Introduction to Data Model Online Training

Abstract:

This training will introduce you to the DELMIA Apriso data model, it will show you how data is structured in the database and how information can be found on the data model.

Detailed objectives. After the training you will:

- Have a general overview of the DELMIA Apriso data model and will be able to access the documentation
- Understand the data model delivered in the standard product
- Know how to identify the specifics of the data model for a given implementation on project

Target audience:

- New consultants working with DELMIA Apriso

Requirements:

- Familiarity with Process Builder user interface
- Familiarity with MS SQL

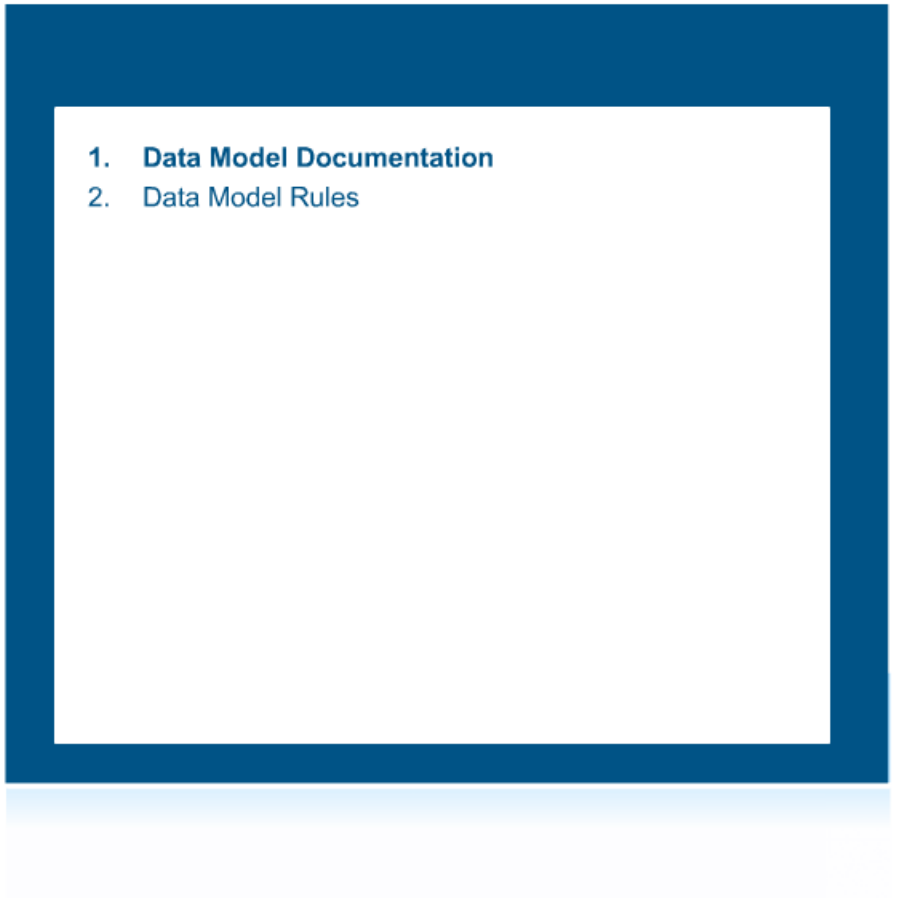
Role and level:

- DELMIA Apriso users that will be working with more advanced screens

Duration: 60 min

Chapter 1: Overview

In this chapter you will get the basic overview of what Data Model is in regards to DELMIA Apriso. Later the Documentation Model will be presented and more details about DM will follow.



Overview

The DELMIA Apriso data model is an abstract scheme that organizes all data in DELMIA Apriso and standardizes how they relate to one another and to the elements they represent in the real world.

Usually, the data model is composed of 1200+ tables that serve all Apriso's modules, e.g.:

- Production
- Logistic
- Quality
- Etc.

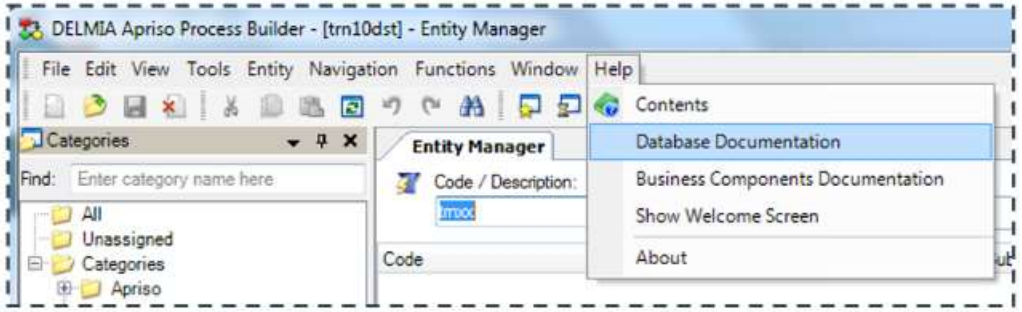
But it can vary between the modules as well.

It supports the usage of both Microsoft SQL servers and Oracle servers, however MS SQL is more frequently used.

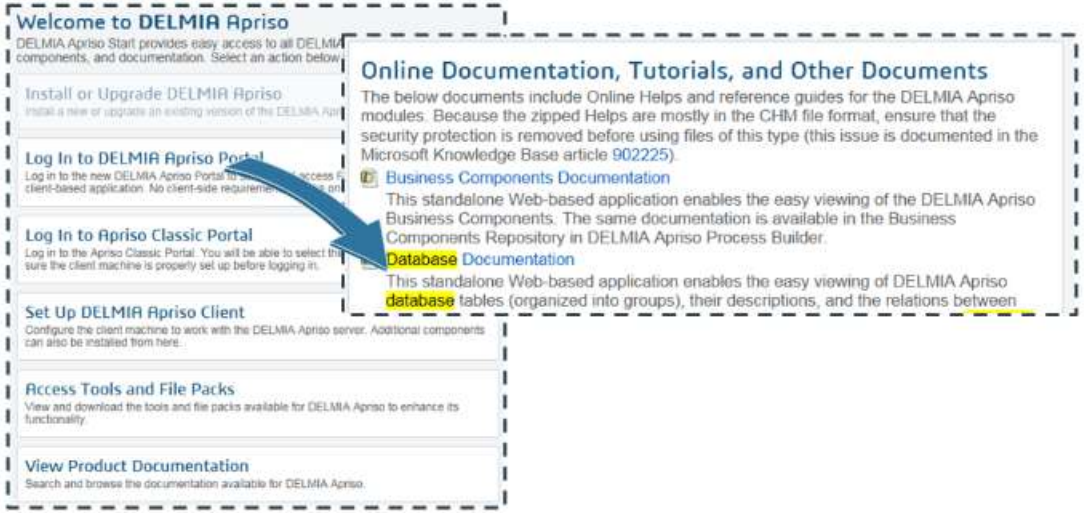
Data Model Documentation

The DELMIA Apriso Data Model is well-documented and can be accessed in multiple ways:

