

# Informatica TDM (1): architecture, the tool tour, discovery and subsetting

How Informatica TDM is built, a guided tour of Test Data Manager, then discovering sensitive data and keys and carving out a data subset.

## Informatica TDM: what it is and how it is built

Informatica Test Data Management, usually shortened to *TDM*, is easiest to work with once the map is clear: what the product is for, which pieces run behind the web page, where each kind of information is stored, and what order of work the product expects.

Almost every error message and permission problem is really a "this piece cannot reach that piece" problem, so knowing the pieces tells you where to look. For the screens, see [A guided tour of Test Data Manager](#).

## 4.1 Words to know first

### Test Data Manager (TDM)

The web page you work in. The same name is also used for the whole product, which is confusing. Here, "Test Data Manager" with spaces means the web client, and "TDM" means the whole product.

### Policy

A named bundle that says "these kinds of data must be protected, and this is how". It holds data domains and holds no actual data.

### Data domain

A group of columns that mean the same thing, for example "all columns that hold a Social Security number", found by looking at column names or at the values.

### **Project**

The workspace for one job: it holds your imported source tables, profiles, masking assignments, subset definitions and plans.

### **Plan**

The final recipe: which tables, which rules, from which connection to which connection. You turn a plan into a workflow and run it.

### **Workflow**

The runnable job that actually copies and changes data. TDM generates it from a plan, and a PowerCenter service runs it.

### **Application service**

A background program in the Informatica installation that does one job, such as running profiles or running workflows. You never see it on the Test Data Manager screens, but TDM calls it.

### **Repository**

A relational database where Informatica keeps its own working information (definitions, metadata, results). It is not your business data.

### **Metadata**

Data about data: table names, column names, data types, keys. Importing "metadata" copies this description, not the rows.

### **Informatica domain**

The family of Informatica services installed together and managed as one unit, with a shared set of users.

---

## **4.2 What Informatica TDM is for**

Start with ShopSphere's problem. The online retailer has a production database, `SHOPSPHERE_PROD`, with real customers. Testers need data that behaves like it, but real names, phone numbers and SSNs must never reach the test lab. Copying the whole database is slow and risky. Writing test rows by hand is slow and unrealistic.

The guide describes TDM as a product that works with PowerCenter, PowerExchange and other Informatica applications to manage *nonproduction data* (anything that is not the live system: test, development, training) (User Guide 10.5.10, p.21). With it an organisation can make a smaller copy of production data, mask the sensitive parts, find the sensitive columns in the first place, or build test data that never touched production. It can also store test data in a central place and reset it later.

The capabilities TDM groups under Test Data Manager (User Guide 10.5.10, pp.21-23), with a ShopSphere example for each

| Capability                      | What the guide says it does                                                                                                           | ShopSphere example                                                                                                                    |
|---------------------------------|---------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| Data discovery                  | Run profiles to find sensitive columns and key relationships; define data domains to group sensitive columns by pattern.              | Find that <code>ssn</code> and <code>tax_id</code> in <code>CUSTOMERS</code> hold identity numbers.                                   |
| Data subset                     | Build a small test database, for example by time, function or geography.                                                              | Copy only customers from the last 6 months with their orders and items.                                                               |
| Data masking                    | Apply rules such as substitution, shuffle, key masking and encryption; results can be repeatable; several rules can go on one column. | Replace <code>first_name</code> and <code>last_name</code> with believable fake names, always the same fake for the same real person. |
| Data generation                 | Create test data from rules, with no production data involved, in a schema you can test against.                                      | Create 5,000 brand-new customers for a load test.                                                                                     |
| Data sets (test data warehouse) | Store versions of test data, edit them, and reset a test system to a version.                                                         | Save "clean masked set v1" and restore it after a destructive test run.                                                               |
| Self-service portal             | A simpler screen for users who only use stored data and do not create it.                                                             | A tester who only needs to reset data, not design masking.                                                                            |
| Data coverage analysis          | Check, across pairs of columns, whether the data set has enough variety for the test cases.                                           | Check that there are customers in every zip-code band with orders of every size band.                                                 |

Two more things are worth knowing now. TDM also includes a command-line program, `ilmcmd`, which can do a subset of the Test Data Manager tasks without the web page (User Guide 10.5.10, p.23). And users have *roles and privileges* (rules about who may do what) that an administrator manages in the Informatica Administrator, so two people can log in to the same Test Data Manager and see different things.

Other items in the guide's overview, which we only name here: exporting flat-file results to an HP ALM test-management server, and masking or moving data on Hadoop clusters (User Guide 10.5.10, pp.22-23).

#### WHY IT MATTERS

The guide's own motivation (p.21) is blunt: organisations guard production tightly but protect test copies far less. TDM exists to close that gap without making developers rewrite code to create test data. When someone asks you "what does TDM do?", the short honest answer is: *find the sensitive data, shrink the data, mask it, or generate it instead, and keep versions of the result.*

## 4.3 Use cases: performance, security, and verified data

The guide groups what TDM is used for into three families (User Guide 10.5.10, pp.23-24). Seeing them as three *goals* helps you choose the right feature later.

### 4.3.1 Performance in testing environments

Here "performance" means the testing team's speed and agility, not database speed. The guide lists these ways TDM helps (p.23):

- **Subsets** keep primary and foreign key relations, so the smaller data is still *referentially intact* (every child row still finds its parent row).
- **Flat file results** can be copied straight to an integrated test tool.
- **Generated data** serves teams who are not allowed to see production at all.
- The **test data warehouse** keeps several versions of test data in one shared place, and teams on different machines that host a Test Data Manager Service in one domain can share it.
- Data sets can be **edited**, with row-level tags to track changes, and **restored** to return a test system to an earlier state.

### SHOPSPHERE EXAMPLE

The Customer Orders team runs a "refund" test that deletes orders. After the run, the test environment is in a strange state. Without TDM, someone re-copies the database (hours). With a stored data set, a tester resets the environment to "masked set v1" from the Data Sets view. This restore idea is the guide's "restore a data set version" use case. The refund story is ours, not the guide's.

## 4.3.2 Security and compliance

The guide frames the challenge as finding sensitive information in huge volumes of production data and making sure it does not appear in test data (p.24). Its three suggestions:

1. Run **profiles** to find sensitive columns, comparing against PII, PCI or PHI regulation standards. (Those three names are explained in the sensitive data and regulations chapter.)
2. Run a **data masking** operation to remove sensitive values, with standard formats (such as credit card or Social Security numbers) or custom formats.
3. Store results in the **test data warehouse** so fewer uncontrolled copies exist.

## 4.3.3 Verified test data with Data Validation Option

*Data Validation Option* (DVO) is a separate Informatica solution used with PowerCenter to check that target data is accurate and that the transformation did not introduce errors (p.24). Used with TDM, it can compare production and development data. The guide's example is a team that must send compliance reports for test data made from production, and needs "verifiable proof" that the test data holds no sensitive values.

The User Guide describes DVO only at this level; configuration is covered in the DVO guide.

### Goal: performance

"Give testers the right data quickly, and let them get it back."

Features: subset, flat-file copy to test tool, data generation, test data warehouse, restore.

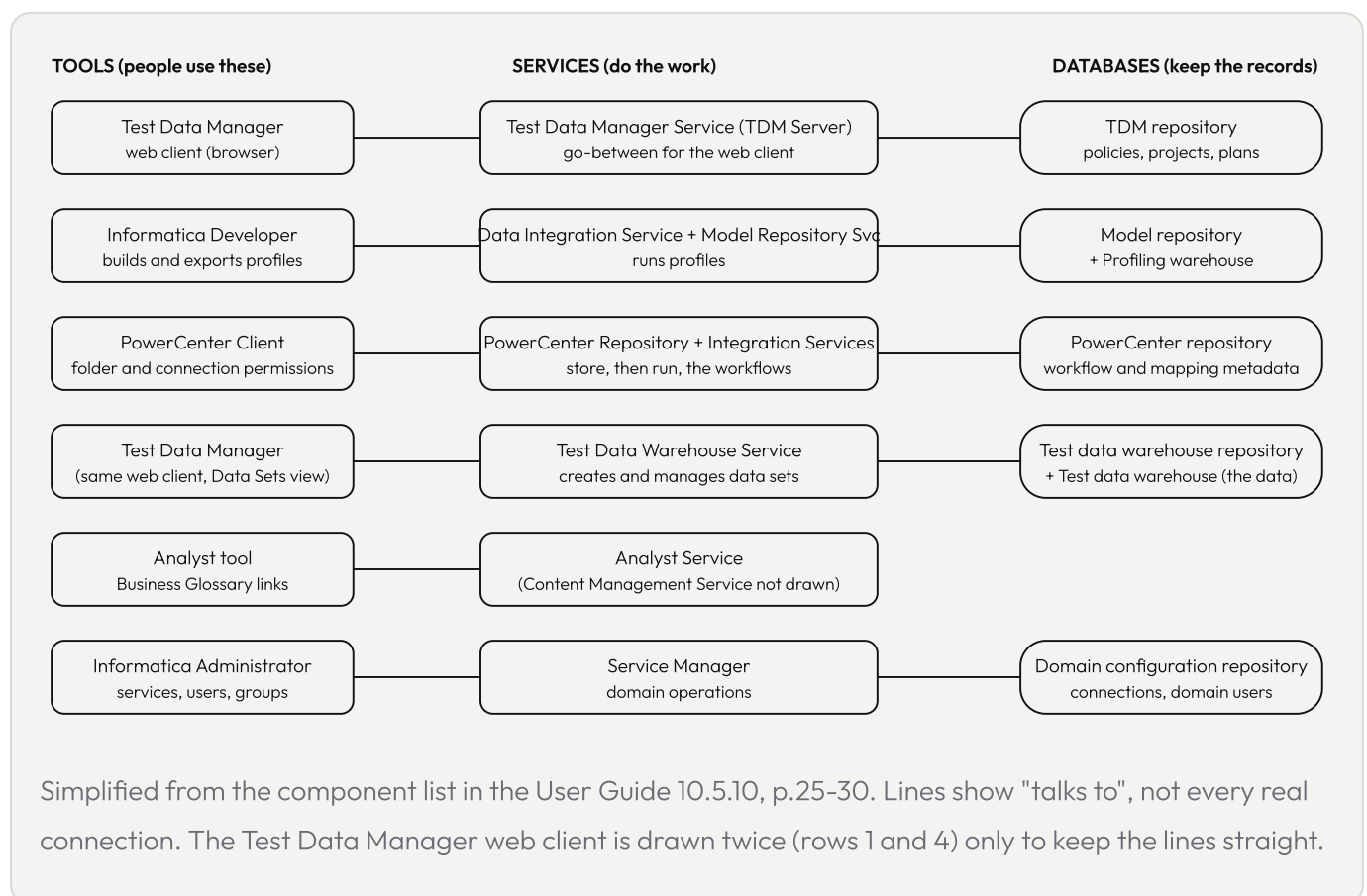
### Goal: security and compliance

"Prove sensitive data is not in the test lab."

Features: profiles with PII, PCI, PHI domains, masking, a single controlled warehouse, DVO for proof.

## 4.4 The architecture: what runs behind the web page

The guide says TDM consists of **tools**, the **Test Data Manager Service and other application services**, and **databases** (User Guide 10.5.10, p.25). A useful picture is a restaurant. Tools are the dining room and the order pad (where people ask). Services are the kitchen stations (grill, pastry, dishwashing), each doing one job. Databases are the pantry and the recipe books. The analogy breaks down in one place: here the "waiter" (the TDM Server) also keeps the plans and definitions in its own book (the TDM repository).



### VISUAL REFERENCE

Informatica TDM 10.5.10 User Guide, p.25, Figure "components of TDM". What to look for: tool boxes down the left (Developer Tool, PowerCenter Client, Test Data Manager Web Client, Analyst Tool, Informatica Administrator), a Test Data Manager Service with its own TDM Repository next to the web client, application services in the middle (grouped Model Repository and Data Integration services, grouped PowerCenter services, Test Data Warehouse Service, Content Management Service, Analyst Service, Service Manager), and cylinders on the right for the Model repository,

Profiling warehouse, PowerCenter repository, Test data warehouse repository, Test data warehouse and Domain configuration repository.

### 4.4.1 The TDM tools

The guide names four tools (User Guide 10.5.10, pp.25-27). You will spend nearly all your time in the first.

| Tool                      | What it is                                                   | What you use it for in TDM                                                                                                                           |
|---------------------------|--------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| Test Data Manager         | Web-based client                                             | Configure masking, subset, generation and discovery profiles; configure connections; manage project permissions; manage data sets.                   |
| Informatica Developer     | Client application ("thick client", installed on a computer) | Create and export profiles for data discovery, and run them.                                                                                         |
| Informatica Administrator | Web-based client for a domain administrator                  | Manage application services; create users and user groups.                                                                                           |
| PowerCenter Client        | Client application                                           | Manage the repository and design mappings, maplets and sessions; set permissions on folders and on connection objects of the PowerCenter repository. |

A *maplet* is a small reusable piece of transformation logic built in PowerCenter. In TDM you can create a data masking rule from a maplet that has been imported (p.31). A step-by-step build is in the maplet walkthrough in the Informatica masking chapter.

### 4.4.2 The TDM Server

One sentence in the guide: the TDM Server is the interface between Test Data Manager and the application services (p.27). Elsewhere the guide says "the TDM Server sends a request to the Data Integration Service" when you run a profile, and that the Test Data Manager Service "creates and manages the TDM repository" (pp.27-29). The guide does not say in one place how the "Server" and the "Service" relate. In the figure on p.25 only the Test Data Manager Service appears. Read "TDM Server" as "the part that takes your click and calls the right service"; how the two names relate is not documented.

### 4.4.3 The application services

The guide says TDM uses Informatica services that you create in the Administrator tool (User Guide 10.5.10, p.27). Here is the list with the plain-English job of each.

| Service                             | Its job (from the guide)                                                                                                                                                                                                      | When you notice it                                                                        |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------|
| Test Data Manager Service           | Creates and manages the TDM repository; runs Test Data Manager and manages connections between service components and users (pp.26, 28).                                                                                      | Every login and every click.                                                              |
| Data Integration Service            | Performs data discovery; stores profile metadata through the Model Repository Service; loads results to the profiling warehouse you chose when creating it; also runs data coverage tasks for the test data warehouse (p.27). | When a profile runs.                                                                      |
| Model Repository Service            | Manages the Model repository for data discovery (p.27).                                                                                                                                                                       | Behind profiles; also when you import metadata or data domains from the Model repository. |
| monitoring Model Repository Service | A Model Repository service that monitors statistics for Data Integration Service jobs (p.27).                                                                                                                                 | Rarely, as an administrator.                                                              |
| PowerCenter Repository Service      | Manages the PowerCenter repository and accepts requests from the PowerCenter Integration Service when a workflow runs (p.27).                                                                                                 | Generating a workflow; importing metadata from PowerCenter.                               |
| PowerCenter Integration Service     | Runs the data subset, generation and masking workflows. You choose which one when you create the Test Data Manager Service, and again when you generate a workflow from a plan (pp.27, 29).                                   | Every plan run.                                                                           |
| Test Data Warehouse Service         | Manages the test data warehouse repository and the warehouse; copies the metadata created with a data set and creates the data set tables (p.27).                                                                             | Creating or resetting data sets.                                                          |
| Analyst Service                     | Manages the Analyst tool TDM uses to link TDM objects to Business Glossary objects; needs a licence that supports Business Glossary (pp.27, 29).                                                                              | Linking glossary terms to policies and domains.                                           |

| Service                    | Its job (from the guide)                                                      | When you notice it                                                    |
|----------------------------|-------------------------------------------------------------------------------|-----------------------------------------------------------------------|
| Content Management Service | Manages reference data (p.26).                                                | The guide does not tie it to a particular TDM screen in this section. |
| Service Manager            | Manages all domain operations and the domain configuration repository (p.26). | Login and domain-level problems.                                      |

#### 4.4.4 The TDM databases (repositories)

These are the "recipe books and pantries". The guide lists seven relational databases that TDM connects to for metadata, profiling, TDM configuration, domain configuration and the test data warehouse (User Guide 10.5.10, p.28). Notice that only one of them holds a copy of *test data*; the rest hold Informatica's own working information.

| Database                        | What it stores                                                                                                                                          | ShopSphere reading                                                                                                            |
|---------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| TDM repository                  | The components you define in Test Data Manager (policies, projects, entities, rules) plus source or target metadata you imported (pp.26, 28).           | The masking policy, the "Customer Orders" project, the imported structure of <code>CUSTOMERS</code> and <code>ORDERS</code> . |
| Model repository                | Table metadata for data discovery profiles; connection information for connections made in TDM; optionally mappings saved for Hadoop operations (p.28). | The profile's view of the tables.                                                                                             |
| PowerCenter repository          | Metadata for PowerCenter sources and targets and for the workflows generated from plans; also permissions and privileges for its users (pp.26, 28).     | The generated masking workflow.                                                                                               |
| Profiling warehouse             | Profile results for data discovery (p.28).                                                                                                              | " <code>ssn</code> " looks like an SSN domain" lives here until you approve it.                                               |
| Domain configuration repository | Connections and metadata for the Informatica domain, including users (pp.26, 28).                                                                       | Who can log in.                                                                                                               |
| Test data warehouse repository  | Project and source table metadata for stored data sets (p.28).                                                                                          | The description of the stored "masked set v1".                                                                                |
| Test data warehouse             | The source data of stored data sets (p.28).                                                                                                             | The masked customer rows themselves.                                                                                          |

### DO NOT CONFUSE THESE WITH SHOPSPHERE\_PROD

Your business database (the source `SHOPSPHERE_PROD` and any test target) is *not* on the list above. TDM needs a connection to it, but it is not one of TDM's own repositories. The guide says that importing copies source *metadata* into the TDM repository (p.31), and that constraints you add in a project are stored in the TDM repository while the TDM server does not update the data sources (p.32).

## 4.4.5 TDM connections

TDM needs connections to databases, repositories and services, and what it needs depends on what you run (User Guide 10.5.10, p.28):

- **Discovery** needs a database source connection and a Data Integration Service.
- **Subset and masking workflows** need connections to services, the TDM repository, external database sources, and Informatica sources and targets.
- **Workflows that store data in the test data warehouse** need external database sources, the test data warehouse repository and the test data warehouse itself.
- To copy flat file results to a test tool, you first integrate the test tool server with TDM.

The guide splits them into service, repository and database connections. The table below keeps its meaning and shortens the words.

| Connection kind        | Examples and purpose (pp.29-30)                                                                                                                                                                                                                                                                                                                                                                                                           |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Service connections    | Data Integration Service (discovery); Model Repository Service; PowerCenter Integration Service (running generated workflows; you pick its name when you generate the workflow); PowerCenter Repository Service (extracts source metadata from the PowerCenter repository on import); Analyst Service; Test Data Warehouse Service; Test Data Manager Service.                                                                            |
| Repository connections | TDM repository (a masking, generation or subset workflow needs it); Model repository; profiling warehouse (the Data Integration Service loads profile results there); PowerCenter repository; test data warehouse repository and test data warehouse.                                                                                                                                                                                     |
| Database connections   | A database source for discovery, a database target for generation, relational sources and targets for subset and masking, and a source plus the test data warehouse for storing data sets. You configure them in the Administrator view of Test Data Manager and do not need a separate driver. You pick them when creating a plan; TDM tests them when it generates the workflow, and the Integration Service uses them when it runs it. |

#### 4.4.6 A readiness checklist before your first job

Most items belong to an administrator, but a TDM engineer needs to know to ask for them.

