HP Test Data Management

Software version: 1.10

Concepts Guide

Document release date: October 2010
Software release date: October 2010

Legal notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted rights legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Licensing

The use of HP products is governed by the terms and conditions of the applicable End User License Agreement (EULA).

Copyright notices

Trademark notices

Microsoft®, Windows®, Windows NT®, Windows XP®, and Windows Vista® are U.S. registered trademarks of Microsoft Corporation.

Java™ is a U.S. trademark of Sun Microsystems, Inc.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates.

UNIX® is a registered trademark of The Open Group.

Contents

About this document

HP Test Data Management provides powerful tools to design a test data management solution that extracts data from your production database for upload into a test database.

This guide is designed to help you understand the fundamental concepts behind HP Test Data Management.

This guide provides information about:

- test data management in general
- HP Test Data Management concepts

Intended audience

This guide is intended for:

- Test data developers

Prerequisites

Prerequisites for using this product include:

- Knowledge of the operating system
- Database knowledge
- Application knowledge

Related documentation

In addition to this guide, please refer to other documents for this product:

- *HP Test Data Management Installation Guide*

  Explains how to use the Installer to install the product.

- *HP Test Data Management Tutorial*

  Provides step-by-step instructions to build a sample project with business flows, deploy them, run them, and troubleshoot errors.

- *HP Test Data Management Designer User Guide*

  Explains how to use the Designer component to design, build, test, and deploy your test data management projects.

- *HP Test Data Management Web Console and Query Server User Guide*

  Explains how to use the Web Console component to run, monitor, and administer business flows that copy data to and from the database. This guide also explains how to install, configure, and use the query server to access data that has been extracted from the database.

- *HP Test Data Management Troubleshooting Guide*

  Explains how to diagnose and resolve errors, and provides a list of common errors and solutions.

- *HP Test Data Management Release Notes*

  Lists any items of importance that were not captured in the regular documentation.

The latest documentation for the most recent HP Test Data Management release can be found on:

http://support.openview.hp.com/selfsolve/manuals

Document conventions and symbols

Convention	Element
`[ ]`	Indicates that the enclosed element is optional and may be left out.
`{ }`	Indicates that you must specify one of the listed options.
|	Separates alternatives.

Convention	Element
<parameter_name>	You must supply a value for a variable parameter.
. . .	• Indicates a repetition of the preceding parameter. • Example continues after omitted lines.
Medium blue text: Figure 1	Cross-reference links and e-mail addresses
Medium blue, underlined text (http://www.hp.com)	Web site addresses
Bold	• Key names • GUI elements that are clicked or selected, such as menu and list items, buttons, and check boxes
Italics	Text emphasis
Monospace	• File and directory names • Text displayed on the screen, such as system output and application messages • Code syntax
Monospace, italic	You must supply a value. • Code variables • Command-line variables

△ CAUTION Indicates that failure to follow directions could result in damage to equipment or loss of data.

NOTE Provides additional information.

TIP Provides helpful hints and shortcuts.

RECOMMENDATION Provides guidance from HP for a best practice or for optimum performance.

Documentation updates

For documentation for all other versions of HP Test Data Management, you can go to:

http://support.openview.hp.com/selfsolve/manuals

NOTE This documentation is written to the latest patch version. If you have not installed the latest patch, there may be items in this documentation that do not apply to your environment.

Subscription service

HP strongly recommends that customers sign up online using the Subscriber's choice web site:

http://www.hp.com/go/e-updates

- Subscribing to this service provides you with e-mail updates on the latest product enhancements, versions of drivers, and firmware documentation updates as well as instant access to numerous other product resources.
- After signing up, you can quickly locate your products under Product Category.

Support

You can visit the HP Software Support web site at:

http://www.hp.com/go/hpsoftwaresupport

HP Software Support Online provides an efficient way to access interactive technical support tools. As a valued support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract.

To find more information about access levels and register for HP Passport, go to:

http://support.openview.hp.com/new_access_levels.jsp

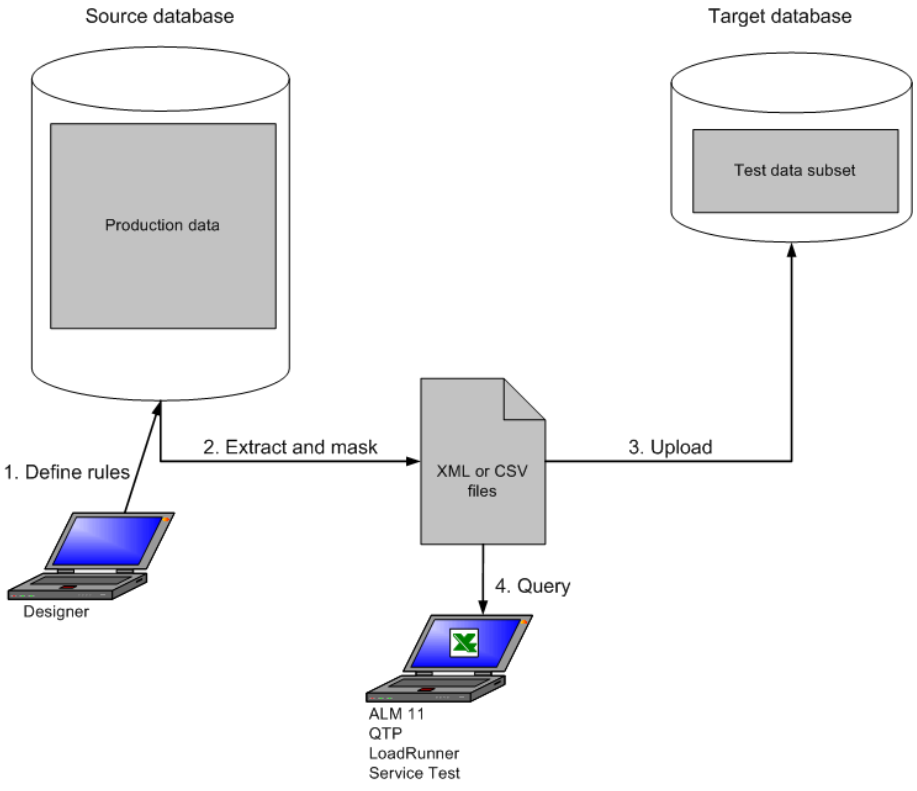
1 Test data management overview

HP Test Data Management provides powerful tools to design, deploy, and implement solutions that extract data from your production databases for the purposes of populating test databases.