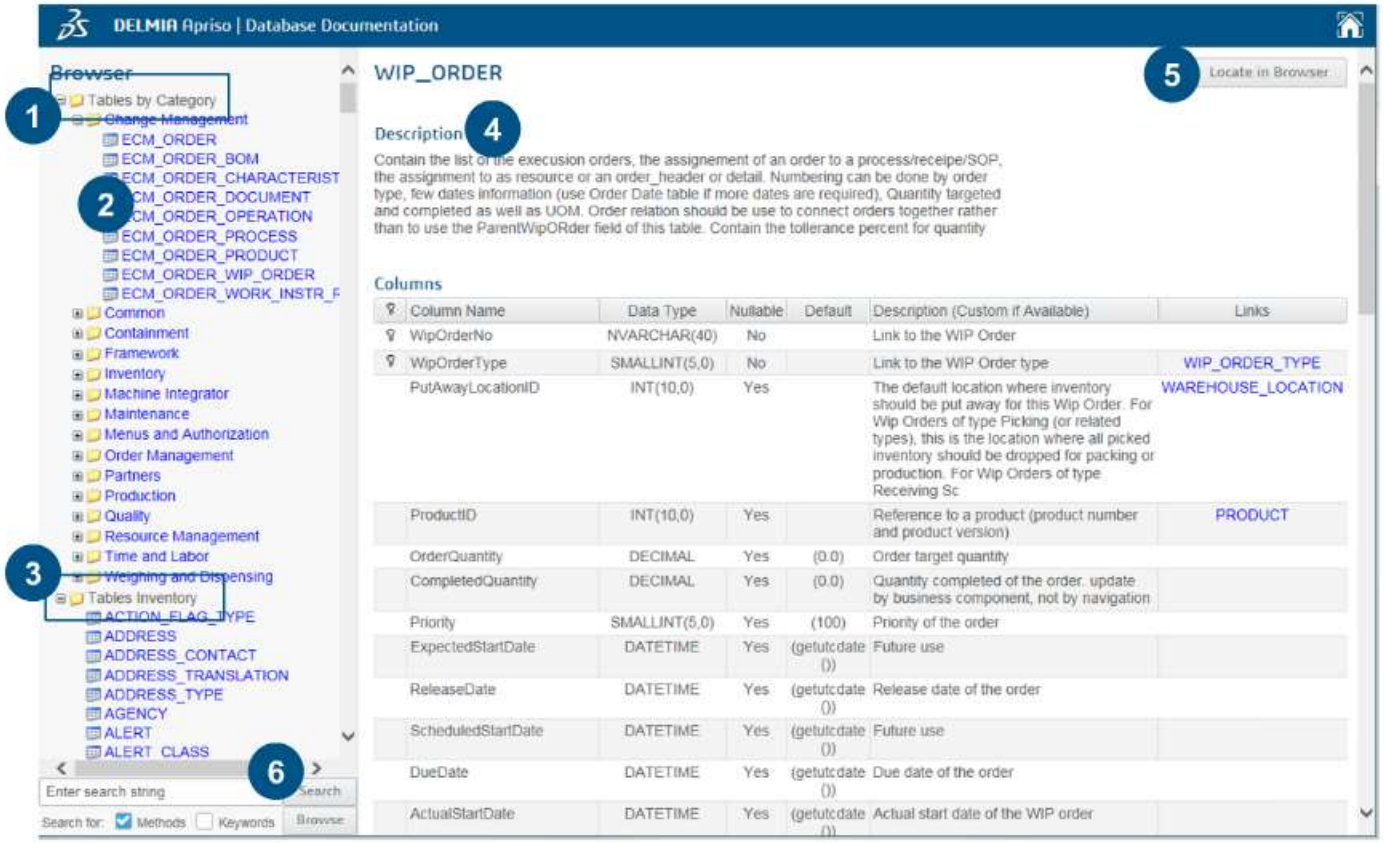
- Process Builder:
  - In Process Builder, the documentation is accessible through the Help -> Database Documentation



- DELMIA Apriso Portal:
  - Use <server_name>/Apriso/Start address and then View Product Documentation -> Database Documentation (link to an online version)



Data Model Documentation Explained



1 - Browse tables by category
2 - Table name
3 - Browse tables in alphabetical order
4 - Description Panel
5 - Switch to see currently displayed table in tree-structure browser (1)
6 - Search window

Table Category Explained, part 1

Under the category folder, you can access the Data Model organized by functional elements.

There are descriptions at the folder level. For instance, when selecting category Common/Alerting you will get:

- A short description of the category that displays information about the purpose of the tables included in this particular category
- A table relation diagram overview of the tables involved in this category

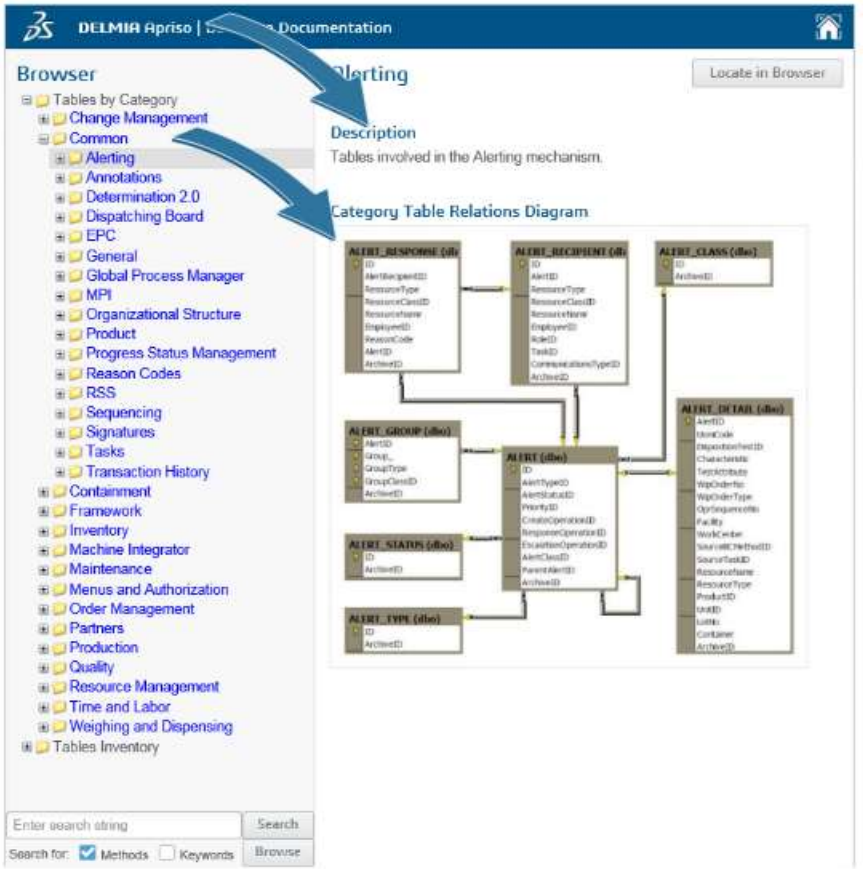
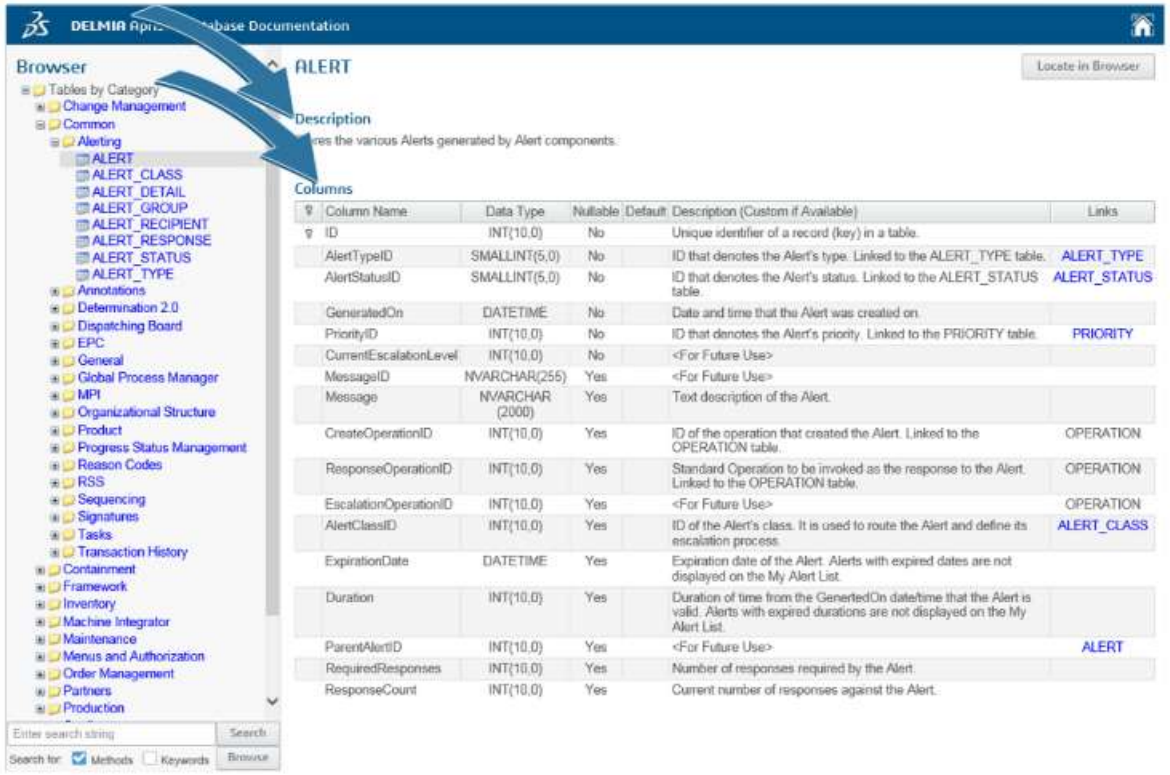


Table Columns Explained, part 2

When selecting a single table you will find:

- A brief description of what the table stores
- Column list:
  - **Key symbol column** – the symbol means the column belongs to table's primary key
  - **Column Name** – the name of the column
  - **Data Type** – column data type (Char, Integer, etc.)
  - **Nullable** – indicates if the column can have a null value
  - **Default** – default column value
  - **Description** – column purpose and possible uses
  - **Links** – if there is a value that should be linked to that column, the table with that value is reflected here and linked (if it is in the documentation)

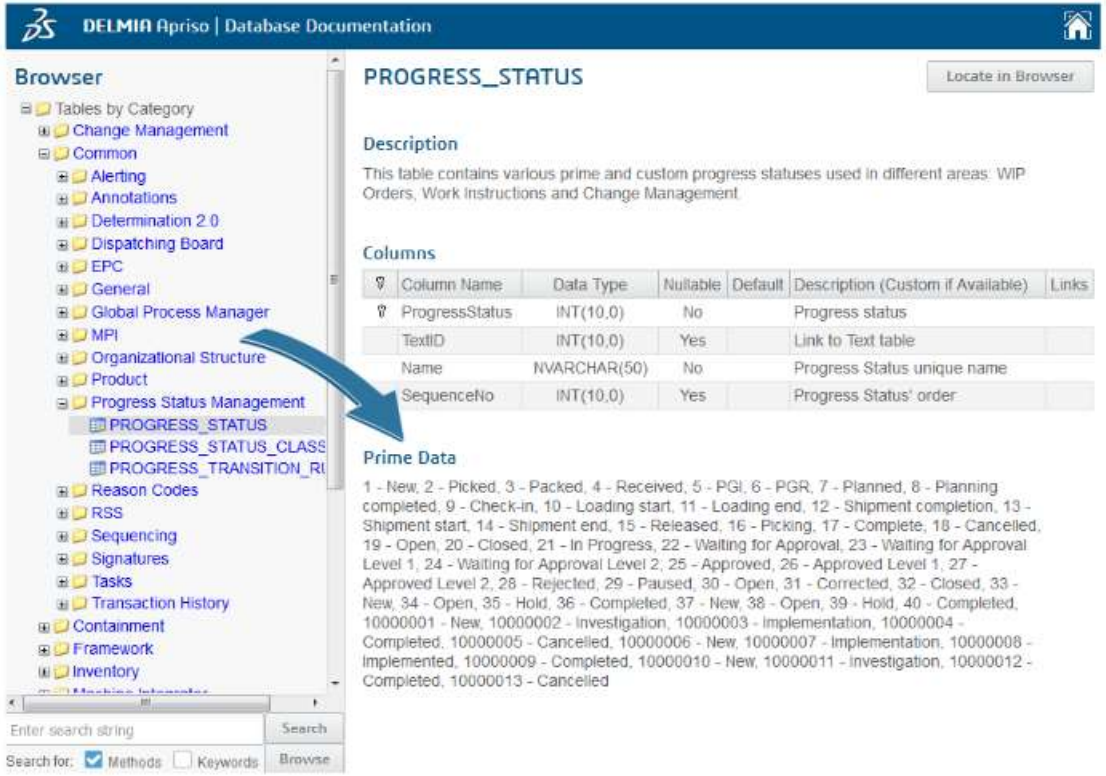


Prime Data, part 1

Some of the tables contain Prime Data. Prime Data is a set of certain statuses and pieces of information that have their numeral representation.

This way, the same information can be included in tables that required different data types.

Example: there is a table that contains production orders, including information about order status. Instead of writing New as the first order status, there is a numeral 1.



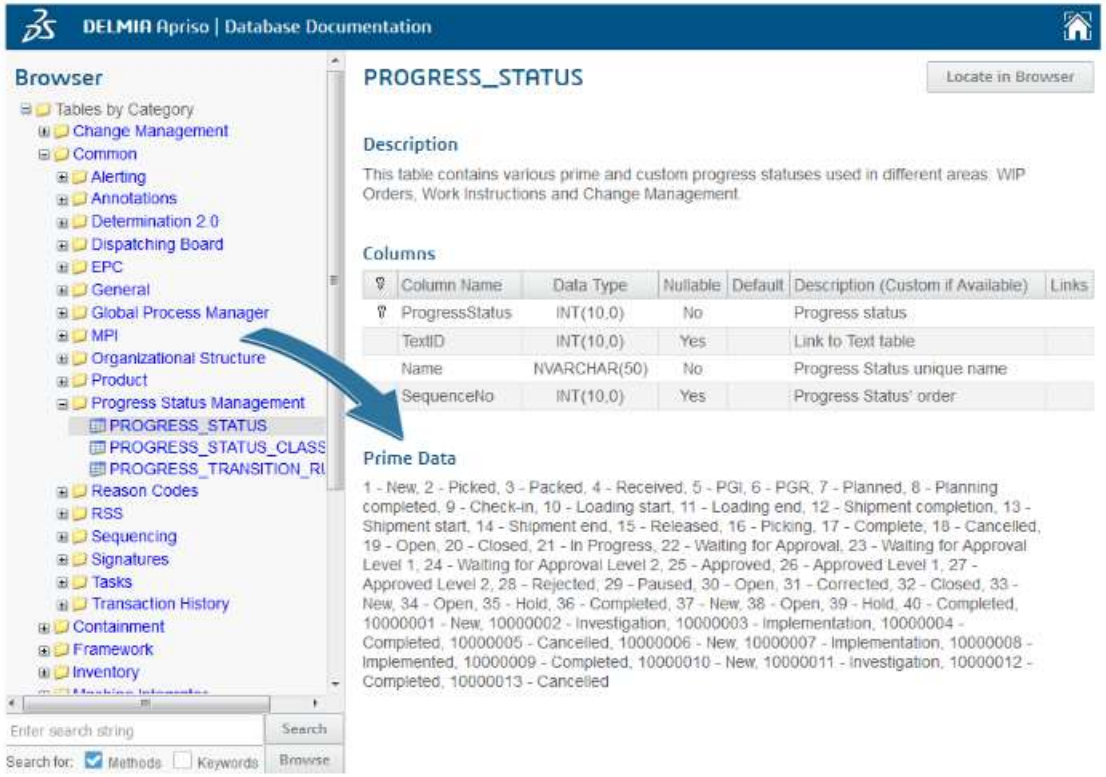
Prime Data, part 2

Reasons for Prime Data:

- Unification of data – there is a standard set of Prime Data for all tables within DELMIA Apriso, marked by the consecutive numbers (1, 2, and so on), as presented on the picture
- If a translation is needed, it is easier to use the numeric code and link it with the correct translation otherwise, each user could write their own type of data
- It is easier to use across the board

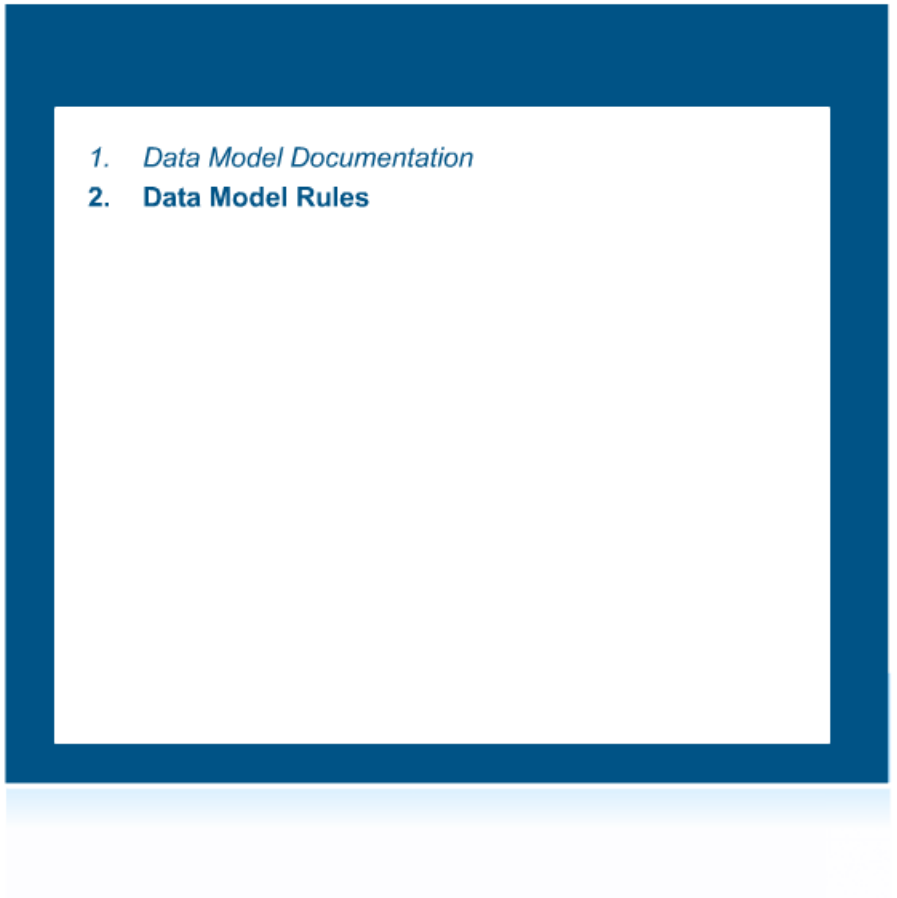
DO NOT TAMPER WITH PRIME DATA!