- 1 A user in the Informatica domain with the right roles and privileges.**  
 Users and groups are created in the Informatica Administrator, and roles and privileges decide which views you can open (pp.23, 27).
- 2 A connection to the source database.**  
 Needed for discovery, and for subset and masking sources. Created in the Administrator view of Test Data Manager, with no extra driver to install (pp.28-30).
- 3 Connections to the Data Integration Service and Model Repository Service.**  
 Needed before any profile can run (p.92).
- 4 A PowerCenter Integration Service chosen for workflows.**

Selected when the Test Data Manager Service is created, and again each time you generate a workflow from a plan (pp.27, 29).

5

#### **A target connection to a test schema.**

For masking, subset and generation plans. Never the production database (p.31).

6

#### **The test data warehouse and its repository, if you will store data sets.**

Needed for plans that target the warehouse (p.28).

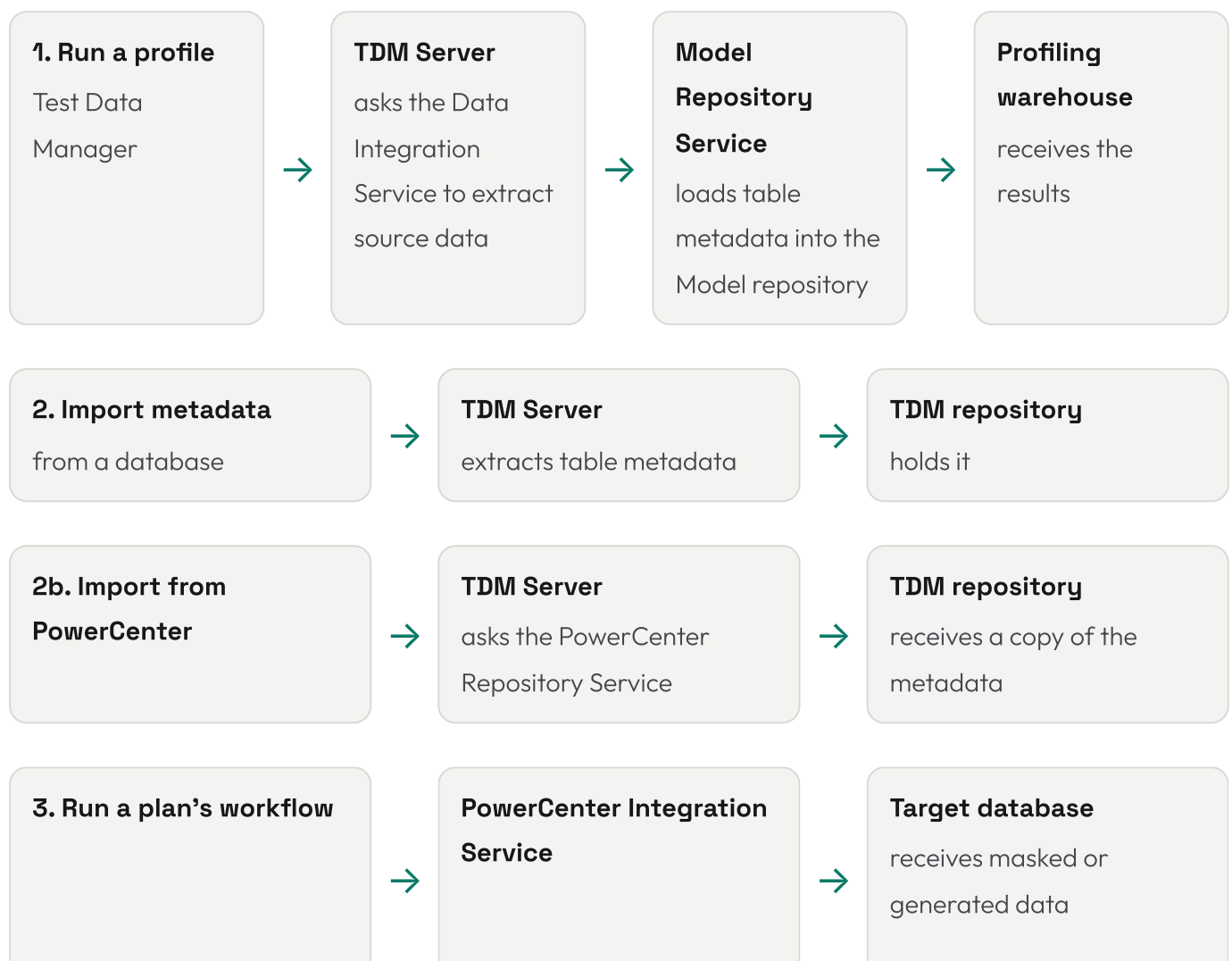
7

#### **Folder and connection permissions in the PowerCenter repository.**

Configured with the PowerCenter Client (p.27).

### 4.4.7 Following a click through the pieces

The guide describes three journeys in words. Putting them side by side shows how the pieces cooperate (pp.29-32).



runs it using the chosen  
connections

Practical lesson: a failed profile points at the Data Integration Service, its connection, or the profiling warehouse. A failed workflow points at the PowerCenter Integration Service or the connections you picked in the plan. A missing table in a project usually means the metadata import did not finish or was rejected.

## 4.5 The TDM process, end to end

The guide gives a ten-step outline: run a profile against source data, create a subset, mask it (User Guide 10.5.10, pp.30-31). Steps 1 to 3 prepare, steps 4 to 8 define, steps 9 and 10 run. Not every project does every step. Step 3 is explicitly optional, and data generation is a separate path from subset and masking.

1

### **Create policies.**

Decide which types of data to mask and which rules to use.

2

### **Create a project and import data sources.**

Bring the table descriptions into TDM.

3

### **Optional: discover information about the source data.**

Profiles find primary keys, entities and data domains.

4

### **Optional: create parameters.**

Global parameters for use across Test Data Manager, or project parameters (criteria, connection and owner) so one plan can make different test data on demand.

5

### **Define subset and masking operations.**

Choose tables and relationships for the subset; assign masking rules to columns.

6

### **Define data generation operations.**

Choose tables and assign generation rules to target columns.

7

### **If needed, enter test-tool (HP ALM) integration properties in the plan.**

To copy flat file results to the integrated test tool server.

8

### **If needed, choose the test data warehouse as the target in the plan.**

To store the result as a data set.

9

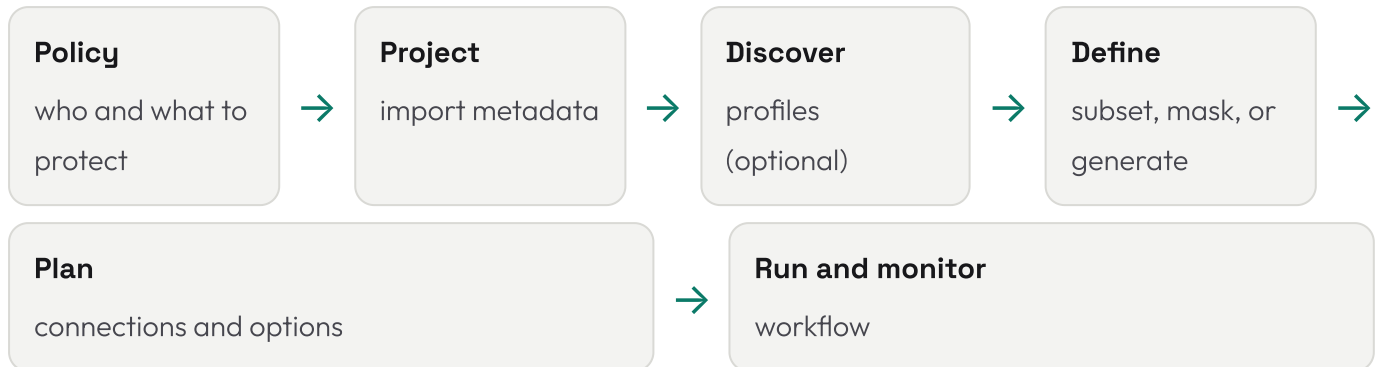
**Generate and run the workflow** for masking, subset or generation.

The PowerCenter Integration Service does the work.

10

**Monitor the workflow.**

Check status and logs.



Notice the split of roles in this order. The *policy* is design work (the guide's example gives it to a compliance officer). Everything from the project onward is build work (a project developer). *Reviewing* profile results and validating the output goes back to compliance.

### 4.5.1 Create a data masking policy

A policy contains **data domains** that describe what you want to mask. It has no data source, so one policy can serve many projects (p.31). A data domain groups sensitive fields by column name or by the data inside, using patterns written as *regular expressions* (a compact way to describe text shapes; the guided tour of Test Data Manager has a short primer). A domain also holds masking rules. A *rule* is a built-in masking technique plus specific settings; rules can also be built from imported maplets.

**ShopSphere:** a policy named for customer personal data, with domains for SSN, email, phone, first name and last name.

### 4.5.2 Create a project and import metadata

A project organises everything for discovery, masking, subset and generation. You import metadata for the sources you want to subset or mask, and target metadata if you want to generate data. You can import from a PowerCenter folder, from the Model repository, or from an external database. You also create a target schema, and the guide warns that TDM overwrites any data that already exists in the target schema (p.31).

### TARGET SCHEMA WARNING

Point your plan at a schema made for test data. "TDM overwrites data that already exists in the target schema" is the guide's own wording (p.31). Never aim a target connection at `SHOPSPHERE_PROD`.

## 4.5.3 Discover source information

Profiles can discover three things (pp.31-32):

- **Primary keys**, when a source has none. A primary key profile suggests candidates.
- **Entities** (related tables), when the project has several sources. An entity profile suggests possible relationships between tables. You select entities from the results to shape the subset.
- **Data domains**. A data domain profile searches columns that fit each domain, so you can mask all of them with the same rules.

Running a profile makes the TDM Server ask the Data Integration Service to extract source data, and the results go to the profiling warehouse. Constraints you add are stored in the TDM repository. The TDM server does not update the data sources (p.32). All of this is the subject of the Informatica discovery and profiling chapter.

## 4.5.4 Define data masking and data subset operations

For **subset**, you define which tables to include and how they relate (p.32). Three building blocks appear:

| Block    | Meaning in the guide                                        | ShopSphere reading                                                                 |
|----------|-------------------------------------------------------------|------------------------------------------------------------------------------------|
| Entity   | A set of tables related by physical or logical constraints. | <code>CUSTOMERS</code> with its <code>ORDERS</code> and <code>ORDER_ITEMS</code> . |
| Group    | A set of <i>unrelated</i> tables.                           | <code>PRODUCTS</code> , copied whole as reference data.                            |
| Template | Optional container holding entities and groups.             | "Customer Orders subset kit".                                                      |

You also assign masking rules to source columns, create a subset plan from entities, groups and templates (with optional filters on parent-table columns), and create a masking plan that carries the policies and

rules. Rules outside policies are allowed too. TDM stores projects, entities, groups, templates and plans in the TDM repository (p.32). The Informatica data subset chapter and 14 teach these in detail.

### 4.5.5 Define a data generation operation

Three tasks (p.32): create the entities for the plan, create generation rules and assign them to target-table columns, then create a generation plan with entities and tables, assigning default generation rules to columns that have none. The Informatica data generation chapter covers it.

### 4.5.6 Create plans

A *plan* holds what a workflow needs. Two rules matter for planning (pp.32-33):

- A masking operation and a subset operation can share one plan or have separate plans.
- A **data generation operation cannot be combined** with masking or subset in the same plan. It needs its own plan.

In either plan you can choose the test data warehouse as a target connection to save the result as a data set, and you can enter test tool integration properties to copy flat file results to an integrated HP ALM server. A generation plan also asks for the number of records. Then you generate the workflow and run it. The Informatica plans, workflows and warehouse chapter covers plans and the warehouse.

### 4.5.7 Monitor the workflow

Each workflow appears as a job in the Monitor view, together with jobs such as metadata imports and profiling (p.33). You can run the **Row Count Report** on a successfully run workflow to see how many rows the plan affected, and open the TDM job log to troubleshoot. In ShopSphere terms: after the masking run, the compliance officer checks that the row counts in the test target match the intended subset, and the QA lead opens the job log if the workflow failed.

---

## 4.6 The guide's TDM example, retold with ShopSphere

The guide's example (User Guide 10.5.10, pp.33-34) is about an organisation that wants to mask sensitive employee stock data for a new employee stock plan. Its test database needs data from several applications: personal data, salary, stock purchases and job information. Several teams must share it and be able to put the original test data back. The company builds a policy and keeps the result in the test data warehouse.

The table maps each of the guide's 13 steps onto ShopSphere. The ShopSphere column is example data.

The guide's 13 steps and a ShopSphere version

| # | Guide (employee stock)                                                                                                                                   | Who                | ShopSphere version                                                                                                                 |
|---|----------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|------------------------------------------------------------------------------------------------------------------------------------|
| 1 | Create an <code>Employee_Stock</code> policy.                                                                                                            | Compliance officer | Create a customer-data policy listing what must be masked.                                                                         |
| 2 | Define data domains; columns named like <code>Employee_Salary</code> , <code>Yearly_Salary</code> , <code>Salary_History</code> share a "Salary" domain. | Compliance officer | An "SSN" domain for <code>ssn</code> , <code>tax_id</code> ; a "Phone" domain for <code>phone</code> .                             |
| 3 | Define masking rules: substitution from a dictionary for names, random masking for salary, Social Security masking for SSNs.                             | Compliance officer | Substitution for names, Social Security masking for <code>ssn</code> .                                                             |
| 4 | Define a project; import sources.                                                                                                                        | Project developer  | Project "Customer Orders"; import the four ShopSphere tables.                                                                      |
| 5 | Run a data discovery profile; it fills the data domains.                                                                                                 | Project developer  | Run a data domain profile; <code>email</code> , <code>phone</code> , <code>ssn</code> appear against their domains.                |
| 6 | The database has no primary or foreign keys, so run a profile for primary keys and entities; create logical keys.                                        | Project developer  | Suppose the test copy of the metadata lacks declared keys; profile them and add <code>customer_id</code> as a logical primary key. |
| 7 | Create entities and a group (Employee, JobHistory, Salary, Employee_Stock as an entity; Stock_History in a group).                                       | Project developer  | Entity: <code>CUSTOMERS</code> , <code>ORDERS</code> , <code>ORDER_ITEMS</code> . Group: <code>PRODUCTS</code> .                   |
| 8 | Approve or reject profile results.                                                                                                                       | Compliance officer | Approve <code>ssn</code> to SSN domain; reject any wrong match.                                                                    |

| #  | Guide (employee stock)                                                                                   | Who                | ShopSphere version                                            |
|----|----------------------------------------------------------------------------------------------------------|--------------------|---------------------------------------------------------------|
| 9  | Verify all sensitive fields are masked, using reports.                                                   | Compliance officer | Check that every personal-data column has a rule.             |
| 10 | One plan for subset and masking; add entities, groups and the policy; run the workflow.                  | Project developer  | A plan that copies a masked subset into a test schema.        |
| 11 | Validate results in the subset database.                                                                 | Compliance officer | Spot-check masked names and SSNs in the test schema.          |
| 12 | Plan with the subset database as source and the test data warehouse as target; run to create a data set. | Project developer  | Store the masked subset as "masked set v1".                   |
| 13 | Reset a data set from the test data warehouse to a required connection.                                  | Project developer  | Restore "masked set v1" into a QA schema after a refund test. |

#### WHAT THE EXAMPLE TEACHES

Two things. First, the *order*: design (policy, domains, rules) before build (project, profile, entities, plan) before run (workflow, data set). Second, the *hand-offs*: the same person does not do everything. The people who understand the regulations choose and approve; the people who understand the tables build. In a real engagement (the chapter on planning real engagements) the same split shows up as roles on your team.

## 4.7 Advanced parameterization at a glance

ShopSphere's QA team needs test data for *sprint 14 regression* (orders from this month, in schema **QA1** ), then for *performance tests* (orders from last year, in schema **PERF** , on a different server). Making a new plan for each request is slow. Editing one plan every time is error-prone.

The guide's answer is *parameters*: placeholders in the plan that point to a value, and the value is read when the workflow runs (User Guide 10.5.10, pp.35, 51). You change the value and run the same plan again

to get different test data.

### 4.7.1 Three kinds of parameter

| Type       | Placeholder for                        | ShopSphere use                                              |
|------------|----------------------------------------|-------------------------------------------------------------|
| Connection | Source or target connections           | "QA database" today, "PERF database" tomorrow.              |
| Owner      | Source or target schema owner names    | QA1 versus PERF .                                           |
| Criteria   | Values in filter criteria of an entity | The order_date cut-off used to choose which orders to copy. |

You can create parameters **inside a project**, or an administrator can create **global parameters** usable across Test Data Manager. Global parameters are marked with an asterisk (\*) in parameter lists. If you import a global parameter into a project, you get a copy: changes you make to the copy do not touch the global parameter (p.35).

#### ONE RULE TO REMEMBER NOW

You must import source metadata from *every* source connection you might need into the project. A workflow whose parameter value points at metadata the project does not contain cannot run (p.35).

### 4.7.2 The sequence in the guide

The guide lists 16 tasks. Grouped into five phases they read like this (pp.35-36):

- 1 Prepare the project.**  
Create a project and import the required source metadata. Create connection, owner and criteria parameters in it with default values, and choose whether the default may be overridden. Optionally import global parameters.
- 2 Use parameters in the entity.**  
Create an entity with the required tables and add filter criteria that use parameters. You can add several criteria; they can be project or global parameters.

3

**Use parameters in the plan.**

Create a plan using parameters. On the Plan Settings page, add the source and target connection types as variables and choose the connection parameters. You must enter a parameter file name and the path to that file on the machine that hosts the PowerCenter Integration Service. In Advanced Settings, add schema owners as variables and select the owner parameters. Save the plan.

4

**Optional: use a parameter file.**

From the plan's Properties tab, Actions > Create Parameter File downloads a file listing every parameter in the plan. You may edit values, and set the "ignore" flags of criteria parameters to "Y" to leave them out of a run.

5

**Run.**

From the plan Properties page, Actions > Generate and Execute, then pick how values are supplied. Choose *Default* (the file entered in Plan Settings), *Use Parameter File* (a different file), or *Add Parameter Values* (edit values in Test Data Manager, and use the Select Parameter button to include criteria parameters). Click Generate and Execute and monitor the workflow.

**Expected result:** What you should see: the same plan, different data each time you change values. The guide's two examples: change owner and criteria values to get data in the same connection but a different schema, or change connection parameters and include different criteria to get data between different connections (p.36).

**SHOPSPHERE EXAMPLE**

Plan "Customer Orders masked subset" uses an owner parameter for the target schema and a criteria parameter for the order date. Sprint run: owner value QA1 , criteria "this month".

Performance run: owner value PERF , criteria "last year". One plan, two results.

Details such as parameter naming prefixes, rules, and editing steps are in the guided tour of Test Data Manager. Using parameters inside plans is covered again in the Informatica plans, workflows and warehouse chapter.

**Lab: trace three ShopSphere problems to the right piece**

Goal: use the architecture map (User Guide 10.5.10, pp.25-30) to trace three failures to the right piece. Paper exercise, no software needed.

**1****Problem A.**

A data domain profile on `CUSTOMERS` finishes, but no results appear in Test Data Manager. Which service ran it, and where should the results be stored?

**2****Problem B.**

A masking plan generates a workflow successfully, but the run fails to start. Which service runs workflows, and which connections did you choose in the plan?

**3****Problem C.**

A tester cannot see the "Customer Orders" project in the list. Which kind of setting decides this, and where does an administrator manage users and roles?

