는 방법에 관계없이, 가져오는 XML 계층은 동일하게 유지됩니다. 요소의 카디널리티 및 데이터 유형을 수정할 수 있지만, 계층의 구조는 수정할 수 없습니다.

PowerCenter에서 XML 크기 조정에 대한 자세한 내용은 "PowerCenter에서 XML 사용 개요" 페이지 28을 참조하십시오. PowerCenter에서 XML 처리에 적용되는 제한에 대한 자세한 내용은 "제한" 페이지 29을 참조하십시오.

다른 요소 유형을 가진 변환을 작성하고 더 큰 XML 입력 파일을 변환하려면 데이터 프로세서 변환을 사용합니다. 데이터 프로세서 변환을 작성하는 방법에 대한 자세한 내용은 *Informatica Data Transformation 사용자 가이드* 및 *Informatica Data Transformation 시작하기 가이드*를 참조하십시오.

XML 대상 작업

XML 대상 작업 개요

다음과 같은 방법으로 XML 대상 정의를 작성할 수 있습니다.

- **XML 스키마 또는 파일에서 정의 가져오기.** XML, DTD 또는 XML 스키마 파일에서 대상 정의를 작성할 수 있습니다. URL 또는 로컬 노드에서 XML 파일 정의를 가져올 수 있습니다. 연결된 DTD와 함께 XML 파일을 가져오는 경우 XML 마법사가 해당 DTD를 사용하여 XML 파일을 생성합니다.
- **XML 소스 정의를 기반으로 XML 대상 정의 작성.** 기존 XML 소스 정의를 대상 디자이너로 끌 수 있습니다. XML 대상 정의를 작성하는 경우 디자이너가 XML 정의 계층을 기반으로 대상 정의를 작성합니다.

- **관계형 파일 정의를 기반으로 XML 대상 작성**. 관계형 또는 플랫 파일 리포지토리 정의에서 XML 대상 정의를 가져올 수 있습니다.

XML 대상 정의 작성뿐만 아니라 대상 디자이너에서 XML 대상을 사용하여 다음 태스크를 완료할 수 있습니다.

- **대상 속성 편집.** XML 대상 정의를 편집하여 대상 XML, DTD 또는 XML 스키마 파일 변경에 대한 설명을 추가하십시오.
- **대상 정의 동기화**. 변경 사항을 지정해야 하는 경우 대상 XML 정의를 스키마에 동기화할 수 있습니다. 정의를 동기화하면, 스키마가 변경되는 경우 스키마를 가져오는 대신 XML 정의를 업데이트합니다.

XML 파일에서 XML 대상 정의 가져오기

XML 스키마, DTD 파일 또는 XML 파일에서 XML 정의를 가져올 수 있습니다. URL을 사용하여 참조하는 파일 또는 로컬 파일을 가져올 수 있습니다. 디자이너가 데이터의 정확한 정의를 제공할 수 있는지 확인하려면 XML 스키마에서 대상 정의를 가져오십시오.

다음 옵션에서 선택하여 XML 보기를 작성할 수 있습니다.

- **항목 관계 작성**. 다중 발생 또는 참조 요소 및 복합 유형에 대한 보기를 작성하려면 이 옵션을 사용하십시오. 하나의 큰 계층을 작성하는 대신 보기 간 관계를 작성합니다.
- **계층 관계 작성.** 루트를 작성하고 이 루트 아래에서 XML 구성 요소를 확장하려면 이 옵션을 사용하십시오. 정규화되거나 비정규화된 보기를 작성하도록 선택할 수 있습니다. 정규화된 보기를 선택하는 경우 모든 요소 또는 특성이 한 번 나타납니다. 일대다 관계는 보기와 관련된 키가 있는 개별 XML 보기로 됩니다. 비정규화된 XML 보기를 작성하는 경우 모든 요소 및 특성이 하나의 계층 그룹에 표시됩니다.
- **사용자 지정 XML 보기 작성.** XML 보기의 루트로 여러 글로벌 요소를 선택하고 메타데이터 확장 감소를 위한 옵션을 선택하려면 이 옵션을 사용하십시오.
- **XML 보기 작성 건너뛰기**. 보기를 작성하지 않고 정의를 가져오려면 이 옵션을 사용하십시오. 이 옵션을 선택하는 경우 XML 편집기를 사용하여 나중에 작업 공간에서 XML 보기를 작성하십시오.
- **XML 정의 동기화.** 기본 스키마가 변경될 때 하나 이상의 XML 정의를 업데이트하려면 이 옵션을 사용하십시오.

  팁: XML 파일 대신 DTD 또는 XML 스키마 파일을 가져오십시오. 연결된 DTD와 함께 XML 파일을 가져오는 경우 XML 마법사가 해당 DTD를 사용합니다.

XML 대상 정의를 가져오려면:

1. 대상 디자이너에서 대상 > Import XML Definition(XML 정의 가져오기)을 클릭합니다.

   Import XML Definitions(XML 정의 가져오기) 창이 열립니다. 기본적으로 로컬 폴더 스키마 파일이 나타납니다.
2. 로컬 파일 또는 URL을 클릭하여 XML 파일을 찾아봅니다.
3. DTD 또는 XML 파일을 찾아보려면 파일 유형 목록에서 해당 파일 확장명을 선택합니다.

XML 소스 정의에서 대상 작성

기존 소스 정의와 매우 유사한 대상 정의를 작성하려는 경우 소스 정의 또는 소스 정의에 대한 바로 가기를 사용하여 대상 정의를 작성합니다. XML 소스 정의를 대상 디저이너로 끌어 XML 대상 정의 또는 관계형 대상 정의를 작성하십시오.

다음 지침에 따라 XML 대상 정의를 작성하십시오.

- XML 소스 정의에서 XML 대상 정의를 작성할 때 XML 소스 정의의 복제본을 작성합니다.
- 올바른 XML 소스 정의가 반드시 올바른 XML 대상 정의를 작성하는 것은 아닙니다. 올바른 대상 정의를 작성하는지 확인하려면 대상 정의의 유효성을 검사하십시오.

참고: XML 대상 정의는 피벗된 열을 포함할 수 없습니다.

다음 지침에 따라 관계형 대상 정의를 작성하십시오.

- 관계형 대상 정의를 작성하면 디자이너가 XML 소스 정의의 그룹을 기반으로 관계형 대상 정의를 작성합니다. XML 소스 정의의 각 그룹이 대상 정의가 됩니다.
- 디자이너가 소스 정의의 그룹 사이처럼 대상 정의 사이에 동일한 관계를 작성합니다.

XML 소스 정의에서 대상 정의를 작성하려면:

1. 탐색기에서 대상 디자이너 작업 공간으로 XML 소스 정의를 끕니다.

   XML Export(XML 내보내기) 대화 상자가 나타납니다.
2. 관계형 또는 XML 대상을 선택하여 작성합니다. 확인을 클릭합니다.

   대상 정의가 대상 디자이너 작업 공간에 나타납니다. 관계형 대상을 선택하면 소스에 따라 두 개 이상의 대상 정의가 작업 공간에 나타날 수 있습니다.

XML 대상 정의 속성 편집

XML 대상 정의를 작성한 후, 속성을 편집하여 대상 데이터의 변경 사항을 적용하거나, 비즈니스 이름과 설명을 추가하거나, 코드 페이지를 변경할 수 있습니다.

XML 대상 정의 속성을 편집하려면:

1. 대상 디자이너에서 XML 대상을 엽니다.
2. 마우스 오른쪽 단추를 클릭하고 편집을 선택합니다.
3. 테이블 탭에서 설정을 편집합니다.

   다음 테이블에는 테이블 설정이 설명되어 있습니다.

테이블 설정	설명
테이블 선택	대상 정의의 이름입니다. 이름을 변경하려면 이름 바꾸기 단추를 클릭하십시오.
비즈니스 이름	대상 테이블의 설명 이름입니다. 이름 바꾸기 단추를 사용하여 비즈니스 이름을 편집하십시오.
제약 조건	XML 대상에는 적용되지 않습니다. 모든 항목이 무시됩니다.
작성 옵션	XML 대상에는 적용되지 않습니다. 모든 항목이 무시됩니다.
설명	대상 테이블의 설명입니다. 문자 제한은 2,000바이트/K이며 여기서 K는 리포지토리 코드 페이지의 각 문자에 대한 최대 바이트 수입니다. 비즈니스 문서에 대한 링크를 입력합니다.

테이블 설정	설명
코드 페이지	대상 정의에서 사용할 코드 페이지를 선택하십시오.
데이터베이스 유형	대상 정의가 XML 대상임을 나타냅니다.
키워드	키워드를 사용하여 대상을 구성하고 식별하십시오. 키워드에는 개발자 이름, 매핑 또는 XML 스키마 이름이 포함될 수 있습니다. 키워드를 사용하여 Repository Manager에서 검색을 수행하십시오.

4. 열 탭에서 XML 열 정의를 볼 수 있습니다.

   다음 테이블에는 열 설정이 설명되어 있습니다.

열 설정	설명
테이블 선택	편집하고 있는 대상 정의를 표시합니다. 편집할 다른 정의를 선택하려면 작업 공간에 있는 사용 가능한 정의 목록에서 정의를 선택하십시오.
열 이름	열의 이름입니다.
데이터 유형	열에 대한 PowerCenter 데이터 유형입니다.
전체 자릿수	열의 크기입니다. 문자열 등 일부 데이터 유형에 대한 전체 자릿수를 변경할 수 있습니다.
배율	숫자 값의 소수점 뒤 최대 자릿수입니다.
Null이 아님	열에 null 값이 있을 수 있는지 여부를 나타냅니다.
키 유형	XML 마법사가 보기를 연결하기 위해 생성하는 키 유형입니다.
XPath	항목을 찾는 XML 파일 계층을 지나는 경로입니다.

5. 속성 탭에서 대상 정의의 변환 특성을 수정할 수 있습니다.

   소스 기반 커밋 세션 또는 트랜잭션 제어 변환을 XML 대상과 함께 사용하는 경우 데이터를 대상에 플러시하는 방법을 정의할 수 있습니다.

   다음 테이블에는 편집할 수 있는 특성이 설명되어 있습니다.

열 설정	설명
테이블 선택	편집하는 소스 정의를 표시합니다. 편집할 다른 소스 정의를 선택하려면 목록에서 소스 정의를 선택하십시오.
중복 그룹 행 처리	다음 옵션 중 하나를 선택하여 대상의 중복 행 처리를 수행하십시오. - 첫 번째 행. 통합 서비스가 첫 번째 중복 행을 대상에 전달합니다. 동일한 기본 키가 뒤어어 나오는 행은 거부됩니다. - 마지막 행. 통합 서비스가 마지막 중복 행을 대상에 전달합니다. - 오류. 통합 서비스가 첫 번째 행을 대상에 전달합니다. 중복 기본 키가 있는 행은 오류 수를 더합니다. 오류 수가 오류 임계값에 도달하는 경우 세션이 실패합니다.

열 설정	설명
DTD 참조	대상 XML 파일에 대한 DTD 또는 XML 스키마 파일 이름입니다. XML 파일을 작성하면 통합 서비스가 문서 유형 선언을 XML 파일에 추가합니다.
커밋 시	통합 서비스는 여러 XML 파일을 생성하거나 커밋 후 하나의 XML 파일에 추가할 수 있습니다. 다음 옵션 중 하나를 사용하십시오. - 커밋 무시. 통합 서비스는 XML 파일을 작성하고 파일 끝에서 이 XML 파일에 씁니다. - 새 문서 작성. 각 커밋에서 새 XML 파일을 작성합니다. - 문서에 추가. 각 커밋 후 동일한 XML 파일에 씁니다.
캐시 디렉터리	XML 대상 캐시 파일의 디렉터리입니다. 기본값은 $PMCacheDir 서비스 프로세스 변수입니다.
캐시 크기	XML 대상 캐시의 전체 크기(바이트)입니다. 기본값은 자동입니다.

6. 메타데이터 확장 탭에서 재사용 불가능 메타데이터 확장을 작성, 수정, 삭제 및 승격하고 해당 값을 업데이트할 수 있습니다. 또한 재사용 가능 메타데이터 확장 값을 업데이트할 수 있습니다.
7. 확인을 클릭합니다.
8. 리포지토리 > 저장을 클릭합니다.

XML 대상 유효성 검사

XML 파일로 데이터를 추출하는 방법에 대해 설명하는 사용자 지정된 XML 보기를 작성할 수 있습니다. 하지만 보기 간에 일부 보기 구조 또는 관계는 XML 정의에서 유효하지 않습니다. 일부 보기 구조는 XML 소스에 대해 유효하지만, XML 대상에 대해서는 유효하지 않을 수 있습니다. 디자이너는 사용자가 모호한 정의를 작성하지 못하게 합니다.

다음 태스크를 수행하면 PowerCenter가 대상 XML 보기의 유효성을 검사합니다.

- 리포지토리에서 XML 대상을 저장하거나 가져오면 디자이너가 제한된 유효성 검사를 수행합니다.
- XML 작업 공간에서 XML을 편집하면 XML 편집기가 각 단계의 유효성을 검사합니다.
- XML 편집기에서 편집하고 있는 대상 정의의 유효성을 검사하도록 선택할 수 있습니다.
- 디자이너가 매핑의 유효성을 검사할 때 XML 대상 연결의 유효성을 검사합니다.

디자이너는 규칙을 사용하여 계층 관계, 유형 관계 및 상속 관계의 유효성을 검사합니다.

참고: 통합 서비스는 세션의 스키마에 대해 대상 XML 인스턴스의 유효성을 검사하지 않습니다. Validate Target(대상 유효성 검사) 세션 속성을 설정하여 데이터의 단순 유형에 대한 유효성을 검사할 수 있습니다.

계층 관계 유효성 검사

디자이너는 다음 규칙을 사용하여 계층 관계의 유효성을 검사합니다.

- 유형에 루트가 있는 보기는 독립 실행형 보기일 수 없습니다. 보기가 상속 관계에서 하위여야 하거나 보기에 다른 보기와의 유형 관계가 있어야 합니다. XML 대상에 요소에 설정된 보기가 없는 경우 이 XML 대상은 유효하지 않습니다.
- 다중 발생 보기 행이 있는 보기를 다른 보기에 연결해야 합니다.

