How-To Library

Informatica

Data Validation Option Best Practices

Abstract

To maximize the benefits that Data Validation Option offers, you must understand how to best set up Data Validation Option in a customer environment and know the best practices for using Data Validation Option.

Supported Versions

- Data Validation Option 9.5.1 Hotfix 2 Update 1
- Data Validation Option 9.6.0
- Data Validation Option 10.0.0

Table of Contents

Elements in Data Validation Option

The Data Validation Option consists of the following elements:

- Data Validation Option repository
- Data Validation Client
- Data Validation Option command line
- Data Validation Option reports

Data Validation Option Repository

The Data Validation Option repository stores metadata related to Data Validation Option and testing. It holds tests generated by users, test results, environment variables, and user folders. PowerCenter metadata involving folders, tables, and database connections are also stored here. The Data Validation Option repository can reside on Oracle, Microsoft SQL Server, or IBM DB2.

Data Validation Client

The Data Validation Client is a desktop (thick) client that can used to build test cases as well as other objects used in testing. It is also used to execute tests, view test results, and generate reports on completed tests.

Data Validation Option Command Line

Data Validation Option also provides a command line utility, DVOcmd. DVOcmd is used for administration and run time execution. There are two versions of DVOcmd, one on Windows, and another on UNIX. The Windows version can be used with the Data Validation Client for certain administrative tasks, as well as on a Windows sever for both administration and execution. The UNIX version is mainly for server administration and run time execution. It supports a subset of the commands that the Windows DVOcmd supports.

Reporting

Data Validation Option provides several methods to create reports concerning tests that are executed. These are the methods:

Built-in reports

Basic reports can be generated directly from the Data Validation Client. These provide both summary and detailed execution reports for Data Validation Option jobs.

Jasper reports

Additionally, Data Validation Option includes a set of extensive dashboards and reports (over 20 in all) that are available from a browser via Jasper Reports Server Integration (Reporting and Dashboards Service in PowerCenter 9.x). These dashboards provide high-level perspectives on test results and allow a drill-down into the most detailed test results for any given run of any test by any user.

Custom reports

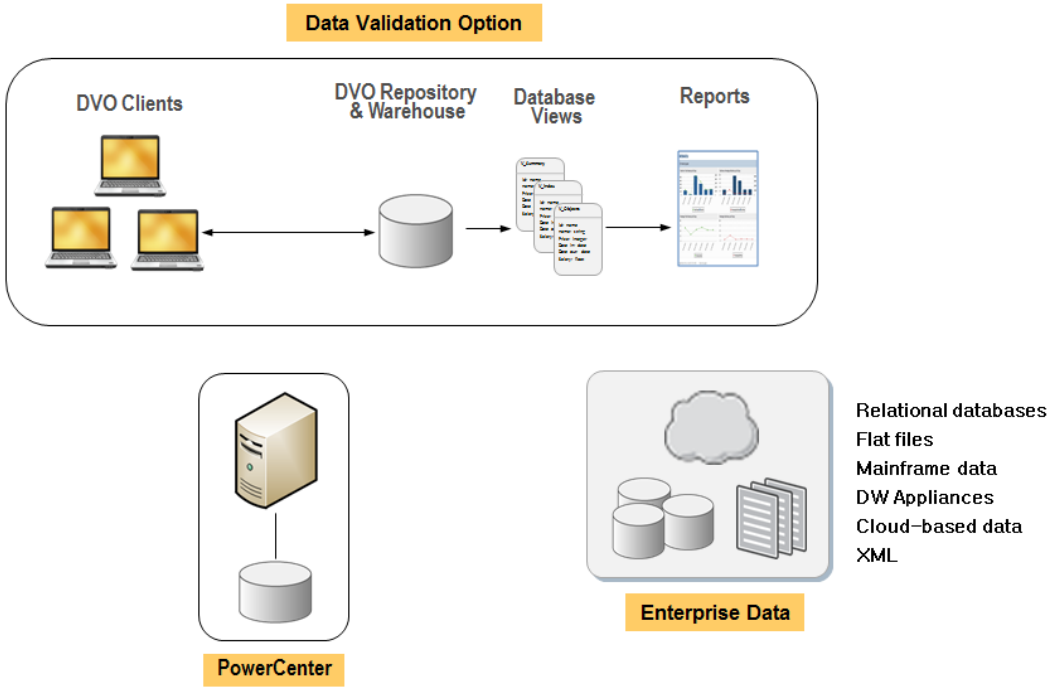
For customers that wish to build their own custom reports, the Data Validation Option repository has many views that can be leveraged via SQL or by reporting products like Tableau, QlikView, MicroStrategy, and so on.

Please see the Data Validation Option documentation for additional information regarding the differences between these reporting solutions.

Detailed Architecture

The overall Data Validation Option architecture consists of a client, a repository, and a server. The client and repository are specific to Data Validation Option. The server (where test execution occurs) is PowerCenter.