This chapter includes:

Test data management overview

Figure 1 TDM Methodology



As shown in Figure 1, test data management is the process of:

1. Using rules to subset production data.
2. Masking sensitive data and extracting the subset from the production database to a structured file (XML or CSV).
3. Uploading the test data subset to a test database.

4 Querying the structured, extract files to create a test data input spreadsheet.

As part of the subsetting process, you need to perform the following functions:

- **Apply eligibility requirements to the data.** In most cases, you do not want all of the data from your production database. You need to define criteria that restrict the records copied. For example, you might only copy records that pertain to a certain period of time or customer.
- **Mask sensitive data.** Typically, your test database is not as secure as your production database. Hence, you need to mask any sensitive data, such as names, addresses, phone numbers, social security numbers, and so on. HP Test Data Management provides a variety of masking capabilities to help you protect confidential data.

Copying qualified records out of your active database and masking them is only the first leg in the lifecycle of test data management.Once you have the desired subset of data in a data extract file, you can upload from the file to a compatible test database as shown in Figure 1 (page 9). HP Test Data Management enables you to upload to heterogeneous databases, if necessary. For example, you may have extracted from a SQL Server database but need to upload to an Oracle database. You can also implement schema mapping in order to change the schema name when you load the data in the test database.

In addition to uploading the data from the extract file, you may also want to create a spreadsheet with some portion of the test data for use with other quality management tools. HP Test Data Management's query server enables you to quickly populate spreadsheets with test data from one or many database tables. Because you are using the extract file, the data you load into the spreadsheet is masked to your specifications and you can join tables, if necessary, without re-querying your source database.

Analyzing your requirements

Before building a test data management solution, you must consider your requirements and plan accordingly. Before you begin building, you should consider all of the following issues:

- **Whether to model the data.** In HP Test Data Management, you can identify tables for selection in two ways. You can model the tables and their relationships (model-based), or you can simply select the tables directly (schema-based). Depending upon your goals, you may choose either one of these approaches:
  - **Schema-based** extraction enables you to easily select the tables to copy and apply subsetting rules to them. While it accelerates the table selection process, a schema-based approach does not include an understanding of the relationships among tables. Hence, the burden is on the test data developer to manage the rules in such a way that no data is orphaned. For example, if they choose only 10 records from a parent table, the test data

developer must ensure that only records corresponding to those 10 are selected from any child tables. Another important consideration for a schema-based approach is that you must upload each table separately.

— A **model-based** approach enables you to copy tables while retaining knowledge of the relationships between them. The data model encapsulates not just the tables but their relationship to each other. Hence, in a model-based solution, HP Test Data Management will ensure that only related records are chosen from any child tables. The test data developer need not manage that aspect themselves. Furthermore, all of the selected tables are uploaded at once rather than singly. The main drawback to a model-based solution is that it typically takes longer to select the tables and build the model because you must also include information about how the tables relate to each other.

- **Upload requirements.** You can upload the test data to a database that is the same or different from the original production database. For example, you might copy from a SQL Server database and upload to an Oracle database. You need to understand the differences in behavior between the disparate databases in this case.
- **How existing data might need to be transformed.** If you are copying data that requires transformation, such as data masking, you need to factor that into your solution as well.

Schema-based extraction requirements analysis

For schema-based extraction, you must consider the following before proceeding to build your schema-based extraction solution:

- **The rules you need to apply to the tables in order to create a meaningful subset.** In all likelihood, you will not limit the records extracted from lookup tables as much as you will the records extracted from transactional tables. Typically, transactional tables contain larger amounts of data and therefore must be subsetted in the interests of creating a manageable test instance.

  When you restrict the records from a transactional table, you may also need to apply rules to some or all of its child tables. Otherwise, you may not get the detail records that you require. For example, suppose you have a parent table, ORDER_HEADER, with a child, ORDER_LINE. If you choose to restrict the records from ORDER_HEADER in some way, you must create a complementary rule on ORDER_LINE to ensure that you get child records from ORDER_LINE that match the parent records you selected from ORDER_HEADER.

  TIP If you are finding it difficult to manage the rules on your tables in a schema-based cartridge, you may want to consider a model-based approach. With a model, HP Test Data Management gains an understanding of the table relationships, which greatly simplifies the formulation of rules.

- **The database types in your environment.** You need to consider the database types with which you are working (Oracle, SQL Server, Sybase, DB2, or JDBC data sources) and whether you need to construct rules that can run

against multiple database types. For example, if you want your project to extract data from both an Oracle and a SQL Server database, you may need to have different versions of your rules for each database type.

Model-based extraction requirements analysis

For model-based extraction, you must also gather and analyze information about the data model and business rules. For example, if you are working with an order and billing management system, you may want to extract only a subset of the tables, such as order entry tables, from the larger application.

Before proceeding to build your model-based extraction solution, you need a good understanding of all of the following:

- **The business entities with which users work.** You must understand how users look at the data. For example, if dealing with an order processing system, the main business entity is an order. Your extraction solution should be designed to reflect these business entities.
- **How the business entities are stored in tables.** Before you begin modelling your extracted files in Designer, you must understand which tables are required for the business entities involved and how those tables are related to one another. To return to the example of an order processing system, you would need to know where the various aspects of a customer order are stored in database tables and how those tables are related to one another (for example, what, if any, foreign keys are used to relate tables to one another). You may find it useful to create an entity relationship diagram (ERD), if you do not already have one.
- **The database types in your environment.** You need to consider the database types with which you are working (Oracle, SQL Server, Sybase, DB2, or JDBC data sources) and whether you need to construct a model that can run against multiple database types. For example, if you want your project to extract data from both an Oracle and a SQL Server database, you may need to have different versions of your rules for each database type.