- 두 보기에는 동일한 유효 보기 행이 있을 수 없습니다.
- XML 대상에 요소의 보기 루트가 없는 경우 이 XML 대상은 유효하지 않습니다.
- 상위 및 하위 보기를 다른 요소로 구분할 수 있지만, 보기에 대한 두 가지 상위 중에서 선택할 수 있는 경우 가장 가까운 상위를 사용해야 합니다. 유효 보기 행의 경로로 가장 가까운 상위를 결정하십시오. 하나의 상위는 경로에서 다른 상위 앞에 나옵니다. 경로에서 두 번째로 나오는 보기를 선택하십시오.
- 모든 보기를 동일한 계층의 동일한 보기 루트와 연결해야 합니다. 정의는 동일한 보기 루트에 대한 여러 트리를 포함할 수 없습니다.
- 보기 행과 보기 루트가 XML 보기에 대해 동일한 경우 해당 XML 보기에 자신에 대한 계층 관계가 있을 수 있습니다.

유형 관계 유효성 검사

유형 관계는 열과 보기 간 관계입니다. 유형 관계는 두 보기 간 관계가 아닙니다. 다음과 같은 규칙이 유형 관계에 적용됩니다.

- 보기 루트가 동일한 유형이거나 V2 보기 루트 유형이 V1 보기 루트에서 파생된 경우, 보기 V1의 열에는 보기 V2에 대한 유형 관계가 있을 수 있습니다. 두 보기 루트는 글로벌 복합 유형이어야 합니다.
- 보기의 열에 다른 보기에 대한 유형 관계가 있는 경우 열을 확장할 수 없습니다.

상속 유효성 검사

XML 보기를 사용하여 두 가지 유형의 상속 관계를 작성할 수 있습니다.

- **보기-보기 상속.** 보기는 다른 보기의 파생된 유형입니다. 두 보기 모두에 글로벌 복합 보기 루트가 있어야 합니다.

  보기 루트가 다른 보기의 보기 루트 유형에서 파생된 복합 유형인 경우 보기에 다른 보기에 대한 상속 관계가 있을 수 있습니다.

  보기는 여러 상속 관계에서 상위일 수 있지만 하나의 상속 관계에서만 하위일 수 있습니다.
- **열-보기 상속.** 열은 로컬 복합 유형 Type1의 요소이며 보기는 글로벌 복합 유형 Type2에서 설정되어 있습니다. Type1은 Type2에서 파생됩니다.

  열이 로컬 복합 유형이고 유형이 다른 보기의 보기 루트 유형에서 파생된 경우 보기의 열에 다른 보기에 대한 상속 관계가 있을 수 있습니다.

  보기 V1의 열에 보기 V2에 대한 상속 관계가 있는 경우 V2의 콘텐츠를 보기 V1에 배치할 수 없습니다.

매핑에서 XML 대상 사용

XML 대상을 매핑에 추가할 때 다중 그룹 변환에 대한 매핑 지침을 따라야 합니다.

다음 구성 요소는 매핑에서 XML 대상을 매핑하는 방법에 영향을 미칩니다.

- 활성 소스
- 루트 요소
- 대상 포트 연결
- 추상 요소
- 트랜잭션 제어점

- FileName 열

활성 소스

활성 소스는 각 입력 행에 대해 다른 행 수를 반환할 수 있는 변환입니다. 통합 서비스는 여러 활성 소스에서 XML 대상으로 데이터를 로드할 수 있습니다. 하지만 XML 대상의 단일 그룹 내 모든 포트가 동일한 활성 소스에서 데이터를 받아야 합니다.

다음 변환은 활성 소스입니다.

- 집계
- 응용 프로그램 소스 한정자
- 사용자 지정, 활성 변환으로 구성됨
- Java, 활성 변환으로 구성됨
- 조이너
- MQ 소스 한정자
- 노멀라이저(VSAM 또는 파이프라인)
- 순위
- 분류기
- 소스 한정자
- SQL
- XML 소스 한정자
- 맵렛, 맵렛에 위 변환 중 하나가 포함된 경우

루트 요소 선택

XML 정의에 두 개 이상의 가능한 루트가 있는 경우 대상 인스턴스에 대한 루트 요소를 지정할 수 있습니다.

루트 요소를 지정하려면:

1. 매핑 디자이너에서 대상 정의를 마우스 오른쪽 단추로 클릭하고 편집을 선택합니다.
2. 속성 탭을 클릭합니다.
3. 루트 요소 값 열에서 화살표를 클릭합니다.

   Select Root(루트 선택) 대화 상자가 나타납니다.
4. 목록에서 요소를 선택합니다.

대상 포트 연결

통합 서비스가 세션 중에 올바른 XML 파일을 작성할 수 있도록 매핑에서 XML 대상 포트를 제대로 연결해야 합니다. XML 대상이 포함된 매핑을 저장하거나 유효성 검사를 하면 디자이너가 대상 포트 연결의 유효성을 검사합니다.

매핑에서 포트를 연결할 때 다음 지침을 따르십시오.

- 그룹에서 하나의 포트를 연결하는 경우 이 그룹에 대한 외래 키와 기본 키 포트를 둘 다 연결해야 합니다.

- 그룹에서 외래 키 포트를 연결하는 경우 다른 그룹의 연결된 기본 키 포트를 연결해야 합니다. 루트 그룹의 기본 키 포트를 연결하지 않는 경우 다른 그룹의 연결된 외래 키 포트를 연결할 필요가 없습니다.
- 기본 특성 값으로 XML 스키마를 사용하는 경우 특성 포트를 연결하여 대상에서 기본 특성을 작성해야 합니다. 연결된 포트를 통해 null 값을 전달하면 통합 서비스가 기본값을 대상에 씁니다.

추상 요소 연결

추상 요소는 XML 인스턴스 파일에서 직접 발생할 수 없습니다. 대신, 추상 요소에서 파생된 요소를 사용해야 합니다. 기본적으로 디자이너는 추상 복합 요소에 대한 보기를 작성합니다. 메타데이터를 줄이기 위해 추상 유형의 요소는 파생된 유형에서 반복되지 않습니다. 추상 유형에 데이터를 매핑하는 경우 하나 이상의 파생된 유형에도 데이터를 매핑해야 합니다.

세션 중에 통합 서비스가 데이터를 추상 유형에 로드하는 경우 통합 서비스에는 추상 유형과 연결된 추상이 아닌 파생된 유형에 대한 데이터가 있어야 합니다. 파생된 유형에 데이터가 없는 경우 통합 서비스는 대상 XML 파일에서 추상 요소를 쓰지 않습니다.

대상에 XML 데이터 플러시

세션에서 각 커밋 지점의 XML 대상에 데이터를 플러시할 수 있습니다. 하지만 각 입력 그룹은 매핑의 동일한 트랜잭션 제어점에서 데이터를 받아야 합니다. 이 매핑에 기반한 세션을 작성하는 경우 각 커밋에서 XML 파일 대상에 데이터를 추가하거나 각 커밋에서 새 파일을 작성할 수 있습니다. 커밋 시 작업 대상 속성으로 옵션을 지정할 수 있습니다.

XML 대상 입력 그룹을 여러 트랜잭션 제어점에 연결하면 통합 서비스가 모든 소스 행을 처리한 후 XML 파일 대상에 데이터를 씁니다.

동적으로 XML 파일 이름 지정

FileName 열을 XML 대상 정의에 추가하여 동적으로 XML 파일에 대한 파일 이름을 작성할 수 있습니다. 통합 서비스가 FileName 열에 데이터를 전달할 때 대상 속성의 출력 파일 이름을 재정의합니다. 예를 들어 문자열 “Harry”를 FileName 열에 전달하면 통합 서비스가 XML 파일에 Harry라고 이름을 지정합니다.

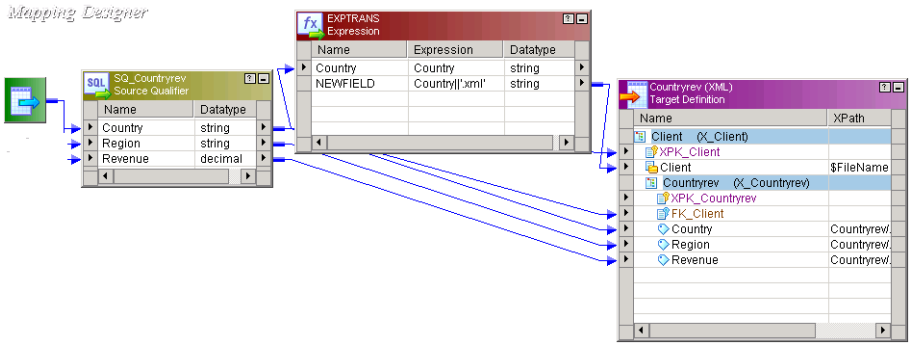
참고: 각 커밋에서 새 XML 파일을 작성할 경우 작성하는 각 XML 파일의 이름을 동적으로 지정해야 합니다. 동적으로 각 XML 파일 이름을 지정하지 않는 경우 통합 서비스가 이전 커밋의 XML 파일을 덮어씁니다.

통합 서비스는 대상의 루트 그룹에 있는 각 고유 기본 키 값에 대한 새 XML 파일을 생성합니다. 각 파일에 대해 다른 이름을 설정하려면 FileName 열을 추가합니다. 각 이름은 세션 속성에서 출력 파일 이름을 재정의합니다.

예제

식 변환은 Country XML 요소에서 파일 이름을 생성하고 해당 값을 FileName 열에 전달합니다. 매핑이 Client라는 대상 루트에 국가를 전달합니다. Client 값이 변경될 때마다 통합 서비스가 새 XML 파일을 작성합니다. 통합 서비스는 각 XML 대상 파일 이름이 포함된 목록 파일을 작성합니다. 통합 서비스에는 해당 목록의 각 파일에 대한 절대 경로가 나열됩니다.

다음 그림에는 FileName 열이 있는 XML 대상이 포함된 매핑이 나와 있습니다.



통합 서비스가 다음 행을 대상에 전달합니다.

```
Country,Region,Revenue
USA,region1,1000
France,region1,10
Canada,region1,100
USA,region2,200
USA,region3,300
USA,region4,400
France,region2,20
France,region3,30
France,region4,40
```

세션에서 국가 이름별로 다음 파일을 생성합니다.

```
Canada.xml
France.xml
USA.xml
```

목록 파일 이름은 세션 속성의 출력 파일 이름입니다.

```
revenue_file.xml.lst
```

XML 대상 문제 해결

XML 파일에서 소스 정의를 가져왔습니다. 그런 다음 동일한 XML 파일에서 대상 정의를 가져왔습니다. 소스 및 대상 정의에 대한 기본 그룹이 동일하지 않습니다.

대상을 가져올 때 옵션 중 일부를 변경하는 경우 XML 마법사가 항상 소스 정의와 대상 정의에 대한 동일한 그룹 구조를 작성하지는 않습니다.

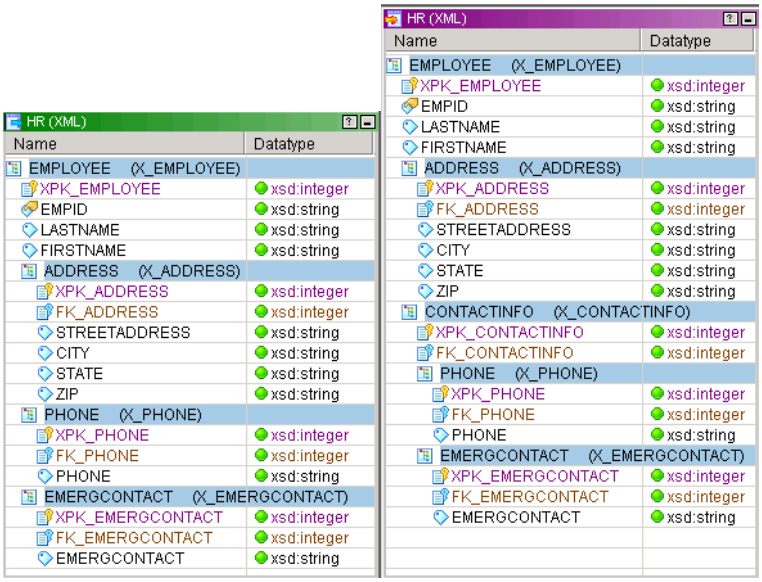
예를 들어 다음 DTD의 ContactInfo 요소는 삽입 요소입니다. 삽입 요소에는 텍스트 콘텐츠가 없지만 maxOccurs가 1보다 큽니다. 하위 요소에서도 maxOccurs가 1보다 큽니다.

```
<!ELEMENT HR (EMPLOYEE+)>
<!ELEMENT EMPLOYEE (LASTNAME,FIRSTNAME,ADDRESS+,CONTACTINFO+)>
<!ATTLIST EMPLOYEE EMPID CDATA #REQUIRED>
<!ELEMENT LASTNAME (#PCDATA)>
<!ELEMENT FIRSTNAME (#PCDATA)>
<!ELEMENT ADDRESS (STREETADDRESS,CITY,STATE,ZIP)>
<!ELEMENT STREETADDRESS (#PCDATA)>
<!ELEMENT CITY (#PCDATA)>
<!ELEMENT STATE (#PCDATA)>
<!ELEMENT ZIP (#PCDATA)>
```

```
<!ELEMENT CONTACTINFO (PHONE+,EMERGCONTACT+)>
<!ELEMENT PHONE (#PCDATA)>
<!ELEMENT EMERGCONTACT (#PCDATA)>
```

소스 정의에서 삽입 요소에 대한 XML 보기를 작성하지 않으면 소스에서 ContactInfo 요소가 작성되지 않습니다.

다음 그림에는 XML 마법사가 작성하는 소스 및 대상 정의가 나와 있습니다.



소스 정의에는 ContactInfo 요소가 포함되어 있지 않습니다. 대상 정의에는 ContactInfo 요소가 포함되어 있습니다. 소스를 작성할 때 삽입 요소에 대한 보기를 작성하지 않도록 선택했기 때문에 마법사가 소스 정의에 ContactInfo 요소를 포함하지 않습니다. 하지만 마법사는 대상 정의에 ContactInfo 요소를 포함합니다.

내 관계형 소스에서 작성한 XML 대상 정의에 모든 요소가 포함되어 있지만 특성은 없습니다. 특정 데이터를 특성으로 표시할 수 있도록 대상 계층을 어떻게 수정할 수 있습니까?

마법사가 관계형 테이블에서 작성하는 구성 요소 유형은 수정할 수 없습니다. 하지만 대상 XML 계층의 XML 스키마 파일 또는 DTD를 볼 수 있습니다. 새 파일 이름으로 DTD 또는 XML 스키마 파일을 저장하십시오. 이 새 파일을 열고 계층을 수정하여 특성 및 요소를 설정하십시오. 그런 다음 이 파일을 사용하여 새 계층과 함께 대상 정의를 가져오십시오.