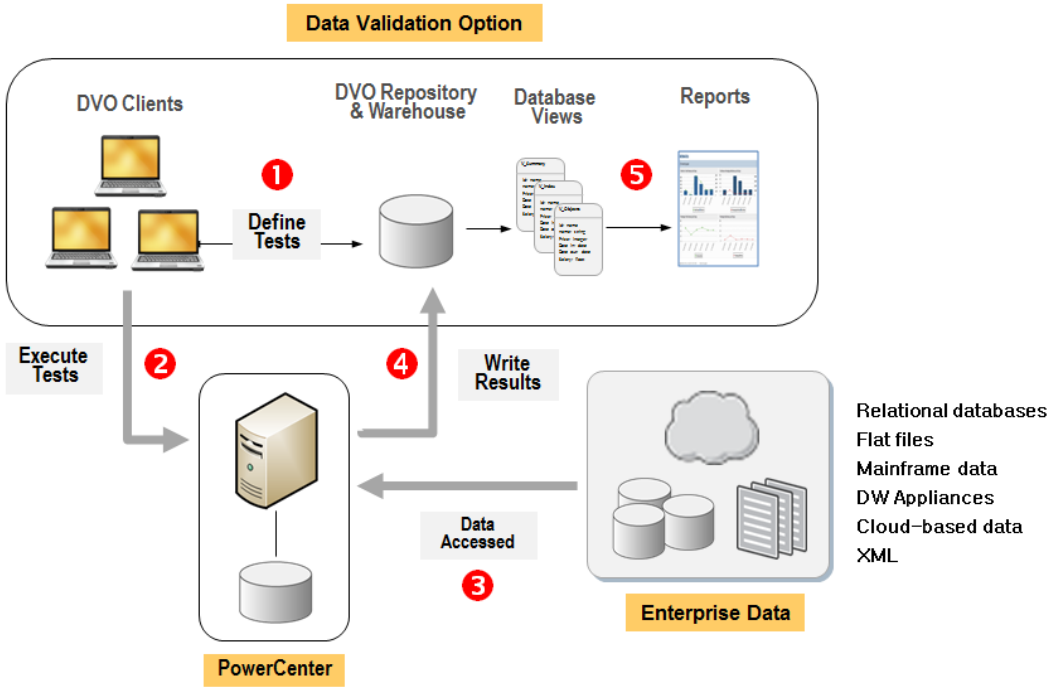
The following image shows the three major elements:



Data Validation Option

Data Validation Option consists of a desktop client, its own combined metadata repository, and a results warehouse. All test definitions and test results are stored in this warehouse. The warehouse has predefined database views against which all reporting occurs.

The following image shows how these elements are used when Data Validation Option tests data:



1. Data Validation Option users define tests in the desktop clients and store those (metadata) definitions in the Data Validation Option repository.
2. Tests are then executed from a GUI or command line. Data Validation Option generates PowerCenter mappings that embody the test conditions runs those mappings in PowerCenter.
3. PowerCenter accesses the data being tested and applies the tests to that data.
4. Test results (like pass or fail, test details, error rows, and data values) are stored in the Data Validation Option warehouse.
5. Users can view test results in reports that are generated from views on the warehouse.

Based on this setup, there is a clear relationship between the Data Validation Option environment (clients, warehouse, repository) and the PowerCenter environment (PowerCenter repository and PowerCenter Integration Service).

The Data Validation Option repository stores metadata needed for testing. This includes both table metadata and connection information so users can define tests, as well as the user-defined test definitions themselves. The table and connection information is imported into the Data Validation Option repository from associated PowerCenter repositories.

During initial configuration, at least one PowerCenter repository must be associated with a Data Validation Option repository. There can be a many-to-one relationship between PowerCenter repositories and a Data Validation Option repository.

This means that one Data Validation Option repository can import metadata from more than one PowerCenter repository, but there are issues to consider.

Execution Repository

For a given Data Validation Option user in a Data Validation Option repository, only one repository can be defined as the target repository. The target repository is where Data Validation Option-generated mappings, sessions, and so on, will be sent for execution. The target repository is typically (but not necessarily) the first repository added to the Data Validation Option configuration.

Execution PowerCenter Integration Service

When the target repository is added, a PowerCenter Integration Service is associated with it. The PowerCenter Integration Service will run all Data Validation Option jobs sent to the target repository. If the PowerCenter Integration Service also runs other jobs, for example, ETL jobs executed by developers, then the load on that service will increase with the new Data Validation Option jobs. In most cases, this is not a problem, but if the PowerCenter Integration Service is already running at capacity on the hardware, then additional CPU or memory resources will be needed, or a different PowerCenter Integration Service will need to be selected.

Metadata Collision

When metadata (source and target definitions and connection information) are imported from multiple PowerCenter repositories into a single Data Validation Option repository, there is the potential for metadata collision.

The collision could be from duplicate source or target names, or duplicate connection names across the repositories. For source or targets definitions, duplicate names do not necessarily mean duplicate definitions because the full path from the source repository is carried forward into the Data Validation Option repository. But, users who are not careful may choose the wrong source or target by not reading the full path and name and thus inadvertently introduce errors in their tests.

Additionally, a given PowerCenter repository can contain hundreds of connections and thousands of sources and targets. Importing objects from multiple PowerCenter repositories into a single Data Validation Option repository can become hard to manage if care is not taken to only bring in (and only refresh) specific folders from the associated repositories.

Setting Up Data Validation Option and PowerCenter Configuration

There are several ways to set up Data Validation Option and PowerCenter. The following are some of the methods:

- One-to-one relationship
- Many-to-one relationship
- Dedicated Data Validation Option and PowerCenter environment