2 Data selection and extraction

When extracting data, an algorithm must first select the data eligible for extraction. Second, the selected data is actually extracted using an extraction method. HP Test Data Management selects data using its standard or advanced selection algorithm and then moves it using its transactional method.

This chapter includes:

Standard selection

Standard selection closely follows the tree model. Starting at the top of the tree, it populates one selection table for each node in the tree. Standard selection does not support the case where a child has multiple parents. Therefore, multiple uses of the same table can break the tree model, if each use of the table does not map to a disjoint set of rows.

- **Driving table.** Most of the selection logic is evaluated in a single query during the population of the driving selection table. This query evaluates all rules that do not have eligibility analytics enabled. The rules on child or grandchild table are evaluated by including them in a subquery to the child table.
- **Child tables.** After the driving table rows are populated, each child table is populated in top-down order. For example, suppose that ORDERS is the driving table and its selection table is already populated. If the table hierarchy were ORDERS, ITEMS, and LEDGERS, the ITEMS selection table (ITEMS_SEL) would be populated next and then the LEDGERS selection table (LEDGERS_SEL).

  The child table is populated by a join between the parent table, the parent selection table, and the base table, as well as any conditional relationships. For example, ITEMS_SEL is populated by joining the driving table (OLTP.ORDERS), the driving table's selection table (OBT_IF.ORDERS_SEL), and the active child table (OLTP.ITEMS).

  NOTE No special handling of null foreign keys is performed. If a child has a null value for its foreign key, it is not related to the parent.

- **Conditional relationships.** A conditional relationship on a virtual foreign key adds an additional condition on the relationship between two tables. For example, if the model defines a relationship of `ORDERS.ORDID = ITEMS.ORDID`, it is possible to add a constraint or filter on the relationship:

  ```
  ITEMS.PARENT_ITEM_ID is null
  ```

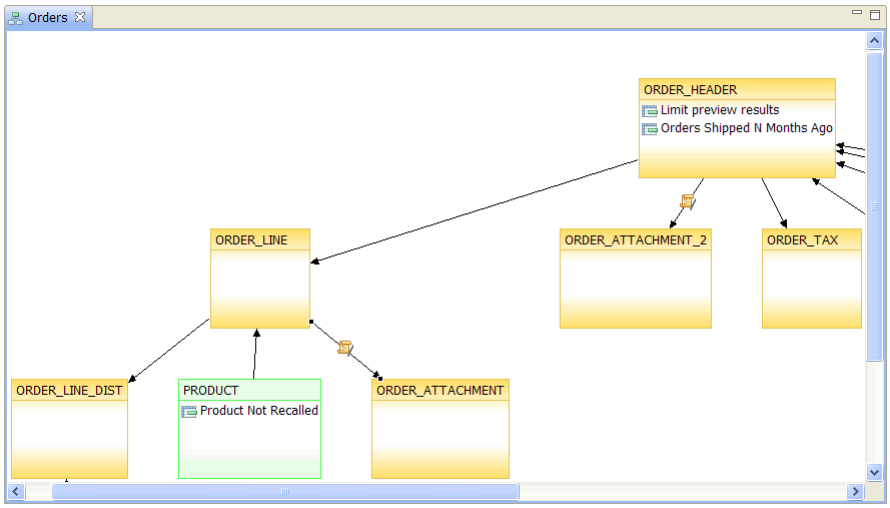
  Any rows which do not pass the original relationship and then the added conditional relationship are not children of the parent table.

- **Multiple table uses.** Creating additional instances of a table in a model indicates that the table is used to store different types of data. Each such instance represents a separate, logical table with its own selection table (one for each node in the model). The selection tables are populated in the order in which they appear in the model.

  With multiple table uses, you must ensure the rows do not overlap, that is they must be disjoint. Otherwise, you will encounter unique constraint errors when moving the data. You can avoid row overlap by creating conditional relationships.

  For example (Figure 2), suppose that you have an ORDER_ATTACHMENT table that contains attachments for both ORDER_HEADER and ORDER_LINE. The ATTTYPE column contains OL or OH to indicate whether the row pertains to ORDER_HEADER or ORDER_LINE.

Figure 2 Multiple table use in a model



To avoid overlapping rows in this case, you could add two conditional relationships. The relationship between ORDER_LINE and ORDER_ATTACHMENT is:

```
${FK_ALIAS}.ATTTYPE = 'OL'
```

The relationship between ORDER_HEADER and ORDER_ATTACHMENT_2 is:

```
${FK_ALIAS}.ATTTYPE = 'OH'
```

When to use

- **When using a schema-based cartridge.** Schema-based cartridges can only use standard selection.
- **When not using an Oracle database.** Advanced selection only works with Oracle databases. Hence, if you are using any other data source, you must use standard selection.
- **When performance is a concern.** Because standard selection does not attempt to discover interrelated rows, it typically runs faster than advanced selection.

Advanced selection

For model-based cartridges, you can choose advanced selection. Advanced selection is a method of data selection that discovers all of the interrelated rows from multiple tables and conceptually places them in the same application partition for extraction. Because advanced selection discovers related rows, it is a powerful method for selecting data for extraction.

NOTE Application partitioning is a concept unique to HP Database Test Management. This notion of partitioning contrasts with the more common table partitioning offered by the database management software. A table partition is only defined in one table. It does not contain all related rows from multiple tables. In order to achieve application integrity during and after data extraction, identifying application partitions is key and table partitioning is often insufficient in this regard.

In advanced selection, the model is considered as a standard entity-relationship model. You can select multiple driving tables for advanced selection. Furthermore, you need not ensure disjointedness for multiple table uses and there is no restriction on null foreign key values.