XML 소스 한정자 변환

XML 소스 한정자 변환 개요

XML 소스 정의를 매핑에 추가할 때 소스 정의를 XML 소스 한정자 변환에 연결해야 합니다. XML 소스 한정자 변환은 통합 서비스가 세션 중에 읽는 데이터 요소를 정의합니다. 소스 한정자에 따라 PowerCenter에서 소스 데이터를 읽는 방법이 결정됩니다. XML 소스 한정자 변환은 활성 변환입니다.

수동으로 소스 한정자 변환을 추가하거나 소스 정의를 매핑에 추가할 때 기본적으로 소스 한정자 변환을 작성할 수 있습니다.

일부 속성을 편집하고 메타데이터 확장을 XML 소스 한정자 변환에 추가할 수 있습니다.

매핑에서 XML 소스 한정자 변환을 연결하는 경우 유효한 매핑을 작성하기 위해 규칙을 따라야 합니다.

매핑에 XML 소스 한정자 추가

XML 소스 한정자 변환은 XML 소스의 모든 열에 대해 하나의 입력/출력 포트가 있습니다. 소스 정의를 위해 XML 소스 한정자 변환을 작성하는 경우 디자이너가 XML 소스 정의의 각 포트를 XML 소스 한정자 변환의 포트에 연결합니다. 링크를 제거하거나 편집할 수 없습니다. 매핑에서 XML 소스 정의를 제거하는 경우 디자이너는 해당 XML 소스 한정자 변환도 제거합니다. 한 개의 XML 소스 정의를 한 개의 XML 소스 한정자 변환에 연결할 수 있습니다.

한 개의 XML 소스 한정자 그룹에 있는 포트를 다른 변환의 포트에 연결하여 별도의 데이터 흐름을 형성할 수 있습니다. 그러나 XML 소스 한정자 변환에서 둘 이상의 그룹에 있는 포트를 동일한 대상 변환의 포트에 연결할 수 없습니다.

둘 이상의 그룹에 있는 열을 하나의 변환으로 끄는 경우 디자이너가 모든 그룹의 열을 변환에 복사합니다. 그러나 디자이너는 처음 그룹의 포트만 변환에 있는 새 열의 해당 포트에 연결합니다.

XML 소스 정의를 매핑 디자이너 작업 공간으로 끌거나 수동으로 소스 한정자를 작성하여 XML 소스 한정자 변환을 매핑에 추가할 수 있습니다.

기본적으로 XML 소스 한정자 변환 작성

XML 소스 정의를 매핑 디자이너 작업 공간으로 끄는 경우 디자이너가 기본적으로 XML 소스 한정자 변환을 작성합니다.

기본적으로 XML 소스 한정자 변환을 작성하려면

1. 매핑 디자이너에서 새 매핑을 작성하거나 기존 매핑을 엽니다.
2. XML 소스 정의를 매핑으로 끕니다.

   디자이너가 XML 소스 한정자 변환을 작성하고 XML 소스 정의의 각 포트를 XML 소스 한정자 변환의 포트에 연결합니다.

수동으로 XML 소스 한정자 변환 작성

소스 한정자가 없는 XML 소스 정의가 포함된 매핑이 있거나 매핑에서 XML 소스 한정자 변환을 삭제하는 경우 매핑에 XML 소스 한정자 변환을 작성할 수 있습니다.

수동으로 XML 소스 한정자 변환을 작성하려면

1. 매핑 디자이너에서 새 매핑을 작성하거나 기존 매핑을 엽니다.

   참고: 매핑에 소스 한정자가 없는 XML 소스 정의가 한 개 이상 있어야 합니다.
2. 변환 > 작성을 클릭합니다.

   변환 작성 대화 상자가 나타납니다.
3. XML 소스 한정자 변환을 선택한 다음 변환에 대한 이름을 입력합니다.

   XML 소스 한정자 변환에 대한 명명 규칙은 XSQ_*TransformationName*입니다.
4. 작성을 클릭합니다.

   디자이너가 해당 XML 소스 한정자 변환이 없는 매핑의 모든 XML 소스 정의를 나열합니다.

5. 소스 정의를 선택하고 확인을 클릭합니다.

   디자이너가 매핑에 XML 소스 한정자 변환을 작성하고 XML 소스 정의의 각 포트를 XML 소스 한정자 변환의 포트에 연결합니다.

XML 소스 한정자 변환 편집

변환 이름 및 설명과 같은 XML 소스 한정자 변환 속성을 편집할 수 있습니다.

XML 소스 한정자 변환을 편집하려면

1. 매핑 디자이너에서 XML 소스 한정자 변환을 엽니다.
2. 변환 탭에서 속성을 편집합니다.

   다음 테이블에는 변환 속성이 설명되어 있습니다.

변환 설정	설명
선택 변환	편집하는 변환을 표시합니다. 다른 변환을 선택하여 편집하려면 목록에서 변환을 선택합니다.
이름 바꾸기	변환의 이름을 편집합니다.
설명	변환을 설명합니다.

3. 포트 탭을 클릭하여 XML 소스 한정자 변환 포트를 봅니다.

   시퀀스 열을 사용하여 XML 그룹의 생성된 키에 대한 시작 값을 설정합니다. 생성된 각 키에 다른 값을 입력할 수 있습니다. 시퀀스 키는 bigint 데이터 유형입니다. 이 값을 변경할 때마다 시퀀스 번호가 다음에 세션을 실행할 때 다시 시작합니다.
4. 속성 탭을 클릭하여 통합 서비스가 세션 중에 매핑을 실행하는 방법에 영향을 미치는 속성을 구성합니다.

   다음 테이블에는 XML 소스 한정자 속성이 설명되어 있습니다.

속성 설정	설명
선택 변환	편집하는 변환을 표시합니다. 다른 변환을 선택하여 편집하려면 목록에서 변환을 선택합니다.
추적 수준	워크플로우를 실행할 때 통합 서비스가 세션 로그에 기록하는 이 변환에 대한 정보의 양을 결정합니다. 세션을 구성할 때 이 추적 수준을 재정의할 수 있습니다.
재설정	세션이 종료될 때 통합 서비스가 시작 값을 현재 세션의 시작 값으로 재설정합니다.
다시 시작	세션이 시작될 때 통합 서비스가 모든 그룹에 대해 생성된 키 시퀀스를 1부터 시작합니다.

5. 메타데이터 확장 탭을 클릭하여 사용자 정의 메타데이터 확장을 작성, 편집 및 삭제합니다.

   재사용 불가능 메타데이터 확장을 작성, 수정, 삭제 및 승격하고 해당 값을 업데이트할 수 있습니다. 또한 재사용 가능 메타데이터 확장 값을 업데이트할 수 있습니다.

6. 확인을 클릭합니다.

생성된 키에 대해 시퀀스 번호 설정

XML 소스 한정자 정의의 각 보기에는 기본 키 및 키에 대한 시퀀스 값이 있습니다. 세션 중에 통합 서비스가 시퀀스 값에서 키를 생성하고 값을 증분합니다.

세션이 종료될 때 통합 서비스가 리포지토리의 각 시퀀스 값을 현재 값+1로 업데이트합니다. 이 값이 다음에 통합 서비스가 시퀀스 생성기 변환을 처리할 때 시작 값이 됩니다.

리포지토리가 다음 시퀀스 값을 유지 관리합니다.

- **기본값.** 처음 소스 한정자를 작성할 때 XML 소스 한정자에 나타나는 키의 시퀀스 값입니다. 각 키의 기본값은 1입니다.
- **시작 값.** 세션을 시작할 때 키의 시퀀스 번호 값입니다. 워크플로우를 실행하기 전에 XML 소스 한정자 변환에서 시작 값을 볼 수 있습니다.
- **현재 값.** 세션 중 키에 대한 시퀀스 값입니다.

생성된 키에 대한 시작 값이 XML 소스 한정자의 시퀀스 열에 표시됩니다.

참고: 포트 탭에서 시퀀스 시작 값을 편집하는 경우 변경 사항을 저장하고 디자이너를 종료한 다음 워크플로우를 실행해야 합니다.

시퀀스 시작 값 변경

XML 소스 한정자 변환 속성 탭에서 다음 옵션을 사용하여 세션 전이나 후에 시퀀스 시작 값을 변경할 수 있습니다.

- **재설정.** 세션을 종료할 때 통합 서비스가 시작 값을 현재 세션의 시작 값으로 다시 재설정합니다. 예를 들어 세션을 시작할 때 키의 시작 값이 2000입니다. 세션을 종료할 때 현재 값이 2500입니다. 세션을 완료하면 리포지토리의 시작 값은 2000으로 유지됩니다. 테스트 중이어서 다음에 세션을 실행할 때 동일한 키 번호를 생성하려는 경우 이 옵션을 사용할 수 있습니다.
- **다시 시작.** 세션을 시작할 때 통합 서비스가 기본값을 사용하여 시작 값을 다시 시작합니다. 예를 들어 키의 시작 값이 1005인 경우 다시 시작을 선택하면 통합 서비스가 시작 값을 1로 변경합니다. 키 값이 커지고 번호 매기기를 다시 시작해도 중복된 키 충돌이 없는 경우 이 옵션을 사용할 수 있습니다.

매핑에서 XML 소스 한정자 사용

XML 소스 정의의 각 그룹은 관계형 테이블과 비슷해서 디자이너가 XML 소스 한정자 변환 내의 각 그룹을 별도의 데이터 소스로 처리합니다.

매핑에서 개체를 연결할 때 디자이너가 연결 규칙을 적용합니다. 그러므로 하나의 파이프라인 분기에서, 요구하는 모든 정보가 각 그룹에 포함될 수 있도록 XML 소스 정의의 그룹을 구성해야 합니다.

매핑에서 XML 소스 한정자 변환을 연결하는 경우 다음 규칙을 고려합니다.

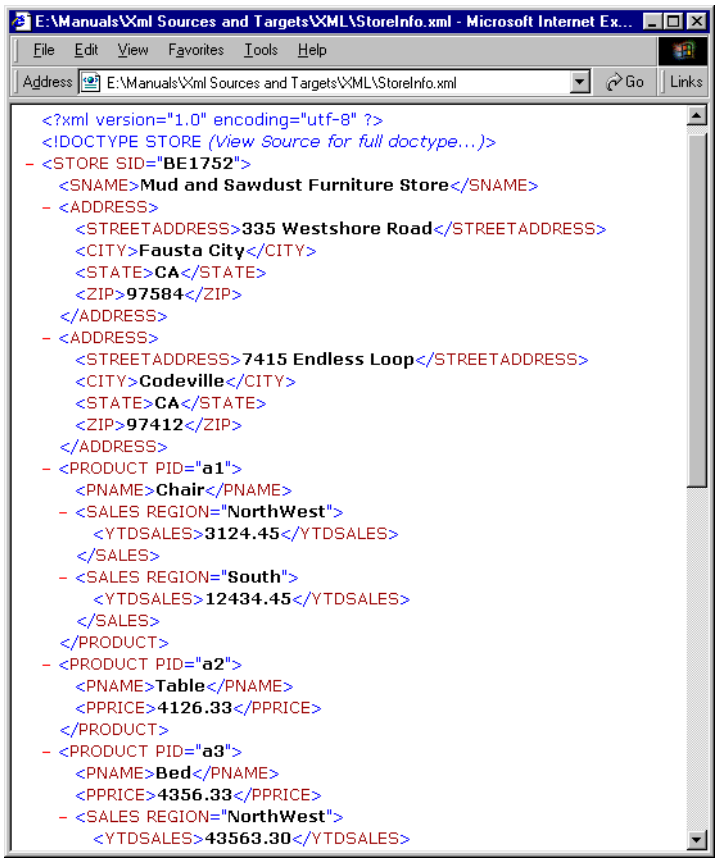
- **XML 소스 한정자 변환의 한 그룹에 있는 포트를 다른 변환의 한 입력 그룹에 있는 포트에 연결할 수 있습니다.** 여러 그룹의 열을 한 변환에 *복사*할 수 있지만 한 그룹의 포트만 변환의 해당 포트에 *연결*할 수 있습니다.

- **XML 소스 한정자 변환에서 한 그룹의 포트를 둘 이상 변환에 있는 포트에 연결할 수 있습니다.** XML 소스 한정자 변환의 각 그룹은 둘 이상 파이프라인 분기에 대한 데이터 소스가 될 수 있습니다. 한 그룹에서 여러 다른 변환으로 데이터를 전달할 수 있습니다.
- **한 XML 소스 한정자 변환의 여러 그룹을 한 변환의 여러 입력 그룹에 연결할 수 있습니다.** 조이너 또는 사용자 지정 변환 같이 한 XML 소스 한정자 변환의 여러 그룹을 대부분의 여러 입력 그룹 변환의 서로 다른 입력 그룹에 연결할 수 있습니다. 그러나 조이너가 입력을 정렬한 경우 한 XML 소스 한정자 변환의 여러 그룹을 한 조이너 변환에 연결할 수 있습니다. 두 XML 소스 한정자 변환 그룹을 정렬되지 않은 입력을 가진 조이너 변환에 연결하려면 동일한 XML 소스의 두 인스턴스를 작성해야 합니다.

XML 소스 한정자 변환 예

이 섹션은 매핑에서 XML 소스 한정자 변환 예를 보여 줍니다. 예는 상점, 제품 및 판매 정보가 포함된 XML 파일을 사용합니다.

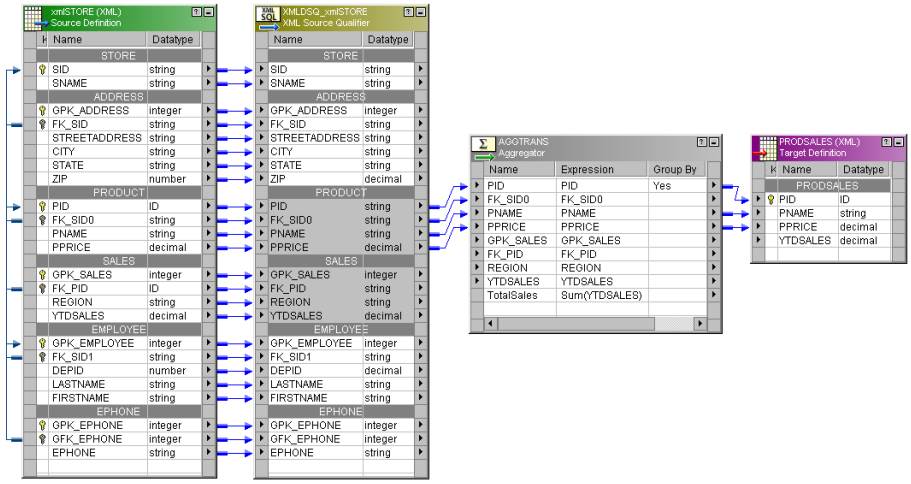
다음 그림은 StoreInfo.xml 파일을 보여 줍니다.