- It is widely used by Business Components – changing it will result in BCs not working
- If there is a need to add additional entries – use further numbers (e.g. 20000001, as presented on the picture)
- Prime Data is fully controlled by DELMIA Apriso, and it may be removed, changed, or added by the Database Upgrader



Chapter 2: Data Model Rules

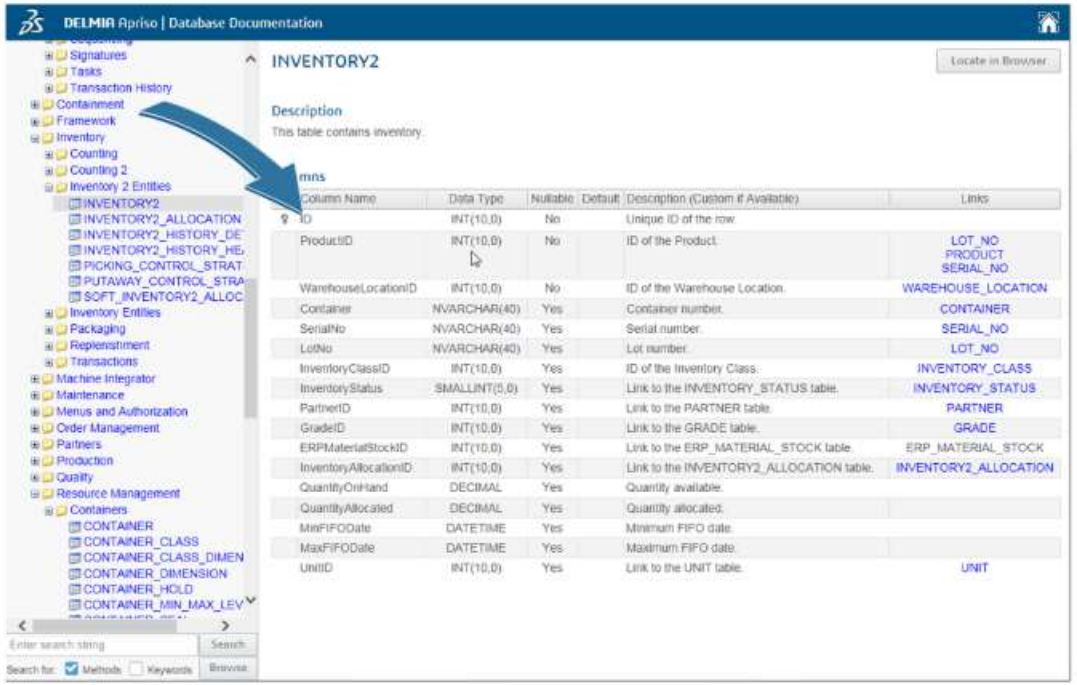
In this chapter you will get to know the most common rules and best practices for using the Data Model and the databases.



Primary Key, part 1

There are three types of the primary keys in the DELMIA Apriso Data Model

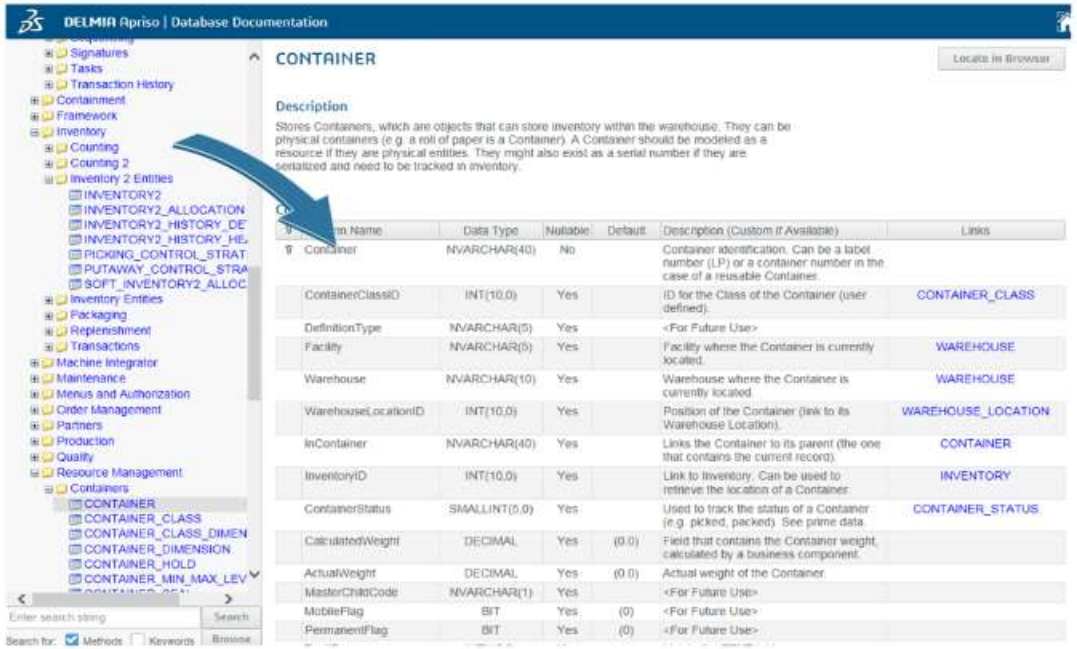
- An automatically incremented ID of the row



Primary Key, part 2

There are three types of the primary keys in the DELMIA Apriso Data Model

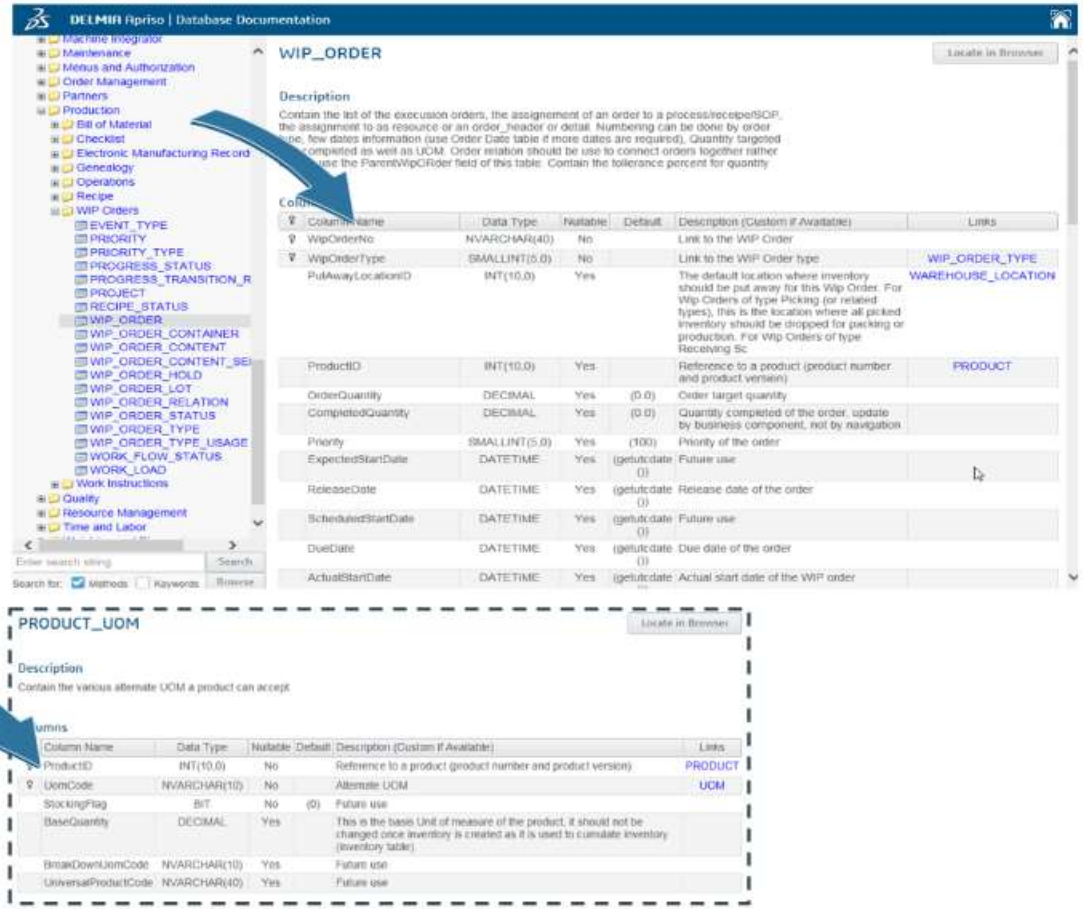
- An automatically incremented ID of the row
- A key based on the column which keeps the main information (e.g. a Container field in the CONTAINER table)