- **Driving table.** In advanced selection, you select as many driving tables as necessary to drive selection. You are not restricted to one driving table as in standard selection.
- **Child tables.** In advanced selection, since the model is viewed as an entity-relationship diagram with potentially many driving tables, the model is not a tree. It has no hierarchical predecessors/successors like you see in tree diagrams. Even though the model editor presents as a tree, this visualization does not apply to advanced selection. Once the driving set of rows is identified, advanced selection follows the relationships in the model to spread the eligibility to other tables. In the process of spreading the selections, the application restricting conditions are evaluated, which can keep or drop a selection.
- **Conditional relationship.** For advanced selection, a condition in a relationship is considered as part of the relationship. In entity-relationship modeling, relationships do not influence the definitions of entities.

For example, adding a condition on the relationship between ORDER_HEADER and ORDER_LINE does not influence which rows are considered part of the ORDER_LINE (or ORDER_HEADER) entity. The simple rule of thumb for designing for advanced selection is to restrict the entity and then add the rule to the entity, rather than the relationship.

- **Multiple table use.** Multiple table use is the key to treating a tree model as an entity relationship model. If a table appears twice in the model, advanced selection can find loops in rows that start from one table, follow relationships in the model to other tables, and eventually return to the starting table. For example, suppose the table ORDERS joins to two tables, NOTES and ITEMS. Now further suppose that we start from a row in ORDERS and that row joins to a row in the ITEMS table. The row in ITEMS in turn joins to a row in NOTES_2, which is a second use of the table NOTES. The row in NOTES_2 is also a row in NOTES, which leads back to the ORDERS table, where the traversal originated.

  If, upon returning to ORDERS, the rows have already been found eligible, nothing changes. However, if new rows are discovered upon the return to ORDERS, these rows are marked eligible for extraction and the looping from ORDERS to ITEMS to NOTES_2/NOTES and back to ORDERS must continue. This kind of data looping is sometimes referred to as chaining.

When to use

- **When using a model-based cartridge.** Advanced selection only applies to model-based cartridges.
- **When application integrity is complex.** If you have a strong requirement to maintain application integrity and the data relationships are more complex, advanced selection is a good option. The application can actually run correctly using the extracted data from advanced selection and, after data extraction, the application is still able to run correctly. Since advanced selection identifies application partitions which encompass all the necessary data for the application, it ensures application integrity.
- **When using an Oracle database.** Advanced selection requires an Oracle database. You cannot use it with any other data source.

Eligibility analytics

If you choose to turn them on, eligibility analytics show you which rows have been selected for movement and which have been excluded. For excluded rows, eligibility analytics will also show you which of your rules caused the exclusion.

Eligibility analytics provide a powerful decision making tool for database extraction. You can review the analytics prior to data movement in order to determine whether you want to proceed or abort the data movement. For example, you might review the analytics to determine if too many or too few rows are

selected for movement. If your review indicates that too many or too few rows are selected for movement, you can abort the operation. Otherwise, you can continue it.

TIP Enabling eligibility analytics can significantly affect the performance of data selection, depending upon the size of your data set. You should consider the possible performance trade-off before enabling eligibility analytics and only enable them if you expect to make use of them.

3 Models, parameters, cartridges, and business flows

In Designer, you use models, parameters, cartridges, and business flows to specify:

- which data to move
- where to move the data
- how to move the data
- what, if any, additional logic to apply

This chapter includes:

Lifecycle

Implementing an extraction solution is an iterative process consisting of multiple, cyclical phases. It requires an understanding of the semantics of the parts that are moved between the active database and the extract data store:

1. Before you begin, analyze the requirements of your users and their environment. Refer to Analyzing your requirements (page 10).
2. Capture metadata about the tables (tables, synonyms, views) to be extracted.
3. Choose whether to create a schema- or model-based cartridge.
4. For a schema-based cartridge:
   a. Create an extraction cartridge. Refer to Cartridges (page 25).
   b. Add tables to the cartridge.
   c. Create rules for the tables to refine which records are eligible for extraction and upload. Refer to Rules (page 23).
5. For a model-based cartridge:
   a. Design a model representing the list of tables to be extracted and uploaded, and their relationships. Refer to Data modeling (page 21).
   b. Create rules for the model tables to refine which records are eligible for extraction and upload. Refer to Rules (page 23).

c Test the model and rules by previewing their execution and navigating the results to understand what data would be copied under this model.

d Create an extraction cartridge, which encapsulates a particular usage of the model you just created. The same model can be reused in multiple cartridges. Refer to Cartridges (page 25).

e Test the cartridge by previewing its execution and navigating the results to understand what data would be extracted under this cartridge.

6 Create a business flow that specifies the sequence and dependencies between one or many cartridges and other activities required to accomplish an extraction solution. Refer to Business flows (page 25).

7 Deploy the business flow to the server where the active database resides.

8 Ensure that the latest version of the business flow you need to run has been deployed.

9 Ensure that the configuration parameters are set to the appropriate values. For example, if you are extracting your data to XML, and you want the XML compressed and zipped, you must set the Compression Algorithm parameter for the business flow to GZIP.

10 Launch the job from the Web Console or the command line.

- Select the desired job, set the runtime parameters to the appropriate values, and launch the job.
- From the command line, review the syntax, ensure you have the required parameter values and environment information, and launch the job.

TIP Ensure that you have enough disk space available before running jobs.

11 Monitor the progress of the job.

12 Repeat this procedure as many times as necessary. For example, when the active database changes or new test eligibility rules come into play, you typically need to update your project, redeploy your revised business flows, and run them.

In HP Test Data Management, the Designer is the main component where an extraction solution is built. You design, test, and deploy your extraction solutions in Designer (step 1 through step 7 in the preceding procedure). You run the extraction solution typically by launching a job from the Web Console or the command line (step 8 in the preceding procedure).