지역에 관계 없이 XML 파일에서 각 제품에 대한 총 YTD 판매 계산을 원할 수 있습니다. 판매 외에도 각 제품의 이름 및 가격을 원할 수 있습니다.

이렇게 하려면 제품 및 판매 정보가 동일한 변환에서 모두 필요합니다. 그러나 StoreInfo.xml 파일을 가져올 때 디자이너가 기본적으로 제품 및 판매에 대해 별도의 그룹을 작성합니다.

다음 그림은 별도의 그룹에 제품 및 판매 정보가 들어 있는 StoreInfo 파일의 기본 그룹을 보여 줍니다.



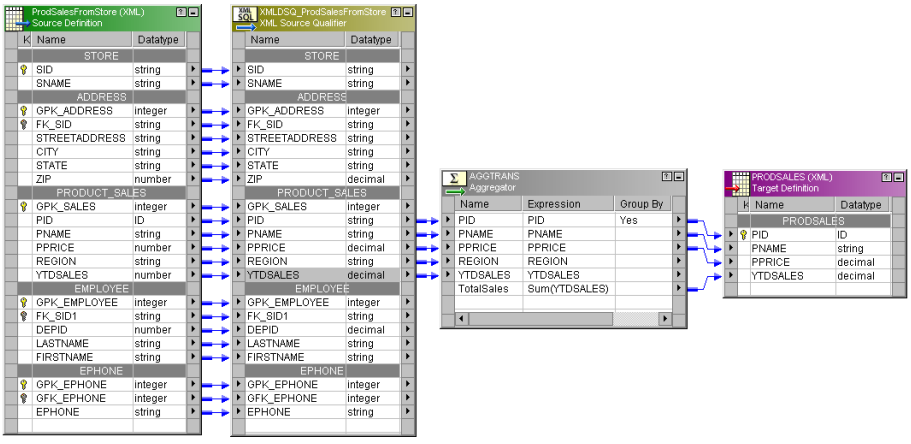
Product 및 Sales 그룹 모두를 동일한 단일 입력 그룹 변환에 연결할 수 없으므로 다음 중 한 가지 방법으로 매핑을 작성할 수 있습니다.

- 필요한 모든 정보가 포함된 비정규화된 그룹을 사용합니다.
- 조이너 변환을 사용하여 두 그룹의 데이터를 조인합니다.

한 개의 비정규화된 그룹 사용

모든 정보를 동일한 그룹에서 가져오도록 소스 정의에서 그룹을 구성할 수 있습니다. 예를 들어 Product 및 Sales 그룹을 소스 정의에서 하나의 비정규화된 그룹으로 결합할 수 있습니다. 한 데이터 흐름을 통해 비정규화된 그룹의 판매 집계에 대해 모든 정보를 처리할 수 있습니다.

다음 그림은 Product 및 Sales 그룹에 모두 속한 열의 조합이 들어 있는 비정규화된 그룹 Product_Sales를 보여 줍니다.



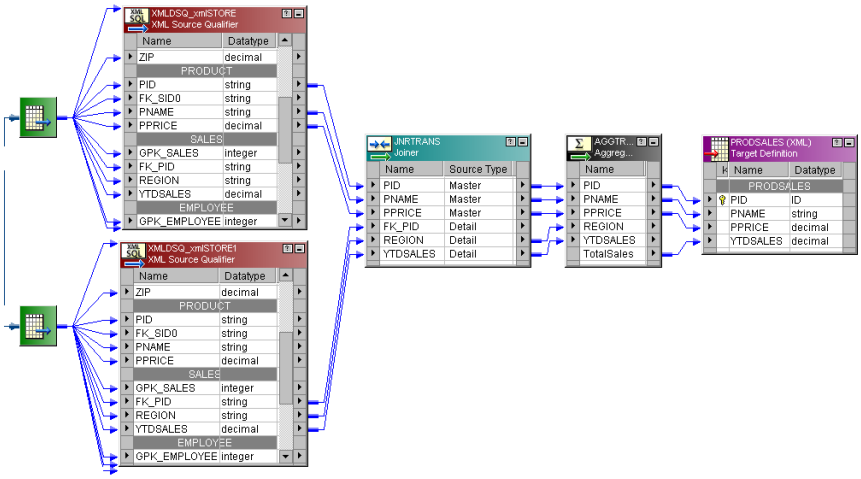
비정규화된 그룹을 작성하려면 소스 분석기에서 소스 정의를 편집합니다. 새 그룹을 작성하거나 기존 그룹을 수정할 수 있습니다. 집계 변환에서 판매 계산을 수행하려면 product 및 sales 열을 그룹에 추가합니다. XML 편집기를 사용하여 그룹을 작성하고 그룹의 유효성을 검사합니다.

두 개의 XML 소스 한정자 변환 그룹 조인

두 소스 그룹의 데이터를 한 데이터 흐름으로 조인할 수 있습니다. 조이너 변환을 사용하여 그룹의 데이터를 조인합니다. 정렬된 입력에 대해 조이너 변환을 구성하는 경우 한 XML 소스 한정자 변환 인스턴스의 두 그룹을 조이너 변환에 연결할 수 있습니다. 정렬되지 않은 입력에 대해 구성된 조이너 변환을 사용하는 경우 동일한 XML 소스의 두 인스턴스를 사용하고 각 XML 소스 한정자 변환 인스턴스의 그룹을 조이너 변환에 연결해야 합니다.

그런 다음 조이너 변환의 데이터를 집계 변환으로 보내 각 제품에 대해 YTDSales를 계산할 수 있습니다.

다음 그림은 동일한 XML 소스의 두 인스턴스를 작성하고 두 XML 소스 한정자 변환의 데이터를 조인하는 방법을 보여 줍니다.



XML 소스 한정자 변환 문제 해결

한 XML 소스 한정자 변환의 두 그룹을 한 변환으로 끄는 경우 디자이너가 열을 복사하지만 모든 포트를 연결하지는 않습니다.

XML 소스 한정자 변환의 한 그룹을 한 변환에 연결할 수 있습니다. 둘 이상의 그룹을 변환으로 끄는 경우 디자이너가 모든 열 이름을 변환에 복사합니다. 그러나 디자이너가 처음 그룹의 열만 연결합니다.

XML 소스 정의와 해당 소스 한정자 사이의 링크를 해제할 수 없습니다.

XML 소스 한정자 변환 열은 해당 XML 소스 정의 열과 일치합니다. XML 소스 정의와 해당 XML 소스 한정자 변환 사이의 링크를 제거하거나 수정할 수 없습니다. XML 소스 정의를 제거하는 경우 디자이너가 해당 XML 소스 한정자 변환을 제거합니다.

미드스트림 XML 변환

미드스트림 XML 변환 개요

XML 정의에서 XML 데이터를 읽거나 작성합니다. 하지만 파이프라인 내에서 XML을 추출하거나 생성해야 하는 경우가 있습니다. 예를 들어 데이터 필드로 XML 문서가 포함된 TIBCO 대상에 메시지를 보내려고 할

수 있습니다. 이러한 경우 TIBCO에 메시지를 보내기 전에 XML 문서를 생성해야 합니다. XML 변환을 사용하여 XML을 생성하십시오.

다음 유형의 미드스트림 XML 변환을 생성할 수 있습니다.

- **XML 파서 변환.** XML 파서 변환은 하나의 입력 포트에서 XML을 읽고 하나 이상의 그룹에 데이터를 출력합니다.
- **XML 생성기 변환.** XML 생성기 변환은 하나 이상의 소스에서 데이터를 읽고 XML을 생성합니다. XML 생성기 변환에는 단일 출력 포트가 있습니다.

미드스트림 XML 변환을 사용하여 WebSphere MQ, TIBCO와 같은 메시징 시스템 또는 파일이나 데이터베이스와 같은 기타 소스에서 XML 데이터를 추출하십시오. 미드스트림 XML 변환이 XML을 구문 분석하거나 파이프라인의 문서를 생성하는 것을 제외하고, XML 변환 기능은 XML 소스 및 대상 기능과 유사합니다.

미드스트림 XML 변환은 XML 마법사 및 XML 편집기가 지원하는 동일한 XML 스키마 구성 요소를 지원합니다. 또한 XML 변환은 다음 기능을 지원합니다.

- **통과 포트**. 통과 포트를 사용하여 미드스트림 변환을 통해 XML이 아닌 데이터를 전달하십시오. 이러한 필드는 XML 스키마 정의의 일부가 아니지만, 비정규화된 XML 그룹을 생성하는 데 사용합니다. 최상위 수준 XML 요소와 같은 방식으로 이러한 필드를 사용합니다. 또한 XML 정의에서 최상위 수준 그룹에 대한 기본 키로 통과 필드를 사용할 수 있습니다.
- **실시간 처리.** 미드스트림 XML 변환을 사용하여 메시징 시스템에서 BLOB로 데이터를 처리하십시오.
- **여러 파티션 지원.** 각 파티션에 대한 여러 XML 문서를 생성할 수 있습니다.

XML 파서 변환

통합 서비스는 XML 파서 변환을 처리할 때 XML 데이터 행을 읽고, XML을 구문 분석하고, 출력 그룹을 통해 데이터를 반환합니다. XML 파서 변환은 통과 포트에서 XML이 아닌 데이터를 반환합니다. JMS 또는 IBM WebSphere MQ와 같은 소스에서 XML 메시지를 구문 분석할 수 있습니다. XML 파서 변환은 활성 변환입니다.

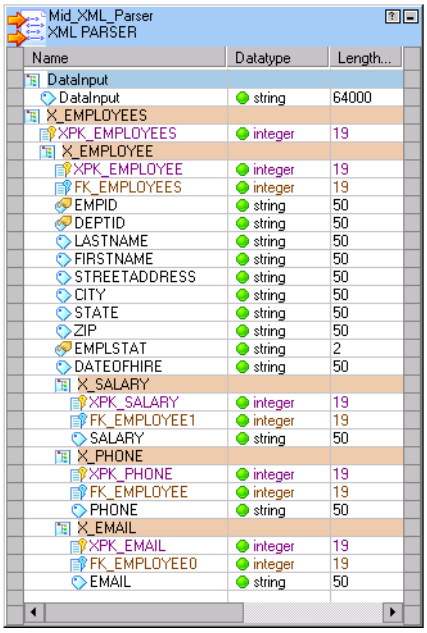
XML 파서 변환에는 하나의 입력 그룹과 하나 이상의 출력 그룹이 있습니다. 입력 그룹에는 하나의 입력 포트인 DataInput이 있으며, 이 포트는 문자열의 XML 문서를 수락합니다.

XML 파서 변환을 작성할 때 XML 마법사를 사용하여 XML, DTD 또는 XML 스키마 파일을 가져오십시오. 예를 들어 다음과 같은 직원 DTD 파일을 가져올 수 있습니다.

```
<!ELEMENT EMPLOYEES (EMPLOYEE+)>
<!ELEMENT EMPLOYEE (LASTNAME, FIRSTNAME, ADDRESS, PHONE+, EMAIL*,   EMPLOYMENT)>
  <!ATTLIST EMPLOYEE EMPID CDATA #REQUIRED
                     DEPTID CDATA #REQUIRED>
<!ELEMENT LASTNAME (#PCDATA)>
<!ELEMENT FIRSTNAME (#PCDATA)>
<!ELEMENT ADDRESS (STREETADDRESS, CITY, STATE, ZIP)>
<!ELEMENT STREETADDRESS (#PCDATA)>
<!ELEMENT CITY (#PCDATA)>
<!ELEMENT STATE (#PCDATA)>
<!ELEMENT ZIP (#PCDATA)>
<!ELEMENT PHONE (#PCDATA)>
<!ELEMENT EMAIL (#PCDATA)>
<!ELEMENT EMPLOYMENT (DATEOFHIRE, SALARY+)>
<!ATTLIST EMPLOYMENT EMPLSTAT (PF|PP|TF|TP|0) "PF">
<!ELEMENT DATEOFHIRE (#PCDATA)>
<!ELEMENT SALARY (#PCDATA)>
```

XML 파서 변환은 루트 보기 X_Employees을 표시하며, X_Employees가 X_Employee의 상위로 나타납니다. X_Employee는 X_Salary, X_Phone 및 X_Email의 상위입니다.

다음 그림은 항목 관계를 작성하도록 선택하는 경우 디자이너가 작성하는 XML 파서 변환을 보여 줍니다.



디자이너는 루트 보기 X_Employees를 작성합니다. X_Employees는 X_Employee의 상위입니다. X_Employee는 X_Salary, X_Phone 및 X_Email의 상위입니다.

XML 파서 변환의 각 보기에는 다른 보기와의 관계를 설정하기 위한 하나 이상의 키가 있습니다. XML 편집기에서 해당 키를 지정하지 않는 경우 디자이너가 각 보기에 대한 기본 키와 외래 키를 작성합니다. 이러한 키의 데이터 유형은 bigint입니다. 통합 서비스가 XML 파서 변환에서 행을 반환할 때마다 키 값을 작성하기 때문에 이러한 키를 생성된 키라고 합니다.

디자이너가 기본 또는 외래 키 열을 작성하는 경우 열 이름에 접두사를 할당합니다. XML 정의에서 접두사는 생성된 기본 키 열의 경우 XPK_, 생성된 외래 키 열의 경우 XFK_입니다. 외래 키는 항상 다른 그룹의 기본 키를 참조합니다. 생성된 외래 키 열은 항상 생성된 기본 키 열을 참조합니다.

예를 들어 그룹 X_Employee에는 XPK_Employee 기본 키가 있습니다. 디자이너는 X_Phone, X_Email 및 X_Salary를 X_Employee 그룹에 연결하는 외래 키 열을 작성합니다. 각 그룹에는 외래 키 열 XFK_Employee가 있습니다.

리포지토리는 키 값을 저장합니다. 리포지토리에서 값을 변경할 수 없지만 세션 이후에 시퀀스 번호를 재설정하거나 다시 시작하도록 선택할 수 있습니다.