► **Show a worked answer**

## 4.8 Common mistakes and gotchas

### WATCH OUT

- **Looking for business data in the TDM repository.** The repositories hold definitions, metadata, results and stored data sets. Your source and target databases are separate and need their own connections.
- **Treating "the TDM Server", "Test Data Manager" and "TDM" as three products.** They are the web client, the go-between, and the product name respectively. When a document is ambiguous, ask which piece is meant.
- **Planning generation and masking in one plan.** The guide says a generation operation needs its own plan (p.33).
- **Forgetting the overwrite.** TDM overwrites existing data in the target schema (p.31). Use a dedicated test schema.
- **Skipping discovery on a database with no declared keys.** The guide's example had none, so profiles created the logical keys the subset needed (p.34).
- **Importing only one connection's metadata, then parameterizing the connection.** The workflow cannot run (p.35).
- **Assuming every feature in the guide exists in your installation.** Licences, privileges and version differ. Confirm in your version.

## 4.9 Recap

### KEY POINTS

- Informatica TDM finds sensitive data, makes smaller copies, masks, generates data, and stores versions of test data (User Guide 10.5.10, pp.21-22).
- Use cases fall into performance, security and compliance, and verified data with DVO.
- Four tools (Test Data Manager, Developer, Administrator, PowerCenter Client), a TDM Server, about ten application services, and seven databases make up the architecture.
- Your business databases need *connections*; they are not TDM's repositories.
- The process is: policy, project and import, optional discovery, define, plan, generate and run, monitor.
- Generation needs its own plan; masking and subset can share one.
- Parameters (connection, owner, criteria) let one plan produce different test data.

## A guided tour of Test Data Manager

The *Test Data Manager* web page has three main working areas: **projects** (the workspace), **data sources** (the tables brought in) and **policies** (what to protect, and how). This section covers where things live, why each task exists and what to expect afterwards.

### 4.10 Words to know first

#### View

One of the main screens of Test Data Manager, chosen from the tab bar at the top (for example Policies or Projects).

#### Content panel and Details panel

The two areas of most screens. The Content panel lists items (rows). The Details panel shows more about the row you selected.

### Actions menu

A menu that changes with the view. Almost every task in this section starts with "click Actions".

### Data source (in a project)

A set of source tables whose *descriptions* (metadata) you imported into the project. Not a copy of the rows.

### Data domain

A named group of columns that mean the same kind of thing, such as "Social Security Number", recognised by column name or by value shape.

### Regular expression

A short text code that describes a shape of text. `[0-9]{3}-[0-9]{2}-[0-9]{4}` describes "three digits, a dash, two digits, a dash, four digits".

### Data pattern and metadata pattern

Two kinds of regular expression on a data domain. A data pattern describes what the *values* look like. A metadata pattern describes what the *column name* looks like.

### Sensitivity level

A label you give a data domain to say how sensitive it is. An administrator defines which levels exist.

### Policy pack

A ready-made policy installed with TDM, covering PII, PHI or PCI data.

### Impacted Objects dialog

A warning window that lists what will break when you delete or change something that is already used.

## 4.11 Who uses which view

The guide ties each view to a typical person (User Guide 10.5.10, p.37). A compliance officer uses **Policies** to create policies, rules and data domains. A developer uses **Projects** to import sources, run profiles, define subset, masking and generation, and generate and run workflows, and uses **Monitor**

to check job status. A test data specialist or test engineer uses **Data Sets** for the test data warehouse. An administrator uses **Administrator** for connections, pass phrases, workflow options and server management. By default an administrator can open the Administrator view; other users need privileges for the other views (p.39).

## 4.12 Logging in to Test Data Manager

You use a normal web browser and the user name and password defined in Informatica Administrator (User Guide 10.5.10, p.48).

1

**Open the address** `https://hostname:portnumber/tdm/` (or `http://` if Transport Layer Security is off).

*hostname* is the machine with the TDM Server. The guide's default port is **6643** with TLS and **6605** without.

**Expected result:** The Login dialog box appears.

2

**Enter your user name and password and choose the security domain, then click Login.**

If the domain uses LDAP, the default security domain is Native.

**Expected result:** Test Data Manager opens. Click Logout to leave.

## 4.13 The views

All views share a header (Search, Quick Links, User Preferences, Help, Logout) and a tab bar. Most screens have a Content panel (a list of rows) and a Details panel (more about the selected row).

### VISUAL REFERENCE

Informatica TDM 10.5.10 User Guide, p.38, Figure "Test Data Manager" (the Policies view).

What to look for: the tab bar, a list of policies above, and Properties and Data Domains tabs below with a pie chart the guide describes as showing unmasked data, a compliance risk.

| View          | What it does                                                                                                                           |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------|
| Overview      | Dashboard reports about projects, with links to project and policy objects.                                                            |
| Policies      | Policies, masking rules, generation rules and data domains that you add to projects.                                                   |
| Data Sets     | Versions of data sets stored in the test data warehouse: view, edit, reset to a target, lock, delete.                                  |
| Projects      | Projects with their source data and subset, masking, profiling and generation work.                                                    |
| Monitor       | Status of import, profiling, subset, masking and generation jobs; stop or abort jobs, read the TDM Server job log, recover workflows.  |
| Parameters    | Global parameters (created outside any project); import them, export them as CSV.                                                      |
| Administrator | Connections, pass phrases, dictionaries, test tool integration, test data warehouse, workflow options, sensitivity levels, log levels. |

The guide's list on p.38 names six views and omits Parameters, but p.45 describes a Parameters view with its own screenshot.

## 4.14 Helpers on every screen

### 4.14.1 Search, filters, Quick Links and preferences

- **Search field.** Finds projects, plans, assignments, data sets and data set tables in the TDM and test data warehouse repositories. **It cannot find connections.** Results include the object plus what it contains and what contains it (p.39).
- **Advanced text filter.** In column filters (Discover > Columns, Define > Data Masking, Define > Data Generation, profile results) type "FIRST\_NAME" in quotes for an exact match, ORDER\* for starts with, \*\_ID for ends with, or plain PHONE for contains (p.39).
- **Data filter.** Type part of a name (the guide uses "Emp") in a column filter and click the Filter icon; click Reset Filter to clear it (p.40).

- **Quick Links.** Shortcuts to create a rule, policy, data domain or project; what you see depends on your permissions (p.40).
- **User Preferences.** The General tab has Show Start Screen; the Projects tab chooses which projects the Overview dashboards show (p.40).
- **Actions menu.** Changes with the view; most tasks below start there (p.40).

## 4.15 The Overview view

Overview is a read-only "management view" of summary dashboards (User Guide 10.5.10, pp.41–42). The main ones: **Project Risk Analysis** (percentage of columns at each domain sensitivity level and the percentage in no domain), **Project Assignments and Project Objects** (counts for the last 10 projects), **Recent Activities** (the last 10 changes, with an Audit Trail report you can download), **Plan Execution** (plans in progress or completed on a date), **Global Components** (share of policies, rules and domains) and **Recent Projects** (the last 10 projects you modified). A sensitive-data distribution graph appears only if sensitivity levels are set.

### VISUAL REFERENCE

Informatica TDM 10.5.10 User Guide, p.41, Figure "Overview view". What to look for: a plan-execution timeline, a Global Components pie chart and a Recent Projects table.

## 4.16 Data Sets, Monitor, Parameters and Administrator

The Policies and Projects views get their own sections below. The others, briefly (pp.43–46):

- **Data Sets** lists stored data sets. A version has Overview, Tables and Components tabs, and the Actions menu can reset a version to a target, delete or lock it (the plans, workflows and warehouse chapter covers this).
- **Monitor** lists jobs with ID, name, status, project, times and user. Select a job and use Actions.
- **Parameters** holds global parameters, usable directly in a plan or added to a project first.
- **Administrator** covers default settings, sensitivity levels, connections, dictionaries, application logs and log levels, test tool integration and pass phrases.

## 4.17 The Expression Builder

An *expression* is a small formula, like a spreadsheet formula. The Expression Builder lists functions by category and columns, so you can build one without memorising names (User Guide 10.5.10, pp.46–47). You open it with the Expression button when you create an Expression-type masking rule, add pre- or post-processing to a rule, create advanced masking rules, assign rules to a column, configure an ad hoc generation rule or set advanced expressions in plans. A Validate Expression button checks your formula.

## VISUAL REFERENCE

Informatica TDM 10.5.10 User Guide, p.47, Figure "Expression Builder". What to look for: function categories on the left, an Expression box and a Join Condition box, and OK, Cancel and Validate Expression buttons.

## EXAMPLE FROM THE GUIDE, WITH SHOPSPHERE NAMES

An advanced masking rule has two variable ports holding the masked first and last names. The output port joins them with a space:

```
CONCAT( CONCAT( VAR1, ' ' ), VAR2 )
```

VAR1 would hold a masked `first_name` and VAR2 a masked `last_name`, giving a full name such as "Priya Das". Function syntax is in the Informatica Transformation Language Reference.

## 4.18 Projects

A *project* is the top-level container for everything you do to one set of data: discovery, masking, subset and generation (User Guide 10.5.10, p.49). Think of a project as a folder on your desk for one job, "Customer Orders for QA". Inside it you keep the table descriptions, your profile results, which rule goes on which column, the subset recipe, and the plans. The analogy has one flaw: in TDM the project folder is also a real folder in the PowerCenter repository with the same name as the project, which is how workflow information lands in the right place (p.60).

### 4.18.1 The six views inside a project

Open a project and you see a second row of tabs (User Guide 10.5.10, pp.49-50).

| Project view | What you do there                                                                                                                                                         |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Overview     | Edit properties; add policies and rules; import data sources (from PowerCenter, the Model repository or a database; you can mix source types); create project parameters. |
| Discover     | Find primary keys, table relationships and data domains (the discovery and profiling chapter).                                                                            |
| Define       | Assign masking rules and policies to columns; define entities, groups and templates for subset; assign generation rules.                                                  |
| Execute      | Define a subset, masking or generation plan; generate and run a PowerCenter workflow.                                                                                     |
| Monitor      | Status of import, profiling, workflow and data jobs; refresh, abort, stop or recover them.                                                                                |
| Permissions  | Set who can do what in the project (see the permissions section).                                                                                                         |

#### VISUAL REFERENCE

Informatica TDM 10.5.10 User Guide, p.49, Figure "a project in Test Data Manager". What to look for: the project tab bar, a General block of project facts and a Data Sources list at the bottom.

## 4.18.2 Project components

A project has at least one data source. The other components depend on the work (pp.50-51):

- **Entity:** a set of related tables (subset, generation). **Group:** a set of unrelated tables copied to the subset (subset). Both are made in Define.
- **Parameters:** values for connections, criteria and schema owners.
- **Plan:** defines the operations; a PowerCenter workflow is generated from it. Made in Execute.
- **Policy assignments:** assigning a policy gives the project its data domains and masking rules.  
**Rule assignments:** which rule applies to which column.
- **Profile:** suggests keys, relationships and data domains. Made in Discover.

### 4.18.3 Parameters in a project

Parameters are introduced in the Informatica overview chapter. Details (pp.51-52, 57-58): the tool adds the name prefix for you: **connection** parameters begin `$DBConnection` , **owner** parameters `$Param` , **criteria** parameters `$$` .

- With a connection parameter on an **Oracle** source, you cannot enable the Oracle Rowid property.
- Connection parameters rule out **batch updates for in-place masking**, because that needs a staging connection and a staging connection cannot be a parameter.
- If a plan uses one connection parameter as both source and target, Update Strategy does not change to Update. Tick Use Source Connection as Target Connection, or set it by hand.

1

#### Open the project, click Parameters, then Add.

A project parameter is usable only in this project.

2

#### Enter a name and choose the type (criteria, connection or owner) and a data type (String, Numeric or Date).

For Date, enter an expression such as `TO_DATE($$CRIT1, 'MM-DD-YYYY')` so a function knows how to use it.

3

#### Enter a default value and choose whether to allow override, then click OK.

If you do not allow override, nobody can change the value when running the workflow, so a default is then required.

**Expected result:** The parameter appears in the project's list.

To edit, click the row (not the checkbox), change values and Save; the parameter and data types cannot change. To delete, tick the box and click Delete; you see the entities and plans that use it and must regenerate and run them again. To import a global parameter, click Add Global Parameter; changes to the copy do not affect the global one (pp.57-58). Global parameters export to CSV from the Parameters view (Actions > Export).

### 4.18.4 Project logs

PowerCenter writes workflow and session logs when you run a plan from a project. Unless you set locations, it uses defaults from the PowerCenter Integration Service: `$PMWorkflowLogDir\` , `$PMSessionLogDir\` and `$PMBadFileDir\` (reject files hold rows that could not be written) (p.52). You can set them in the Edit Project dialog but **not when creating** the project: open the project,

Actions > Edit, Default Log File tab, fill in the three directories with a trailing slash, Save (p.59). **The directory must already exist.** A wrong workflow log directory makes the workflow fail without an error log.

## 4.18.5 Data masking transformation session properties

You can tune run-time properties of the masking *session* (pp.52-53). The guide ties better performance to a bigger **Cache Size** (the dictionary cache in memory; default 8 MB, at least 32 MB recommended for 100,000 records) and a bigger **Storage Commit Interval** (rows committed per batch to the storage table; default 100,000). Other properties: **DMO Cache Directory** (default `$PMCacheDir`, needs write permission), **SSN High Group File** (path to the Social Security Administration high-group list) and **Shared Storage Table** (default disabled; share it only when two masking instances or ports use the same dictionary column, connection, seed and locale). The Informatica masking chapter explains the techniques.

## 4.19 Managing projects

When you create a project you become its **owner** and have access to its folder by default. You can edit its name, description, folder location and associated sources, and the place where its logs are stored (User Guide 10.5.10, p.53). You can also export a project, with its components, to an XML file and import it elsewhere. People do this to back a project up or to move it from a development system to a production system.

### 4.19.1 Creating a project

#### Task: create the "Customer Orders" project

Goal: make an empty workspace for the ShopSphere work. (pp.53-54)

1

##### Click Projects.

The Projects view lists existing projects. Always check here first that a project for this job does not already exist.

**Expected result:** A list of projects appears.

2

##### Click Actions > New.

This opens the Create Project dialog.

3

##### Fill in the properties.

Use the table below. For ShopSphere: Name `CustomerOrders` (no spaces, which the guide requires if you will use Hadoop sources), a short description, keep the default Folder (the project name), and keep the default Target Conflict Resolution Strategy (Rename) unless your team says otherwise.

4

Click OK.

**Expected result:** The project opens in a new page.

► Show the property table

## 4.19.2 Editing a project

1

Click Projects and click the project name.

It opens in its own tab, with properties and data sources.

2

Click Edit, change options, click Save.

You can change name, description, folder, owner, target conflict resolution strategy and target empty value (the PowerCenter Repository field is not in the edit list). **You cannot remove a source that entities or groups in the project use.** (pp.54-55)

## 4.19.3 Project default property configuration

An administrator with the **Manage Preferences** privilege can add default fields to every new project, on Administrator > Preferences > Project Configuration (p.55). Two examples from the guide: `Enable Old Dictionary Optimization` (True keeps the pre-10.5 substitution-masking optimisation so output stays the same; False or unset uses the new algorithm, pp.55-56) and `Target Empty Value` (a replacement for empty strings that would break a non-null target column, for example when resetting an Oracle-based data set to SQL Server; the workflow still fails if that column is unique, p.56).

## 4.19.4 Copying and deleting a project

1

**Copy:** in Projects click a project *description* to select it without opening it, then Actions > Duplicate, rename, Save.

The copy does **not** import the metadata or objects (p.56), so it starts as an empty shell.

2

**Delete:** select it the same way, Actions > Delete, Yes.

All sources associated with the project are deleted (p.57).

## 4.19.5 Exporting and importing a project

An exported project is an XML file containing source information, connections, constraints, data domains and assignments (p.59).

1

**Export:** Projects, click a project, Actions > Export, choose the XML file name and path, OK.

2

**Import:** Projects, Actions > Import, browse for the XML file, click Finish.

**Before importing, import all the global components that the project assigns** (policies, domains and rules; see the policies section). Otherwise the project refers to objects the target repository does not have (p.59).

**Expected result:** Test Data Manager imports the project.

## 4.20 Data sources

A project cannot do anything until it knows what tables it is working on. To do subset, masking or discovery, you import **source metadata** into the TDM repository. To generate data, you import **target** metadata instead (User Guide 10.5.10, p.60). A project can hold more than one type of source, for example a flat file and a relational database, and you can create constraints that relate them and add filter criteria for subset and masking.

### A NUMERIC PRECISION LIMIT

If a numeric column has a precision above 28, you cannot apply a masking or subset condition to it. The Integration Service passes that row and column to the target without applying the condition (p.60). Check unusual numeric columns before you trust a run.

### 4.20.1 TDM source definitions

You can import source metadata from PowerCenter, from the Model repository, or directly from the source database (p.60). The guide's points:

- The TDM project and the repository folders share the same name, so workflow information goes to the correct folder.
- If the source is relational and has **many tables**, importing straight from the database is faster than going through PowerCenter or the Model repository.
- Importing from the PowerCenter or Model repository brings in the definitions Informatica services support, by importing the repository folder that holds them; TDM makes a copy of that folder in the TDM repository.
- MongoDB sources imported into PowerCenter need unique column names of at most 80 characters.
- **Flat files** must come from the PowerCenter or Model repository; you cannot create flat file connections in Test Data Manager.
- `rowid` is a reserved keyword. You cannot import Oracle sources that use it as a column or table name.

## 4.20.2 Importing data sources

### Task: import the ShopSphere tables into the project

Goal: have the four tables visible in the project's Data Sources view. (pp.66-67). Assumes an administrator already created a connection to the ShopSphere database.

1

#### Click Projects, then click your project to open it.

Imports belong to a project, so the tables arrive in the right workspace.

2

#### Click Actions > Import Metadata.

**Expected result:** The Import Metadata window appears.

3

#### Choose the source type.

Options: *PowerCenter Repository*, *Model Repository*, or *Datasource Connection*.  
For a plain relational database, Datasource Connection is the direct route. To import metadata from a flat file or a JDBC connection, you must use the Model repository.

4

#### Decide whether to review metadata changes before the import.

If you choose to review, TDM shows the metadata and its impact on rules and domain assignments, or entities and groups. You can skip this option. Reviewing is

wise when re-importing into a project that already has assignments.

5

**Click Next and choose where to import from.**

For PowerCenter or the Model repository, pick the folder (filter by name or description). For Datasource Connection, pick the schema (filter by name). Click Next.

6

**Select the tables or files to import and click Next.**

You can filter by data source, table name or description. For a Hive database, do not select temporary "junk" tables created when mappings run.

7

**Choose when to import and click Finish.**

*Import Now* (tick Run import in the background if you want to keep working) or *Schedule Later* (pick a date and time).

**Expected result:** The Import Progress dialog box shows a progress bar. If you chose to review, the Impact Review dialog box appears with metadata, assignments and project objects; click Approve to import or Reject to cancel. Afterwards the tables are in the Data Sources details view.