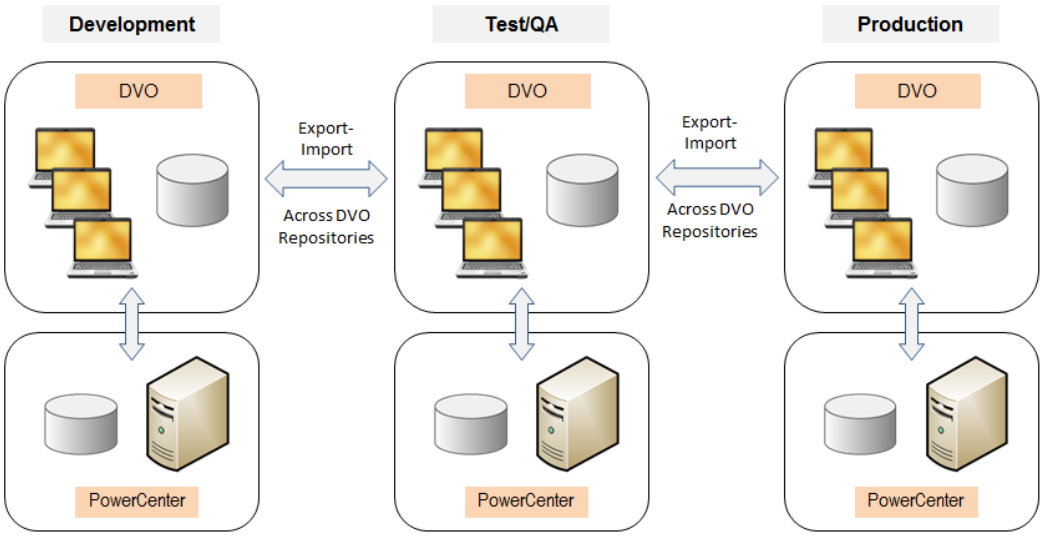
One-to-One Relationship

The most common setup is to have a one-to-one relationship between a Data Validation Option repository and a PowerCenter repository. For example, if a PowerCenter environment has development, test, and production repositories, then defining parallel development, test, and production Data Validation Option repositories makes sense.

This eliminates issues of metadata collision from multiple PowerCenter repositories. Also, any changes to table definitions or connections made in the PowerCenter environment can be imported directly into the Data Validation Option repository.

Additionally, this allows a simple user workflow with Data Validation Option across repositories, similar to that of PowerCenter. For example, a user can export or import metadata as needed across environments. Data Validation Option jobs will add load to each PowerCenter Integration Service in this setup.

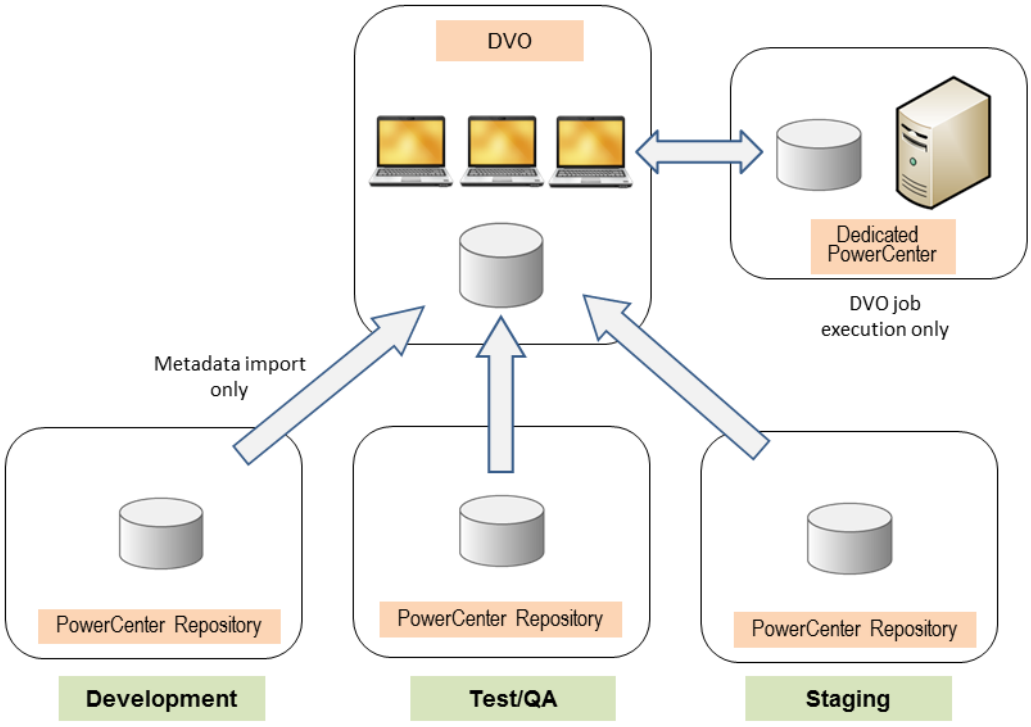
The following image shows a typical one-to-one relationship where each Data Validation Option environment is associated with a single PowerCenter environment:



Many-to-One Relationship

Another configuration between PowerCenter and Data Validation Option is to have a many-to-one relationship, with Data Validation Option having a dedicated PowerCenter execution environment.

The following image illustrates this configuration:



In this scenario, there is no need to migrate Data Validation Option metadata across Data Validation Option repositories. Also, the dedicated PowerCenter environment for executing Data Validation Option jobs means there is no

additional load placed on the regular development, test, and staging PowerCenter environments used for ETL. But, care must be taken when importing metadata from the multiple PowerCenter repositories to avoid metadata collision and to ensure that all metadata is kept updated from across all associated PowerCenter repositories. Also, note that connections from all associated PowerCenter repositories will be imported so there may be a very large number of connections to choose from for the Data Validation Option user.

Folders (or users) can be used in the Data Validation Option environment to separate the work (table pairs, single tables, views, and so on) associated with each of the development, test, and staging PowerCenter repositories.