Primary Key, part 3

There are three types of the primary keys in the DELMIA Apriso Data Model

- An automatically incremented ID of the row
- A key based on the column which keeps the main information (e.g. a Container field in the CONTAINER table)
- A key based on a few columns as a main information (e.g. WipOrderNo, WipOrderType from the WIP_ORDER), or as a property for the primary key from another table (e.g. ProductID, UomCode from the PRODUCT_UOM)



Foreign Key, part 4

A foreign key in one table points to a primary key in another table. The foreign key constraint is used to prevent actions that would destroy links between tables.

The foreign key constraint also prevents invalid data from being inserted into the foreign key column, because it has to be one of the values contained in the table it points to.

Foreign key is a constraint on table level. It is commonly mentioned as a link to another table but is not.

In this illustration, the system generated ID column for Products becomes the foreign key that products are known by in the INVENTORY2 table

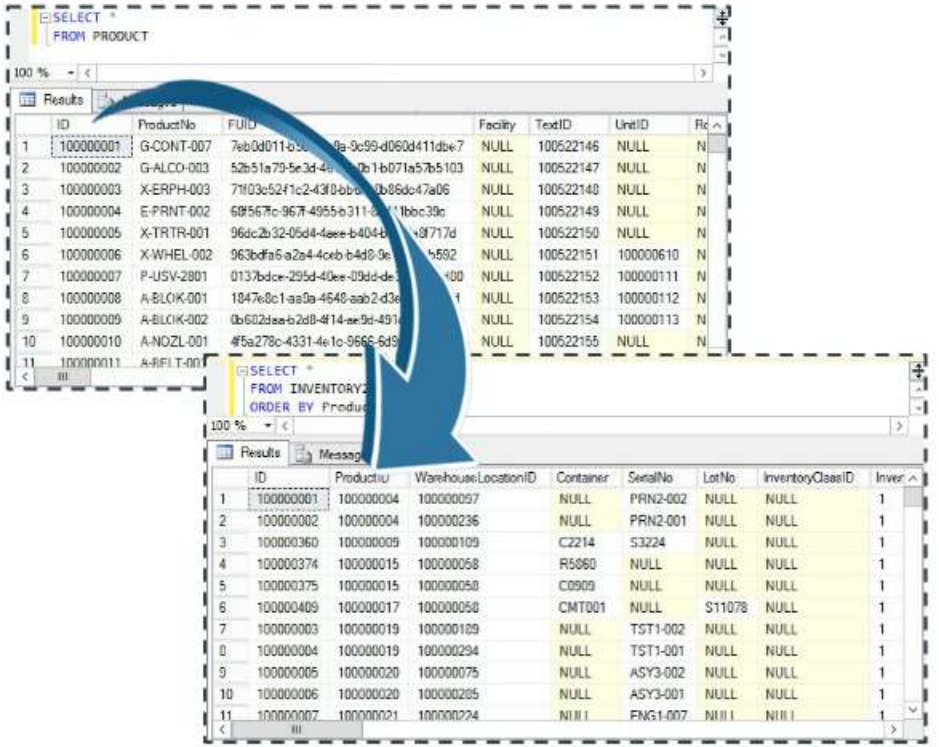


Table Joining, part 1

Foreign key data is located in many tables and very frequently there is a need to cross-reference the information from different tables (join tables). You can always use the Data Model Documentation to check which tables can be joined.

```
SELECT *
FROM
        INVENTORY2
        INNER JOIN PRODUCT
                ON INVENTORY2.ProductID = PRODUCT.ID
        INNER JOIN CONTAINER
                ON INVENTORY2.Container = CONTAINER.Container
```

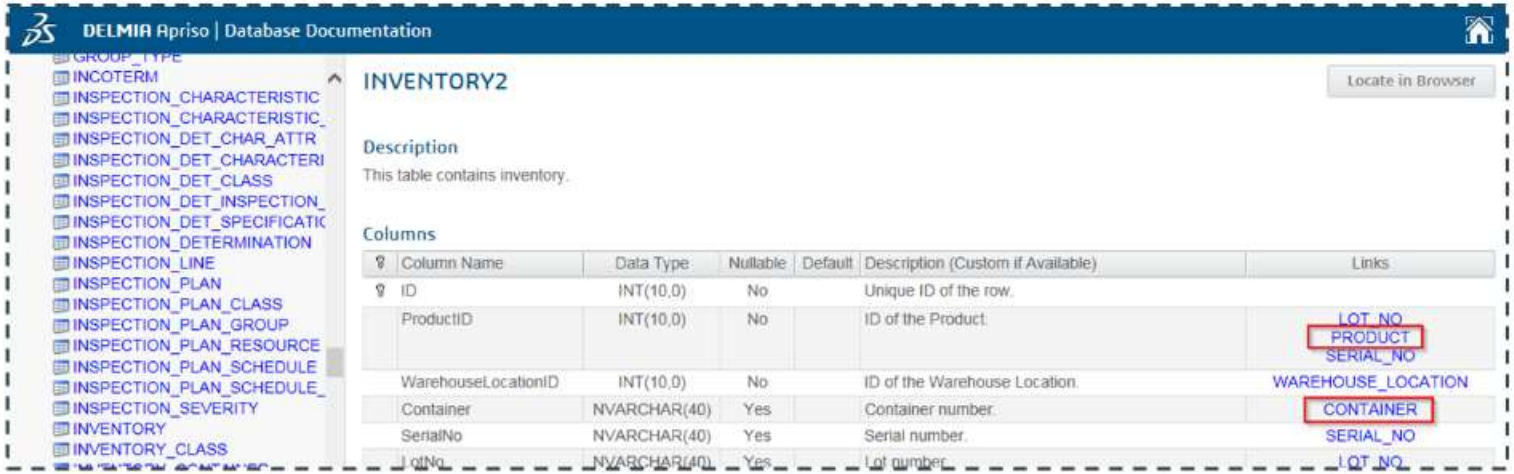


Table Joining, part 2

Please take into account that in the instance in which the primary key in the other table that consists of few columns, you should use all of them in the query.

```
SELECT *
FROM
        INVENTORY2_ALLOCATION
        INNER JOIN WIP_OPERATION
                ON INVENTORY2_ALLOCATION.WipOrderNo = WIP_OPERATION.WipOrderNo
    AND INVENTORY2_ALLOCATION.WipOrderType = WIP_OPERATION.WipOrderType
    AND INVENTORY2_ALLOCATION.OprSequenceNo = WIP_OPERATION.OprSequenceNo
```

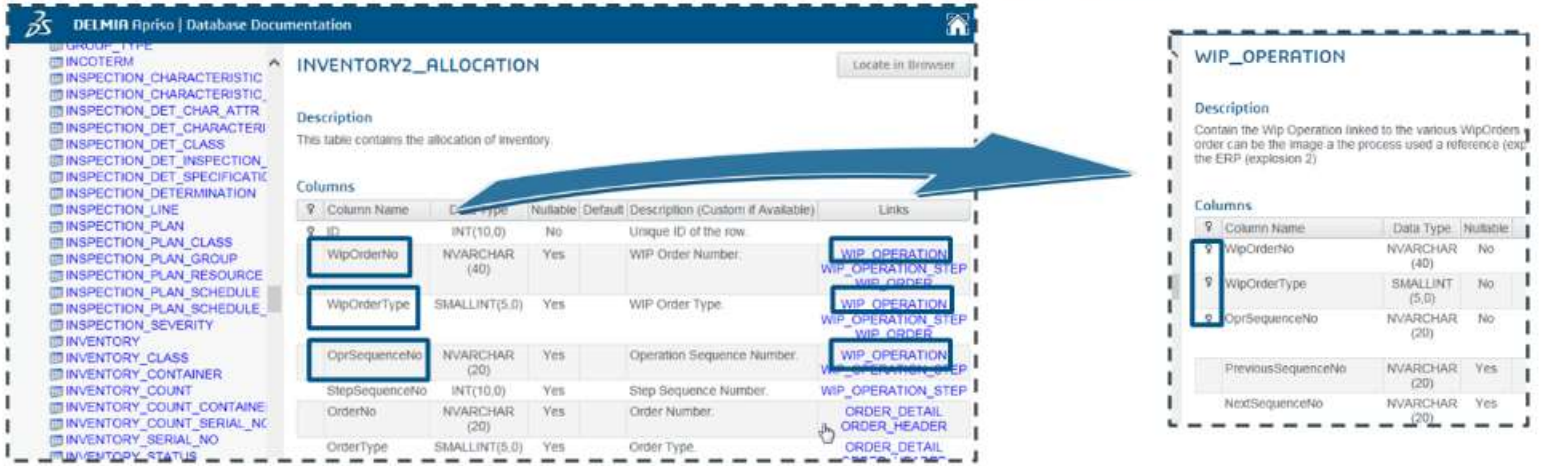


Table Joining, part 3

There are exceptions though:

- In the RESOURCE_ table the primary key consists of only one column - ID
- If you join the EQUIPMENT table with it the ResourceID column is used to join with the 'ID' of the RESOURCE_ table

```
SELECT *
FROM
        EQUIPMENT
        INNER JOIN RESOURCE_
                ON EQUIPMENT.ResourceID = RESOURCE_.ID
```

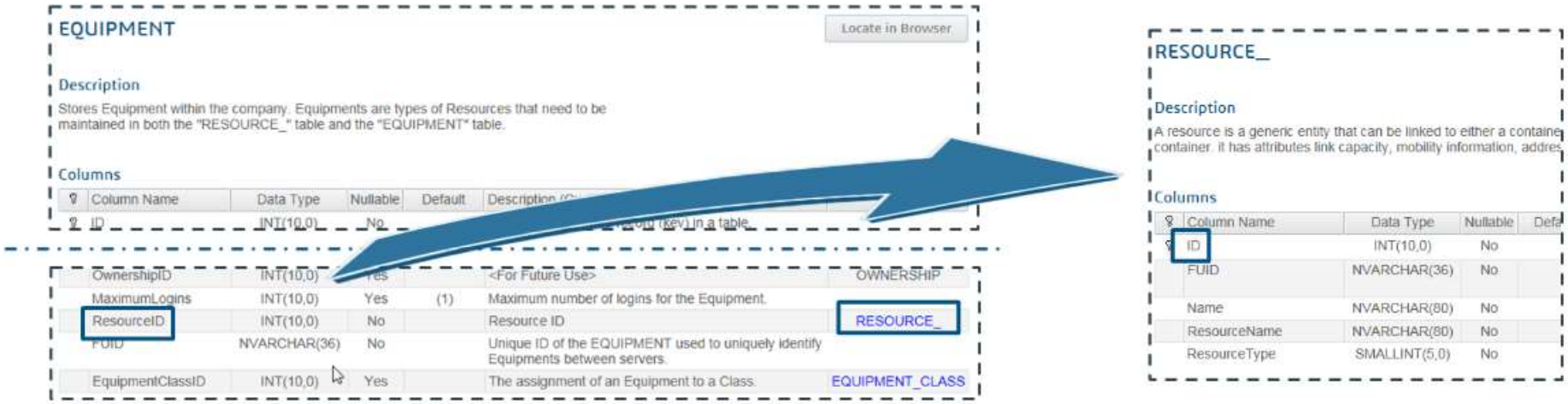
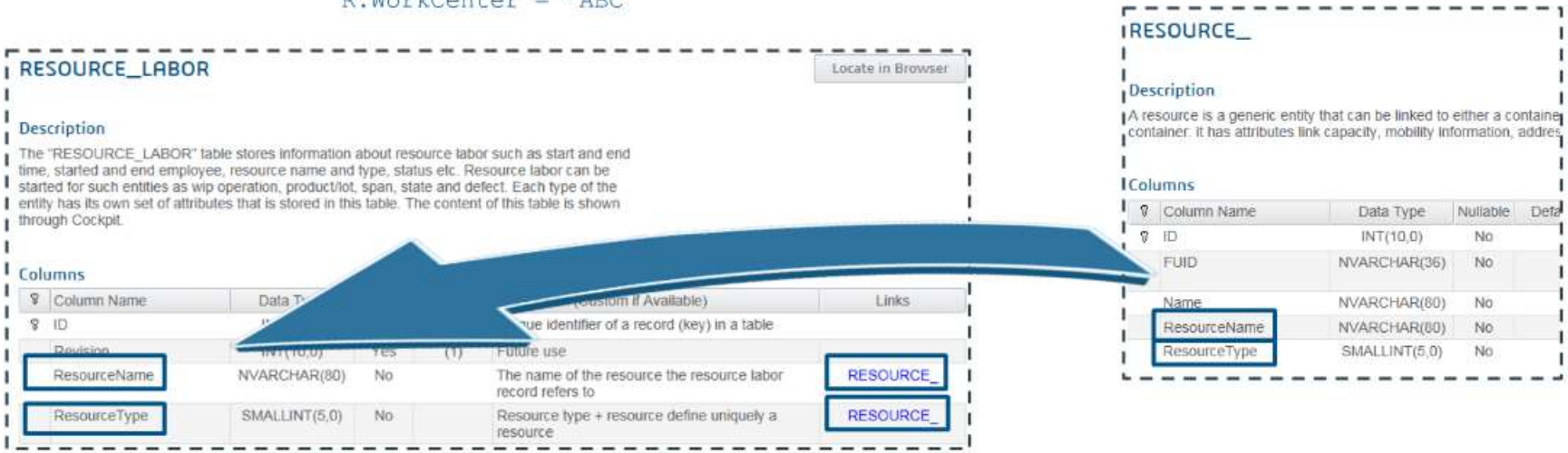


Table Joining

- In some of the other tables i.e. RESOURCE_LABOR the reference to the RESOURCE_ table is done by the two columns: ResourceName and ResourceType

```
SELECT *
FROM
        RESOURCE_LABOR RL
        JOIN RESOURCE_ R
                ON RL.ResourceName = R.ResourceName
                AND RL.ResourceType = R.ResourceType
        WHERE
                R.WorkCenter = 'ABC'
```



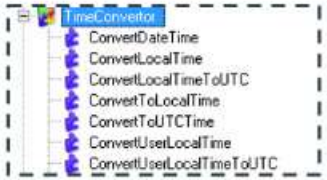
Data Model Rules

The columns of the datetime type are stored in database in UTC format (unless specified otherwise). When handling the dates, conversions should always be done.

Main rules of conversion apply:

- Local → UTC: when the user inserts a value in the database
- UTC → Local: when the user reads a value from database. That should be done in a query by an individual creating a DELMIA Apriso Operation

There are a few BCs that can convert datetime back and forth based on employee, resource and database configuration. Furthermore there are standard SQL functions that also can do that as well.



Columns

Column Name	Data Type	Nullable	Default	Description (Custom if Available)	Links
WipOrderNo	NVARCHAR(40)	No		Link to the WIP Order	
WipOrderType	SMALLINT(5,0)	No		Link to the WIP Order type	WIP_ORDER_
PutAwayLocationID	INT(10,0)	Yes		The default location where inventory should be put away for this Wip Order. For Wip Orders of type Picking (or related types), this is the location where all picked inventory should be dropped for packing or production. For Wip Orders of type Receiving Sc	WAREHOUSE_LC
ProductID	INT(10,0)	Yes		Reference to a product (product number and product version)	PRODUCT
OrderQuan	DECIMAL	Yes	(0.0)	Order target quantity	
CompletedQuan	DECIMAL	Yes	(0.0)	Quantity completed of the order. update by business component, not by navigation	
Priority	SMALLINT(5,0)	Yes	(100)	Priority of the order	
ExpectedStartDate	DATETIME	Yes	(getutcdate())	Future use	
ReleaseDate	DATETIME	Yes	(getutcdate())	Release date of the order	
ScheduledStartDate	DATETIME	Yes	(getutcdate())	Future use	
DueDate	DATETIME	Yes	(getutcdate())	Due date of the order	
ActualStartDate	DATETIME	Yes	(getutcdate())	Actual start date of the WIP order	
ActualCompletionDate	DATETIME	Yes	(getutcdate())	This is the actual completion date of the WipOperation	
ScheduledDurationSeconds	INT(10,0)	Yes	(0)	Future use	
UomCode	NVARCHAR(10)	Yes		UOM of the quantity ordered	UOM
WorkOrderStatus	SMALLINT(5,0)	Yes		The status of the wip order	WIP_ORDER_S

Technical Columns

In the DELMIA Apriso Data Model you will find identical columns at the end of each table. They are called **technical columns** or **WHO columns**.