► Show what ShopSphere should look like afterwards

### 4.20.3 Special source types in the guide

Headline rules for the less common source types:

| Source type                 | Headline facts (pp.60-66)                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hive and HDFS               | You can do data movement, data domain discovery and masking. A cluster configuration must exist in the Administrator tool. Use a <i>Hadoop plan</i> . You cannot do subset or generation on Hive and HDFS. Do not select temporary junk tables. Data domain discovery works, but primary key and entity profiles do not run on Hive. Before in-place masking on Hive, back up the source tables, because a failed move from staging can truncate them. |
| Avro and Parquet            | Semi-structured formats for single-level hierarchy files, usable when you choose an HDFS target. With Parquet you cannot use null or repeated constraints. You cannot run profiles on Avro and Parquet.                                                                                                                                                                                                                                                |
| XSD (XML) sources           | Subset, masking and generation work on XML. Import the <b>XSD source definition</b> from the PowerCenter repository; importing the XML itself makes the workflow fail. View files under Discover > Files. On a file page, the Properties, View and Details tabs let you assign data domains, mark elements Restricted or Sensitive, and assign masking rules, generation rules or subset criteria. You cannot delete XML files after import.           |
| PowerExchange nonrelational | Adabas, IMS unload, Sequential, and hierarchical and nonhierarchical VSAM. Define datamaps in PowerExchange, import into PowerCenter, then import into the project. You cannot import a PowerExchange source directly into the TDM repository.                                                                                                                                                                                                         |

## 4.20.4 Identity columns

An identity column is one the database fills in automatically with a counter. Three rules from the guide (p.67):

- On **Sybase**, you cannot move data from a source DBO schema to a target non-DBO schema if the data has an identity column and indexes.
- If auto identity is enabled in Sybase and a table has no primary or unique keys, Sybase adds a `SYB_IDENTITY_COL` column. **Mark it Restricted** (restricted means the Integration Service does not write the value to the target), or the workflow fails.
- To disable an identity column on **Microsoft SQL Server**, the database user needs execute permission on `sp_help`.

## 4.20.5 Deleting a table

You can remove a table from a project if you added too many or no longer need some (pp.67-68). Rules: an *orphan* table (related to no other table and with no assignments) deletes directly. If the table is part of profile results, a warning appears and it is removed from them too. If it is part of an entity, group, masking plan or generation plan, the **Impacted Objects** dialog lists what will be affected; a table in a template that is part of a plan does not list the template and plan. You delete one table at a time.

1

Open the project and click Discover > Tables.

2

Select the table and click Actions > Delete Table.

**Expected result:** A confirmation message appears if the table has no assignments; otherwise the Impacted Objects dialog appears.

3

Click Continue to confirm.

Delete only tables not used in a TDM operation. Regenerate and run affected plans afterwards.

## 4.21 Project permissions and security

When you create a project you become its owner and can add users and user groups with permission levels. The domain administrator can also add and edit project permissions (User Guide 10.5.10, p.68). A project you have **no permission on does not appear** in your list, and you cannot open it even from search results. There are three levels, and they sit *on top of* privileges: you need the permission on this project **and** the minimum privilege for the task (p.50).

## Permission levels and tasks (p.69)

| Permission | What you can do                                                                                                                                                                       | Minimum privileges (as listed)                                                            |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------|
| Read       | Open and view the project; monitor logs for project workflows.                                                                                                                        | View project, Monitor project, Audit project                                              |
| Write      | Everything in Read, plus import metadata, delete tables, create entities, groups and templates, assign rules, generate workflows, run profiles, copy the project, delete the project. | View, Monitor, Audit, Import metadata, Generate project, Manage project, Discover project |
| Execute    | Open and view the project; monitor logs; run workflows.                                                                                                                               | View, Monitor, Audit, Execute project                                                     |

Notice that "run workflows" is separate from "generate workflows". That lets a team give a QA engineer Execute to re-run a prepared plan without letting them change it. Write does *not* list Execute in the guide's table; if a developer must both build and run, grant both.

1

**Open the project and click Permissions.**

**Expected result:** Lists of users and user groups with their permissions appear (Users and User Groups tabs).

2

**Click Edit on the Users or User Groups tab.**

The Edit Project Permissions dialog opens.

3

**Change, delete or add.**

To change, select the user or group and edit the permission; save for *each* user or group. To delete, select and click Delete. To add, click Add Users or Add User Groups, select one or more, optionally pick the permission to give all selected, then OK; if you did not pick, select each one and set permissions, saving each.

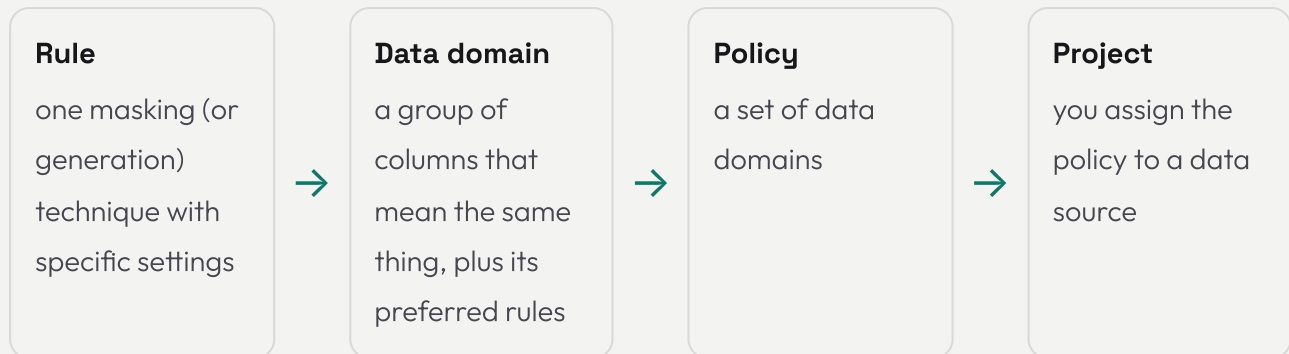
4

**Click OK.**

Changes take effect from the user's **next login**.

## 4.22 Policies

A *policy* is a data masking component that describes how to protect specific types of source data (User Guide 10.5.10, p.71). It is built from three smaller things that you can create separately, in any order:



The ShopSphere reading: a *rule* is "replace first names from a dictionary of fake names". A *domain* is "First Name", the group of every column that holds first names. The *policy* is the compliance officer's list of domains that must be masked in any customer-data project. Once the policy exists, the Customer Orders project simply receives it and every first-name column gets the first-name rule. You can apply one policy to many projects (p.71).

**Policies task flow** (p.72): create masking rules, define data domains that describe which columns can receive the same rules, assign the rules to the domains, create a policy, assign the domains to the policy. The order is flexible; you can create a policy first and fill it later.

### 4.22.1 Rules

A rule defines logic to mask sensitive data *or* to generate test data (p.72). You can create rules for string, numeric and date data types. You can add rules to data domains and then use the domains in policies and plans. After you add rules to a project, you still assign them to columns, matching the column's data type. The built-in masking techniques (substitution, shuffle, key masking and so on) are the subject of the Informatica masking chapter; generation rules are in the Informatica data generation chapter.

### 4.22.2 Data domains

A data domain represents the functional meaning of a column, based on its data or its name. Assign a masking rule to the domain and **all columns in that domain are masked with the same rule**. You can also give it generation rules so generated data follows the same logic (p.73). The guide says to assign at least one masking rule to each domain.

The guide's Social Security example: the number might be in a column called `SSN`, in a column called `SOCIAL_SECURITY` in another table, or buried in a `COMMENTS` column. One domain describes all three: a data expression for the format `999-99-9999` and metadata expressions for the likely column names such as `SSN` or `Social` (p.73).

If you do not have Data Discovery, you can still use domains to group data, but you must associate columns with domains **manually** (p.73). With discovery, a profile does the assignment for you (the discovery and profiling chapter).

## Preferred rules and the default rule

Rules you attach to a domain are called **preferred rules**. If there are several, you mark one as the **default**, which applies to every column in the domain. You may change an individual column to another preferred rule, and you can even apply more than one rule to a column (p.73).

### GUIDE EXAMPLE, WITH A SHOPSPHERE TWIST

The guide's `Last_Name` domain has a shuffle rule as default and a substitution rule as another preferred rule: shuffle for customer last names in one table, substitution in another (p.73). In ShopSphere: `last_name` in `CUSTOMERS` uses the default, while a hypothetical `shipping_contact_last_name` in another table uses the second preferred rule. Generation rules work the same way: in the guide, `Date_of_Birth` defaults to date random generation and also offers date sequence generation (p.74).

## 4.22.3 Data patterns and metadata patterns

A *regular expression* (regex) is a text string that describes a search pattern; it matches strings of characters in the source data (p.74). TDM uses regexes in two ways, and a domain may hold several of each. A column belongs to the domain if **any** expression matches (pp.74-75).

### Data pattern (look at values)

Describes what the data looks like. SSN example from the guide: `[0-9]{3}-[0-9]{2}-[0-9]{4}`.

Use when you do not trust column names (an SSN hidden in a `COMMENTS` column).

### Metadata pattern (look at column names)

Describes the column name. Guide example: `.*Name*` matches `Name`, `Employee_Name` and `Organization_Name`; `.*SSN*`, `.*SOCIAL*`, `.*SECURITY*` find SSN columns.

Use when names are reliable and values vary.

## A regex primer, using only what the guide lists

Regexes are **case sensitive**. The guide lists these building blocks (pp.74-75), and gives a tutorial site for more.

| Piece               | Guide's meaning                                                                                      | Reading                                                                                                                                      |
|---------------------|------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| Ordinary characters | All except <code>[\^\$. ?*\+()]</code> match themselves; <code>abc</code> matches <code>abc</code> . | <code>ssn</code> matches the letters <code>s</code> , <code>s</code> , <code>n</code> in order (and not <code>SSN</code> , because of case). |
| <code>\</code>      | Escapes a special character so it loses its special meaning.                                         | <code>\.</code> means a real dot.                                                                                                            |
| <code>*</code>      | Matches the previous token zero or more times.                                                       | <code>.*SSN*</code> style names.                                                                                                             |
| <code>[ ]</code>    | One character from a set; a hyphen gives a range, such as <code>[a-zA-Z0-9]</code> .                 | <code>[0-9]</code> is one digit.                                                                                                             |
| <code>?</code>      | Makes the previous item optional.                                                                    | <code>-?</code> an optional dash.                                                                                                            |
| <code>{n}</code>    | Repeats the previous item <code>n</code> times ( <code>n</code> is 1 or more).                       | <code>[0-9]{4}</code> is four digits.                                                                                                        |

### CASE AND SHAPE TRAPS

Because regexes are case sensitive, a pattern in capitals may not match a lower-case column name such as `ssn`. Whether TDM treats metadata expressions case-insensitively is not stated in the guide, so test it on your own data. And a 10-digit phone pattern also matches any other 10-digit column. This is why you review profile results rather than approving everything blindly.

## 4.22.4 Data domain options

A data domain has four options (p.75): **Name**; **Sensitivity level**, applied to every column in the domain and chosen from the levels the administrator defines (the Overview dashboards use it); **Description**; and **Status**, enabled or disabled, where an enabled domain is included in discovery profiles (default enabled).

## 4.22.5 Creating a data domain

### Task: create a "Customer SSN" data domain

Goal: a domain that finds SSN-like data and SSN-like column names in ShopSphere. (p.76)

Assume a Social Security masking rule already exists (see the Informatica masking chapter).

1

**Click Policies.**

**Expected result:** The Policies view lists policies, data domains and rules in the TDM repository.

2

**Click Actions > New > Data Domain.**

This starts the domain wizard.

3

**Enter the name, sensitivity level and description, then click Next.**

For example Name `Customer_SSN` , a high sensitivity level (your administrator's list decides what exists), description "US-style SSN in customer data".

4

**On the data pattern page, optionally enter a regular expression; click the + icon to add more.**

Enter `[0-9]{3}-[0-9]{2}-[0-9]{4}` . This finds the SSN even when the column name is odd.

5

**Click Next for the column-name page (or Finish to stop here), then enter name expressions.**

Enter `.*SSN*` and `.*SOCIAL*` . You can add several.

6

**Click Next to add preferred rules, or Finish to stop.**

To add rules, click Add Rules, select masking and generation rules in the Add Rules dialog, click OK.

7

Enable one default masking rule and one default generation rule, then click **Finish**.

**Expected result:** The new data domain appears in the Policies view.

► Show a worked answer

#### 4.22.6 Copying, editing and deleting a domain

- **Copy:** in Policies, click a data domain *description* to select it without opening it, Actions > Duplicate, change the name and description, Save (p.76).
- **Edit:** click the domain name so it opens in a tab, Actions > Edit, then use the Data Patterns, Metadata Patterns and Preferred Rules tabs, Save. If you delete a rule that has assignments, the Impacted Objects dialog lists affected columns and plans. Click Export to download the list as CSV, then Continue. **Regenerate and run the affected plans** for changes to take effect (pp.76-77).
- **Delete:** select it, Actions > Delete, confirm. Deleting a domain deletes its *assignments* but **not** the rules inside it. A domain with assignments shows Impacted Objects; Continue to proceed, Export for the CSV (p.77).

#### 4.22.7 Policy packs: PII, PHI and PCI

You do not have to start from an empty Policies view. TDM includes three ready-made **policy packs**: PII, PHI and PCI. After installation they are visible in the Policies view, with the data and metadata patterns of their domains, so you can read the actual regular expressions (User Guide 10.5.10, p.77). To view and work with policies you need the Test Data Manager Service **View Policies** and **Manage Policies** privileges, and without View Policies you cannot see policies at all.

The three names are explained in the sensitive data and regulations chapter: PII is personally identifiable information, PHI is protected health information, PCI is payment card information.

## Policy packs and their data domains with default rules (pp.78-80)

| Pack | Industry focus (guide)        | Data domains (each has a default rule of a matching name)                                                                                                                                                                                                                                                                                                |
|------|-------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PII  | Personal information          | Age; Birth Date; Birth Place; Country; Credit Card; Drivers License Number; Email Address; First Name; Full Address UPPER; Full Name; Gender; Grades; IP Address; Job Position; Last Name; Organization Name; Passport; Phone; Salary; School Name; Social Security Number; State; Street Name; UK National Identifier; Vehicle Registration Number; ZIP |
| PHI  | Healthcare and pharmaceutical | Account Number; Birth Date; Certificate License Number; Device Identifier Serial Number; Email Address; First Name; Health Plan Beneficiary Number; IP Address; Last Name; Medical Record Number; Phone; Primary Account Number; Social Security Number; State; Unique Identifier; Web URL; ZIP                                                          |
| PCI  | Banking and finance           | Account Number; Birth Date; Credit Card; Expiration Date; First Name; Last Name                                                                                                                                                                                                                                                                          |

In ShopSphere, the PII pack already has First Name, Last Name, Email Address, Phone, Social Security Number, Birth Date and ZIP, which cover almost every personal column in CUSTOMERS ( `first_name` , `last_name` , `email` , `phone` , `ssn` , `date_of_birth` , `zip_code` ). Nothing in the packs obviously covers `tax_id` : that is a case where you would add your own domain or assign one manually. The guide's pack tables do not list a "tax ID" domain.

### 4.22.8 Importing and exporting policy components

You can export policies, domains and rules to XML files and import them into another TDM repository (pp.80-81). The details matter:

- Exporting a **policy** exports *references* to its domains. Exporting domains exports references to rules. Neither exports the referenced objects unless you choose to export them as well.
- Policies, domains and rules go into **separate XML files**, each with a default name and a date number such as `Policy_121112141309.xml` or `Domain_121112141309.xml` . You can rename them.

- If a rule or policy in the target TDM already **has assignments**, you cannot import the file. Clear the assignment or change the name in the file first.
- If the object already exists, import **overwrites** it.

1

**Export:** in the Policies view tick the objects, click Actions > Export, then Open or Save. Test Data Manager prompts once for each type of file.

2

**Import:** in the Policies view click Actions > Import, browse for the XML file, click Finish.

## Importing data domains from the Model repository

Data domains created in the Informatica Developer tool live in the Model repository. Import them **before** you import a data domain profile (pp.81-82).

1

**Click Policies, then Actions > Import Data Domains.**

**Expected result:** A window lists data domains in the Model repository.

2

**Select the domains, set a sensitivity level for each in the Sensitivity Level column, click Save, then Import.**

If a same-name domain exists, you can overwrite or cancel.

What comes across: **name, description and links to business terms**. What does *not*: data patterns, metadata patterns and preferred rules. So an imported domain is a shell that you finish in Test Data Manager. If you overwrite, TDM updates the description and linked terms but keeps the patterns and rules you set in TDM. Linked business terms are merged, and a term deleted in Developer is kept in TDM until you delete it by hand (p.81).

### 4.22.9 Linking Business Glossary terms

You can link Informatica Business Glossary terms to global objects: policies, domains, masking rules and generation rules (p.82). A business term has a business definition and examples of use, stored centrally so people can look things up. Link a term to a TDM object and you can see the object from the term in the Analyst tool. You need an Analyst Service configured in the Test Data Manager Service properties, as the Informatica architecture chapter listed.

1

**In Policies, select the object** (policy, domain, masking rule or generation rule) and open the **Linked Business Terms** tab.

The tab lists the term, its glossary and a description.

2

**Click Add**, choose the glossary on the Link Asset to Business Term page, select terms and click OK.

You need Manage privileges on the object.

3

**To remove a link:** select the term on the tab, click Delete, then Yes. You can also delete it from the Business Glossary UI (p.83).

Side effects: deleting the TDM object deletes the link; renaming a linked object shows the new name in the glossary; deleting a term from a glossary deletes the link.

## 4.22.10 Policy management: create, copy, edit, delete

You can create a policy before or after you define rules and domains, and add domains later. A copy contains the rules and domains of the original, and a policy can be exported to XML and imported elsewhere (p.83).

### Task: create the ShopSphere customer-data policy

Goal: one policy holding the domains your project will mask. (p.83)

1

**In the Policies view, click Actions > New > Policy.**

**Expected result:** The New Policy dialog box appears.

2

**Enter a name and optional description, click Next.**

For example ShopSphere\_Customer\_PII , "Personal data in customer tables". A clear name matters because projects choose policies by name.

3

**Click Add Data Domains and select domains from the list.**

Choose First Name, Last Name, Email Address, Phone, Social Security Number, Birth Date and ZIP from the PII pack, plus your own Customer\_SSN domain if you made one.

4

**Click Finish.**

**Expected result:** The policy appears in the Policies view. Select it and look at its Data Domains tab to confirm the list.

► Show a worked answer

- **Copy a policy:** in Policies, click the policy *description* (do not open it), Actions > Duplicate, change name and description, Save (p.84).
- **Edit a policy:** click the name, Actions > Edit; change name, description or status; on the Data Domains tab click Add, select, OK, then Save. Deleting a domain that has rule assignments shows Impacted Objects, with Export for a CSV and Continue to proceed. Regenerate and run affected plans afterwards (p.84).
- **Delete a policy:** open it, Actions > Delete, confirm. The domains and rules inside are **not** deleted, but all assignments made through the policy **are**. If assignments exist you see Impacted Objects; Export, then Continue (pp.84-85).

## 4.23 Common mistakes and gotchas

### WATCH OUT

- **Opening a project when you meant to select it.** For Copy and Delete, select the row by clicking its Description or Created On column and do not open it (pp.56-57).
- **Expecting a project copy to contain the tables.** Copy does not import metadata or objects (p.56).
- **Setting log directories that do not exist.** TDM will not create them, and a wrong workflow log directory makes the workflow fail without an error log (p.59).
- **Changing a policy, domain or rule and forgetting to regenerate the plan.** The guide repeats "generate and run the plan again" after every such change.
- **Importing a project before its policies.** Import global components first (p.59).
- **Importing a Model repository data domain and expecting patterns.** Patterns and preferred rules do not come across (p.81).
- **Treating a policy pack as complete.** Check that columns like `tax_id` are covered.