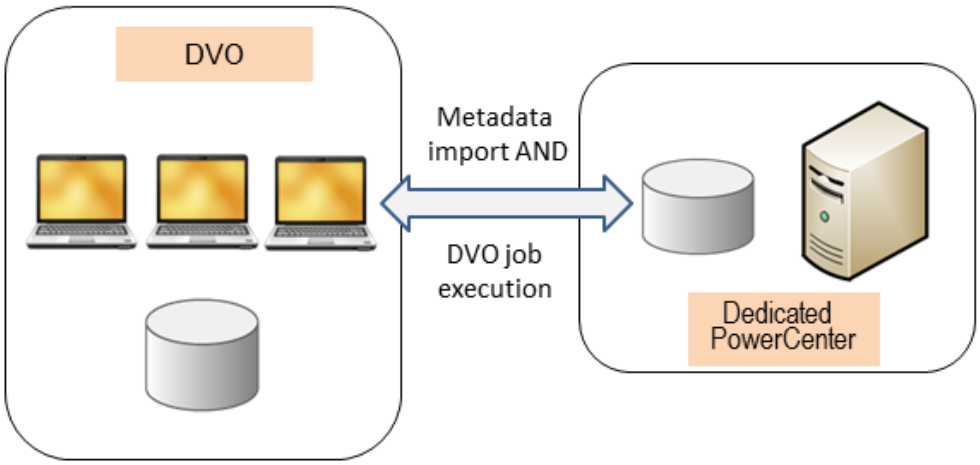
For example, unit testing associated with development, regression testing associated with QA, integration testing associated with staging, and so on.

Dedicated Data Validation Option and PowerCenter Environment

This configuration option is conceptually the simplest, and involves setting up a Data Validation Option and a dedicated PowerCenter environment. The PowerCenter environment supplies all the needed metadata to Data Validation Option and is also the execution environment.

While conceptually simple, this configuration has both advantages and disadvantages. By having a dedicated PowerCenter environment, there is no load placed on the other PowerCenter environments used by developers, and so on. Given that all metadata is coming from a single PowerCenter repository, there is no risk of metadata collision.

The following image shows the relationship between Data Validation Option and a dedicated PowerCenter environment:



However, all metadata must be imported into the PowerCenter environment directly from the sources, or exported from other PC repositories and imported into this one. Additionally, all necessary connections must be created in the PowerCenter repository. Both connections and table metadata must be kept update to date in the PowerCenter repository.

The benefit of this configuration is that only those tables and connections that are needed for testing are exposed to testers. This connection may also use different database accounts (versus using normal database connections), and

allows for different security to be applied to database connections as compared to what developers have in the PowerCenter repositories.

PowerCenter Repository Setup

While Data Validation Option tests are created with the Data Validation Client tool, the test execution is handled by PowerCenter. Data Validation Option will generate the necessary mappings, sessions, and workflows to accomplish the tests and write these objects in the target folder specified in the Data Validation Option configuration for target repository.

The following are issues to consider when setting up Data Validation Option:

- Version controlled repository
- Target folder setup for users
- PowerCenter Integration service
- Domain configuration

Version Controlled Repository

If the Data Validation Option target repository is an existing, versioned PowerCenter repository, then the following should be kept in mind:

- Data Validation Option creates at least five different PowerCenter metadata objects for each test that is created and executed.
- These include a mapping, session, workflow, source, and target.
- As Data Validation Option usage grows, the number of generated objects in the target repository will increase as well.
- Each time a table pair or single table is changed and executed, a new version of the associated mapping, session, and workflow is generated and written to the Data Validation Option repository
- In a versioned repository, a new version of each of these objects is written. It is possible to end up with hundreds of objects, each with dozens of versions in the target folder.
- Data Validation Option regenerates the mapping, session, and workflow when changes are made, so there is no need to maintain unnecessary older object versions in the PowerCenter repository.

Target Folder Setup for Users

Each Data Validation Option user can be assigned their own target folder in the target PowerCenter repository. Having separate target folders will help organize the Data Validation Option generated metadata and allows for easier maintenance.

For example, when an employee leaves the company, that employee's target folder can be purged without impacting other users.

On the other hand, if multiple users share the same target folder, it is nearly impossible to determine which objects are owned by different users. Additionally, in highly active Data Validation Option environments, multiple users might import the same objects at the same time. This causes the test for one user to fail with a PowerCenter locking error message.

PowerCenter Integration Service

If Data Validation Option has its own PowerCenter Integration Service (for example, many-to-one or Dedicated Data Validation Option-PowerCenter environment) for executing tests, then the following should be considered:

- The infa_shared directory can be placed into a different location than the PowerCenter Integration Services used for PowerCenter development. This allows easier purging of log and data files, which also can be done in a more aggressive time frame than a typical PowerCenter Integration Service environment.
- This allows changes to other PowerCenter Integration Service settings or custom properties, which might not be possible in a shared environment.
- This PowerCenter Integration Service can be set up to use an Informatica grid, or can be assigned to specific Informatica nodes to avoid conflicts with other Informatica application services.

Domain Configuration

Setting up Data Validation Option with its own PowerCenter repository and PowerCenter Integration Service allows for greater security on who can use Data Validation Option and what metadata they can access.

Data Validation Option users who are not PowerCenter users can be granted restricted access to primary PowerCenter repositories, which they will need to access source and target metadata

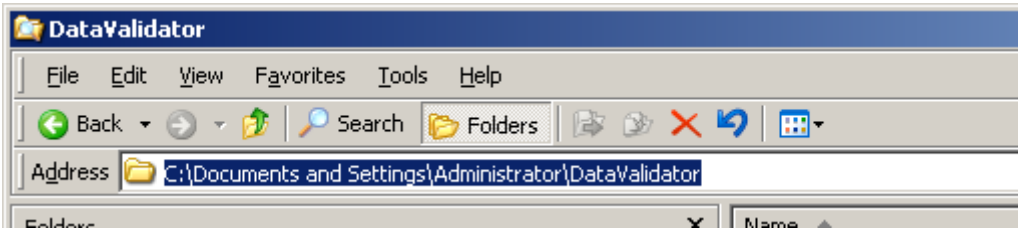
Since new database connections are created in the new Data Validation Option PowerCenter repository, these connections can have increased security privileges to restrict data access to groups of users.