See also *HP Test Data Management Designer User Guide*

HP Test Data Management Web Console and Query Server User Guide

HP Test Data Management Troubleshooting Guide

Data modeling

Based on your analysis of requirements and understanding of your database, you must first decide on the basis for choosing data for your extracted files:

- **Schema-based**. If you choose the schema-based approach, you simply create cartridges and choose tables or views for extraction. You can choose to extract all data, no data (table structure only), or selected data using a WHERE clause. Schema-based extraction is best for rapid table selection with minimal rules.
- **Model-based**. In the model-based approach, you design a data model in Designer. This data model governs the choice of data for inclusion in the test data subset. Unlike the schema-based approach, model-based extraction chooses data from the specified tables based upon the relationships and rules of the model. Model-based extraction is best when you have a number of complex rules. It also simplifies the upload process.

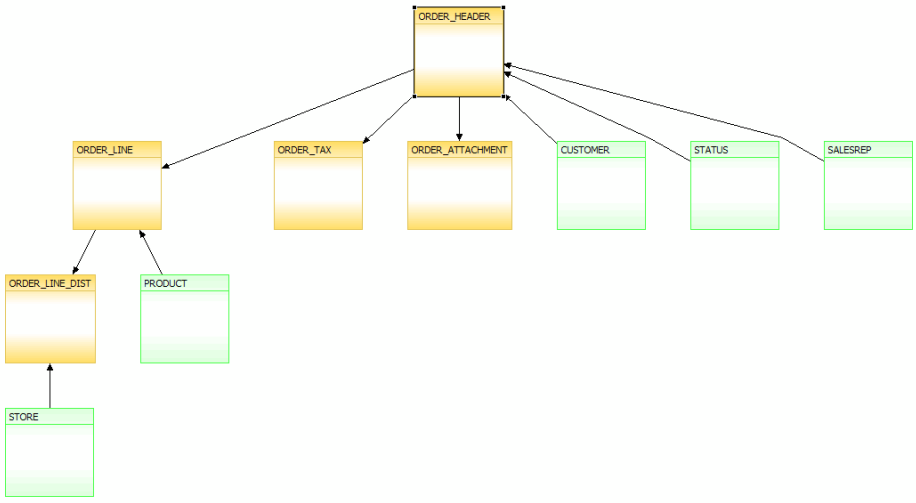
Models

In HP Test Data Management, a model, like the one in Figure 3, can be used to identify which data to include in a test database. The model graphically represents the tables (including views and synonyms) and how they are all linked through relationships.

Tables and table uses

In models, we use the term tables for the sake of simplicity, but it is important to understand that tables mean table uses or instances and do not necessarily have a one-to-one correspondence to the actual tables stored in the database. That is, you might have a model where the same table is used and appears multiple times. This concept of multiple table uses is described more completely in Multiple table use (page 23).

Figure 3 Sample model



A model can include tables of the following types:

- Driving table

The driving table is the root of the model hierarchy. Its relationship to the child tables drives the selection of transactions. Typically, a model only has one driving table, but, when using advanced selection, you can specify multiple driving tables. If you use advanced selection, you can mark any table as a driving table. When a table is selected as a driving table, its selections are then propagated to other tables, thus impacting their row selections.

In Figure 3, ORDER_HEADER is a driving table. The driving table can have zero to many children. The children may be transactional, lookup, or chaining tables. In Designer, child tables are related to the parent by foreign keys (database or virtual), optionally with a conditional relationship. A unique key (database or virtual) is the referencing constraint for the foreign key.

- Lookup table

  A lookup table contains helpful non-transaction information. You might need these lookup values present for the purposes of a rule or for the sake of making an extracted file more complete. For example, nontransaction information could be status definitions, or the names of the sales representatives. In Figure 3, STORE, PRODUCT, CUSTOMER, STATUS, and SALESREP are all lookup tables.

- Transactional table

  A transactional table contains information about the business transaction. For example, a transactional table might contain detailed tax or payment information related to each business transaction. In Figure 3, ORDER_LINE, ORDER_LINE_DIST, ORDER_TAX, and ORDER_ATTACHMENT are all transactional tables. Transactional tables are always part of the extracted files.

- Chaining table

  If a many to one (or many to many) relationship exists between a higher level table and a lower level table, the lower level table is labeled as a chaining table.

- Relationship

  All tables in a model are linked through relationships. These relationships are derived from captured foreign keys or defined by you at design time.

- Unique constraints

  A unique constraint is a column or a list of columns that enables you to identify rows in the database. For example, in a table of employees, employee numbers or Social Security Numbers are often unique constraints. Database unique constraints are defined in the database. If you have database unique constraints, HP Test Data Management discovers them. Thus, you can make use of these database constraints in Designer. If you do not have such database constraints, you can define your own, known as virtual unique constraints, in Designer. Virtual unique constraints only apply within HP Test Data Management. They are not added to your database and are only available for use inside of Designer. You must ensure yourself that virtual unique constraints are correctly defined. Furthermore, virtual constraints should be indexed, otherwise you risk significant performance issues.

TIP Unique keys perform the same function as primary keys and can be defined for any table in the model. Note, though, that unique keys are less strict than primary keys. For example, in a table, you could define a primary key on column A and a unique key on column B. In that case, column A is mandatory, but column B is not unless the column is defined as not null. The unique key enforces the rule that you cannot enter duplicate values for column B, but null values are not considered duplicates.

- Foreign key constraints

  A foreign key constraint is a column or a list of columns that enables you to relate rows in the database. For example, in a table of sales orders, employee number might be a foreign key that enables you to relate to the employee table. Database foreign key constraints are defined in the database. If you have database foreign key constraints, HP Test Data Management discovers them. Thus, you can make use of these database constraints in Designer. If you do not have such database constraints, you can define your own, known as virtual foreign key constraints, in Designer. Virtual foreign key constraints only apply within HP Test Data Management. They are not added to your database and are only available for use inside of Designer. You must ensure yourself that virtual foreign key constraints are correctly defined. Furthermore, virtual constraints should be indexed, otherwise you risk significant performance issues.

Multiple table use

In some cases, you may find it necessary to reference the same table more than once in your model. Each use of the table may have different rules associated with it.