XML 파서 입력 유효성 검사

XML을 구문 분석하기 전에 XML의 유효성을 검사하도록 XML 파서 변환을 구성할 수 있습니다. XML 파서 변환은 스키마에 대해 XML의 유효성을 검사합니다. XML이 스키마에 대해 유효하지 않은 경우 행 오류가 발생합니다. XML 파서 변환은 XML 및 관련된 오류 메시지를 별도의 출력 그룹에 반환합니다. 사용자는 올바르지 않은 XML 및 오류 메시지를 대상에 전달할 수 있습니다.

예를 들어 실시간 PowerCenter 세션이 WebSphere MQSeries 소스에서 XML 메시지를 읽습니다. 이 세션은 소스 기반 커밋으로 실행됩니다. 커밋 트랜잭션의 메시지에는 올바르지 않은 XML 페이로드가 있습니다.

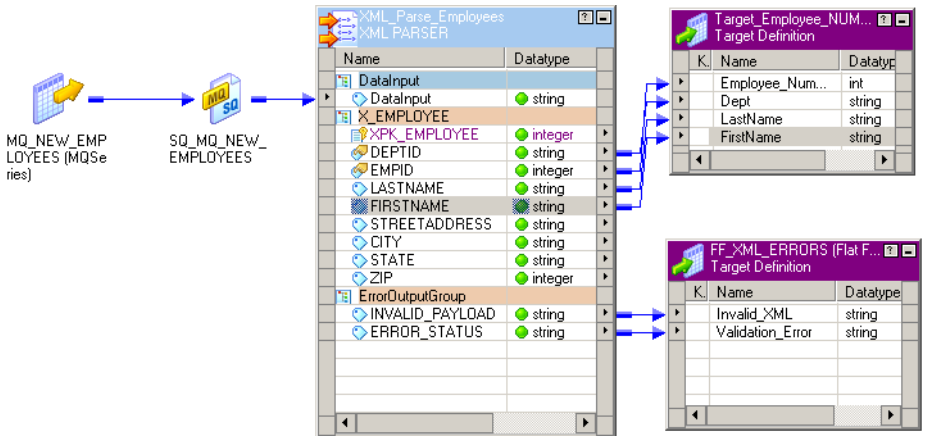
커밋의 실패를 방지하도록 올바른 데이터에서 별도의 출력 그룹으로 올바르지 않은 XML을 반환하도록 XML 파서 변환을 구성할 수 있습니다. XML 파서 변환은 올바른 XML 메시지를 처리하고 트랜잭션을 완료합니다.

세션 로그에는 Route Invalid Payload Through Data Flow(데이터 흐름을 통해 잘못된 페이로드 라우팅)가 활성화되는 시기를 나타내는 메시지가 포함되어 있습니다. 세션 추적 수준을 일반으로 설정하는 경우 통합 서비스는 유효성 검사가 성공적인지 여부를 나타내는 세션 로그에 메시지를 씁니다. 로그 메시지에는 XML 파서가 XML의 유효성을 검사하기 위해 액세스한 스키마 위치가 포함되어 있습니다. XML 스트리밍이 활성화되어 있고 XML이 올바르지 않은 경우 통합 서비스에서는 XML을 잘라서 Invalid_Payload 포트에 전달합니다. 통합 서비스는 세션 로그에 올바르지 않은 XML을 기록합니다.

XML의 유효성을 검사하도록 XML 파서 변환을 구성하려면 미드스트림 XML 파서 탭에서 Route Invalid Payload Through Data Flow(데이터 흐름을 통해 잘못된 페이로드 라우팅) 옵션을 활성화하십시오. 디자이너가 다음 포트를 XML 파서 변환에 추가합니다.

- **Invalid_Payload**. 올바르지 않은 XML 메시지를 파이프라인에 반환합니다. XML 페이로드가 올바른 경우 Invalid_Payload 포트에는 null 값이 포함됩니다. 이 포트에는 DataInput 포트와 동일한 전체 자릿수가 있습니다.
- **Error_Status**. XML 유효성 검사에서 반환된 오류 문자열 또는 상태를 포함합니다. XML이 현재 행에 대해 올바른 경우 Error_Status에는 null 값이 포함됩니다. 이 포트에는 DataInput 포트와 동일한 전체 자릿수가 있습니다.

다음 그림에서는 올바르지 않은 XML 메시지를 오류 대상 테이블에 라우팅하는 XML 파서 변환을 보여 줍니다.



이 매핑에는 다음 개체가 포함됩니다.

- **MQSeries 소스 정의.** 메시지 데이터 필드의 직원 XML 데이터를 포함합니다.
- **소스 한정자 변환**. WebSphere MQ에서 데이터를 읽습니다. 메시지 헤더 필드 및 메시지 데이터 필드를 나타내는 포트 집합을 포함합니다.
- **XML 파서 변환**. DataInput 포트의 XML 메시지 데이터를 받습니다. XML이 올바른 경우 XML 파서 변환은 직원 데이터를 반환하고 이 데이터를 대상에 전달합니다. XML이 올바르지 않은 경우 XML 파서 변환은 Invalid_Payload 포트의 XML을 반환합니다. Error_Status 포트의 오류 메시지를 반환합니다.
- **직원 대상 정의**. 올바른 직원 데이터 행을 받습니다.
- **XML_Errors 대상 정의.** 올바르지 않은 XML 및 오류 메시지를 받습니다.

변환에 대한 세션 속성에서 XML 스키마 위치 특성을 구성하십시오. 스키마의 이름과 위치를 입력하여 XML의 유효성을 검사하십시오. XML 스키마 정의에 대한 워크플로우, 세션 또는 매핑 변수 및 매개 변수를 구성할 수 있습니다. 세미콜론으로 스키마를 구분하면 유효성 검사를 위해 여러 스키마를 구성할 수 있습니다.

입력 XML 페이로드에 DTD를 포함하면 유효성 검사에 이 DTD를 사용할 수 있습니다. XML 스키마 위치 특성에서 DTD를 구성하거나 이 DTD를 사용하여 올바르지 않은 XML 데이터를 잘못된 페이로드 포트에 라우팅할 수 없습니다.

XML 스트리밍을 활성화하는 경우 Invalid_Payload 포트에 대한 전체 자릿수가 최대 메시지 크기와 일치하는지 확인하십시오. 포트 전체 자릿수가 메시지 크기보다 작은 경우 XML 파서 변환에서 Invalid_Payload 포트의 잘린 XML을 반환하고 세션 로그에 오류를 기록합니다.

XML 파서 변환으로 XML 스트리밍

구조화되지 않은 Data Transformation, JMS 소스 또는 WebSphere MQ 소스에서 XML 파서 변환으로 XML을 스트리밍하도록 세션을 구성할 수 있습니다. PowerCenter 통합 서비스는 XML 데이터를 스트리밍할 때 XML 데이터를 여러 세그먼트로 분할합니다.

XML 파서 변환에서 더 작은 입력 포트를 구성하고 XML 파서 변환이 큰 XML 파일을 처리하는 데 필요로 하는 메모리 양을 줄일 수 있습니다. 100MB보다 큰 XML 파일을 구문 분석할 수 있습니다.

XML 스트리밍을 활성화하면 XML 파서 변환이 포트 크기보다 작거나 같은 세그먼트의 데이터를 받습니다. XML 파일이 포트 크기보다 큰 경우 PowerCenter 통합 서비스가 두 개 이상의 행을 XML 파서 변환에 전달합니다. 각 XML 행에는 스트리밍의 행 유형이 있습니다. 마지막 행에는 삽입의 행 유형이 있습니다.

입력 포트 전체 자릿수는 XML을 XML 파서 변환으로 전달하는 개체의 출력 포트 전체 자릿수보다 크거나 같아야 합니다. 대부분의 XML 문서가 작지만 일부 메시지가 큰 경우 최상의 성능을 위해 XML 파서 변환 포트 크기를 더 작은 메시지 크기로 설정하십시오.

XML 스트리밍을 활성화하는 경우 XML 데이터를 XML 파서 변환으로 전달하고 있는 소스 또는 변환에 대해서도 XML 스트리밍을 활성화해야 합니다. 스트리밍을 활성화하지 않는 경우 XML 파서가 한 행의 XML을 받으며, 이로 인해 성능이 저하될 수 있습니다.

XML 파서 변환에서 XML 스트리밍을 활성화하려면 XML 파서 변환 세션 속성에서 Enable XML Input Streaming(XML 입력 스트리밍 활성화)을 선택하십시오. 소스 또는 변환에서 XML 스트리밍을 활성화하지만 XML 파서 변환에 대해서는 XML 스트리밍을 활성화하지 않는 경우 XML 파서 변환이 XML 파일을 처리할 수 없습니다.

XML 스트리밍을 활성화하고 XML 문서에서 오류가 발생하는 경우 PowerCenter 통합 서비스는 기본적으로 XML 문서를 세션 로그에 씁니다. 오류가 발생하면 XML 문서를 오류 로그 파일에 쓰도록 세션을 구성할 수 있습니다.

세션 속성에서 로그 소스 행 데이터를 활성화하십시오. 로깅을 활성화하고 XML 문서에서 오류가 발생하는 경우 PowerCenter 통합 서비스는 행 오류를 생성합니다. PowerCenter 통합 서비스는 XML 문서를 오류 로그 파일에 쓰고 오류 수를 증분 처리합니다.

PowerCenter에서 XML 크기 조정에 대한 자세한 내용은 "PowerCenter에서 XML 사용 개요" 페이지 28을 참조하십시오. PowerCenter에서 XML 처리에 적용되는 제한에 대한 자세한 내용은 "제한" 페이지 29을 참조하십시오.

다른 요소 유형을 가진 변환을 작성하고 더 큰 XML 입력 파일을 변환하려면 데이터 프로세서 변환을 사용합니다. 데이터 프로세서 변환을 작성하는 방법에 대한 자세한 내용은 *Informatica Data Transformation 사용자 가이드* 및 *Informatica Data Transformation 시작하기 가이드*를 참조하십시오.

XML 10진수 데이터 유형

XML 10진수 요소의 전체 자릿수를 34보다 큰 자릿수로 정의하는 경우 통합 서비스는 XML 파서 변환에서 XML 10진수 데이터 유형을 배정밀도로 변환하기 위한 외부 함수를 호출합니다. 이 함수는 통합 서비스가 실행되고 있는 노드에 종속된 전체 자릿수 길이와 함께 배정밀도를 반환합니다. 모든 플랫폼에서 숫자가 반올림되기 전에 전체 자릿수는 17자릿수로 보장되지만 일부 플랫폼에서는 전체 자릿수가 더 클 수 있습니다.

예를 들어 Windows 32비트에서 통합 서비스는 17자릿수 이후 숫자를 반올림합니다.

```
1234.567890123456789 is converted to 1234.567890123460800.
```

HP-UX 32비트에서 통합 서비스는 34자릿수 이후 숫자를 반올림합니다.

XML 생성기 변환

XML 생성기 변환을 사용하여 XML 문서를 작성하기 위해 여러 소스에서 가져오는 입력을 결합하십시오. 예를 들어 이 변환을 사용하여 두 개의 TIBCO 소스에서 하나의 TIBCO 대상으로 XML 데이터를 결합할 수 있습니다. 하나의 소스에 직원 및 급여 정보가 포함될 수 있고, 다른 소스에는 직원 전화 및 전자 메일 정보가 포함될 수 있습니다. XML 생성기 변환이 연결된 활성 변환입니다.

XML 생성기 변환은 XML 대상 정의와 유사합니다. 통합 서비스가 XML 생성기 변환을 처리할 때 XML 데이터 행을 씁니다. 또한 통합 서비스는 해당 변환에서 XML이 아닌 데이터를 포함하는 통과 필드를 처리할 수 있습니다.

XML 생성기 변환에는 하나 이상의 입력 그룹과 하나의 출력 그룹이 있습니다. 출력 그룹에는 하나의 포트 “DataOutput”이 있으며, 이 포트는 문자열 데이터 BLOB XML 문서를 생성합니다. 통과 필드를 작성하면 출력 그룹에 통과 포트가 포함됩니다.

미드스트림 XML 변환 작성

미드스트림 XML 변환을 작성할 때 XML 마법사와 XML 편집기를 사용하여 XML 그룹을 정의합니다. 변환 개발자 및 매핑 디자이너에서 변환을 작성할 수 있습니다.

미드스트림 XML 변환을 작성하려면:

1. 변환 개발자 또는 매핑 디자이너를 엽니다.
2. 변환 > 작성을 클릭합니다.

   변환 작성 대화 상자가 나타납니다.
3. XML 파서 또는 XML 생성기 변환 유형을 선택합니다.
4. 변환 이름을 입력하고 작성을 클릭합니다.

   XML 정의 가져오기 대화 상자가 나타납니다.
5. 가져올 파일을 선택하고 열기를 클릭합니다.

   XML 마법사가 나타납니다.
6. XML 마법사를 사용하여 XML 정의를 작성합니다.
7. XML 마법사에서 마침을 클릭합니다.

   작업 공간에 미드스트림 XML 변환이 표시됩니다.
8. 미드스트림 XML 변환 속성을 편집하려면 작업 공간에서 해당 변환을 두 번 클릭합니다.

미드스트림 XML 정의 동기화

미드스트림 XML 변환을 작성하기 위해 가져온 스키마 또는 소스 파일의 다른 버전을 사용하여 해당 변환을 동기화할 수 있습니다. 예를 들어 XML 파서 변환을 작성하기 위해 가져온 스키마 파일에 새 요소를 추가할 수 있습니다. 변환을 삭제한 후 다시 작성하는 대신 새로운 스키마로 XML 파서 변환을 업데이트할 수 있습니다.

미드스트림 XML 변환을 동기화하려면 변환 개발자 또는 매핑 디자이너를 사용하십시오.

미드스트림 XML 변환을 동기화하려면:

1. 변환 개발자 또는 매핑 개발자를 엽니다.
2. 업데이트하려는 미드스트림 XML 변환이나 매핑을 작업 공간으로 끕니다.
3. 미드스트림 XML 변환의 상단을 마우스 오른쪽 단추로 클릭합니다.
4. Synchronize XML transformation(XML 변환 동기화)을 선택합니다.
   XML 정의 가져오기 대화 상자가 나타납니다.
5. XML 파서 또는 XML 생성기 변환을 작성하는 데 사용한 리포지토리 정의나 파일로 이동합니다.
6. 열기를 클릭하여 변환을 업데이트합니다.