Data Validation Client Overview

The Data Validation Client is primarily designed with a single-installation, single-environment usage in mind. Since most customers have multiple environments in which they want to run tests, user desktops need to be configured to support these multiple environments.

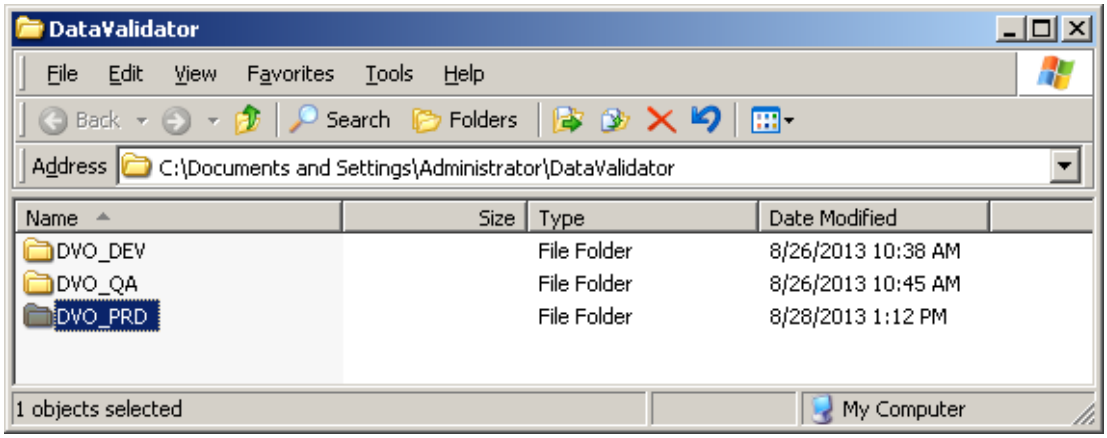
When a user installs the Data Validation Client onto their machine, a DataValidator directory is created under their Window's user area. In Windows 7, this is typically located in C:\Users\<username>

For example C:\Users\<username>\DataValidator, as shown in the following image:



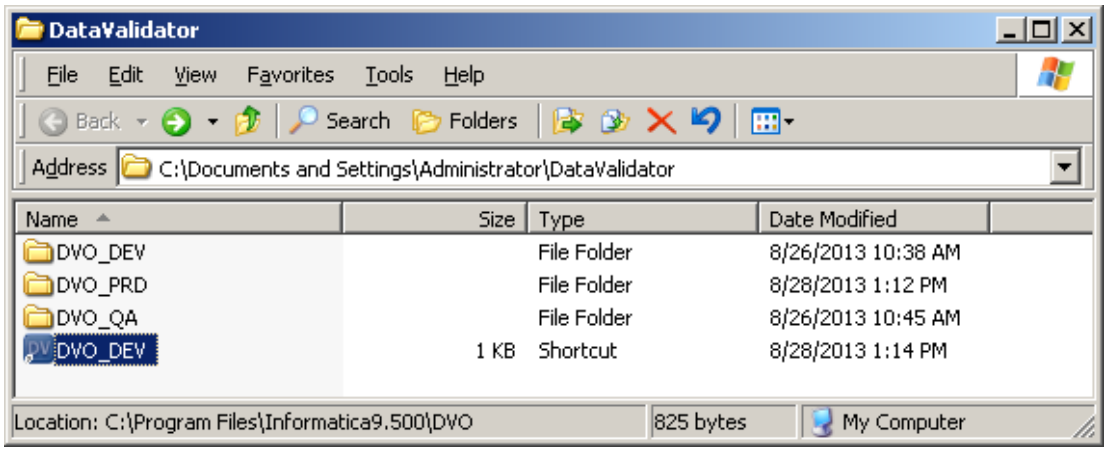
For the Data Validation Client to access multiple Data Validation Option repositories, the preference file that is associated with the Data Validation Option environment needs to be provided when the Data Validation Client is started. So, under this directory, there should be a subdirectory for each environment a user would need access to.

The image below shows a sample subdirectory:



To start Data Validation Option properly, a shortcut to Data Validation Client.exe should be created for each environment.

The following image shows a sample shortcut:



This shortcut should run Data Validation Client.exe and pass in the directory associated with the environment that needs to be accessed. In this directory, Data Validation Option would create a preference file and support directories that it needs to function properly.

A sample shortcut and path is: "C:\Program Files\Informatica9.500\Data Validation Option \Data Validation Client.exe" "C:\Documents and Settings\Administrator\DataValidator\Data Validation Option _QA"

Note: Quotes are needed if you have spaces in your directory path.

Suggested Naming Conventions

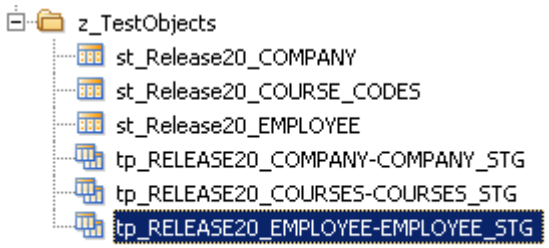
While Data Validation Option does not require any specific naming standards for metadata objects, having standard conventions for object names makes managing repository objects easier.