In most situations, such multiple table use poses no problems. In some cases, though, you need to proceed with caution:

- If you have multiple table uses in your model that do not correspond to disjoint (non-overlapping) sets of rows in the table, you must use advanced selection. If the uses are not disjoint, standard selection can lead to duplicate data in your extraction data store. Hence, you can only use standard selection when the multiple uses are disjoint from each other.
- For advanced selection, you can use the same table in a model multiple times and each use can be of a different type (transactional, lookup, or chaining).

Rules

Each table within a cartridge or model can contain rules. Rules define the extent of the extraction operations or further refine the subset by excluding certain records. Rules typically fall into two broad categories:

- A rule can articulate a condition, often parameterized, to determine the rows eligible for extraction. This type of rule defines the scope of the data to be extracted. For example, if you have ten years of data and you want to use the last four of those years for testing purposes, you can define a rule that restricts the data extracted to the last four years.
- A rule can be a requirement or restriction for an application or business entity that ensures application data integrity. For example, a rule might specify that an order come from a particular customer or business unit in order to be included in the test subset.

You can create rules on any table and incorporate parameters into them.

Parameters

You can set up parameters that provide values within your project. In HP Test Data Management, you have three types of parameters:

- **Runtime parameters** have their values set at runtime by the user running the job. Runtime parameters tend to be best for operational values that tend to change with each execution of a job. For example, if your extract file is based on a specified cutoff date, you most likely need to update that date every time you run the job.
- **Configuration parameters** have their values set by an administrator (someone who has repository privileges from the Web Console) through the administrator interface. Typically, this type of parameter represents values that should be changed very infrequently, perhaps only at deployment time.
- **Dynamic parameters** have their values computed by every run of a Groovy script that executes at runtime. For example, this type of parameter can supply the type or version of a database or application, which can be obtained programmatically at runtime.

Regardless of a parameter's type, it can be referenced from several places within your project:

- WHERE clauses within your rules
- Groovy scripts in runtime parameter validations
- Groovy scripts in business flows
- SQL for runtime parameter list of values
- Interrupts in your business flows

Cartridges and business flows

A cartridge includes tables that you added directly via schema or tables that you modeled. You can incorporate the cartridge into a business flow that runs it, as well as other cartridges or logic you wish to implement.

Cartridges

Cartridges capture the rules to ensure integrity of the data. Cartridges allow test data to be extracted from production data based on the defined rules.

There are two types of cartridges:

- **Model-based cartridges** copy data, based upon a model, from an active database to an XML or CSV file from where it can be uploaded to a test database.
- **Schema-based database to file cartridges** extract selected tables, without a model, from an active database to file an XML or CSV file from where it can be uploaded to a test database.

A cartridge has a number of characteristics. Some of the more common ones are:

- Transactional tables to be extracted. In the case of model-based cartridges, transactional tables typically refer to the driving and transactional tables in the model.
- Lookup tables to be copied by the cartridge.
- Which rules to apply for subsetting the data to be extracted
- Columns to include or exclude
- Whether to apply data masking to a column and, if so, what type of masking

Business flows

You can deploy a cartridge by itself or as part of a business flow that performs some additional processing. A business flow is a series of activities, such as extraction operations and scripts, that run in a sequence. Business flows enable you to split your cartridge into multiple activities and insert Groovy scripts to perform additional processing in between those activities. You can create a business flow to run one or more cartridges and perform any additional operations you require. You construct business flows in the Designer with drag and drop gestures.

TIP You can even create a business flow that includes no cartridges at all. Such a business flow might only include Groovy scripts.

A business flow can also include activities other than cartridges that are necessary to accomplish an entire extraction solution. Such activities might include:

- Operating system commands, such as copying a file, deleting a file, compressing a file, or transferring a file (ftp).
- Generating an email message or other notification.

- Accomplishing a database operation, such as dropping indexes, making a tablespace read-only, or analyzing statistics for a table.

Groovy scripts

HP Test Data Management enables you to include Groovy code to extend your extraction solution for the following purposes:

- Groovy scripts can be inserted in business flows to perform additional processing.
- Interrupts and conditions in a business flow use Groovy code to define their branching criteria.
- Parameter validations are performed in Groovy code.
- Dynamic parameters are assigned their values by Groovy code.
- Groovy scripts can be set to execute before and after business flow deployment.
- Groovy scripts can be set to run before and after running a business flow.

The Groovy code in these places is processed as follows:

- The Groovy in dynamic parameters executes at runtime.
- The Groovy in validation and dynamic parameters runs just before execution (during job initialization).
- The Groovy in interrupts, conditions, and business flow scripts runs when the business flows are executing.

See also: *HP Test Data Management Designer User Guide*

4 Repository, deployment environments, and Web Console

After your cartridge or business flow is complete and tested, you are ready to try running it on the actual active database, outside of Designer. In order to run against your active database, you must first deploy the business flow or cartridge in the deployment environment.

Repository

The HP Test Data Management repository contains the metadata required to perform and monitor operations on the active (source) databases. The repository is stored in a database of your choice (any database, including the active database, or the embedded Java database installed with HP Test Data Management).

In addition to the repository, each active database in a standard environment contains interface schema. The interface schema contains some additional metadata use by HP Test Data Management. If you cannot have any schema installed in your active database for some reason, you can use a non-intrusive environment instead. Non-intrusive environments do not create any interface schema in the source database.

Deployment environments

An environment is a named deployment environment associated with a source (active) database. From an environment, users with the necessary privileges can run business flows against the source database. You can deploy as many environments as you wish, thus supporting multiple source databases from a single installation of HP Test Data Management.

HP Test Data Management can service several database environments, each with its own defined characteristics. Depending upon their assigned privileges, various users may access one or more of these environments to perform actions such as deploying business flows, running jobs, or administering the system.

Non-intrusive environments

When you create a standard deployment environment, Test Data Management creates interface schema in the source database. The interface schema stores metadata that enables Test Data Management to function more efficiently. In some cases, though, you may be unable or unwilling to install such schema into your databases. For example, suppose that your source database is an older, read-only system that does not support basic SQL statements, such as DELETE. On the other hand, you may simply prefer not to create additional schema in your production databases.