디자이너는 사용자가 변환을 작성하는 데 사용하지 않은 파일과 변환을 동기화할 수 없습니다.

소스 및 대상 XML 정의를 동기화하려면 소스 분석기 및 대상 디자이너를 사용합니다.

미드스트림 XML 변환 속성 편집

미드스트림 XML 변환 속성 중 일부를 편집할 수 있습니다. 하지만 XML 마법사와 XML 편집기를 사용하여 변환을 정의하기 때문에 이러한 도구를 사용하여 XML 정의를 변경해야 합니다.

매핑 디자이너에서 미드스트림 XML 변환을 작성하는 경우 다음 규칙이 적용됩니다.

- 변환을 재사용 가능하게 만드는 경우 매핑 디자이너에서 변환 속성 일부를 변경할 수 있습니다. 통과 포트 또는 메타데이터 확장을 추가할 수 없습니다.
- 재사용 불가능한 변환을 작성하는 경우 매핑 디자이너에서 변환을 편집할 수 있습니다.

미드스트림 XML 변환을 구성할 때 다음 탭의 구성 요소를 구성할 수 있습니다.

- **변환 탭.** 변환 탭에서 변환 이름을 바꾸고 설명을 추가합니다.
- **포트 탭.** XML 파서 또는 XML 생성기 탭에서 작성하는 변환 포트 및 특성을 표시합니다.
- **속성 탭.** 추적 수준을 업데이트합니다.
- **초기화 속성 탭.** 외부 절차에서 초기화 중에 사용하는 런타임 속성을 작성합니다.
- **메타데이터 확장 탭.** XML 변환 등 리포지토리 개체와 정보를 연결하여 리포지토리에 저장된 메타데이터를 확장합니다.
- **포트 특성 정의 탭.** 변환의 모든 포트에 적용되는 포트 특성을 정의합니다.
- **미드스트림 XML 파서 또는 XML 생성기 탭.** 이 탭을 사용하여 통과 포트를 작성합니다. 통과 포트를 사용하면 해당 변환을 통해 XML이 아닌 데이터를 전달할 수 있습니다. XML 파서 변환의 경우, 시퀀스 번호 지정을 사용하여 XML 열 이름을 생성하면 시퀀스 번호를 재설정하도록 선택할 수 있습니다. XML 생성기 변환의 경우, 커밋 시 새 XML 문서를 작성하도록 선택할 수 있습니다.

속성 탭

속성 탭에서 미드스트림 XML 변환 속성을 구성하십시오.

다음 테이블에는 속성 탭에서 변경할 수 있는 옵션이 설명되어 있습니다.

변환	설명
런타임 위치	DLL 또는 공유 라이브러리가 포함된 위치입니다. 기본값은 $PMExtProcDir입니다. XML 세션이 실행되는 통합 서비스 노드에 상대적인 경로를 입력하십시오. 이 속성이 비어 있는 경우 통합 서비스는 통합 서비스 노드에 정의된 환경 변수를 사용하여 DLL 또는 공유 라이브러리를 찾습니다. 런타임 위치 또는 통합 서비스 노드에 정의된 환경 변수에 모든 DLL이나 공유 라이브러리를 복사해야 합니다. 통합 서비스가 DLL, 공유 라이브러리 또는 참조된 파일을 찾을 수 없으면 해당 절차를 로드하지 못합니다.
추적 수준	이 변환에 대해 세션 로그에 표시되는 세부 정보의 양입니다. 기본값은 일반입니다.
변환 범위	통합 서비스가 수신 데이터에 변환 논리를 적용하는 방법을 나타냅니다. XML 파서 변환에 대한 다음 변환 범위 값 중 하나를 선택할 수 있습니다. - 행. 한 번에 데이터의 한 행에 변환 논리를 적용합니다. 다음 행을 처리하기 전에 모든 출력 그룹에 대해 생성된 행을 플러시합니다. - 트랜잭션. 트랜잭션의 모든 행에 변환 논리를 적용합니다. 트랜잭션 경계에서, 출력 블록이 채워질 때 그리고 파일 끝에서, 생성된 행을 플러시합니다. - 모든 입력. 모든 수신 데이터에 변환 논리를 적용합니다. 출력 블록이 채워질 때 그리고 파일 끝에서만, 생성된 행을 플러시하십시오. XML 생성기 변환의 경우, 커밋 시 작업 설정을 커밋 무시로 설정하면 디자이너는 변환 범위를 모든 입력으로 설정합니다. 커밋 시 작업을 새 문서 작성으로 설정하면 디자이너는 변환 범위를 트랜잭션 수준으로 설정합니다.
출력은 반복 가능합니다.	출력 데이터의 순서가 세션 실행 간에 일치하는지 여부를 나타냅니다. - 사용 안 함 출력 데이터 순서는 세션 실행 간에 일관되지 않습니다. - 입력 순서 기반. 입력 데이터 순서가 세션 실행 간에 일관된 경우 입력 순서는 세션 실행 간에 일관됩니다. - 항상 입력 데이터 순서가 세션 실행 간에 일관되지 않더라도 출력 데이터 순서는 세션 실행 간에 일관됩니다. XML 파서 변환에 대한 기본값은 입력 순서 기반입니다. XML 생성기 변환에 대한 기본값은 항상입니다.
파티션당 단일 스레드 필요	통합 서비스가 하나의 스레드로 각 파티션을 처리해야 하는지 여부를 나타냅니다.
확정 출력입니다.	변환에서 세션 실행 간에 동일한 출력 데이터를 생성하는지 여부를 나타냅니다. 이 변환을 사용하는 세션에서 복구를 수행하려면 이 속성을 활성화해야 합니다. 기본값은 활성화됨입니다.

경고: 변환을 반복 가능 및 확정으로 구성하는 경우 데이터가 반복 가능 및 확정인지 확인하는 것은 사용자의 책임입니다. 세션과 복구 간에 동일한 데이터를 생성하지 않는 변환으로 세션을 복구하려는 경우 복구 프로세스로 인해 손상된 데이터가 발생할 수 있습니다.

미드스트림 XML 파서 탭

미드스트림 XML 파서 탭을 사용하여 DataInput 포트 크기를 수정하십시오. 또한 이 탭에서 통과 포트를 추가할 수 있습니다.

미드스트림 XML 파서 탭에서 XML 편집기에 액세스할 수 있습니다. XML 편집기 단추를 클릭하십시오.

참고: XML 편집기에 액세스하는 경우 XML 편집기를 종료할 때까지 Edit Transformations(변환 편집)를 업데이트할 수 없습니다.

다음 테이블에는 미드스트림 XML 파서 탭에서 변경할 수 있는 옵션이 설명되어 있습니다.

변환	설명
전체 자릿수	열의 길이입니다. 기본 DataInput 포트 전체 자릿수는 64K입니다. 통과 포트에 대한 기본 전체 자릿수는 20입니다. 전체 자릿수를 늘릴 수 있습니다.
다시 시작	항상 생성된 키 시퀀스를 1에서 시작하십시오. 세션을 실행할 때마다 XML 정의의 모든 그룹에서 키 시퀀스 값은 1에서 다시 시작됩니다.
재설정	세션 끝에서 모든 그룹의 생성된 키 모두에 대해 값 시퀀스를 재설정하십시오. 시퀀스 번호를 세션 전에 있었던 위치로 다시 재설정하십시오.
Route Invalid Payload Through Data Flow(데이터 흐름을 통해 잘못된 페이로드 라우팅)	스키마에 대해 XML의 유효성을 검사하십시오. XML이 스키마에 대해 유효하지 않은 경우 행 오류가 발생합니다. XML 파서 변환은 XML 및 관련된 오류 메시지를 별도의 출력 그룹에 반환합니다.
설명	변환을 설명합니다.

참고: 재설정 또는 다시 시작을 선택하지 않는 경우, 생성된 키의 시퀀스 번호가 세션 간에 늘어납니다. 재설정 또는 다시 시작 옵션을 선택하는 경우 초기화 속성 탭에 나타나는 재설정 또는 다시 시작 속성을 업데이트합니다. 하지만 초기화 속성 탭에서 이러한 옵션을 변경할 수 없습니다.

미드스트림 XML 생성기 탭

XML 생성기 탭을 사용하여 DataOutput 포트 크기를 수정하십시오. 또한 이 탭에서 통과 포트를 추가할 수 있습니다.

미드스트림 XML 생성기 탭에서 XML 편집기에 액세스할 수 있습니다. XML 편집기 단추를 클릭하십시오. XML 편집기에 액세스하는 경우 XML 편집기를 종료할 때까지 변환 속성을 업데이트할 수 없습니다.

다음 테이블에는 XML 생성기 변환 탭에서 변경할 수 있는 옵션이 설명되어 있습니다.

변환 설정	설명
전체 자릿수	열의 길이입니다. 기본 DataOutput 포트 전체 자릿수는 64K입니다. 통과 포트에 대한 기본 전체 자릿수는 20입니다. 전체 자릿수를 늘릴 수 있습니다.
커밋 시	통합 서비스는 커밋 후 여러 XML 문서를 생성할 수 있습니다. 다음 옵션 중 하나를 사용하십시오. - 커밋 무시. 통합 서비스는 XML 문서를 작성하고 파일 끝에서 이 문서에 데이터를 씁니다. 두 개의 다른 소스가 XML 생성기 변환에 연결된 경우 이 옵션을 사용하십시오. - 새 문서 작성. 각 커밋에서 새 XML 문서를 작성합니다. 실시간 세션을 실행하는 경우 이 옵션을 사용하십시오. 세션이 여러 파티션을 사용하는 경우, 커밋 시 작업 설정에 관계없이 통합 서비스에서 각 파티션에 대한 별도의 XML 문서를 생성합니다. 새 문서 작성을 선택하는 경우 통합 서비스에서 각 파티션에 대한 새 문서를 작성합니다.
설명	변환을 설명합니다.

참고: 커밋 시 작업 설정을 커밋 무시로 설정하면 디자이너는 변환 범위를 모든 입력으로 설정합니다. 커밋 시 작업을 새 문서 작성으로 설정하면 디자이너는 변환 범위를 트랜잭션 수준으로 설정합니다.

통과 포트 생성

통과 포트는 미드스트림 XML 변환을 통해 XML이 아닌 데이터를 전달하는 열입니다. 예를 들어 MQSeries 소스 및 대상에 대한 XML을 사용하여 메시지 ID를 전달할 수 있습니다. 메시지 ID를 사용하여 요청 및 회신에 대한 입력 및 출력 메시지를 연관시키십시오.

미드스트림 변환에서 통과 포트를 정의할 때 XML 파서 변환의 DataInput 그룹 또는 XML 생성기 변환의 DataOutput 그룹에 통과 포트를 추가합니다.

포트를 생성한 후에는 XML 편집기를 사용하여 XML 정의의 다른 보기에 해당 참조 포트를 추가합니다. XML 파서 변환에서 통과 포트는 입력 포트이고 해당 참조 포트는 출력 포트입니다. XML 생성기 변환에서 통과 포트는 출력 포트이고 연결된 참조 포트는 입력 포트입니다.

미드스트림 XML 변환에서 통과 포트를 작성하려면:

1. 변환 개발자 또는 매핑 디자이너에서 변환을 엽니다.
2. 변환을 두 번 클릭하여 Edit Transformations(변환 편집)를 엽니다.
3. 미드스트림 XML 생성기 또는 미드스트림 XML 파서 탭을 클릭합니다.

   변환 유형에 따라 DataInput 또는 DataOutput 포트가 나타납니다.
4. 추가 단추를 클릭하여 통과에 대한 출력 포트를 추가합니다.

   기본 필드가 필드 이름 열에 나타납니다.
5. 필드 이름을 수정합니다. 또한 정의를 작성하는 데 사용한 파일에 따라 유형, 전체 자릿수 및 배율을 수정할 수도 있습니다.
6. XML 편집기를 클릭하여 변환에 대한 XML 정의를 엽니다.

   해당 정의의 XML 보기가 작업 공간에 나타납니다.
7. 보기 상단을 마우스 오른쪽 단추로 클릭하여 참조 포트를 추가합니다.
8. Add a Reference Port(참조 포트 추가)를 선택합니다.

   Reference Port(참조 포트) 대화 상자가 열립니다.

   이 대화 상자에는 변환에서 추가한 통과 포트가 나열됩니다.
9. 보기의 새 참조 포트에 해당하는 통과 포트를 선택하고 확인을 클릭합니다.

   해당 출력 참조 포트가 보기에 나타납니다. 열 창에서 포트의 이름을 의미 있는 이름으로 바꿀 수 있습니다.
10. Apply Changes(변경 사항 적용)를 클릭하고 XML 편집기를 종료합니다.
11. 변환에서 확인을 클릭합니다.

XML이 아닌 데이터가 Pass_thru_field라는 입력 포트를 통해 나오고 해당 COL_0 참조 출력 포트를 통해 전달됩니다.

미드스트림 XML 변환 문제 해결

XML CLOB가 포함된 데이터베이스 테이블에서 XML 파일을 추출해야 합니다. 각 XML 파일은 최대 2GB일 수 있습니다. XML 파서 변환을 작성하는 경우 CLOB 열에 대한 고정 최대 길이를 정의해야 합니다. 하지만 CLOB 데이터 유형의 최대 길이는 104MB입니다.

XML 데이터가 너무 커서 테이블에서 XML 파서 변환으로 테이블을 직접 전달할 수 없습니다. 플랫 파일에 대한 CLOB 테이블 데이터를 준비하고 이 파일에서 XML 소스 정의를 작성해야 합니다.

PowerCenter에서 XML 크기 조정에 대한 자세한 내용은 "PowerCenter에서 XML 사용 개요" 페이지 28을 참조하십시오. PowerCenter에서 XML 처리에 적용되는 제한에 대한 자세한 내용은 "제한" 페이지 29을 참조하십시오.

다른 요소 유형을 가진 변환을 작성하고 더 큰 XML 입력 파일을 변환하려면 데이터 프로세서 변환을 사용합니다. 데이터 프로세서 변환을 작성하는 방법에 대한 자세한 내용은 *Informatica Data Transformation 사용자 가이드* 및 *Informatica Data Transformation 시작하기 가이드*를 참조하십시오.