The following table a suggested naming convention that has been used at some customer sites:

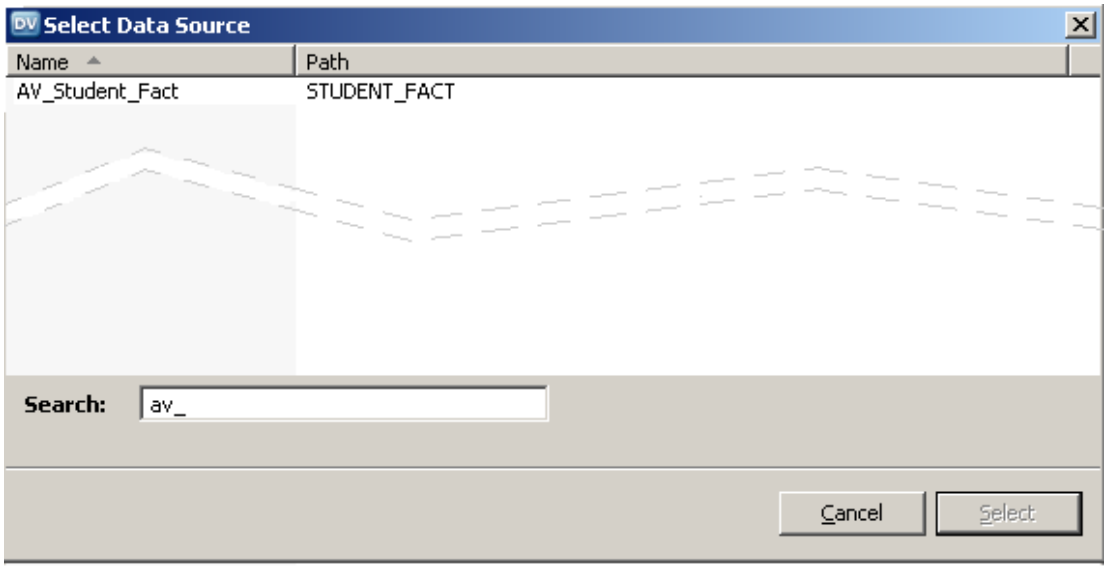
OBJECT TYPE	SUGGESTED NAMING STANDARD
Table Pair	tp_<project>_<TableA_name>-<TableB_name>
Single Table Constraint	st_<project>_<Table_name>
Lookup View	lv_<project>_<source_table>-<lookup_table>
SQL View	sv_<project>_alias_name
Join View	jv_<project>_alias_name
Aggregate View	av_<project>_alias_name

Prefixing objects makes it very easy to identify them in the GUI, or filter them when possible.

For example, in the folder shown in the image below, it is easy to separate the table pairs from the single tables:



It is also easy to filter to find a view in the data source selection dialog box, as shown in the image below:



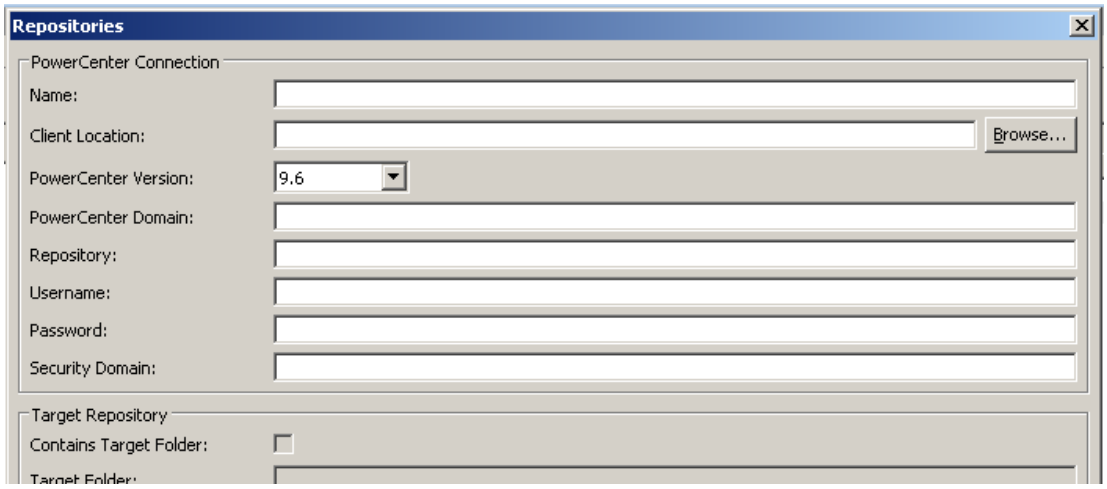
User Accounts

To start, each user must have their own Data Validation Option account. Even though it is tempting to share accounts among users to work around the lack of a true shared multiuser repository, this is not recommended as it opens up the potential for repository corruption. If multiple Data Validation Clients are logged in with the same Data Validation Option user credentials, they could overwrite objects at the same time.

Each Data Validation Option user is associated with a corresponding PowerCenter user for each PowerCenter repository that is accessed by that user.

For example, in the image below, User1 has access to metadata from two PowerCenter repositories, **DEV Repo**and **QA Repo**:

Each added repository is configured with a user name, a password, and some other values. For example, see the following dialog box:



The user name and password are the same as the user's user name and password for the PowerCenter domain for that particular PowerCenter repository. The privileges provided to the user in the PowerCenter domain determine the folders and objects in the PowerCenter repository that can be accessed by the user. This means that the metadata (like source and targets) that can be imported into Data Validation Option for testing is based on the permissions that user has in that PowerCenter repository

User Security

Data Validation Option supports two types of user security. They are:

- Native security (default)
- Domain authentication

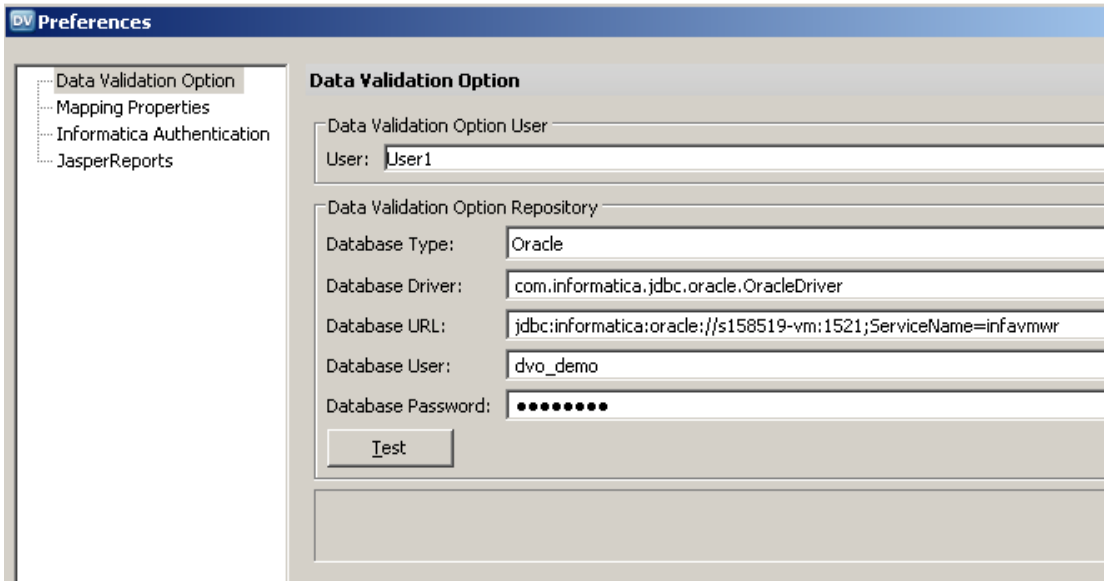
Native Security

In native security, the user name (along with other preference settings) for a given Data Validation Client is stored in the preferences.xmlfile for that client.

There is no password required for native users. When the Data Validation Client starts, it reads this file and sets the preferences accordingly in the Data Validation Client.

This information can be seen in the .**File** > **Settings** > **Preferences**

For example, see the following image:

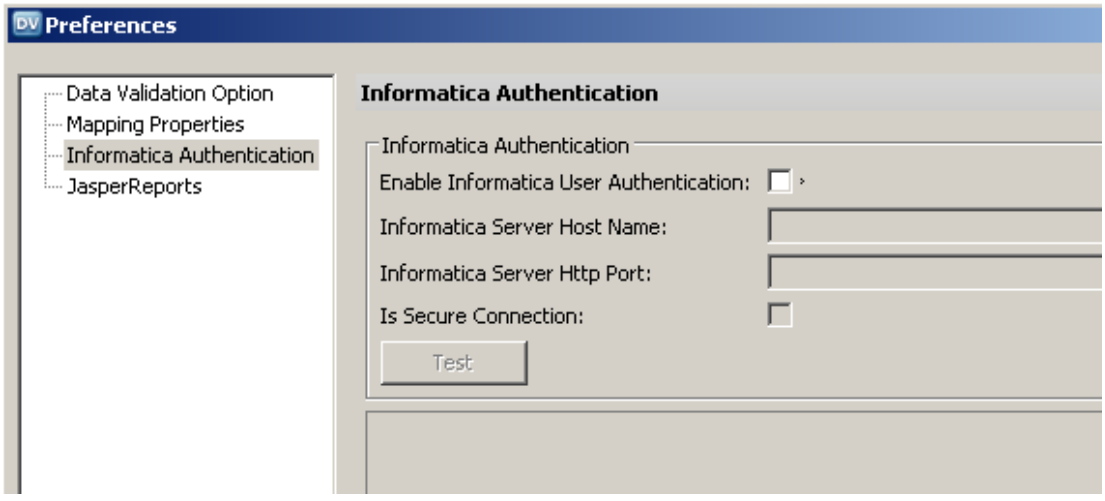


This model is used by many Data Validation Option customers because it is simple and works. But if a company has security concerns or requirements, then this model, particularly with users and no passwords, falls short.

Domain Authentication

Domain authentication allows Data Validation Option users to be associated with users in a PowerCenter domain. It requires the Data Validation Option user to log in as that user when they start the Data Validation Client.

Domain Authentication is set via the **Informatica Authentication** panel in the **Preferences** dialog box, as shown in the following image:



Authentication can be set via any client associated with a given Data Validation Option repository. However, after it is enabled in one client, it is enabled in all clients; that is, it is a repository setting. All users for a given Data Validation Option repository must connect via native security or domain authentication.

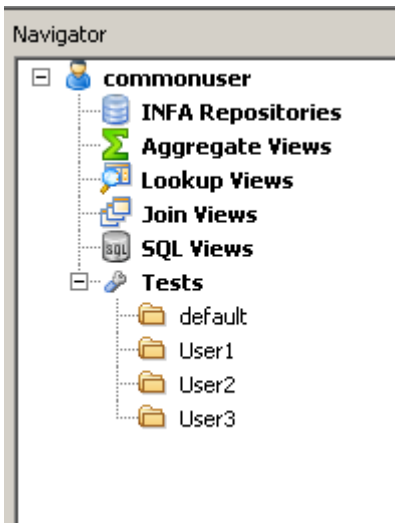
Strategies for User Collaboration

Given that the Data Validation Option repository is multiuser and unshared the question arises as to how to collaborate with other users.

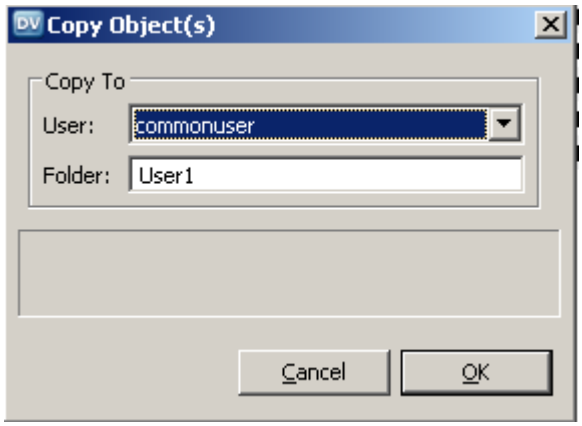
Imagine that there are three Data Validation Option users working together, say, User1, User2, and User3. Each has their own metadata (source and target definitions), Data Validation Option folders, and objects (like views, table pairs, single tables, and so on) that they have created.