- They are:
  - ReferenceID → Column used to reference another table .
  - LastUpdateOn
  - LastUpdatedBy
  - CreatedOn
  - CreatedBy

  Columns showing who and when a record was created and by whom.
  - Active → Active column allow to mark the record as 'deleted' without physically removing it.
  - LastDeleteOn
  - LastDeletedBy
  - LastReactivateOn
  - LastReactivatedBy
  - ArchiveID
  - LastArchiveOn
  - LastArchivedBy
  - LastRestoreOn
  - LastRestoredBy

  Columns planned to be used by the archiving tool - not currently used.
  - RowVersionStamp → The current version of the record, used for concurrency validation.

'Active' Column

Active columns let you exclude data from queries you don't want to see without deleting the record entirely.

In some tables, such as 'CONTAINER' or 'PRODUCT', the standard course of action is to mark the record as 'deleted' without physically removing it.

The good practice is to check them when joining tables and in where clause.

```
SELECT *
FROM dbo.EMPLOYEE e
JOIN dbo.EMPLOYEE_ROLE er
  ON er.EmployeeID = e.ID
  AND er.Active = 1
WHERE
  e.Active = 1
```

UnitID

When the column is named UnitID, it is automatically treated as linking to the UNIT table.

The UNIT table holds all unitID information across whole database.

It is not possible to have two different entities that have same unitID.

In the UNIT_CHARACTERISTIC table, an entity is identified only by unitID, not type of entity.

The unitID is used to link to a record in a table like FACILITY, PRODUCT, etc. with a Characteristic (in the UNIT_CHARACTERISTIC table), Annotation (in the UNIT_ANNOTATION), etc.

UNIT

Locate in Browser

Description

This table stores Unit Ids that can be assigned to such entities as Facility, Product, Group, Product Group, Order Shipment Stage, Resource Life Cycle, Lot, Warehouse Location, Resource, Container, Work Center, Order, Wip Operation, etc. Once a unit is created, it can be assigned multiple characteristics.

Columns

Column Name	Data Type	Nullable	Default	Description (Custom if Available)	Links
ID	INT(10,0)	No		Unique ID of a record (key) in a table.	

Multi Language

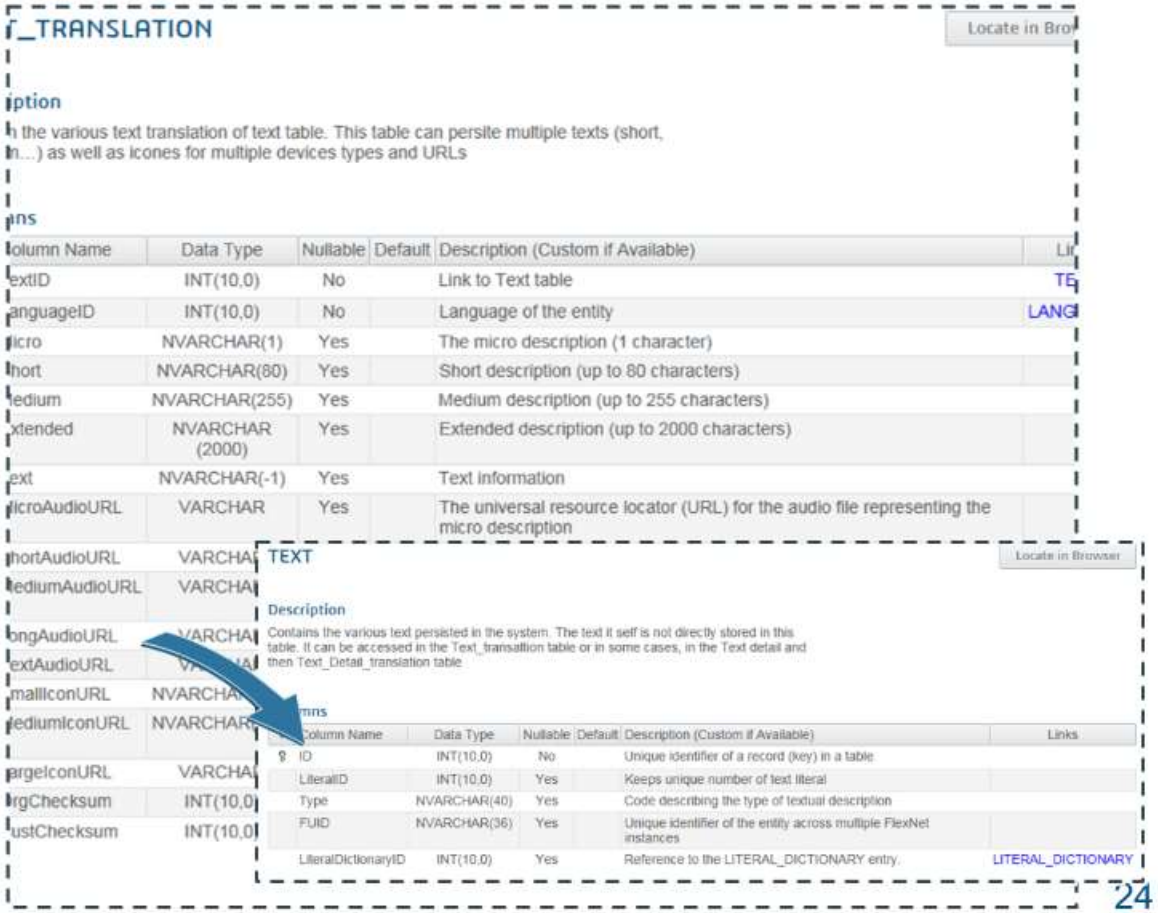
When the column is named TextID, it is automatically treated as a reference to the TEXT table.

Such columns don't have any foreign keys constraints.

In the TEXT_TRANSLATION table there are translations of the text for different LanguageIDs.

The LanguageID = 1033 (English USA) is the default language, if there is no translation to a specific language.

TEXT hold texts (similarly like Unit), TEXT_TRANSLATION is a translation.



Types, Classes, and Groups, part 1

Types, Classes, and Groups are commonly used to classify entities:

- Type is always an integer data type field
  - The name of the field is usually created by adding 'Type' to the entity name (i.e. ResourceType, WipOrderType, etc.)
  - That field will be a foreign key to a table which keeps the types dictionary. The table has '_TYPE' added to the main table name (i.e. WIP_ORDER_TYPE)
  - The primary key of the table sometimes consists of the entity name, and the type (i.e. WipOrderNo, WipOrderType)

WIP_ORDER_TYPE

Locate in Browser

Description

This table contains the various order types. This list can be extended.