XML 데이터 유형 참조

XML 및 변환 데이터 유형

PowerCenter는 2001년 5월 2일 W3C 권장 사항에서 지정된 모든 XML 데이터 유형을 지원합니다. 다음 테이블에는 XML 데이터 유형이 나열되어 있으며 XML 소스 한정자 변환의 변환 데이터 유형과 비교합니다. XML 데이터 유형에 대한 자세한 내용은 http://www.w3.org/TR/xmlschema-2에서 XML 데이터 유형에 대한 W3C 사양을 참조하십시오.

XML 파일을 가져와 정의를 작성하는 경우 XML 정의 및 미드스트림 XML 변환에서 데이터 유형을 변경할 수 있습니다. XML 데이터 유형을 XML 스키마에서 가져오는 경우 변경할 수 없습니다. 매핑 내에서 XML 소스에 대한 변환 데이터 유형을 변경할 수 없습니다.

다음 테이블에는 XML 및 해당 변환 데이터 유형이 설명되어 있습니다.

데이터 유형	변환	범위
anySimpleType	문자열	1 ~ 104,857,600자
anyURI	문자열	1 ~ 104,857,600자
base64Binary	이진	1~104,857,600바이트
부울	작은 정수	전체 자릿수 5; 소수 자릿수 0
바이트	작은 정수	전체 자릿수 5; 소수 자릿수 0
날짜	날짜/시간	A.D. 0001년 1월 1일 ~ A.D. 9999년 12월 31일(전체 자릿수는 나노초)
날짜/시간	날짜/시간	A.D. 0001년 1월 1일 ~ A.D. 9999년 12월 31일(전체 자릿수는 나노초)

데이터 유형	변환	범위
10진수	10진수	전체 자릿수 1 ~ 28, 소수 자릿수 0 ~ 28
배정밀도	배정밀도	정밀도 15, 배율 0
기간	문자열	1 ~ 104,857,600자
ENTITIES	문자열	1 ~ 104,857,600자
ENTITY	문자열	1 ~ 104,857,600자
부동 소수점 수	배정밀도	정밀도 15, 배율 0
gDay	정수	-2,147,483,648 ~ 2,147,483,647 전체 자릿수 10; 소수 자릿수 0
gMonth	정수	-2,147,483,648 ~ 2,147,483,647 전체 자릿수 10; 소수 자릿수 0
gMonthDay	날짜/시간	A.D. 0001년 1월 1일 ~ A.D. 9999년 12월 31일(전체 자릿수는 나노초)
gYear	정수	전체 자릿수 10; 소수 자릿수 0
gYearMonth	날짜/시간	A.D. 0001년 1월 1일 ~ A.D. 9999년 12월 31일(전체 자릿수는 나노초)
hexBinary	이진	1~104,857,600바이트
ID	문자열	1 ~ 104,857,600자
IDREF	문자열	1 ~ 104,857,600자
IDREFS	문자열	1 ~ 104,857,600자
int	Bigint	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807 전체 자릿수 19; 소수 자릿수 0
정수	Bigint	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807 전체 자릿수 19; 소수 자릿수 0
언어	문자열	1 ~ 104,857,600자
긴 정수	Bigint	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807 전체 자릿수 19; 소수 자릿수 0
이름	문자열	1 ~ 104,857,600자
Ncname	문자열	1 ~ 104,857,600자
negativeInteger	Bigint	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807 전체 자릿수 19; 소수 자릿수 0
NMTOKEN	문자열	1 ~ 104,857,600자

데이터 유형	변환	범위
NMTOKENS	문자열	1 ~ 104,857,600자
nonNegativeInteger	Bigint	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807 전체 자릿수 19; 소수 자릿수 0
nonPositiveInteger	Bigint	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807 전체 자릿수 19; 소수 자릿수 0
normalizedString	문자열	1 ~ 104,857,600자
NOTATION	문자열	1 ~ 104,857,600자
positiveInteger	Bigint	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807 전체 자릿수 19; 소수 자릿수 0
QName	문자열	1 ~ 104,857,600자
짧은 정수	작은 정수	전체 자릿수 5; 소수 자릿수 0
문자열	문자열	1 ~ 104,857,600자
시간	날짜/시간	A.D. 0001년 1월 1일 ~ A.D. 9999년 12월 31일(전체 자릿수는 나노초)
토큰	문자열	1 ~ 104,857,600자
unsignedByte	작은 정수	전체 자릿수 5; 소수 자릿수 0
unsignedInt	Bigint	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807 전체 자릿수 19; 소수 자릿수 0
unsignedLong	Bigint	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807 전체 자릿수 19; 소수 자릿수 0
unsignedShort	정수	-2,147,483,648 ~ 2,147,483,647 전체 자릿수 10; 소수 자릿수 0

XML 날짜 형식

PowerCenter는 날짜, 시간 및 날짜/시간 데이터 유형에 대해 다음 형식을 지원합니다.

```
CCYY-MM-DDThh:mm:ss:sss
```

XML 파일에서 이 형식 또는 이 형식의 일부를 사용합니다. PowerCenter는 날짜/시간 형식에 대해 음수 날짜를 지원하지 않습니다.

세션 내에서 다음 형식 중 하나로 날짜, 시간 또는 날짜/시간 요소를 사용합니다.

```
CCYY-MM
```

-또는-

```
CCYY-MM-DD/Thh
```

XML 파일의 첫 번째 날짜/시간 요소 형식이 요소의 이후 모든 값에 대한 형식을 결정합니다. 통합 서비스가 서로 다른 형식을 가진 동일한 날짜, 시간 또는 날짜/시간 요소에 대한 값을 읽는 경우 통합 서비스가 행을 거부합니다.

예를 들어 날짜/시간 요소의 첫 번째 값이 다음 형식인 경우

```
CCYY-MM-DDThh:mm:ss
```

통합 서비스는 다음 형식의 요소가 포함된 행을 거부합니다.

```
CCYY-MM-DD
```

XML 파서는 입력 XML의 날짜/시간 값을 통합 서비스를 호스트하는 시스템의 로컬 시간대의 값으로 변환합니다. Windows에서 일광 절약 변경을 위해 시계를 조정하는 옵션을 활성화하는 경우 XML 파서가 한 시간을 날짜/시간 값에 추가합니다. 일관된 날짜/시간 값 변환을 위해 Windows에서 일광 절약 변경을 위해 시계를 조정하는 옵션을 활성화하지 마십시오.

XPath 쿼리 함수 참조

XPath 쿼리 함수 개요

XPath는 XML 문서에서 항목을 찾는 방법을 설명하는 언어입니다. XPath는 루트 구성 요소에서 XML 계층 경로를 기반으로 하는 주소 지정 구문을 사용합니다. 보기 행이 포함된 XPath가 있는 보기 행 또는 열의 요소에 대해 XPath 쿼리 조건자를 작성할 수 있습니다.

XML 보기의 XPath 쿼리 조건자를 사용하여 XML 소스 데이터를 필터링합니다. 세션에서 통합 서비스가 쿼리를 기반으로 소스 XML 파일에서 데이터를 추출합니다. 모든 쿼리가 TRUE를 반환하는 경우 통합 서비스가 보기를 위한 데이터를 추출합니다.

XPath 쿼리 조건자는 추출할 요소 또는 특성 및 조건을 확인하는 쿼리를 포함합니다. 요소 또는 특성의 값을 확인하거나 요소 또는 특성이 소스 XML 데이터에 있는지 확인할 수 있습니다.

이 부록은 XPath 쿼리 조건자에서 사용하는 각 함수에 대해 설명합니다. 함수는 인수를 허용하고 값을 반환합니다. 함수를 작성할 때 XML 보기에 요소 및 특성의 구성 요소를 포함시킨 다음 리터럴 값을 추가할 수 있습니다. 리터럴을 지정하는 경우 작은따옴표 또는 큰따옴표로 리터럴을 묶어야 합니다.

함수 빠른 참조

XPath 쿼리 조건자에서 다음 유형의 함수를 사용합니다.

- **문자열.** 문자열 함수를 사용하여 하위 문자열 값을 테스트하거나, 문자열을 연결하거나, 문자열을 다른 문자열로 변환합니다. 예를 들어 다음 XPath 쿼리 조건자는 직원의 전체 이름이 성과 이름을 연결한 것과 동일한지 확인합니다.

  ```
  EMPLOYEE[./FULLNAME=concat(./ENAME/LASTNAME,./ENAME/FIRSTNAME)]
  ```

- **숫자.** 요소 및 특성 값이 포함된 숫자 함수를 사용합니다. 숫자 함수는 숫자에서 작업을 수행하고 정수를 반환합니다. 예를 들어 다음 XPath 쿼리 조건자는 할인을 반올림하여 결과가 15보다 큰지 테스트합니다.

  ```
  ORDER_ITEMS[round(./DISCOUNT) > 15]
  ```

- **부울.** 부울 함수를 사용하여 요소를 테스트하거나, 언어 특성을 확인하거나, true 또는 false 결과를 강제로 적용합니다. 예를 들어 다음 XPath 쿼리 조건자는 값이 0보다 큰 경우 true를 반환합니다.

  ```
  boolean(string)
  ```

다음 테이블에는 XPath 쿼리 조건자 문자열 함수가 설명되어 있습니다.

함수	구문	설명
concat	concat (string1, string2)	두 문자열을 연관시킵니다.
포함	contains (string, substring)	문자열이 다른 문자열을 포함하는지 확인합니다.
normalize-space	normalize-space (string)	문자열에서 선행 및 후행 공백을 제거합니다.
starts-with	starts-with (string, substring)	string1이 string2로 시작하는지 확인합니다.
문자열	string (value)	숫자 또는 부울을 문자열로 변환합니다.
string-length	string-length (string)	문자열 중 후행 공백을 포함하여 모든 문자의 개수를 반환합니다.
하위 문자열	substring (string, start [,length])	지정된 위치에서 시작하는 문자열의 일부를 반환합니다.
substring-after	substring-after (string, substring)	지정된 위치에서 시작하는 문자열의 일부를 반환합니다.
substring-before	substring-before (string, substring)	하위 문자열 앞에서 발생하는 문자열의 문자를 반환합니다.
변환	translate (string1, string2, string3)	문자열의 문자를 다른 문자로 변환합니다.

다음 테이블에는 XPath 쿼리 조건자 숫자 함수가 설명되어 있습니다.

함수	구문	설명
ceiling	ceiling (number)	숫자를 전달된 숫자보다 크거나 같은 정수 중 가장 작은 정수로 반올림합니다.
floor	floor (number)	숫자를 전달된 숫자보다 작거나 같은 정수 중 가장 큰 정수로 반올림합니다.
숫자	number (value)	문자열 또는 부울 값을 숫자로 변환합니다.
round	round (number)	숫자를 가장 가까운 정수로 반올림합니다.

다음 테이블에는 XPath 쿼리 조건자 부울 함수가 설명되어 있습니다.

함수	구문	설명
부울	boolean (object)	개체를 부울로 변환합니다.
false	false ( )	항상 FALSE를 반환합니다.
lang	lang (code)	요소에 코드 인수와 일치하는 xml:lang 특성이 있는지 확인합니다.

함수	구문	설명
not	not (condition)	부울 조건이 FALSE이면 TRUE를 반환하고 부울 조건이 TRUE이면 FALSE를 반환합니다.
true	true ( )	항상 TRUE를 반환합니다.

부울

값을 부울로 변환합니다.

구문

```
boolean ( object )
```

다음 테이블에는 부울 인수가 설명되어 있습니다.

인수	설명
개체	숫자 또는 문자열 데이터 유형입니다. 테스트할 숫자 또는 문자열을 전달합니다.

반환 값

부울.

함수가 다음과 같이 부울을 반환합니다.

- 문자열 길이가 0이 아닌 경우 문자열은 TRUE를 반환하고 그렇지 않은 경우 FALSE를 반환합니다.
- 숫자가 0이거나 NaN(숫자가 아님)인 경우 숫자는 FALSE를 반환하고 그렇지 않은 경우 TRUE를 반환합니다.

예

다음 예는 이름에 문자가 있는지 확인합니다.

```
boolean ( NAME )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

NAME	RETURN VALUE
Lilah	TRUE
-	FALSE

다음 예는 우편 번호가 숫자인지 확인합니다.

```
boolean ( ZIP_CODE )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

ZIP_CODE	RETURN VALUE
94061	TRUE

ZIP_CODE	RETURN VALUE
94005	TRUE
9400g	FALSE

ceiling

숫자를 전달된 숫자보다 크거나 같은 정수 중 가장 작은 정수로 반올림합니다.

구문

```
ceiling ( number )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
숫자	숫자 값. 반올림하려는 숫자입니다.

반환 값

정수.

예제

다음 식은 가장 작은 정수로 반올림된 가격을 반환합니다.

```
ceiling ( PRICE )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

PRICE	RETURN VALUE
39.79	40
125.12	126
74.24	75
NULL	NULL
-100.99	-100
100	100

concat

두 문자열을 연관시킵니다.

구문

```
concat ( string1, string2 )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
string1	문자열 데이터 유형. 연결할 첫 번째 문자열을 전달합니다.
string2	문자열 데이터 유형. 연결할 두 번째 문자열을 전달합니다.

반환 값

문자열.

문자열 중 하나가 NULL인 경우 concat는 이 문자열을 무시하고 다른 문자열을 반환합니다.

예제

다음 식은 FIRSTNAME과 LASTNAME을 연결합니다.

```
concat( FIRSTNAME, LASTNAME )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

FIRSTNAME	LASTNAME	RETURN VALUE
John	Baer	JohnBaer
NULL	Campbell	Campbell
Greg	NULL	Greg
NULL	NULL	NULL

팁

concat 함수는 문자열에 공백을 추가하지 않습니다. 두 문자열 사이에 공백을 추가하려면 중첩된 concat 함수가 포함된 식을 기록할 수 있습니다. 예를 들어 다음 식은 공백을 이름의 끝에 추가하여 이름을 성에 연결합니다.

```
concat ( concat ( FIRST_NAME, " " ), LAST_NAME )
```

다음 테이블은 인수 및 반환 값 예를 보여 줍니다.

FIRST_NAME	LAST_NAME	RETURN VALUE
John	Baer	John Baer
NULL	Campbell	Campbell *(includes leading space)*
Greg	NULL	Greg

포함

문자열이 다른 문자열을 포함하는지 확인합니다.

구문

```
contains( string, substring )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
문자열	문자열 데이터 유형. 검사할 문자열을 전달합니다. 인수는 대/소문자를 구분합니다.
하위 문자열	문자열 데이터 유형. 문자열에서 검색할 문자열을 전달합니다. 인수는 대/소문자를 구분합니다.

반환 값

부울.

예제

다음 식은 NAME에 SHORTNAME이 포함된 경우 TRUE를 반환합니다.