You can configure a non-intrusive environment that does not require any interface schema in the source database. The data is extracted without interface schema by using a generic JDBC driver.

Web Console

The Web Console provides a browser-based interface to the HP Test Data Management repository and your deployment environments. You use the Web Console to:

- Perform the initial setup of the repository.
- Define deployment environments.
- Create and provision users to perform actions in the Web Console
- Deploy, run, administer, and monitor your business flows
- Generate XML and CSV files for upload
- Upload from XML and CSV files to test database

Deployment options

When deploying your cartridge or business flow from Designer, you have three options:

- **Local deployment.** If Designer and runtime are installed on the same machine, you can do local deployment, and generate and deploy your cartridge or business flow on your local client. In this case, HP Test Data Management generates the deployment file and then immediately deploys it in the chosen local environment.
- **Remote deployment.** If Designer and runtime are remote from each other but still accessible across a network, you can deploy from Designer to the remote runtime client. In this case, Test Data Management generates the deployment files locally and moves the deployment files to the remote system, where they are deployed in the chosen environment.

- **Generate.** If Designer and runtime are remote from each other and you do not have network access to the remote machine, you must generate the deployment files in Designer without actually deploying them. After the files have been generated locally, you can hand them off to someone with access to the remote system. Once the deployment files are moved to a location accessible to the runtime system, they can be deployed to an environment using the Web Console.

Deployment and runtime history

When you create or modify a cartridge or business flow, you can associate a version with it. Only one version of a cartridge or business flow may be deployed at a time. If you deploy a new version of a cartridge or business flow that was previously deployed, the previous version is uninstalled before the new version is deployed.

HP Test Data Management maintains the runtime history of the cartridge or business flow across multiple deployments, unless you specifically indicate that you want the history dropped at deployment time. In general, it is a best practice to maintain the runtime history across deployments. Otherwise, when the history is dropped, you can no longer view status information for previous executions of the cartridge or business flow.

5 Next steps

Once you have completed this guide, here is a roadmap to follow to learn more about HP Test Data Management:

1. Install HP Test Data Management according to the instructions in *HP Test Data Management Installation Guide*.
2. Go through *HP Test Data Management Tutorial*. The tutorial enables you to get hands on with the product quickly and exposes you to many of the most commonly used features.
3. Review the advanced tutorials in the *HP Test Data Management Web Console and Query Server User Guide*.
4. Plan your extraction solution. Use the considerations in Analyzing your requirements (page 10) as a starting point and gather information about your own database environment before you begin.
5. Develop your extraction solution.
   - Refer to *HP Test Data Management Designer User Guide* and *HP Test Data Management Tutorial* to help you perform the necessary tasks for developing your project.
   - Refer to the *HP Test Data Management Troubleshooting Guide* to help you diagnose problems and resolve common issues.
6. When your business flow or cartridge is ready, generate and deploy it according to the information in *HP Test Data Management Designer User Guide*.
7. Run your deployed business flow.
   - Refer to the *HP Test Data Management Web Console and Query Server User Guide* to help you perform the necessary tasks for running your job from the Web Console or command line.
   - Refer to the *HP Test Data Management Troubleshooting Guide* to help you diagnose problems and resolve common issues.
8. As necessary, repeat step 5 through step 7 to modify your project, re-deploy the business flow, and run the job.

Glossary

active database

The database from which you plan to extract data. Typically, this database is your online transaction processing (OLTP) or production database. In a two-tiered configuration, the active database resides on tier one and is the source for data movement operations.

active environment

The Web Console views and acts upon only one environment at a time, the active environment. To switch the active environment, you use the Change Active option in the Web Console.

activity

In Designer, a component of a business flow, which is added by using the toolbar. Note, activities in a business flow are different from what you see at runtime and therefore do not necessarily map directly to what you see in Console.

advanced selection

A method of data selection that discovers all of the interrelated rows from multiple tables and conceptually places them in the same application partition for extraction.

annotation

In Designer, a comment associated with the project, or one of its objects or components. These comments are collected and published in a PDF file when you right click a project or business flow and select Generate Documentation.

application partitioning

The concept of partitioning related rows together during data selection, regardless of whether they are in one or more tables. Application partitioning is unique to HP Test Data Management and contrasts with the more common table partitioning offered by the database management software, which only groups related rows from one table.

business flow

A series of activities, such as extraction operations and scripts, that run in sequence. You build business flows in Designer.

business flow status

The Web Console shows the last run of each business flow. The states are Complete/Error/Running.

cartridge

An instance of model- or schema-based eligibility criteria used to copy data from one location to another. Cartridges capture the application and business rules to ensure referential integrity of the data. For any one model in your project, you may have many cartridges that use it.

chaining table

The lower level table in a many-to-one or a many-to-many relationship between higher level and lower level tables in the model hierarchy.

collection

The configuration of a directory location and file pattern to match a set of extracted XML files, thus allowing SQL access to the extracted data.

comma separated values (CSV) A database to file output format that stores the data as values separated by commas and a metadata file. Each line in the CSV file corresponds to a row in a table. Within a line, fields are separated by commas, each field belonging to one table column. CSV files provide a simple format that many applications can import.

command Command files or JavaScript files launched by the Web Console on your behalf with status displays.

condition In Designer, the way you branch your business flow to run or skip an activity based on some criteria.

configuration parameter A type of parameter that has its values set by an administrator (someone who has repository privileges from Console) through the administrator interface. Typically, this type of parameter represents values that should be changed very infrequently, perhaps only at deployment time.

console user The Web Console identifies individual users, who are distinct from database users. The properties for a Console user are User Name, Full Name, Password, Enabled, Description, Email, Phone, and Privileges.

console user name The login name associated with a Console user.

constraint A column or a list of columns that enables you to identify rows in the database and relate them to one another.

customization A change that an administrator or DBA makes to a project provided by a third party, typically for a packaged application like Oracle PeopleSoft or Oracle E-Business Suite. As long as the customization is allowable by the project, the user can merge the customization into newer revisions of the third party project.

customization mode A Designer mode that provides visual cues to indicate customizations in the model. In a project with locked files, customization mode is on by default, but you can toggle it on and off from the toolbar in the model editor.

data masking The process of replacing private or confidential data during movement with a specified mask. You can choose from pre-defined masks that are part of HP Test Data Management or create your own mask.

data movement The method used by HP Test Data Management to actually copy data.

database constraint A constraint that exists in the database and can be discovered and referenced from Designer.

database to file A movement in which data goes from an active database to a file (XML or CSV format).