## 4.24 Recap

### KEY POINTS

- Test Data Manager has seven views: Overview, Policies, Data Sets, Projects, Monitor, Parameters and Administrator, with search, quick links, preferences, an Actions menu, a data filter and an Expression Builder on top.
- You log in with an Informatica domain user at `http(s)://host:port/tdm/` ; the guide's defaults are ports 6643 (TLS) and 6605.
- A project holds data sources, entities, groups, parameters, plans, policy and rule assignments and profiles, and has six views including Permissions.
- Import metadata with Actions > Import Metadata from PowerCenter, the Model repository or a database; importing does not copy rows.
- Permissions are Read, Write and Execute, and each task also needs a minimum privilege.
- A policy is a set of data domains; a domain finds columns by data patterns and metadata patterns (regular expressions) and carries preferred masking and generation rules, one marked default.

## Informatica: discovering and profiling your data

Before you can protect data you have to know where it is. ShopSphere's `CUSTOMERS` table has nine columns, so it is easy to see that `ssn` is sensitive. A real production database has hundreds of tables, and a Social Security number can hide in a column called `NOTES` . *Data discovery* is Informatica's name for the automatic search that finds sensitive columns and table relationships.

## 4.25 Words to know first

### Profile

A saved search definition: which tables, which kind of analysis, which sampling limits. You can run it again later.

### **Profiling**

Reading the data (or the column names) to describe it: which columns are unique, which look like SSNs, how many are empty.

### **Primary key (PK)**

A column, or combination of columns, whose values uniquely identify each row. No empty values, no duplicates. Like a library card number.

### **Foreign key (FK)**

A column in a child table that holds the primary key value of a parent row, such as `ORDERS.customer_id` pointing to `CUSTOMERS.customer_id`.

### **Referential integrity**

The rule that every foreign key value has a matching parent row. Without it you get orphan orders.

### **Logical key or constraint**

A key relationship defined only inside TDM, not in the real database. The guide stresses that TDM does not update the source tables.

### **Entity**

A set of related parent and child tables, used to define a subset.

### **Uniqueness percentage**

The share of values in a column that are unique. 100 means no duplicates.

### **Conformance**

The percentage of rows that fit a rule or pattern. A threshold lets a few rows disagree.

### **Curation (review)**

Looking at suggested profile results and approving or rejecting each one.

## 4.26 What data discovery is for

The guide's overview (User Guide 10.5.10, p.86): you run profiles to discover source data for **masking** and **subset**. Before masking, you discover which columns to mask with the same rule. Before building a subset, you discover relationships between tables and the entities that shape the subset database. You can then *apply the results* instead of configuring entities by hand or applying a masking rule one column at a time.

### WHY IT MATTERS

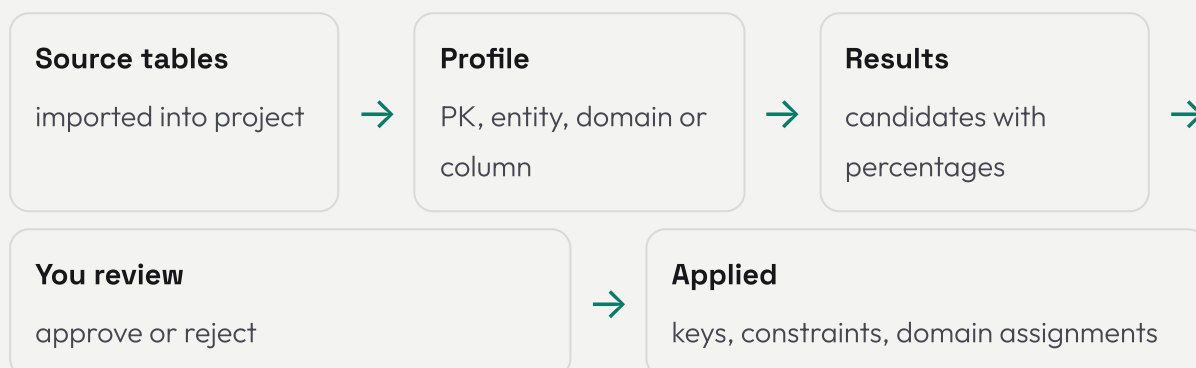
Discovery turns "someone remembers which columns are sensitive" into a repeatable, reviewable record. A compliance officer can see which columns were found, with what confidence, and who approved them. It also supplies the keys that subsetting needs when the database does not declare them.

### 4.26.1 The profile types

The guide names four types on p.86-87. In the screens you will see three of them listed as the options when you create a profile (primary key, entity, data domain) plus column. Note a naming wrinkle: the overview calls the relationship profile "Foreign key", while the creation steps and result screens call it an **entity** profile. They describe the same relationship discovery.

| Profile              | Question it answers                                                   | What you get                                                                                                                                                                       | Used for                          |
|----------------------|-----------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------|
| Primary key          | Which column (or combination) could uniquely identify rows?           | Candidate keys with a uniqueness percentage. You approve those you want, and TDM adds them to the table in the TDM repository, not to the source database.                         | Subset, relationships             |
| Foreign key / entity | Which tables are related, and which could be entities?                | Possible entities and logical constraints; you select entities and approve constraints. TDM stores logical constraints in its repository and does not apply them to source tables. | Subset with referential integrity |
| Data domain          | Which columns hold data of a given kind (SSN, phone, name)?           | Column-to-domain suggestions based on column values or names.                                                                                                                      | Masking, sensitive-data inventory |
| Column               | What do the values in this column look like (range, patterns, lists)? | Inferred rules (range, pattern, list of values) you can reuse to generate data.                                                                                                    | Data generation                   |

You can run data domain and primary key profiles at the same time, and you can import and run profiles you built in Informatica Developer (p.87).



## 4.26.2 Which sources can be profiled

Profiles run on **relational sources**: Oracle, Microsoft SQL Server, IBM DB2 for Linux, UNIX and Windows, MySQL, Sybase and Teradata (ODBC for MySQL, Sybase and Teradata). Data domain profiles also run on Hive and HDFS (User Guide 10.5.10, p.87). Rules (pp.87–88, 94):

- You cannot profile nonrelational sources, but you can profile in PowerCenter or Informatica Data Quality, export the results and import them into TDM. Imported mapplet-based results must use simple regular expressions (no labeler, tokenizer, content set or reference tables).
- A table must use the **same connection** in the profile as when imported, or results may be unexpected.
- One profile cannot hold **two tables with the same name**; make a profile per table.
- No primary key or entity profiles on Hive, and no profiles on Avro or Parquet.

## 4.27 The Discover view

Inside a project, the **Discover** tab is where profiling happens (User Guide 10.5.10, pp.88–89). Its sub-views: **Tables** (tables, relationships, columns; define keys; delete a table; disable a physical primary key you do not need; edit a column's precision), **Profile** (define profiles), **Columns** (maintain domain assignments, mark columns Restricted or Sensitive, assign domains in bulk, define value cascades) and **Files** (XSD files).

### VISUAL REFERENCE

Informatica TDM 10.5.10 User Guide, p.88, Figure "Discover view in a Project". What to look for: the Tables, Profile, Columns and Files sub-tabs, a table list, and a Relationship Overview pane that draws the selected table with its related tables.

## 4.27.1 Column properties

Discover > Columns shows Table Name, Column, Owner (the schema), Data Type, **Domain**, and three flags (p.89):

- **Restricted**: the Integration Service does not write the value to a target.
- **Computed**: the value is calculated from other columns, for example `Net_Pay = Basic_Salary + Allowances - Tax_Deduction`. You cannot assign masking or generation rules or cascades to it, and TDM does not copy data into it because the database recomputes it (p.90).

- **Sensitive:** a label for people only; it does not affect processing. Assigning a data domain marks the column sensitive automatically.

A **Similar Value Columns** count shows how many columns share a value cascade with this one.

## 4.27.2 Value cascades and auto cascades

Both make sure the *same original value* becomes the *same masked value* in several places. That is the difference between a customer whose id looks right everywhere and one whose id no longer matches across tables.

### Auto cascade (keys exist)

Masks a parent table's primary key and the related foreign keys in child tables with the same value. Applied **by default** to tables with key constraints; you can enable or disable it. It needs key constraints between parent and child. Not available for XSD, MongoDB and Cassandra sources (p.91).

ShopSphere: mask `CUSTOMERS.customer_id` and `ORDERS.customer_id` receives the same masked value.

### Value cascade (no keys)

Masks *similar columns in different tables* the same way when they are not linked by key constraints. You group similar columns and mark one as the **driving column**; a rule assigned to it goes to all the others. The columns must have the same type (p.90).

ShopSphere: an `email` column in `CUSTOMERS` and a `contact_email` column in some marketing table.

The guide's example: `employee_num` in one table and `EmpID` in another are similar value columns, with `employee_num` driving. Assign a rule to it and `EmpID` receives the same rule (p.90). The driving column shows a number in Similar Value Columns; the others show none.

1

**Create:** in Discover > Columns select the columns, click Actions > Set as similar value columns, and click the driving-column checkbox to change No to Yes for one column (p.90).

2

**Change:** select the driving column (it has a number), open the Similar Value Columns view in the Details panel, click Edit Similar Value Columns. Set Driving Column to Yes on another column (the previous one switches to No), use Add to add columns or Delete to remove them (p.91).

3

**Delete the group:** select the driving column, open Similar Value Columns, select any column in the group, click Delete, confirm (p.91).

## 4.28 The discovery task flow

Before profiles can run, the **administrator** must configure a source database connection for discovery plus connections to the Data Integration Service and Model Repository Service (User Guide 10.5.10, p.92). The steps (p.92):

1

**Create a profile and choose its type** (primary key, entity or data domain; primary key can combine with the others).

2

**For a data domain profile, choose domains or policies, then choose the sampling size.**

More rows is more reliable and slower.

3

**Run the profile and watch the job in Monitor.**

4

**Close and reopen the profile, review the results, then approve the ones to use.**

The guide says results show only after you reopen it (p.103). Approval applies them.

## 4.29 Primary keys: the basics, then discovery

Nearly every later task, subset above all, depends on keys.

### 4.29.1 Primary and foreign keys in plain words

A library gives each member a card number. No two members share one, and no member is without one. That number is a *primary key*: a column (or combination of columns) that uniquely identifies each row. The guide's rule for keys you add by hand: a primary key column cannot contain null (empty) or duplicate values, and a table has only one primary key constraint (User Guide 10.5.10, p.109).

When a library loan record stores the member's card number, that stored number is a *foreign key*: it points at a row in another table. In ShopSphere:

```
CUSTOMERS (customer_id PK) 1 ----< ORDERS (order_id PK, customer_id FK)
ORDERS    (order_id PK)    1 ----< ORDER_ITEMS (item_id PK, order_id
FK)
PRODUCTS  (product_code PK) (ORDER_ITEMS.product_code refers to it)
```

**Data integrity** means these links stay true: every `ORDERS.customer_id` exists in `CUSTOMERS`, every `ORDER_ITEMS.order_id` exists in `ORDERS`. Test data that breaks this is useless. A subset that copies an order but not its customer produces an orphan. Masking that changes `CUSTOMERS.customer_id` but not `ORDERS.customer_id` breaks the link. TDM relies on keys to avoid both: subset follows constraints "to create subset data with referential integrity" (p.86), and auto cascades mask parent and child keys together (p.91).

#### WHY DISCOVERY OF KEYS MATTERS

Real source systems often have no declared keys, for performance or legacy reasons. In the guide's own example (pp.33-34), the database "does not contain primary and foreign keys", so the project developer runs profiles to *find* them and then creates logical keys. If ShopSphere's imported tables came with no declared keys, you would have to do the same.

## 4.29.2 Primary key discovery

To discover possible keys, create a primary key profile (p.92). It also helps find data quality problems: a profile that finds *non-unique* rows where you expected unique ones shows duplicates. It is useful when the key is a **combination of columns** that together identify a row. After discovery you can add keys to the source table in the project.

The guide's example: an Employee table has an EmpID column that is 100% unique, so the profile identifies it as a possible primary key to add to the project table (p.92). ShopSphere equivalent: `CUSTOMERS.customer_id`.

### Primary key profile options

The Data Integration Service infers candidates and you can limit them (pp.92-93).

| Option<br>(guide<br>label) | Meaning                                                                                                                             | Default                     |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------|-----------------------------|
| Primary<br>Key Rows        | Maximum number of columns the profile may combine to form a primary key. (The label says "rows", but the description says columns.) | 3                           |
| Max Rows                   | Maximum number of rows to profile.                                                                                                  | 1,000                       |
| Conformance<br>Criteria    | Minimum percentage, or maximum number, of key violations allowed.                                                                   | Minimum<br>percentage<br>99 |

## Creating and running a primary key profile

1

**Open the project, click Discover, then click Profile.**

You are in the profile list of this project.

2

**Click Actions > New Profile.**

The New Profile dialog box opens.

3

**Enter a name and description and choose to create a primary key profile.**

For ShopSphere: `pk_shopsphere` . A clear name makes it findable later.

4

**Click Next, select the tables to profile, click OK, then Next.**

Include every table you want keys for.

5

**On the Sampling Options page, enter the maximum number of columns in the key, the maximum rows to profile, and the minimum conformance (or the maximum number of rows that may violate).**

Fewer columns in the key means a faster, simpler search.

6

**Click Finish, then Actions > Execute.**

The run is a job you can follow in Monitor (p.92).

**Expected result:** After it completes, close and reopen the profile to see results.

### Reading primary key results

Click the **Profile > Primary Key** view. Each row is a candidate (pp.103-104).

**VISUAL REFERENCE**

Informatica TDM 10.5.10 User Guide, p.103, Figure "sample primary key profile results".

What to look for: a table of candidates with columns for table, column, relationship type (Primary Key), uniqueness percentage, verified status (NOT\_VERIFIED), existing constraints, and status (for example Approved), with a Properties pane below for the selected row.

| Field                | Meaning                                                                                      |
|----------------------|----------------------------------------------------------------------------------------------|
| Table Name           | The table holding the candidate.                                                             |
| Column Name          | The column or combination inferred to be a primary or unique key.                            |
| Relationship         | The column's role: primary key.                                                              |
| Uniqueness           | Percentage of unique values. 100 means all unique. Use it to decide.                         |
| Verified             | Whether the candidate has been verified as not containing another constraint (for tracking). |
| Existing constraints | Any primary or foreign key constraint already on the column.                                 |
| Status               | Whether a user verified, approved or rejected the candidate.                                 |

## PRIMARY KEY RESULTS FOR SHOPSPHERE (EXAMPLE DATA)

| Table        | Column       | Uniqueness | Existing constraints | Decision                                                                           |
|--------------|--------------|------------|----------------------|------------------------------------------------------------------------------------|
| CUSTOMERS    | customer_id  | 100        | none                 | Approve. It is the identifier and ORDERS points to it.                             |
| CUSTOMERS    | email        | 100        | none                 | Reject as key. Unique today, but a customer can change email, and it is sensitive. |
| CUSTOMERS    | ssn          | 100        | none                 | Reject as key. Sensitive; it will be masked, so it is a poor link.                 |
| CUSTOMERS    | phone        | 99.6       | none                 | Reject. Not fully unique and not an identifier.                                    |
| ORDERS       | order_id     | 100        | none                 | Approve.                                                                           |
| ORDERS_ITEMS | item_id      | 100        | none                 | Approve.                                                                           |
| PRODUCTS     | product_code | 100        | none                 | Approve.                                                                           |

### UNIQUE IS NOT THE SAME AS A KEY

A profile only tests uniqueness within the sample. `email` and `ssn` may score 100 on a small sample and still be unsuitable keys. Pick the column that is the business identifier and is what child tables refer to. The guide's decision aid is the uniqueness percentage plus knowledge of the data.

## Applying the primary key results

Select a row and choose **Verify, Approve or Reject** from the Actions menu (p.104).

- **Approve:** TDM adds the primary key to the table in the TDM repository.
- **Reject:** does not affect the subset operation.
- **Verified** column: only for tracking your review; it does not affect subset operations.

**Expected result:** Expected result: approved keys appear on the table in Discover > Tables, under Constraints as primary key constraints. The source database is unchanged (p.87).

You can skip discovery and add keys by hand. The guide says when you manually add keys, you do not need a primary key discovery profile, and you define constraints for subset, masking and generation without updating the source database (p.109). The step list is in the manual keys section below.

## 4.30 Entity (foreign key) discovery

An *entity* is a parent table plus its related child tables. An **entity profile** finds possible entities by *comparing column names* in the source tables, and returns a list you choose from. The entities you define for a subset determine the structure and content of the subset database (User Guide 10.5.10, p.93).

The guide's example: Employee, Employee\_Detail and Employee\_Address all contain an EmpID column, so the profile suggests Employee as an entity with two direct relationships and two child tables. You select Employee as the entity. Later, a subset filter on EmpID (for example EmpID less than 2000) filters each table in the entity relationship (p.93).

**ShopSphere reading:** `customer_id` appears in `CUSTOMERS` and `ORDERS` ; `order_id` appears in `ORDERS` and `ORDER_ITEMS` ; `product_code` appears in `ORDER_ITEMS` and `PRODUCTS` . The profile can therefore suggest CUSTOMERS to ORDERS to ORDER\_ITEMS as a chain, and ORDER\_ITEMS to PRODUCTS as another link.

### NEEDS KEYS, OR RUN BOTH TOGETHER

If you choose the Entity type and the physical database has **no primary keys**, the profile returns **no results**. If you run it as a *primary key and entity* profile, it discovers possible primary and foreign keys and returns results even without physical keys (p.93).

The creation steps repeat the advice: select primary key and entity profile if the database has no primary keys (p.99).

## 4.30.1 Entity profile options and creation

TDM infers foreign key candidates, and options limit the results (pp.93-94, 99): **Datatypes used in comparisons** (inferred or documented), **Comparison case sensitivity** (the guide warns it can slow the profile), **Trim spaces before comparison** (None, Both, Leading or Trailing), a **minimum confidence or conformance percent** and **Regenerate signature** (clears existing object information in the profiling warehouse first). The p.94 summary and the p.99 dialog list slightly different options, so read the labels on your screen.

Create it like a primary key profile (Discover > Profile > Actions > New Profile, choose entity, select tables, set sampling options, Save, Actions > Execute). Select *primary key and entity* if the database has no primary keys (p.99).

## 4.30.2 Reading entity results

Close and reopen the profile, then open **Profile > Entity** (p.104). Each row is a possible entity. Select one to see the **Constraints** view (parent columns, child columns, relationship) and approve the constraints and the entity you want.

### VISUAL REFERENCE

Informatica TDM 10.5.10 User Guide, p.104, Figure "results of an entity discovery profile". What to look for: a list of profiled relationships, an Actions menu (Approve, Reject, Clear status, Verify checked items) and a Constraints tab with parent and child key columns.

ShopSphere: approve CUSTOMERS.customer\_id to ORDERS.customer\_id ,  
ORDERS.order\_id to ORDER\_ITEMS.order\_id and PRODUCTS.product\_code to  
ORDER\_ITEMS.product\_code . Approved relationships become **logical constraints**: TDM  
stores them in its repository and does not apply them to the source tables; a subset plan uses  
them to keep referential integrity (p.86).

## 4.31 Data domain discovery

A data domain profile finds the source columns that hold similar data and assigns each the same data domain name, so you can apply the same masking rules to all of them at once (User Guide 10.5.10, p.94). Domains are defined with regular expressions on data and on column names (see the data domains section of the Test Data Manager tour). A project must contain **policies** before you create a data domain profile, because policies hold the domains the profile can search for (p.100).