If they are all working on the same project and want to create a common area to review and run their tests, they can create an additional user, say CommonUser, and use that to store and execute finished work.

For convenience, create folders in the CommonUser workspace for each of the other users. For example, create folders named User1, User2, and User3 as shown in the image below:



Objects can be copied across users via the option available at **right-click > Copy**. These folders provide an easy way to remember the destination folder of each user who will copy objects to the CommonUser, as shown in the image below:



After the objects are copied to the CommonUser workspace, they can be moved, copied or edited as needed.

If domain authentication is enabled, User1, User2 ,and User3 will need to have the password to CommonUser if they need to log in and modify anything in that workspace. They do not need the password if they are only copying to that user.

This approach (or slight variations) is used by a number of Data Validation Option customers to work around the lack of a truly multiuser shared repository.

Data Validation Option Performance in PowerCenter Environment

One performance consideration is to identify the resource requirements, like CPU, memory, and so on that Data Validation Option mappings and sessions will require in each PowerCenter environment. Another consideration is to ensure that the additional load will not be a problem for either the environment or other users in that environment.

Data Validation Option, on its own, does not place any load on your PowerCenter environment. Instead, the load on PowerCenter depends on how many jobs are run, how much data is tested, the kind of tests run, how things are configured, what else is running on the server, and so on. As such, users themselves decide the load placed on the PowerCenter environment by defining and executing jobs with Data Validation Option.

The following include ways to optimize performance that are directly supported by Data Validation Option:

- Sending WHERE clauses, counts, and aggregations into databases, where possible.
- Having presorted data, where possible, as joins on presorted data are significantly faster than those on unsorted data.
- Using data sampling (in database, if possible) to reduce the amount of data tested.

Performance Testing

The following tests are taken from a detailed white paper titled *Optimizing Testing Performance in the Data Validation Option* that is available from your Informatica account team. It covers performance issues and performance optimization techniques, and provides detailed execution benchmarks to help you understand the load that Data Validation Option places on the PowerCenter environment.

The following table lists the server configurations that all tests were performed on:

Component	Configuration
Operating system	Linux version 2.6.18-308.el5 (Red Hat 4.1.2-50)
CPU	AMD Opteron 6220 Processor
Speed	3000 MHz
Cores	4
Memory	128GB

Also,

- All tests used Oracle data for both tables, table A and table B, in the table pair.
- All tests used default configurations and were performed entirely in PowerCenter. That is, no database optimization, caching, sampling, or other performance enhancements were made.

The intention is to provide sample baseline performance numbers on a small server. Larger or more powerful servers and enabling Data Validation Option optimizations would likely result in significantly improved performance.

Methodology and Metrics Calculations

All tests were conducted five times, and the elapsed time for each test run was recorded. The lowest and highest times were discarded, and the average of the middle three were computed and rounded to the nearest second. This is the number displayed in the following tables under **Average Time**.

Rows/Second is shown to provide a normalized number to compare test performance across different row counts and test types. The ideal situation is to have a consistent (that is, flat) measure for rows per second as the number of rows increases for a given test scenario. This shows linear scalability for the system.

The joined column in the following tables indicates whether a join condition was defined across the table pair.

Count Tests

Count tests are common tests performed on data. They count all non null values in a column and provide a simple check to see if the expected number of values are present. In the following table, five count tests were performed on a table pair with 50 data columns. The performance is consistent, averaging about 57,000 rows per second across all tests, as shown in the following table:

Joined	Rows	Columns	Test Types	Average Time	Rows/Sec
No	1,000,000	50	5 Count	16 seconds	63,830
No	2,000,000	50	5 Count	36 seconds	56,604
No	5,000,000	50	5 Count	1 minutes 35 seconds	52,817
No	10,000,000	50	5 Count	3 minutes 1 second	55,351

Outer Value Tests

Outer value tests perform a full outer join across table A and table B in the table pair and display any orphan from either side. In general, outer value tests are performed across the key columns of the data sets and only one outer value is typically needed in given table pair. Thus, the following results show the performance for only one outer value test:

Joined	Rows	Columns	Test Types	Average Time	Rows/Sec
Yes	1,000,000	50	1 Outer Value	10 seconds	103,448
Yes	2,000,000	50	1 Outer Value	18 seconds	111,111
Yes	5,000,000	50	1 Outer Value	38 seconds	131,579
Yes	10,000,000	50	1 Outer Value	1 minute 26 seconds	116,279

Fifty Value Tests

This scenario contains 50 value tests, one count, and one outer value. This is a typical set of tests that can be automatically generated for a table pair in Data Validation Option. Note that the rows per second drops by about 10 times as compared to the 5-value test scenario. This is expected given the fixed capacity of the server and the 10-times

increase in the number of columns testing. But comparisons per second remains high. On average, comparisons per second remains slightly higher in the 50-value tests when compared to the 5-value test scenario.

Joined	Rows	Columns	Test Types	Average Time	Rows/ Second	Comparisons/ Second
Yes	1,000,000	50	Fifty value, one count, one outer value	3 minutes 49 seconds	4,367	218,340
Yes	2,000,000	50	Fifty value, one count, one outer value	12 min 14 sec	2,725	136,239
Yes	5,000,000	50	Fifty value, one count, one outer value	30 min 20 sec	2,747	137,337
Yes	10,000,000	50	Fifty value, one count, one outer value	53 min 29 sec	3,117	155,827

Authors

Brian Shepherd

Saeed Khan