Deployment Assistant The user interface component used to deploy or generate business flows. You invoke Deployment Assistant from within Designer.

description A technical description created by the developer for her own reference. These descriptions do not appear in the generated PDF file for the cartridge or business flow.

Designer The user interface component used to develop, test, and deploy your extraction solution. Designer is a powerful graphical development environment for extraction solutions.

driving table A driving object is a root of a model hierarchy. Its relationship to the child tables drives the selection of transactions.

dynamic list of values A list of values for a parameter that obtains its members from a SELECT statement that returns identifiers and labels.

dynamic parameter A type of parameter that has its value set by a Groovy script that runs at deployment time to obtain a value. For example, this type of parameter can supply the type or version of a database or application, which can be obtained programmatically at deployment time.

embedded repository A Java database, installed with HP Test Data Management, that can act as your repository database, where you store your HP Test Data Management metadata. Alternatively, your source database or another database can act as the repository database.

environment The source and (optional) target credentials against which you plan to run commands. You can define multiple environments within your installation to identify various source databases.

error One of the ways in which you can interrupt a business flow. Error indicates that the business flow failed for some reason.

exclusive rules One of the ways in which HP Test Data Management determines whether to include or exclude rows from the extract operation. Exclusive rules require all rows in the constraint table to match for inclusion. Exclusive rules exclude the instance if the condition on any child is false, like STATUS='CLOSED'.

exit One of the ways in which you can interrupt a business flow. You can exit successfully or with a warning.

export The way that you save an HP Test Data Management project to an exchange format (.hdp) from the File menu. See also *import*.

export data The way that a user can send data to CSV format from Preview using the toolbar item.

extract data store The location where the data is to be copied. Can be an XML or CVS file.

generate documentation The process of collecting and grouping all annotations into a PDF file that also describes the business flow or cartridge structure.

import The way that you transfer projects from exchange format (.hdp) into the Project Navigator.

inclusive rules One of the ways in which HP Test Data Management determines whether to include or exclude rows from the extract operation. Inclusive rules require only one row in the constraint table to match the rule and be included. Inclusive rules include the instance if the condition on any child is true, like PRODUCT_RECALLED='Y'.

interrupt The way to stop or pause a business flow (pause, error, exit with warning, exit successfully).

local cache A capture of the metadata for your databases, schemas, and tables used when working offline in Designer.

local deployment The generation and deployment of your cartridge or business flow to an environment on your local, Designer client. Deployment files are generated locally and then deployed to the designated, local environment.

lookup table A table that contains helpful non-transactional information. For example, non-transactional information could be status definitions, or the name of the sales representative.

model A model identifies the tables and table relationships representing a business entity or related business entities. A project can have multiple models. Each model contains a driving table and all of its child and descendent tables.

model compatibility Each model in your project can have one or more dynamic parameters associated with it to verify the compatibility with the target environment. If the compatibility parameter returns false, then the cartridge referencing the model will not deploy or run and throw an error. For example, the script could return false for Oracle 10.2 and true for Oracle 11.1 to indicate that a cartridge referencing the model can only deploy and run against Oracle 11.1.

model-based cartridge A cartridge that moves data based upon a defined data model with relationships. This type of cartridge is typically used for ongoing extract operations.

OLTP database The online transaction processing database that typically is your active or source database.

pause One of the ways in which you can interrupt a business flow. Pausing suspends the business flow while awaiting operator intervention.

query server The component that provides SQL access to XML or CSV files.

remote deployment The generation and deployment of your cartridge or business flow to an environment on a system that is remote from your Designer client. Deployment files are generated locally and then deployed to the designated, remote environment.

repository	The location that holds business flow metadata, product configuration data, and data collected during runtime. The repository can be located on your active database, another logical database, or can be embedded database.
rule	Qualifications added to the model in order to include or exclude data based on certain criteria. For example, you might add a rule to exclude from extracting any orders that are not yet closed.
runtime parameter	A type of parameter that has its values set by the operator executing the job in Console or on the command line. Typically, this type of parameter represents operational values that tend to change frequently and therefore need to be set each time the job is run.
schema-based cartridge	A cartridge that moves data based upon the database schema rather than a defined data model with relationships. This type of cartridge is typically used for database retirement or the cleanup of orphan tables.
selection	The form of data selection to use (standard or advanced) for choosing data. When creating a cartridge or adding it to a business flow, you must specify the selection method.
source	The location (database) from which you are copying or moving data.
standard selection	A method of data selection that restricts itself to the rows identified by the model. Unlike advanced selection, it does not attempt to traverse related rows across multiple tables.
table use	A database table, view, or synonym that is referenced in Designer, for example, in the model. The same table can be used multiple times in a model. For example, a table could be appear as a transactional table and a lookup table in the same model.
target	The location (XML) to which you are copying data.
transactional data movement	Transactional movement uses set-based data movement and is the default method of movement.
transactional table	A table that contains information about the business transaction. For example, a transactional table might contain detailed tax or payment information related to each business transaction.
unique identifiers (UIDs)	A 16 hexadecimal identifier calculated based on the content of a Designer file. This value is used to determine if the user has customized key pieces of a project.
virtual constraint	A constraint that you define in Designer that only exists within HP Test Data Management as opposed to a database constraint, which exists within the database.
Web Console	A browser-based interface where you can create and manage your deployment environments, and deploy, run, administer, and monitor your business flows.

Index

T

U

W


Java™
COMPATIBLE