### 4.31.1 Sampling options for a domain profile

| Option                      | Meaning (p.95)                                                               |
|-----------------------------|------------------------------------------------------------------------------|
| Data                        | Search for patterns in the data only.                                        |
| Column Name                 | Search for patterns in the column name only.                                 |
| Data and Column Name        | Search both.                                                                 |
| Maximum Rows to Profile     | Limit rows profiled. Default 1,000.                                          |
| Minimum Conformance Percent | Minimum percentage of rows in which the data or metadata matches the domain. |

Why a conformance percent? Real data is messy. If 97 of 100 rows look like SSNs and three are blank or mistyped, a threshold lets the column still match. The guide says all rows might not conform, so you enter the minimum percentage of profiled rows that must (p.100).

### 4.31.2 Creating and running a data domain profile

1

**Open the project, click Discover, then Profile, then Actions > New Profile.**

You need a policy in the project first.

2

**Enter a name and description; choose a data domain profile.**

For ShopSphere: `dd_customers` .

3

**Select the tables to profile and click OK, then Next.**

Start with `CUSTOMERS` .

4

**In Select Sampling Options, choose to add policies or data domains, then select them.**

Selecting a policy includes all its domains, which is faster to set up than ticking each one.

5

**In the Sampling panel, choose Data, Column Name, or Data and Column Name.**

You can run the profile for column metadata and then again for source data.

6

**Enter the maximum rows to profile and the minimum conformance percent.**

Defaults are a sensible start.

7

**Click Save, then Actions > Execute.**

Watch Monitor; when done close and reopen the profile.

### 4.31.3 Reading and curating data domain results

Click **Profile > Data Domain**. Each row is a source column with a suggested domain (pp.105-106).

#### VISUAL REFERENCE

Informatica TDM 10.5.10 User Guide, p.105, Figure "data domain discovery results".

What to look for: a table of columns with conformance and null percentages and pattern counts, a Current Data Domain and a Profiled data domain column with drop-down lists, and a lower Data Preview tab listing matching data values.

| Column               | Meaning                                                                 |
|----------------------|-------------------------------------------------------------------------|
| Table; Source        | The table and the column to mask.                                       |
| % Data Conformance   | Percentage of rows whose data or metadata pattern matches the domain.   |
| % Null               | Rows that contain NULL in the column (the guide says "number of rows"). |
| Patterns             | How many of the domain's data or metadata patterns match the column.    |
| Column Name Match    | Whether the column name is the matching pattern.                        |
| Current Data Domain  | The previous domain, shown if you profiled more than once.              |
| Profiled data domain | The domain returned by the latest run.                                  |
| Status               | Whether the assignment is verified or approved.                         |

#### DATA DOMAIN RESULTS FOR CUSTOMERS (EXAMPLE DATA)

| Source     | % Data Conformance | % Null | Patterns | Column Name Match | Profiled domain        | Decision                         |
|------------|--------------------|--------|----------|-------------------|------------------------|----------------------------------|
| first_name | 100                | 0      | 1        | Yes               | First Name             | Approve                          |
| email      | 100                | 0      | 2        | Yes               | Email Address          | Approve                          |
| phone      | 100                | 1      | 2        | Yes               | Phone                  | Approve                          |
| ssn        | 100                | 0      | 2        | Yes               | Social Security Number | Approve                          |
| tax_id     | 0                  | 0      | 0        | No                | none                   | Assign by hand (or add a domain) |
| zip_code   | 100                | 0      | 1        | Yes               | ZIP                    | Approve                          |

In the `tax_id` row nothing matched, because no domain describes it. Either create a domain (see the create-a-data-domain steps) or assign one manually below.

- **Data Preview tab:** select a column and click Data Preview to see its source values. The data viewer shows the **first 200 records** of columns returned in the profile (p.105). Use it to sanity-check before approving.
- **Approve:** select rows and approve the domain for each column. Approved domains let you assign the domain's rules to those columns in the **Define > Data Masking** view.
- **Verify:** a tracking column; it does not affect the profile.
- **Mark classification completed:** when you finish approving, mark the data domain classification as completed to confirm you reviewed everything. It affects no process.

## 4.31.4 Assigning domains by hand, and by CSV

Discovery only suggests, so you can correct it (pp.95–96). For one column, click its *Domain* cell in Discover > Columns. For many, select the columns and use Actions > Edit Assignments (a blank domain removes an assignment). You can also use Actions > Export Data Domain Assignments or Import Data Domain Assignments with a CSV (you need read and write permission and the Discover Project privilege). The CSV columns must be, in order, *Data Source Name*, *Owner Name*, *Table Name*, *Column Name*, *Is Sensitive*, *Domain Name*. **The first row is treated as a header and skipped**, so a file without a header loses its first data row. A failed record is logged (Administrator > Application Logs) and the import continues; existing assignments are overwritten. The guide gives no sample values, so export a file first and copy its style.

## 4.32 Column profile (for data generation)

A column profile reports the range of values, list of values and data patterns of the columns you choose, so you can **generate** test data that looks like the real data without copying it (User Guide 10.5.10, pp.87, 97). It runs on Oracle and Microsoft SQL Server. TDM infers three kinds of rule: **Range** (min and max, first and last values, percent null; numeric and date), **Patterns** (patterns with percentage distribution, summing to 100%; string and numeric) and **List of Values** (unique and non-unique values with distribution; string, numeric and date). Inferred rules become **auto rules** you can assign in Define > Data Generation (the Informatica data generation chapter uses them).

Sampling options (p.98): minimum conformance for patterns (default 5), maximum rows (default All), minimum data coverage to show List of Values (default 80) and maximum values in the list (default 20). With a cap of 20 values, those 20 must cover at least 80% of the data.

### 4.32.1 Creating, running and reading a column profile

1

**Discover > Profile > Actions > New Profile, select Column, click Add to choose tables, set the sampling options, Finish.**

2

**Actions > Execute and choose a connection.**

The run fails with a native Oracle ODBC connection (use an Oracle ODBC connection with the DataDirect driver) and fails if a Microsoft SQL Server connection has no schema name (p.101).

3

**Follow it in Monitor, then open Profile > Column Discovery.**

Each column shows % null, an Inferred Rule (select Yes to apply) and a Selected Inferred Type (pattern, list of values or range, depending on data type). Accept, reject or edit the patterns, then save (pp.106-107).

#### VISUAL REFERENCE

Informatica TDM 10.5.10 User Guide, p.106, Figure "column profile results". What to look for: an Inferred Rule drop-down per column and a Properties pane showing range, patterns and list of values with percentages.

### 4.33 Managing profiles

- **Run many times.** Edit a profile between runs; each run **overwrites** the earlier results (p.101).
- **Edit:** open the profile and click Edit. You can rename it (not change its type), add or remove tables, override key inference options, or change domains, policies and what to search. **Delete:** select the row without opening it, Actions > Delete, Yes (pp.101-102).
- **Import a profile from the Model repository** (p.102), for example one run in Informatica Developer on a flat file. Import the source metadata and data domains first, from a connection of the same name as the profiling connection. Use Discover > Profile > Actions > Import Profile, optionally import the *curation information* (saved approvals, which TDM then applies), choose the profile types and Finish. Table names may contain only alphanumerics and underscore. If you delete the project metadata afterwards, the profile shows no results.

### 4.34 Constraints and relationships

Discovery suggests; constraints are what TDM actually uses. Foreign key constraints define parent-child relationships between source tables, and TDM uses them to decide which tables go into a subset and to limit the values in subset columns (User Guide 10.5.10, p.107). When you create an entity you pick one table as the **driving table**, and TDM adds other tables based on the constraints of that table (p.107). The Informatica data subset chapter builds on this.

#### 4.34.1 Table classification

In Discover > Tables you can classify tables as *Temporary*, *Master*, *Configuration*, *Transactional*, *Log* or *Seed*. It does not affect profiles, but you can filter tables by it when creating entities or plans. The guide's reason: knowing which tables are master helps you decide driving tables, and transaction and log tables are candidates for extra filter criteria

(p.107). ShopSphere reading: CUSTOMERS and PRODUCTS look like master tables; ORDERS and ORDER\_ITEMS look transactional.

### 4.34.2 Constraint types

| Type           | Meaning (pp.108-109)                                                                                                     | Notes                                                           |
|----------------|--------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------|
| Primary Key    | Column or combination that uniquely identifies a row; a table can have one.                                              | No nulls or duplicates.                                         |
| Foreign Key    | A column or combination in a table related to a parent column.                                                           | Defined on the child table.                                     |
| Logical Manual | Parent-child relationship based on columns that are not keys. Either accepted from discovery profiles or defined in TDM. | You can delete both kinds.                                      |
| Unique Key     | Column(s) that define a unique key; may contain nulls; you can add more than one.                                        | Also helps bulk generation rule assignment.                     |
| Conditional    | An expression that restricts generated data in a column based on another column.                                         | For data generation; becomes part of the target table metadata. |

### 4.34.3 Major and minor constraint severity

When you define a foreign key constraint you choose a **severity**, which decides how much data a subset pulls in (pp.108-109). The severity does not change the structure of the entity; you can change it after creating an entity and at run time in a subset plan (the run-time change does not change the entity).

#### Major

Moves in both directions, child to parent and parent to child. If a filtered child row selects a parent, **all other children of that parent come along too**. This honours referential and transactional integrity. You can apply a major constraint only from within an entity.

## Minor

One direction only, child to parent. The parent row is included, but **not all its other children**.

**The guide's example:** Customer is the parent of Leads. You filter Leads to high-quality leads. With a major constraint the subset includes the customer for each high-quality lead and all that customer's leads. With a minor constraint it includes the high-quality leads and their customers only (pp.108-109).

**ShopSphere version:** you filter `ORDERS` to `total_amount > 500` . Major: you get those orders, their customers, and *all other orders* of those customers. Minor: you get those orders and their customers only.

- Selecting more records in the child can pull in more parents, which pull in more children, and so on until no related records remain (p.108).
- For a smaller subset, choose the **Referential Integrity Only** option in a plan at run time. The data stays referentially intact, but it may not include every child of a parent, so transactional integrity might not be kept.
- If constraints form **cycles**, or an object has more than one major parent constraint, TDM does recursive evaluation and needs a **staging connection**. Using the source connection as the staging connection is recommended (p.109).

### 4.34.4 Adding keys by hand

You can add keys yourself and skip discovery (p.109). In every case the source database is not updated. Steps are the same up to the type.

#### Task: add a logical primary key and a logical foreign key in ShopSphere

Goal: tell TDM that `ORDERS.customer_id` refers to `CUSTOMERS.customer_id` .  
(pp.109-110)

1

##### Primary key on CUSTOMERS.

In the project click Discover > Tables, click `CUSTOMERS` , click Constraints, then Create New Constraint. Select Primary Key, Next, click Add, choose `customer_id` , Finish.

**Expected result:** The constraint appears in the table's Constraints list.

2

#### Foreign key on ORDERS.

Click **ORDERS** , Constraints, Create New Constraint. Enter a Name, set Constraint Type to Foreign Key, choose Severity Level (Major or Minor), choose Parent Table **CUSTOMERS** , and keep Enable Constraint ticked. Click Next.

3

#### Map the columns.

In the left pane click the child column `customer_id` , in the right pane click the parent column `customer_id` , click the Link icon, then Finish.

**Expected result:** The relationship shows in the table's Relationship Overview.

► Show a worked answer

For a **unique key**, choose Unique Key instead of Primary Key and add columns (pp.110-111). For a **conditional constraint**, enter a name, choose Conditional, click Next, build an expression in the Expression Builder (see the Expression Builder section), and click Finish (p.111).

### 4.35 Performance and accuracy tips from the guide

- **Cap the sample.** Primary key and data domain profiles default to 1,000 rows; column profiles default to *All*, the heaviest setting (pp.93, 95, 98). A small sample is fast but can call a column "unique" or "an SSN" wrongly.
- **Limit key columns** (default 3, p.93) and avoid case-sensitive entity comparisons, which can slow the profile (p.99).
- **Search names first, then data.** You can run a domain profile on column metadata and then on the data (p.100).
- **Pick only the domains you need:** disabled domains are skipped (p.75), and choosing a policy includes all its domains at once.
- **Import metadata directly** from the database when there are many tables (p.60), and use the same connection for import and profile (p.88).

## 4.36 Hands-on labs: profiling, masking and subset on ShopSphere

Starting point for all three labs: a TDM environment where an administrator has already created the database connection to ShopSphere, the Data Integration Service and the Model Repository Service connections (p.92), and you have Write permission on the project (see the permissions section of the tour). The sample rows are the shared ShopSphere rows: customers 1001 Asha Menon, 1002 Ravi Kumar, 1003 Meera Nair.

### Lab 1: TDM and data profiling

Goal: bring ShopSphere into a project, discover sensitive columns, and approve the results. This lab uses the policy and domain built in the guided tour, or the PII policy pack that ships with TDM (p.77).

1

#### Create the project.

Projects > Actions > New; Name `CustomerOrders` , then OK (pp.53-54). A project is where profiles and results live.

**Expected result:** The project opens in a new page.

2

#### Import the four tables.

In the project click Actions > Import Metadata, choose Datasource Connection, pick the ShopSphere schema, select `CUSTOMERS` , `ORDERS` , `ORDER_ITEMS` , `PRODUCTS` , choose Import Now (pp.66-67).

**Expected result:** Import Progress finishes and the tables appear in the Data Sources view.

3

#### Add the policy to the project.

Use the Policies tab on the project Overview (p.50) to add `ShopSphere_Customer_PII` or a PII pack policy. A data domain profile can only use domains from policies in the project (p.100).

**Expected result:** The policy and its domains are listed in the project.

4

#### Create a data domain profile.

Discover > Profile > Actions > New Profile; name `dd_customers` ; type data domain; select `CUSTOMERS` ; choose the policy; sampling Data and Column Name; Maximum Rows 1,000; a Minimum Conformance Percent such as 80; Save (p.100).

Why both data and name: a mismatch between them (an SSN-shaped value in an oddly named column) is exactly what profiling exists to catch.

5

**Execute and monitor.**

Actions > Execute. Open the project's Monitor view and wait for the job to finish (pp.92, 101).

**Expected result:** The job shows a completed status.

6

**Close and reopen the profile, then open Profile > Data Domain.**

The guide says results appear only after reopening (p.103, 105).

**Expected result:** Rows for CUSTOMERS columns with a Profiled data domain column and conformance percentages.

7

**Preview, then approve.**

Select `ssn`, open Data Preview, and confirm the first values look like 123-45-6789. Approve the rows you agree with, and Reject or fix any wrong match (p.105). Then mark the classification completed.

► Show a worked answer

## Lab 2: data profiling and masking

Goal: use the approved domains from Lab 1 to mask `CUSTOMERS` in a test schema. The masking steps are outlined here; the Informatica masking chapter covers rule types and plan settings.

1

**Check that the domains are approved.**

Open Discover > Columns and confirm each personal column has a Domain and is marked sensitive (p.89). If not, finish Lab 1 step 7 first.

**Expected result:** Columns such as `ssn` and `email` show a domain name.

2

**Assign the domain rules in Define > Data Masking.**

The guide says that once you approve a domain for a column, you assign the domain's rules in the Define > Data Masking view (p.105). The default rule of

each domain is applied to its columns; you may switch a column to another preferred rule (p.73).

**Expected result:** Each approved column shows a masking rule in that view.

3

### Handle keys so masking does not break links.

Check that `customer_id` has primary key and foreign key constraints (add them by hand if profiling did not). Auto cascade then masks the parent and child keys with the same value (p.91). If you will mask `customer_id` itself, this is essential.

4

### Create a masking plan.

Project > Execute: create a data masking plan and assign the policy, with a target connection to a *test* schema, never production (pp.31-32). Remember that TDM overwrites existing data in the target schema (p.31).

5

### Generate and run the workflow, then monitor.

Generate the workflow from the plan and run it, then watch Monitor (pp.32-33).

**Expected result:** The workflow job completes; a Row Count Report on the successful run shows rows affected (p.33).

6

### Verify.

Query the test target. The guide's example has the compliance officer validate the results (p.34).

► Show a worked answer

## Lab 3: data profiling and subset

Goal: discover keys and relationships, approve them, and set up a subset of ShopSphere customers. The subset plan itself is outlined; the Informatica data subset chapter gives the details.

1

### Pretend the metadata has no declared keys.

This matches the guide's example database (pp.33-34) and gives profiling something to find. If your imported tables already show keys under Discover > Tables > Constraints, you can disable the physical keys on the Columns tab (p.88-89).

2

### Create a primary key and entity profile.

Discover > Profile > Actions > New Profile; name `pk_entity_shopsphere` ; choose primary key and entity; select all four tables; Sampling Options: key columns maximum 3, Max Rows 1,000, conformance 99 (the guide's defaults, pp.93, 99); Save.

Why combined: an entity profile alone returns nothing when there are no physical keys (p.93).

3

### Execute, monitor, close and reopen.

Actions > Execute; wait in Monitor; reopen the profile (p.103).

**Expected result:** Primary Key and Entity views show candidates.

4

### Approve primary keys.

On Profile > Primary Key, approve the true identifiers ( `customer_id` , `order_id` , `item_id` , `product_code` ) and reject unique-but-wrong candidates like `email` or `ssn` (p.104).

**Expected result:** Approved rows show status Approved; the keys are on the tables in the TDM repository.

5

### Approve entity constraints.

On Profile > Entity, select an entity, check the Constraints view for parent and child columns, and approve the constraints and the entity you want (p.104).

**Expected result:** The logical relationships `CUSTOMERS` to `ORDERS` to `ORDER_ITEMS` exist.

6

### Set the severity.

On the constraints, choose Major for `ORDERS` to `CUSTOMERS` if you want a customer's other orders to follow a filtered order, Minor if not (pp.108-109).

7

### Define the subset structure.

In Define, create an entity with `CUSTOMERS` as the driving table (p.107) so TDM adds `ORDERS` and `ORDER_ITEMS` by constraint, and a group for `PRODUCTS` , so the whole product list is copied as reference data (a group holds tables without tying them to the entity; p.32, p.51). Create a subset plan with a filter on a parent-table column and run it as described in the subset chapter.

► **Show a worked answer**

## 4.37 Common mistakes and gotchas

### WATCH OUT

- **Not reopening the profile.** Results only show after you close and reopen it (User Guide 10.5.10, p.103).
- **Running an entity-only profile on tables without keys.** It returns nothing. Combine it with a primary key profile (p.93).
- **Approving every unique column as a key.** Uniqueness is evidence, not proof. Choose the real identifier that child tables use.
- **Trusting a domain match without a preview.** Use the Data Preview tab (first 200 records) and the conformance percentage before approving (p.105).
- **Expecting profiling to change the source database.** Keys and constraints live in the TDM repository only (pp.86–87, 109).
- **Profiling with a different connection from the one used for import,** or putting two same-named tables in one profile (p.88).
- **Importing a CSV of domain assignments without a header row.** The first row is skipped (p.96).
- **Forgetting that each run overwrites results.** Export or note decisions you want to keep before editing and re-running (p.101).
- **Column profile failing on Oracle or SQL Server.** Native Oracle ODBC connections and a missing SQL Server schema name both cause failures (p.101).
- **Expecting profiles on every source.** Not nonrelational sources, Avro or Parquet, and Hive only for data domains (pp.87–88, 94).
- **Assuming a pattern is case-insensitive.** The guide says regular expressions are case sensitive (p.74).

## 4.38 Recap

### KEY POINTS

- Discovery profiles find what to mask and how tables relate before you configure masking or subset (User Guide 10.5.10, p.86).
- Four profile types: primary key, entity (called foreign key in the overview), data domain and column. Primary key and entity, or primary key and data domain, can run together.
- Primary keys uniquely identify rows; foreign keys point at them. Discovery proposes keys with a uniqueness percentage, and you approve or reject.
- Data domain profiles match column values or names to domains defined by regular expressions; sampling options set the rows and the conformance threshold.
- Results are suggestions: preview, approve or reject, correct by hand or by CSV, and close and reopen the profile to see them.
- Constraints you approve or add (primary, foreign, logical, unique, conditional) live in the TDM repository. Major severity brings all siblings; minor does not.
- Auto cascade keeps parent and child keys masked the same way; value cascades do it for similar columns without keys.
- Labs 1 to 3 chain profiling into masking and subset on ShopSphere.