```
contains( NAME, SHORTNAME )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

NAME	SHORTNAME	RETURN VALUE
John	Baer	FALSE
SuzyQ	Suzy	TRUE
WorldPeace	World	TRUE
CASE_SENSITIVE	case	FALSE

false

항상 FALSE를 반환합니다. 이 함수를 사용하여 부울을 FALSE로 설정합니다.

구문

```
false ()
```

false 함수는 인수를 허용하지 않습니다.

반환 값

FALSE.

예제

false 함수를 다른 함수와 결합하여 FALSE 결과를 강제로 적용합니다.

다음 테이블에는 FALSE를 반환하는 식이 포함되어 있습니다.

EXPRESSION	RETURN VALUE
(salary) = false()	FALSE

EXPRESSION	RETURN VALUE
A/B = false ()	FALSE
starts-with (name, 'T') = false()	FALSE

floor

숫자를 전달된 숫자보다 작거나 같은 정수 중 가장 큰 정수로 반올림합니다.

구문

```
floor( number )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
숫자	숫자 값. 숫자 식을 사용합니다.

반환 값

정수.

함수에 전달된 값이 NULL인 경우 NULL입니다.

예제

다음 식에서는 BANK_BALANCE의 값보다 작거나 같은 정수 중 가장 큰 정수가 반환됩니다.

```
floor( BANK_BALANCE )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

BANK_BALANCE	RETURN VALUE
39.79	39
NULL	NULL
-100.99	-101
5	5

lang

요소에 코드 인수와 언어가 동일한 xml:lang 특성이 있는 경우 TRUE를 반환합니다. lang 함수를 사용하여 언어별로 XML을 선택합니다. xml:lang 특성은 요소 콘텐츠의 언어를 식별하는 코드입니다. 요소는 여러 언어로 된 텍스트를 포함할 수 있습니다.

구문

```
lang ( code )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
코드	문자열 데이터 유형. 요소 콘텐츠 언어 코드를 전달합니다.

반환 값

부울.

예제

다음 식은 요소 콘텐츠의 언어 코드를 검사합니다.

```
lang ( 'en' )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

XML	RETURN VALUE
<Phrase xml:lang="es"> El perro esta en la casa. </Phrase>	FALSE
<Phrase xml:lang="en"> The dog is in the house. </Phrase>	TRUE

normalize-space

문자열에서 선행 및 후행 공백을 제거합니다. 공백은 공백 문자 및 탭 문자와 같이 표시되지 않는 문자를 포함합니다. 이 함수는 공백의 시퀀스를 단일 공백으로 대체합니다.

구문

```
normalize-space ( string )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
문자열	문자열 데이터 유형. 공백이 포함된 문자열을 전달합니다.

반환 값

문자열.

문자열이 NULL인 경우 NULL입니다.

예제

다음 식은 이름에서 초과 공백을 제거합니다.

```
normalize-space ( NAME )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

NAME	RETURN VALUE
Jack Dog	Jack Dog
Harry Cat	Harry Cat

not

부울 조건의 역을 반환합니다. 함수는 조건이 FALSE인 경우 TRUE를 반환하고 조건이 TRUE인 경우 FALSE를 반환합니다.

구문

```
not ( condition )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
조건	부울 식 또는 값입니다.

반환 값

부울.

조건이 NULL인 경우 NULL입니다.

예제

다음 식은 부울 조건의 역을 반환합니다.

```
not ( EMPLOYEE = concat ( FIRSTNAME, LASTNAME ))
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

EMPLOYEE	FIRSTNAME	LASTNAME	RETURN
Fullname	Full	Name	FALSE
Lastname	Lastname	First	TRUE

숫자

문자열 또는 부울 값을 숫자로 변환합니다.

구문

```
number ( value )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
값	부울 또는 문자열 값을 사용합니다.

반환 값

함수는 다음 데이터에 대해 숫자를 반환합니다.

- 문자열에 숫자가 포함된 경우 문자열이 숫자로 변환됩니다. 문자열은 공백을 포함할 수 있으며 빼기 기호 다음에 숫자를 포함할 수 있습니다. 문자열에서 공백이 숫자 뒤에 나올 수 있습니다. 다른 문자열은 NaN(숫자가 아님)입니다.
- 부울 TRUE는 1로 변환됩니다. 부울 FALSE는 0으로 변환됩니다.

함수에 대한 인수로 전달된 값이 숫자가 아닌 경우 함수가 NaN(숫자가 아님)을 반환합니다.

예제

다음 식은 불입을 숫자로 변환합니다.

```
number ( PAYMENT )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

PAYMENT	RETURN VALUE
'850.00'	850.00
TRUE	1
FALSE	0
AB	NaN

round

숫자를 가장 가까운 정수로 반올림합니다. 숫자가 두 정수 사이에 있는 경우 더 높은 숫자로 반환값을 반올림합니다.

구문

```
round ( number )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
숫자	숫자 값. 숫자 데이터 유형 또는 결과가 숫자인 식을 전달합니다.

반환 값

정수.

예제

다음 식은 BANK_BALANCE를 반올림합니다.

```
round( BANK_BALANCE )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

BANK_BALANCE	RETURN VALUE
12.34	12
12.50	13
-18.99	-19
NULL	NULL

starts-with

첫 번째 문자열이 두 번째 문자열로 시작하는 경우 TRUE를 반환합니다. 그렇지 않으면 FALSE를 반환합니다.

구문

```
starts-with ( string, substring )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
문자열	문자열 데이터 유형. 검색할 문자열을 전달합니다. 문자열은 대/소문자를 구분합니다.
하위 문자열	문자열 데이터 유형. 문자열에서 검색할 하위 문자열을 전달합니다. 하위 문자열은 대/소문자를 구분합니다.

반환 값

부울.

예제

다음 식은 NAME이 FIRSTNAME으로 시작하는지 확인합니다.

```
starts-with ( NAME, FIRSTNAME )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

NAME	FIRSTNAME	RETURN VALUE
Kathy Russell	Kathy	TRUE
Joe Abril	Mark	FALSE

문자열

숫자 또는 부울을 문자열로 변환합니다.

구문

```
string ( value )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
값	숫자 또는 부울 값입니다. 숫자 또는 부울 값을 전달합니다.

반환 값

문자열.

값을 전달하지 않는 경우 빈 문자열을 반환합니다. NULL 값을 전달한 경우 NULL을 반환합니다.

문자열 함수는 다음과 같이 숫자를 문자열로 변환합니다.

- 숫자가 정수인 경우 함수는 소수점 및 선행 0이 없는 10진수 형식으로 문자열을 반환합니다.
- 숫자가 정수가 아닌 경우 함수는 소수점 앞에 최소 1자리가 있고 소수점 뒤에 최소 1자리가 있는 소수점이 포함된 문자열을 반환합니다.
- 숫자가 음수인 경우 함수는 빼기 기호(-)가 포함된 문자열을 반환합니다.

문자열 함수는 다음과 같이 부울을 문자열로 변환합니다.

- 부울이 FALSE인 경우 함수는 문자열 “false”를 반환합니다.
- 부울이 TRUE인 경우 함수는 문자열 “true”를 반환합니다.

예제

다음 식은 숫자 인수 SPEED에서 문자열을 반환합니다.

```
string( SPEED )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

SPEED	RETURN VALUE
10.99	‘10.99’
15.62567	‘15.62567’
0	‘0’
10	‘10’
50	‘50’
1.3	‘1.3’

다음 식은 부울 인수 STATUS에서 문자열을 반환합니다.

```
string( STATUS )
```

다음 테이블은 인수 및 반환 값 예를 보여 줍니다.

STATUS	RETURN VALUE
TRUE	'true'
FALSE	'false'
NULL	NULL

string-length

문자열 중 후행 공백을 포함하여 모든 문자의 개수를 반환합니다.

구문

```
string-length ( string )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
문자열	문자열 데이터 유형. 평가할 문자열입니다.

반환 값

정수.

함수에 전달된 값이 NULL인 경우 NULL입니다.

예제

다음 식은 고객 이름의 길이를 반환합니다.

```
string-length ( CUSTOMER_NAME )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

CUSTOMER_NAME	RETURN VALUE
Bernice Davis	13
NULL	NULL
John Baer	9

하위 문자열

지정된 위치에서 시작하는 문자열의 일부를 반환합니다. 하위 문자열은 공백을 문자열의 문자로 포함합니다.

구문

```
substring ( string, start [ ,length ] )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
문자열	문자열 데이터 유형. 검색할 문자열입니다.
시작	정수 데이터 유형. 세기 시작할 문자열의 위치를 전달합니다. 시작 위치가 양수인 경우 하위 문자열은 문자열의 처음부터 세는 방식으로 시작 위치를 찾습니다. 처음 문자는 1입니다. 시작 위치가 음수인 경우 하위 문자열은 문자열의 끝부터 세는 방식으로 시작 위치를 찾습니다.
길이	정수 데이터 유형. 0보다 커야 합니다. 문자열에서 반환할 문자 수를 전달합니다. 길이 인수를 생략하는 경우 하위 문자열은 문자열의 시작 위치부터 끝까지 모든 문자를 반환합니다.

반환 값

문자열.

문자열에 숫자 값이 포함된 경우 함수는 문자열을 반환합니다.

음의 정수 또는 0을 전달하는 경우 함수는 빈 문자열을 반환합니다.

함수에 전달된 값이 NULL인 경우 NULL입니다.

예

다음 식은 PHONE에서 지역 번호를 반환합니다.

```
substring( PHONE, 1, 3 )
```

PHONE	RETURN VALUE
809-555-3915	809
NULL	NULL

다음 식은 지역 번호 없이 PHONE을 반환합니다.

```
substring ( phone, 5, 8 )
```

다음 테이블에는 지역 번호가 없는 인수 및 반환 값 예가 포함되어 있습니다.

PHONE	RETURN VALUE
808-555-0269	555-0269
NULL	NULL

문자열의 오른쪽에서 시작할 음수 시작 값을 전달할 수 있습니다. 식은 *길이* 인수에 대해 왼쪽에서 오른쪽으로 소스 문자열을 읽습니다.

```
substring ( PHONE, -8, 3 )
```

다음 테이블에는 식이 왼쪽에서 오른쪽으로 소스 문자열을 읽는 경우의 인수 및 반환 값 예가 포함되어 있습니다.

PHONE	RETURN VALUE
808-555-0269	555
809-555-3915	555
357-687-6708	687
NULL	NULL

길이 인수가 문자열보다 긴 경우 하위 문자열은 문자열의 시작 위치부터 끝까지 모든 문자를 반환합니다. 예:

```
substring ( 'abcd', 2, 8 )
```

'bcd'를 반환합니다.

```
substring ( 'abcd', -2, 8 )
returns 'cd.'
```

substring-after

하위 문자열 뒤에서 발생하는 문자열의 문자를 반환합니다.

구문

```
substring-after ( string, substring )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
문자열	문자열 데이터 유형. 검색할 문자열을 전달합니다.
하위 문자열	문자열 데이터 유형. 문자열에서 검색할 하위 문자열을 전달합니다.

반환 값

문자열.

하위 문자열이 없는 경우 빈 문자열입니다.

함수에 전달된 값이 NULL인 경우 NULL입니다.

예제

다음 식은 지역 번호(415) 뒤에서 발생하는 PHONE의 문자열을 반환합니다.

```
substring-after ( PHONE, (415) )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

PHONE	RETURN VALUE
(415)555-1212	555-1212

PHONE	RETURN VALUE
(408)368-4017	-
NULL	NULL
(415)366-7621	366-7621

substring-before

하위 문자열 앞에서 발생하는 문자열의 일부를 반환합니다.

구문

```
substring-before ( string, substring )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
문자열	문자열 데이터 유형. 검색할 문자열을 전달합니다.
하위 문자열	문자열 데이터 유형. 문자열에서 검색할 하위 문자열을 전달합니다.

반환 값

문자열.

하위 문자열이 없는 경우 빈 문자열입니다.

함수에 전달된 값이 NULL인 경우 NULL입니다.

예제

다음 식은 Third Street 주소에서 발생하는 숫자를 반환합니다.

```
substring-before ( ADDRESS, Third Street )
```

다음 테이블에는 특성 및 반환 값 예가 포함되어 있습니다.

ADDRESS	RETURN VALUE
100 Third Street	100
250 Third Street	250
600 Third Street	600
NULL	NULL

변환

문자열의 문자를 다른 문자로 변환합니다. 함수는 변환 쌍으로 두 개의 다른 문자열을 사용합니다.

구문

```
translate ( string1, string2, string3 )
```

다음 테이블에는 이 함수의 인수가 설명되어 있습니다.

인수	설명
string1	문자열 데이터 유형. 변환할 문자열을 전달합니다.
string2	문자열 데이터 유형. 변환할 문자를 정의하는 문자열을 전달합니다. Translate가 *string1*의 각 문자를 *string2*의 위치를 나타내는 숫자로 대체합니다.
string3	문자열 데이터 유형. 암호화된 *string1*의 문자가 어떤 문자로 변환되어야 하는지 정의하는 문자열을 전달합니다. Translate가 암호화된 *string1*의 각 문자를 *string2*의 위치 숫자에서 *string3*의 문자로 대체합니다.

반환 값

문자열.

예제

다음 식은 두 개의 다른 문자열을 사용하여 문자열을 변환합니다.

```
translate ( EXPRESSION, STRING2, STRING3 )
```

다음 테이블에는 인수 및 반환 값 예가 포함되어 있습니다.

EXPRESSION	STRING2	STRING3	RETURN VALUE
A Space Odissei	i	y	A Space Odyssey
rats	tras	TCAS	CATS
bar	abc	ABC	BAr

문자가 string2에서 발생하지 않는 경우 Translate가 EXPRESSION에서 문자를 변경하지 않습니다. 문자가 EXPRESSION 및 string2에서 발생하지만 string3에서 발생하지 않는 경우 반환된 문자열에서 문자가 발생하지 않습니다.

true

항상 TRUE를 반환합니다. 이 함수를 사용하여 부울을 TRUE로 설정합니다.

구문

```
true ()
```

true 함수는 인수를 허용하지 않습니다.

반환 값

부울 TRUE입니다.

예제

다음 테이블에는 TRUE를 반환하는 식의 예가 포함되어 있습니다.

EXPRESSION	RETURN VALUE
`( decision ) = true ()`	TRUE
`A/B = true ()`	TRUE
`( starts-with ( name, 'T' ))= true`	TRUE