### Informatica: creating a data subset

A *data subset* is a small slice of production data that still hangs together: if you keep a customer, you also keep that customer's orders and the order lines inside them.

A full copy of production is slow to load, expensive to store and full of real people's data. A random sample of rows is cheap but useless, because a sampled order may point to a customer that was not sampled. A subset gives you the middle path: small *and* consistent.

## 4.39 Words to know first

### Data subset

A small, targeted copy of production data that keeps related rows together, so it can be used in development, test or training.

### Referential integrity

The rule that every foreign key points at a row that exists. An order whose customer is missing breaks it.

### Entity

In TDM, a set of tables that are related by physical or logical constraints, built around one starting table. You filter the entity to say "which customers".

### Driving table

The one table you pick as the starting point of an entity. Also called the "driver table".

### Group

A set of tables that are *not* related to each other (or that you want copied without filtering), added to a plan as they are.

### Template

An optional container that holds entities and groups. The guide mentions it but gives almost no detail (see 13.5).

### Constraint

A rule linking two tables, usually a foreign key from a child table to a parent table. It can be *physical* (declared in the database) or *logical* (added in TDM because the database does not declare it).

### Major / minor constraint

A severity setting on a foreign key. Major moves in both directions (child to parent and parent to child). Minor moves only child to parent.

### Criteria parameter

A named placeholder (for example `$$CRITERIAVALUE` ) used in an entity filter so the value can change at run time without editing the entity.

### Plan / workflow

A plan lists what to do (which entities, groups, rules, connections). A workflow is the runnable job TDM generates from the plan.

## 4.40 What a data subset is, and what TDM can subset

### 4.40.1 The idea in one example

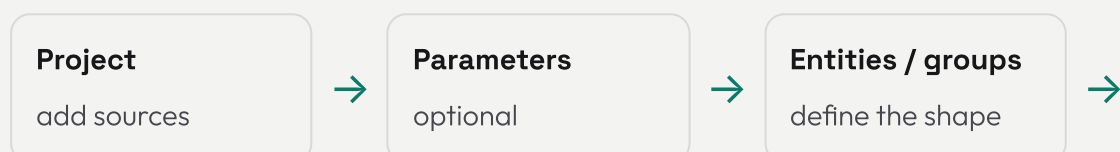
ShopSphere's production database holds 2 million customers. The QA team needs a test environment that holds about 10 percent of them, with every order and order line for those customers, and the products those lines refer to. Nothing else. If you did this by hand you would write SQL joins for each table and hope you did not miss a foreign key. TDM lets you describe the shape once (an entity) and the filter once (criteria), then repeats it on demand.

### 4.40.2 What the guide says

- You create a subset when you need "a small, targeted, and referentially intact copy" of production data for a non-production environment such as development, test or training. The guide's own example is financial data for one region or one time period (User Guide 10.5.10, p.112).
- Sources can be **relational** databases, **flat files** or **XSD** (XML structure) files (p.112).
- You define **data subset components** and add them to a **plan**. You then generate a workflow from the plan, and the workflow loads the subset into a target database (p.112).

## 4.41 The big picture: process flow and components

### 4.41.1 The tasks, in order



**Optimize**

optional

**Plan**

add components

**Run + monitor**

workflow

## 4.41.2 The three component types

A component is a building block you define before you build the plan. The guide names three (User Guide 10.5.10, p.113 to 114) :

| Component | What it defines                                                                           | Source types          | Use it when                                                                                 |
|-----------|-------------------------------------------------------------------------------------------|-----------------------|---------------------------------------------------------------------------------------------|
| Entity    | A set of tables related by physical or logical constraints, starting from a driving table | Relational, flat file | You want related data copied together, filtered by criteria                                 |
| Group     | A set of unrelated tables                                                                 | Relational, flat file | You want some tables copied with no filtering, or tables with no relationship to the entity |
| XSD files | Criteria applied to elements or attributes, plus a "scope"                                | XSD                   | You subset XML data                                                                         |

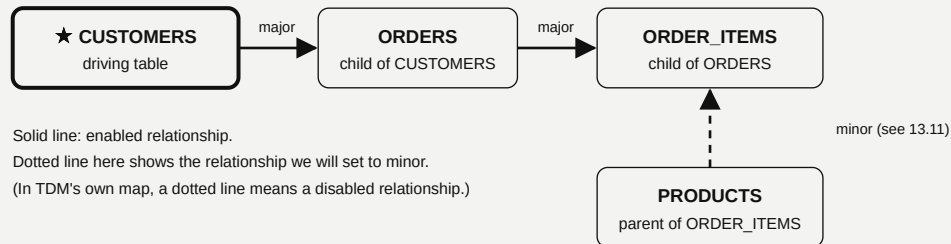
## 4.42 Entities and the driving table

### 4.42.1 What an entity is

An entity is a *shape*: "start at this table, and include everything connected to it". The guide says an entity "consists of a driving table and related tables" (User Guide 10.5.10, p.114) . The driving table is the starting point. TDM works out the other tables from the constraints that involve the driving table and adds them by default. You then put a filter on columns of the tables in the entity to say which rows you want.

Here is the same idea on ShopSphere. We choose `CUSTOMERS` as the driving table because the business question is "give me some customers and everything they did". `ORDERS` points at `CUSTOMERS` through `customer_id` , `ORDER_ITEMS` points at

ORDERS through `order_id`, and ORDER\_ITEMS points at PRODUCTS through `product_code`. (The `product_code` link is assumed for this example: in the source it is a column that names a product, treated here as a foreign key.)



The ShopSphere entity with CUSTOMERS as the driving table. The star marks the driving table, as in TDM's own entity map. Here the dotted line marks "minor"; in the real tool, dotted means disabled (p.120).

#### 4.42.2 The three views of an entity

Once saved, an entity can be seen three ways in the Define | Data Subset view of a project (User Guide 10.5.10, p.114 to 116).

| View          | What it shows                                                                                                                                                     | What you can do there                                                                                                                           |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| Entity Map    | A picture of the tables and links.<br>The driving table has a star.                                                                                               | Click a table to edit its filter.<br>Click a link to edit the constraint. Enable or disable relationships. Pick a sub-view (below).             |
| Tables        | A list: table name, description, database name, and a status (included or excluded).                                                                              | Edit filter criteria for each table.                                                                                                            |
| Relationships | A list of constraints: parent table, child table, database names, and enabled or disabled status. A table with more than one relationship appears more than once. | Enable or disable relationships. Select a relationship and set minimum and maximum child records (the guide ties this to generating test data). |

The Entity Map can show **All tables**, **All parents** of the selected table, **All immediate relationships** (one level up and down), **All children**, or **All cyclic relationships** (TDM shows at most 1000). By default it shows only enabled relationships; tick **Show excluded tables** to see disabled ones (p.115). Selecting a link in the map highlights the matching row in the Relationships view, and the other way round.

### 4.42.3 Where the relationships come from: physical and logical constraints

TDM can only follow links it knows about. Those links are the **constraints** stored in the project (User Guide 10.5.10, p.107 to 109) . Foreign keys that already exist in the database are physical. If the database does not declare a foreign key (common in older systems), you can accept one that data discovery inferred (see the Informatica discovery and profiling chapter), or add one by hand as a *logical* constraint. Adding keys in TDM does not change the source database (p.109). The guide's advice before you build an entity is: run data discovery to identify constraints, and add constraints to the source to define the child tables (p.116).

## 4.42.4 Creating an entity, step by step

Procedure (User Guide 10.5.10, p.116 to 117), with the reason for each step.

1

### **Open the project, then click Define > Data Subset.**

The Define area is where all the "design" objects live: subset components here, masking assignments in Define | Data Masking (see the Informatica masking chapter).

**Expected result:** You see a list of entities and groups in the project, possibly empty.

2

### **Click Actions > New > Entities.**

This starts the New Entity dialog.

3

### **Enter a name and an optional description.**

Names matter later: plans list components by name. Something like

ShopSphere\_Customer\_Entity beats Entity1 .

4

### **Click Select Table, choose the driving table, click OK.**

This is the key decision. Choose the table that answers your business question: "customers" for ShopSphere.

5

### **Click Save.**

TDM now looks at the constraints of the driving table and adds every related table.

**Expected result:** A diagram of the tables and their relationships appears, and you can switch to the list views.

6

### **To filter, click Edit in the Properties pane. The Edit Criteria dialog opens.**

Without a filter, the entity copies everything it reaches. The filter is what makes it a subset.

7

### **On the Entity Criteria tab, click Add Criteria, select the columns to filter on, click OK.**

The columns you chose now appear in the list.

8

**For each column, pick the attribute: Value (type a value) or Parameter (pick a parameter). Pick an operator. Then type the value or choose the parameter. Click Save for each criterion.**

Parameters let you change the value later without editing the entity (13.7).

9

**Click Save to leave the Edit Criteria dialog.**

**Expected result:** The entity is ready to add to a plan.

When you have several criteria in one entity the guide says the filter conditions act as "AND" (p.117). So `customer_id ≤ 1002` and `zip_code = '560001'` together would keep only rows that satisfy both.

## 4.43 Groups, residual tables and templates

### 4.43.1 Groups

A **group** is "one or more unrelated tables" you want copied to the subset database. The guide gives two uses: add unrelated tables to a plan, or "copy unfiltered data to a target" (User Guide 10.5.10, p.116) . When you add tables to a group you can include *residual tables*: tables that are not yet in any group or entity. Tick them when you want every table of the source to be in the subset plan.

An entity only reaches tables linked by constraints, so unlinked tables go in a group. The guide's example: an employee-skills table and an organization-location table are unrelated, you want both, so you put them in one group and add the group to the plan (p.116).

### 4.43.2 Creating a group

1

**In the project, open Define | Data Subset and click Actions > New > Groups.**

Same place as entities, different menu item (User Guide 10.5.10, p.123).

2

**In the Create Group dialog, enter a name and optional description, then Next.**

3

**Click Add Tables, optionally filter the list to search, select tables, click OK.**

The filter box helps when a schema has hundreds of tables.

**Expected result:** The group appears in the Define | Data Subset list next to your entities. Editing a group later lets you add or remove tables (p.124).

### 4.43.3 Templates: what the guide actually says

The guide mentions templates only in passing: "an optional component that contains the entities and groups" (User Guide 10.5.10, p.32 and p.293) . The guide gives no steps for creating a template.

## 4.44 How related rows are pulled

This is the part that makes subsetting more than a filter. When your filter selects customer 1001, which other rows must come with it? The answer depends on the **constraint severity** and on the **data integrity option** you pick in the plan. Both are covered in the guide (User Guide 10.5.10, p.108 to 109, 119 to 123).

### 4.44.1 Major and minor constraints

When a foreign key is defined, you give it a severity. The guide's rules :

- **Major** constraints move in both directions, child to parent and parent to child. A selected child pulls in its parent, and the parent then pulls in *all* of its children. This keeps referential and transactional integrity (p.108).
- **Minor** constraints move one way only, child to parent. The parent row comes along, but the parent's other children do not (p.108 to 109).
- You define the constraint on the child table. Severity does not change the entity's structure. You can change it after the entity exists, and also at run time in a subset plan, which leaves the entity unchanged (p.108).
- **By default every constraint in an entity is major** (p.119).

The guide's own example (p.108 to 109) uses Customer as parent and Leads as child, with a filter on Leads that keeps only high-quality leads. With a *major* constraint, the subset receives each qualifying lead, its customer, and *all* leads of that customer. With a *minor* constraint it receives only the qualifying leads and their customers.

#### MAJOR (both ways)

CUSTOMER 1001

| ^  
v | parent pulls ALL its children  
ORDER 5001 (selected)  
ORDER 5002 (comes along)  
ORDER 5003 (comes along)

#### MINOR (child to parent only)

CUSTOMER 1001

^  
| parent comes along,  
| siblings do not  
ORDER 5001 (selected)  
ORDER 5002 (left out)

### 4.44.2 The loop effect

With major relations the pulling can chain. The guide describes it (p.108, p.122): a selected child record pulls in its parents; each parent then selects all its related children; those children might point to *other* parents not yet in the subset, which pull in more children, "in a loop until there are no more related records to select". This is how a small filter can grow into a big subset. You will see exactly this in the ShopSphere example when `PRODUCTS` is related by a major constraint (13.11).

For schemas with cycles, or objects that have more than one major parent, TDM uses *recursive evaluation* to compute the subset, and that needs a staging connection. The guide recommends using the source connection as the staging connection (p.109).

### 4.44.3 The four data integrity options

The entity says how tables are connected. The **data integrity option** in the plan says how aggressively TDM follows those connections at run time. This lets you reuse one entity to produce different subsets. You choose it in the Advanced Settings area of the Plan Settings page when you create the plan; the field is called Data Integrity for Subset and its default is Transactional Integrity for Major Relation Type (User Guide 10.5.10, p.121, p.287) .

Two terms first. (A *staging connection*, mentioned below, is a database connection TDM uses for temporary working tables.) **Referential integrity** means no child row points at a missing parent. **Transactional integrity** goes further: if a parent is in the subset, *all* of its children are too, so a business transaction (a customer with all their orders) is complete.

| Option                                                    | What it does (guide, p.122 to 123)                                                                                                                                             | Size of result            | Needs / caveats                                                                                                                                                                                                                    |
|-----------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Transactional Integrity for Major Relation Type (default) | Follows major constraints both ways in a loop until no more related rows. Applies criteria to the entity as a whole, not each table. Several criteria are ANDed.               | Can be large              | Recursive evaluation (cycles, more than one major parent) needs a staging connection. If criteria hit mutually exclusive tables the result can be empty; you can switch to OR at plan time (Advanced on the Custom tab).           |
| Referential Integrity Only                                | Picks the fewest rows needed to keep foreign keys valid. A parent may arrive without all of its children.                                                                      | Smaller                   | Meant for an entity with one criterion. With more than one criterion TDM uses the Transactional option anyway. Self-cyclic relations or objects with more than one major parent also fall back to Transactional.                   |
| Two-way                                                   | Criteria propagate downward, then upward, alternately, until every table is covered; the upward step uses OR. Every table that meets the criteria has some data in the result. | Larger than the first two | Needs a staging connection (it may or may not be used). Ignores optimized or manually changed relationship types. Not available with custom Advanced or Limit criteria. With a cyclic relationship it falls back to Transactional. |
|                                                           |                                                                                                                                                                                |                           |                                                                                                                                                                                                                                    |

| Option  | What it does (guide, p.122 to 123)                                                                                                                               | Size of result    | Needs / caveats                                                                                                                                                                                     |
|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| One-way | Applies criteria to individual tables, not the entity as a whole, and propagates downward only.<br>Tables above the topmost filtered table are included in full. | More than Two-way | Same caveats as Two-way about optimization and custom criteria. Falls back to Transactional for cyclic relationships.<br>Recursive evaluation for self-cyclic relations needs a staging connection. |

#### A WORDING PUZZLE IN THE GUIDE

The One-way description (p.123, repeated in the plan settings table on p.287) says One-way returns more data than Two-way, and in the same breath recommends it for "the smallest required subset that does not break referential integrity". Those two statements only fit together if "smallest" means "smallest among the ways that keep every row of the tables above the filtered table". When size matters, run a trial plan and count rows rather than trust the labels.

#### 4.44.4 Which option for ShopSphere?

- **One criterion on CUSTOMERS, want complete customer histories:** keep the default Transactional option. Every order and line of a selected customer comes along.
- **Same criterion, want the smallest consistent database for a quick smoke test:** Referential Integrity Only. You may get a customer with only some of their orders. For a test that checks "order total equals sum of lines" that is fine; for one that counts "all orders for this customer" it is not.
- **Criteria on both CUSTOMERS and ORDERS (for example zip code and order value):** consider Two-way so that both tables end up with data, or edit the criteria to OR (see the table).

Verify the choice on a trial run.

## 4.44.5 Optimizing relations

Because every relationship starts as major (p.119), a subset can pick up more rows than you need. **Optimize Relations** asks TDM to analyse the entity and propose which relationships can be minor while still reaching every table. The guide's procedure (User Guide 10.5.10, p.119 to 120) :

1

### **Make sure the entity already has a criterion.**

TDM raises an error if you optimize an entity with no criteria at entity level (p.119).

2

### **Open the entity and click Actions > Optimize Relations.**

TDM analyses the constraints. The Optimize Relationships page lists each parent and child with the current and the proposed type.

3

### **Optional: Download Optimizations.**

Saves the proposal as a CSV so you can review it with a colleague.

4

### **Click Optimize to accept all proposals.**

You cannot accept some and reject others (the guide says results cannot be applied selectively). A list of plans that include the entity appears; regenerate and rerun them.

5

### **Click Continue.**

**Expected result:** The entity is updated. You can still edit single constraints by hand afterwards.

You can also optimize at run time in the plan (the "Optimize Relationship Type for Entities" setting, p.283). That changes the run, not the saved entity, and it also requires criteria at the plan level (p.283). When you choose Two-way or One-way integrity, optimization is ignored (p.122 to 123).

## 4.44.6 Disabling and enabling relationships

Sometimes you do not want TDM to follow a link at all, for example a link to a huge audit table. In the Entity Map or the Relationships view you can disable a relationship (p.120) . In the map, choose to view all relationships and show excluded tables; enabled links are solid, disabled links are dotted. Click the link, then Disable Relationship or

Enable Relationship. If a table loses every link to the entity this way, it is excluded from the subset database (p.115). If you still want its data, put it in a group.

#### 4.44.7 When constraints change after you built the entity

Entities do not update themselves. If you add a constraint after building the entity, the entity list marks it as having a change impact. Open it, click **Review Changes**, accept the relationship changes you want, then use Actions > Mark Changes Review as Complete so the notice goes away (p.120 to 121) . The guide's figure on p.121 shows a dialog with Added, Changed and Deleted groups and an Accepted action.

### 4.45 Choosing rows: criteria, operators, parameters and limits

#### 4.45.1 Entity filter criteria and operators

You met entity criteria while creating an entity (13.4). A criterion is: a column, an *attribute* (Value or Parameter), an *operator*, and the value or parameter. The guide's examples use "greater than", "equals" and "IN" (p.118, p.119, p.127). The full operator list is not documented, so do not assume more than those three.

##### SHOPSPHERE EXAMPLES

- `customer_id ≤ 1002` : keeps the first two customers. Easy to predict, good for a lab.
- `zip_code = '560001'` : keeps all customers in one postal area, a "region" subset like the guide's financial-data-for-one-region example.
- `customer_id > $$CRITERIAVALUE` : the same, but the number comes from a parameter (next section).

#### 4.45.2 Subset with parameters

Typing the value into the entity means editing the entity every time the requirement changes. A **criteria parameter** lets you change the value when you run the workflow. The guide (User Guide 10.5.10, p.113, p.118, p.57) says:

- Create criteria parameters in the project (Projects, open the project, Parameters tab, Add) or use *global* parameters an administrator created. Global ones show an asterisk (\*) in lists, and you can import one into the project first (p.57 to 58).
- Parameter types are criteria, connection and owner. Criteria parameter names begin with `$$` , connection ones with `$DBConnection` , owner ones with `$Param` .

Data types are String, Numeric or Date. For a Date parameter you enter an expression that says how it is used, for example `TO_DATE($$CRIT1, 'MM-DD-YYYY')` (p.57).

- You choose a default value and whether overriding is allowed. If you do not allow override, you cannot change the value when you run the workflow (p.57).
- Values can live in the project or in an external **parameter file**. The guide's example: a parameter `CRITERIAVALUE`, a parameter file line `$$CRITERIAVALUE=100`, and an entity criterion `EMPID > CRITERIAVALUE`. To get EMPID greater than 500, change the file to 500 and run the same workflow (p.118).
- For a string or date parameter on a relational source, put the value in **single quotes** in the parameter file (p.118).

**Ignore flags.** If you keep values in a parameter file you create yourself, add a flag for each criteria parameter, for example `$$CRITERIAVALUE_IGNORE`. "N" means use the criterion, "Y" means the workflow skips it. A file created from Test Data Manager already contains the flags, set to "N" (p.118, p.309 to 310).

To make the plan accept parameter values, tick **Treat the subset criteria as a variable** in the plan's connection options and give the parameter file name and path (p.279). At run time, in the Execute Workflow window, you choose where values come from (p.310 to 311): the default file named in the plan settings, a parameter file you browse to, or *Add Parameter Values* from the project. Criteria parameters do not appear in that list by default: click Select Parameter and add them, because "a workflow does not include parameters that do not appear in the list" (p.311).

#### **TWO SILENT WAYS TO GET THE WRONG SUBSET**

(1) You forget to select the criteria parameter in the Execute Workflow window, so the criterion is not included. (2) You edit a parameter that was created without "allow override": the change does not take effect and the workflow uses the default value and logs an error (p.311). Check the workflow log if the row counts look wrong.

### **4.45.3 Database functions in entity criteria**

You may want a date rule such as "hired in the last three months". The guide (p.118 to 119) warns that writing a database function directly in the criterion makes workflow generation fail, because TDM does not recognise it as a function and cannot pass it on.

The documented workaround: choose the operator **IN** and write a SELECT query in the Value field, for example `select HIREDATE from <table> where HIREDATE > add_months(current_date, -3)` . TDM then passes the query to the database. The ShopSphere version of this idea would be a query on `ORDERS.order_date` .

#### 4.45.4 Filtering again inside the plan

You can also set criteria in the plan itself without editing the entity (User Guide 10.5.10, p.294, p.304) . In the plan's component list you pick a subset component, then a column in the Criteria panel. TDM shows existing criteria, which you can change or add to. You then choose a method:

| Method | Option         | What it does                                                                                                                                                                                         |
|--------|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Custom | Basic          | A simple expression with an operator and an operand. You can use parameters.                                                                                                                         |
| Custom | Advanced       | An expression typed in the Expression Editor. It can use several columns, functions and parameters, including global ones (a global parameter inside an expression is written without the asterisk). |
| Limit  | (a limit type) | Limit the subset result by <b>percentage</b> , by <b>absolute value</b> or by an <b>interval of rows</b> to write to the target.                                                                     |

The plan creation steps put it like this: "To limit the subset results, click Limit and choose to limit by percentage, by absolute value, or by defining an interval of rows to create" (p.304). You cannot override the subset criteria of an XSD component in the plan (p.304).

##### WHAT THE GUIDE DOES NOT TELL YOU ABOUT LIMIT

The guide names the three limit types and nothing more. It does not say which table the percentage applies to, how rows are chosen (first N, random?), what the "interval" arithmetic is (every Nth row? a range?), or how a Limit interacts with parent-child pulling. Test on a small table and count the result before relying on it. The guide also says Two-way and One-way integrity are not available when a plan uses custom Advanced or Limit criteria (p.123).

- **Percentage:** for example 10 percent. Natural fit for the ShopSphere goal, but check how TDM applies it to related tables.
- **Absolute value:** for example 500 rows. Handy when the test environment has a hard size cap.
- **Interval:** a defined interval of rows. The guide gives no example.

Ways of choosing rows: by *condition* (entity criteria and plan Basic/Advanced criteria), by *parameterised condition*, and by *quantity* (Limit).

#### 4.45.5 Subset and masking in one plan

A plan can hold both subset components and masking components. The guide says the subset component gets priority: TDM does the subset first, then applies masking rules to the columns of the subset data. Masking is not applied to source data that is outside the subset, "even if the data is part of a cascade" (User Guide 10.5.10, p.275) . You can also keep them in separate plans (p.304). the Informatica masking chapter covers the masking half.

#### 4.46 Subsetting XML: XSD sources in brief

XML data is a tree, not a set of tables, so entities and driving tables do not apply. Instead you put a filter on an *element* or *attribute* and set a **scope**: the level of the tree at which the filter is applied. You can apply one filter per hierarchy, and the scope changes the outcome (User Guide 10.5.10, p.114, p.124) .

The procedure (p.124): import the XML source definition from the PowerCenter repository into the project; click Discover > Files; open the XSD file; select an element or attribute; on the Criteria tab of the Details tab click Edit; choose an operator; enter a value (you *cannot* use a parameter here); choose the scope; click OK. Then add the XSD file to a plan and run it.

The guide's worked example (p.127 to 134): a master XML file lists departments, each with an address, employees and contractors, each with a name, address and skill. Test group A wants employees skilled in Java; group B wants departments located in BLR.

- **Group A:** criteria on the element `Skill` on the Employee branch, operator Equals, value JAVA, scope Employee. The output has all Java employees. Because Department is an *ancestor* of Employee, the departments that have such employees come along with all their child elements.
- **Group B:** delete the Skill criterion, then put a criterion on `City` : Equals, BLR, scope Department. The output has every department whose address city is BLR,

with all child elements.

#### **VISUAL REFERENCE**

Informatica TDM 10.5.10 User Guide, p.127, Figure (Criteria tab of the file page). Shows a dialog titled "Edit - Skill" with tabs General, Data Masking, Data Generation and Criteria. Criteria is selected and shows three fields: Operator set to Equals, Value set to JAVA, Scope set to Departments/Department/Employee, with OK and Cancel buttons.

The lesson carries over to relational data: the level where you filter decides which ancestors and descendants appear. For the ShopSphere tables the equivalent of "scope" is your choice of driving table and relationship types.

### **4.47 Managing subset components**

Components live in the TDM repository and are reused by plans, so changing them has side effects. The guide's housekeeping tasks (User Guide 10.5.10, p.124 to 126) :

| Task   | How                                                                                                                                                                  | Watch out for                                                                                                                                            |
|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Edit   | Define   Data Subset, open the entity or group, Actions > Edit.<br>General tab: name, description, status. Entity Criteria tab (entity) or Tables tab (group).       | After saving, a list of plans that use it appears. Those plans must be generated and run again to pick up the change. Optionally Export the list to CSV. |
| Export | In the project, tick the component, Actions > Export, choose a file name and path.<br>Output is an XML file; the default name has the component type, date and time. | Use it to move a component to another TDM repository.                                                                                                    |
| Import | In the Projects view select the target project without opening it, Actions > Import, browse to the XML, Finish.                                                      | The guide says to import into a different TDM repository's project.                                                                                      |
| Copy   | Select the component without opening it, Actions > Duplicate, change name and description, Save.                                                                     | Good for creating a "10 percent" and a "1 percent" variant.                                                                                              |
| Delete | Select, Actions > Delete, OK.                                                                                                                                        | If the component is assigned to a plan, that plan becomes invalid. A list of affected plans is shown, which you can export.                              |

## 4.48 The whole workflow, start to finish

The guide's own summary of "Creating a Data Subset" is seven steps (User Guide 10.5.10, p.126) .

1

### Create a project and add the data sources.

A project is the workspace that holds sources, parameters, components and plans. The guide advises separate projects for XSD sources and relational

sources. If you ran discovery in the Informatica discovery and profiling chapter, it is the same project.

**Expected result:** The tables of the source appear in the project, with constraints from physical keys.

2

**Optional: create connection, owner or criteria parameters.**

Do this now if you want the same plan to serve different values later (13.7).

3

**Create the subset components for your source type.**

Relational and flat file: entities and groups. XSD: criteria with scope on elements and attributes.

**Expected result:** Entities and groups listed under Define | Data Subset.

4

**Optional: optimize the relationships in the entity.**

Reduces over-pulling by turning unneeded major relationships into minor ones (13.6).

5

**Create a plan and add the subset components and data sources.**

The plan ties together what to copy, where from and where to. Choose the integrity option, connections (source, target and a staging connection if needed), and any plan-level criteria or Limit.

6

**Run the workflow.**

Generate a workflow from the plan (the PowerCenter Repository Service generates the mappings) and execute it (the PowerCenter Integration Service does the extraction and load). You can do both in one step with Actions > Generate and Execute (p.308, p.310).

7

**Monitor progress.**

Watch status in the Workflow Executions view: In Queue, Running, Success or Error. Click the Job ID to open the workflow log (p.311 to 312).

1 Project



2 Parameters



3 Entities /  
groups



4 Optimize



5 Plan



6 Run



7 Monitor

Related: the Informatica plans and workflows chapter (plans and workflows), the Informatica discovery and profiling chapter (connections and constraint discovery) and the Informatica TDM tour (screens).

## 4.49 Worked example: ShopSphere down to 10 percent

### 4.49.1 The goal

ShopSphere's QA team wants a test database with about 10 percent of customers, with all their orders and order lines, and the products on those lines. The source has 20 customers (1001 to 1020), 36 orders, 80 order lines and 12 products. Target: 2 customers.

### 4.49.2 Before: the source rows we care about

SHOPSPHERE\_PROD.CUSTOMERS, FIRST THREE OF 20 ROWS (FAKE DATA)

| customer_id         | first_name        | last_name | email                      | zip_code |
|---------------------|-------------------|-----------|----------------------------|----------|
| 1001                | Asha              | Menon     | asha.menon@examp<br>le.com | 560001   |
| 1002                | Ravi              | Kumar     | ravi.k@example.com         | 600017   |
| 1003                | Meera             | Nair      | meera.n@example.c<br>om    | 682001   |
| ... 1004 to<br>1020 | 17 more customers |           |                            |          |

#### ORDERS AND ORDER\_ITEMS ROWS FOR CUSTOMERS 1001 AND 1002 (EXAMPLE DATA)

| order_id | customer_id | item_id | product_code | quantity | unit_price |
|----------|-------------|---------|--------------|----------|------------|
| 5001     | 1001        | 9001    | P-100        | 1        | 999.00     |
| 5001     | 1001        | 9002    | P-205        | 2        | 249.50     |
| 5002     | 1001        | 9003    | P-205        | 1        | 249.50     |
| 5003     | 1001        | 9004    | P-310        | 3        | 40.00      |
| 5004     | 1002        | 9005    | P-100        | 1        | 999.00     |
| 5004     | 1002        | 9006    | P-402        | 1        | 59.00      |
| 5005     | 1002        | 9007    | P-310        | 2        | 40.00      |

(Order totals in ORDERS would be 1498.00, 249.50, 120.00, 1058.00 and 80.00. Customer 1003 and the others have their own orders that we want to leave behind.)

### 4.49.3 Steps

1

#### Confirm the constraints exist.

An entity follows constraints. Open the project's tables and check that ORDERS to CUSTOMERS, ORDER\_ITEMS to ORDERS and ORDER\_ITEMS to PRODUCTS are listed. If the PRODUCTS link is not declared in the database, add it as a logical foreign key in TDM (p.109 and p.116; this does not alter the source).

**Expected result:** Three relationships visible, each with Input Type physical or logical.

2

#### Create a criteria parameter.

Projects, open the project, Parameters, Add. Name `MAX_CUST_ID`, type criteria (TDM builds the name `$$MAX_CUST_ID`), data type Numeric, default 1002, override allowed (p.57).

**Expected result:** The parameter is listed in the project. Setting 1002 gives 2 of 20 customers, which is 10 percent.

3

**Create the entity.**

Define | Data Subset, Actions > New > Entities, name

ShopSphere\_Customer\_Entity , Select Table = CUSTOMERS, Save (p.116 to 117).

**Expected result:** The map shows a star on CUSTOMERS and links to ORDERS, ORDER\_ITEMS and PRODUCTS. Every link is major by default (p.119).

4

**Add the filter.**

Edit, Entity Criteria, Add Criteria, choose `customer_id` . Attribute

Parameter, operator less-than-or-equal, parameter `MAX_CUST_ID` . Save (p.117).

5

**Deal with PRODUCTS before you run anything.**

See "Option A versus a major relationship" below. Set the ORDER\_ITEMS to PRODUCTS relationship to **minor** by clicking the link in the Entity Map and changing Type (p.117), or use Actions > Optimize Relations (p.119).

**Expected result:** The link is marked minor; the entity needs a criterion first for Optimize Relations to work.

6

**Create the plan.**

Execute, Actions > New, enter a name, click Add Subset Components, select the entity, Next. Set source and target connections and a staging connection (recommended: the source connection). Tick "Treat the subset criteria as a variable" and give the parameter file name and path. Keep Data Integrity for Subset at its default, Transactional Integrity for Major Relation Type (p.279, p.287, p.304 to 305).

**Expected result:** The plan appears in the project.

7

**Generate and execute.**

Open the plan, Actions > Generate and Execute, choose the Integration Service and run now. In the parameter section, select `MAX_CUST_ID` using Select Parameter so it is included, and keep the value 1002 (p.310 to 311).

**Expected result:** Status moves from In Queue to Running to Success in the Workflow Executions view.

**Verify the target.**

Count rows and check there are no orphans (SQL below). TDM's job is to keep integrity, but you confirm.

```
-- Row counts in the target (general SQL, adjust schema names)
SELECT 'CUSTOMERS' t, COUNT(*) n FROM CUSTOMERS
UNION ALL SELECT 'ORDERS', COUNT(*) FROM ORDERS
UNION ALL SELECT 'ORDER_ITEMS', COUNT(*) FROM ORDER_ITEMS
UNION ALL SELECT 'PRODUCTS', COUNT(*) FROM PRODUCTS;

-- Orphan check: order lines whose order is missing (expect 0)
SELECT COUNT(*) FROM ORDER_ITEMS i
LEFT JOIN ORDERS o ON o.order_id = i.order_id
WHERE o.order_id IS NULL;
```

**4.49.4 After: what lands in the target****ROW COUNTS, SOURCE VERSUS TARGET (EXAMPLE DATA)**

| Table       | Source | Target with minor<br>PRODUCTS link | Target with major<br>PRODUCTS link |
|-------------|--------|------------------------------------|------------------------------------|
| CUSTOMERS   | 20     | 2 (1001, 1002)                     | 15                                 |
| ORDERS      | 36     | 5 (5001 to 5005)                   | 31                                 |
| ORDER_ITEMS | 80     | 7 (9001 to 9007)                   | 70                                 |
| PRODUCTS    | 12     | 4 (P-100, P-205, P-310, P-402)     | 12                                 |

With the minor link, the target holds exactly the two customers, their five orders, seven lines and the four products on those lines. Meera (1003) and the other 17 customers are gone, and nothing points at a missing row.

#### 4.49.5 Option A versus a major relationship

Why change PRODUCTS to minor? With the default (all major), the loop from 13.6 applies: line 9001 pulls in its parent product P-100. As a major parent, P-100 then selects *all* order lines for P-100, from every customer. Those lines pull their orders, the orders pull their customers, and so on, until nothing new is added. A popular product like P-100 can connect nearly everyone, so "10 percent of customers" turns into most of the database (the right-hand column above). Minor means: bring the product along, but do not let the product pull its other lines.

```
Major PRODUCTS link
customer 1001
  order 5001 → line 9001
    product P-100 --pulls→
here)
  ALL lines with P-100
    → other orders
    → other customers
```

```
Minor PRODUCTS link
customer 1001
  order 5001 → line 9001
    product P-100 (stops
```

#### 4.49.6 Option B: reference table as a group

PRODUCTS is small and looks like reference data. Another valid design: disable the ORDER\_ITEMS to PRODUCTS relationship in the entity (p.120) so the entity is just customers, orders and lines, then create a **group** containing PRODUCTS (tick it from the residual tables, p.116) and add both the entity and the group to the plan. The target gets all 12 products (unfiltered) and still has no orphan lines. The trade-off: 8 unused products travel along, which is harmless here but would matter for a 2-million-row table. Pick Option A for big parent tables and Option B for small reference tables.

To mask the customers just selected, see the Informatica masking chapter.

#### 4.50 Lab: subsetting ShopSphere

##### Lab: subset ShopSphere to about 10 percent of customers

Goal: after profiling the source for constraints (see the Informatica discovery and profiling chapter), build an entity with CUSTOMERS as the driving table, filter with a parameter and run a plan so the target has the first 10 percent of customers with their orders, order lines and products. Use the lab environment's real database and table names.

1

Check the source.

Query counts on the four tables and write them down. You need them for the before and after comparison.

**Expected result:** For example 20, 36, 80, 12 on the sample, or your lab's own numbers.

2

### **Confirm or add the constraints.**

Open the project's tables and check the three relationships. Add any missing logical foreign key.

3

### **Create parameter `MAX_CUST_ID` , then the entity and its criterion.**

Follow steps 2 to 4 above. In your lab replace 1002 with the 10th percentile ID of your own data.

4

### **Switch the `PRODUCTS` link to minor (or use Option B).**

Record which you picked and why.

5

### **Create the plan, generate and execute it.**

Use Generate and Execute and select the criteria parameter in the run window.

**Expected result:** Status Success.

6

### **Verify.**

Run the count and orphan queries on the target and fill in a before/after table.

7

### **Experiment.**

Change the parameter value to 1004 and rerun the *same* workflow: the point of parameters is that nothing else needs editing. Then try the default (all major) `PRODUCTS` link and note how the counts change.

► **Show a worked answer**

## 4.51 Common mistakes and gotchas

### WATCH OUT FOR THESE

- **No criterion, huge subset.** An entity without a filter copies everything it reaches.
- **Everything is major by default,** so a popular parent table can balloon the result (p.119, 13.11). Optimize relations or set minor.
- **Missing constraints.** If the database does not declare a foreign key, TDM will not follow it until you add a logical constraint (p.109, p.116).
- **Editing without rerunning.** After you edit or delete an entity, group or rule, the plans that use it do not pick up the change until you generate and run them again (p.124, p.275).
- **Using a database function directly in a criterion.** Workflow generation fails; use IN with a SELECT clause (p.118 to 119).
- **Parameter not selected or not overridable.** Criteria parameters must be added in the Execute Workflow window, and non-overridable ones ignore your edit (p.311).
- **Forgetting quotes on string or date parameter values** in the parameter file (p.118).
- **Cycles and a missing staging connection.** Cyclic or multi-major-parent structures need one (p.109, p.122).
- **Mutually exclusive criteria give an empty subset.** Several criteria are ANDed; use OR in the plan's Advanced criteria if you want either (p.122).
- **Treating the Limit like a magic 10 percent.** The guide does not describe how it interacts with related rows. Test it.

## 4.52 Recap

### KEY POINTS

- A data subset is a small, referentially intact copy of production data; TDM can build one from relational, flat file or XSD sources.
- An entity is a driving table plus the tables related to it by constraints; a group is a set of unrelated tables copied as they are; a template is an optional container the guide barely describes.
- Filter criteria, with an operator and either a value or a parameter, choose rows. A plan can add Custom (Basic, Advanced) criteria or a Limit by percentage, absolute value or interval.
- Major constraints move both ways and can pull in far more than you expected; minor constraints go child to parent only. Optimize Relations suggests which can be minor.
- Integrity options (Transactional, Referential only, Two-way, One-way) trade completeness against size.
- The workflow: project, parameters, components, optional optimization, plan, run, monitor.