Columns

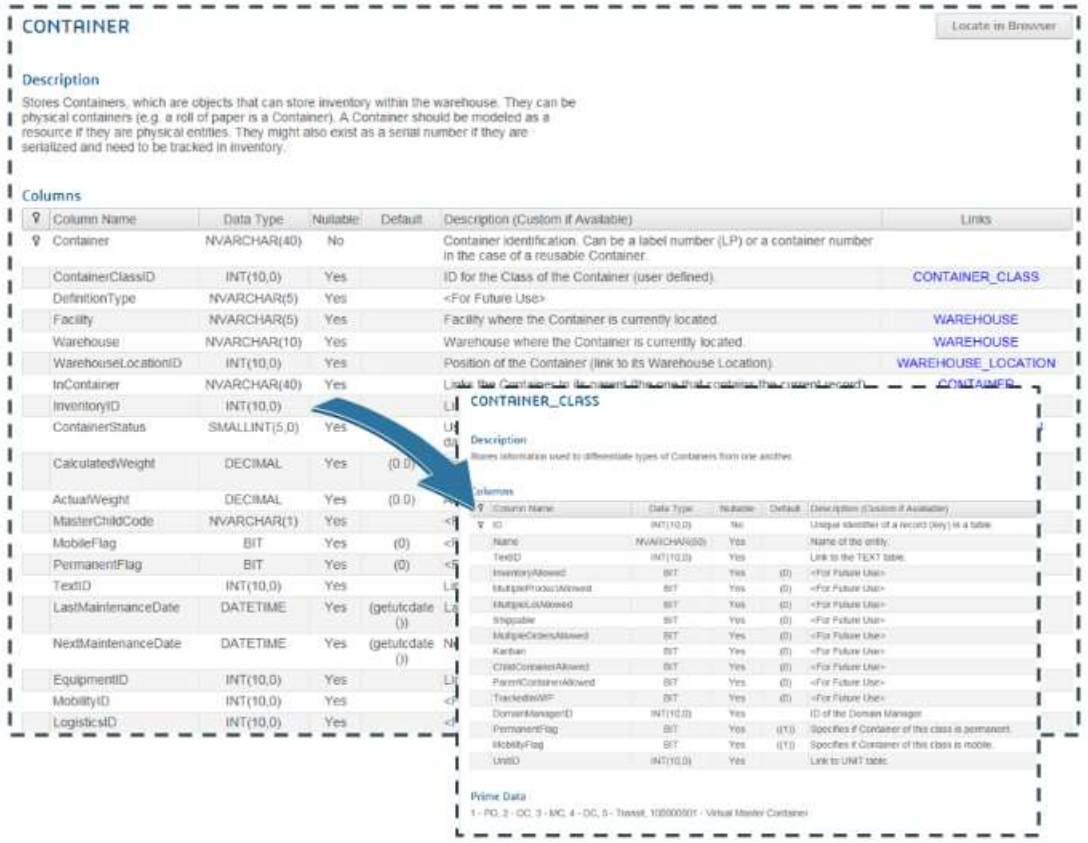
	Column Name	Data Type	Nullable	Default	Description (Custom if Available)	Links
🔑	WipOrderType	SMALLINT(5,0)	No		Link to the WIP Order type	
	TextID	INT(10,0)	Yes		Link to Text table	
	WipOrderTypeUsageID	INT(10,0)	Yes		ID of WIP Order Type Usage	WIP_ORDER_TYPE_USAGE

Prime Data

1 - Production Order, 2 - Maintenance Order, 3 - Inspection Order, 4 - Requisition, 5 - Replenishment Order, 6 - Material Order, 7 - Delivery Order, 8 - Transportation Order, 9 - Campaign, 10 - Return Material Authorization, 11 - Service & Repair Authorization, 12 - Disposal Order, 13 - Customer Order, 14 - Project, 15 - Inventory Count, 16 - Purchase Order, 17 - ASN Inbound Delivery Order, 18 - Customer Return Order, 19 - Packaging Order, 20 - Picking Order, 21 - Putaway Order, 22 - Driven Picking Order, 23 - Master Schedule Order, 24 - Production Picking Order, 25 - Sequenced Picking Order, 26 - Preventive Maintenance Order, 27 - Reactive Maintenance Order, 28 - Tooling Order

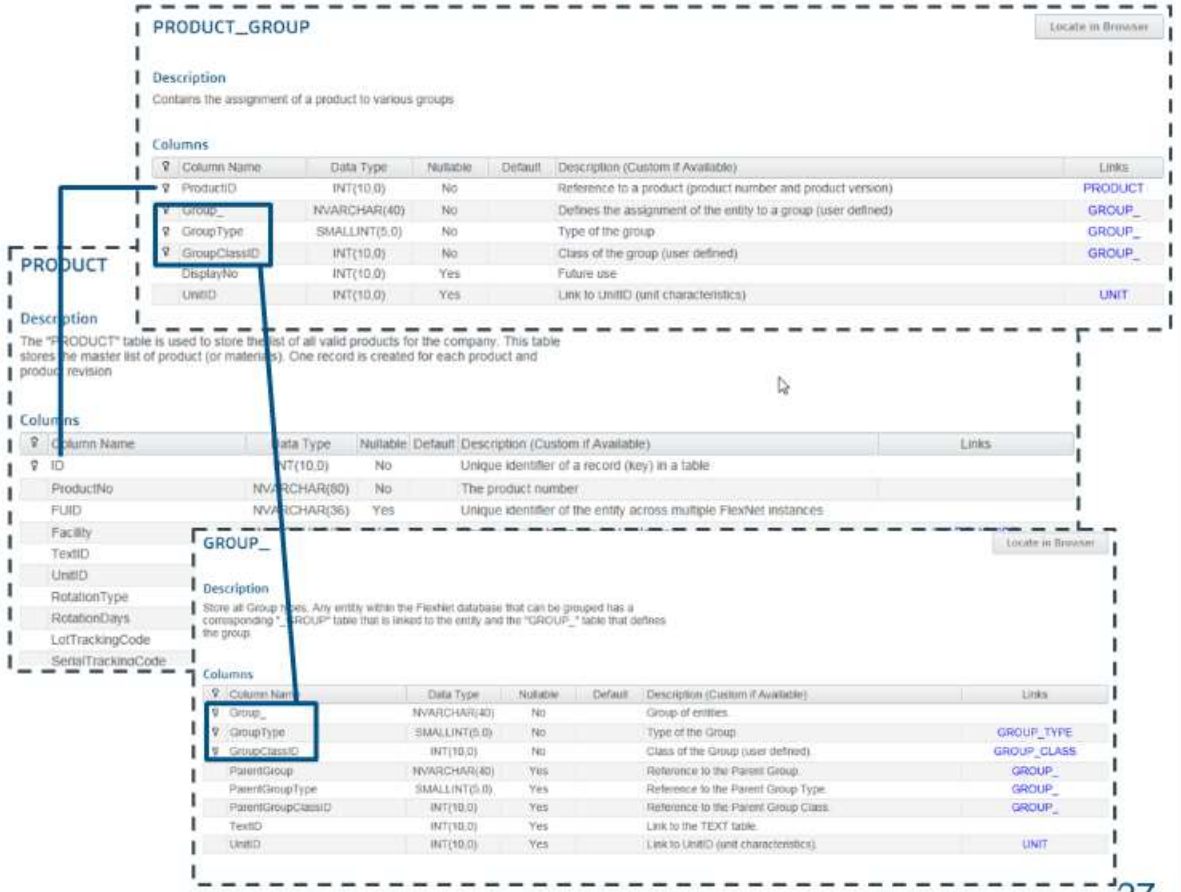
Types, Classes, and Groups, part 2

- An entity class is always referred by a foreign key to the particular class table (i.e. in the CONTAINER table there is a ContainerClassID field which is a foreign key to the CONTAINER_CLASS table)
- The class table can have a prime data (standard DELMIA Apriso classes), but you can extend it, by adding more classes



Types, Classes, and Groups, part 3

- Groups of entities are created in tables with 'GROUP_' added to the entity name
- Unlike classes, for groups there is no reference in the entity table to the group table, but the opposite - in the group table there is a foreign key to the entity table (i.e. ProductID field in the PRODUCT_GROUP table points to a record in PRODUCT table)
- Also unlike classes, entities can belong to more than one group
- The GROUP_ table has a primary key which consists of three fields (Group_, GroupType, and GroupClassID)
- Entity group tables typically have a primary key which consists of four fields: foreign key to the entity, and foreign key to the GROUP_ table



Other Rules

Here are few less important - but still vital - rules:

- ID rows are automatically incremented
  - The ID column is always the first column in the table
  - The initial value is 100000000
- A column with the GUID will have the name FUID and the type of nvarchar(36), not null
- The data processing in DELMIA Apriso Business Components is written using the “optimistic concurrency control”. This means that the RowVersionStamp column is checked to determine if the record used in the transaction has been changed. If a change is detected, it returns the “Table XYZ concurrency violation” error code

LAB 1: Querying a Database

Task:

- Write a query retrieving data from the database

What you will learn:

- How to use the Database Documentation
- How to join tables according to the DELMIA Apriso Data Model

Requirements:

- Use Microsoft SQL Server Management Studio.
  - Login: sa
  - Password: Apriso2017
  - Use the APRISO database
- In case of any technical problems, please contact DELMIA.Apriso.training@3ds.com

Remember to use the following to login and name Screens thorough this entire training:
- TRN<yourinitials> if your are an external self-paced learner
- TRN<yourtrigram> if you are a 3DS employee self-paced learner

Duration: 45 min

LAB 1: Querying a Database

- Retrieve the following data from the database:
  - ProductNo (from the PRODUCT table)
  - SerialNo (from INVENTORY2 table)
  - AvailabilityDate (from SERIAL_NO table - pay attention to the fact, that there are two column in the primary key in that table)
  - Warehouse (from the WAREHOUSE table)
  - Location (from the WAREHOUSE_LOCATION table)
  - Extended translations for product and warehouse location (get them from the TEXT_TRANSLATION table for the 1033 LanguageID - it's English)
  - QuantityOnHand (from the INVENTORY2 table)
  - DafaultUomCode (from the PRODUCT table)
- Sample script can be found on the training server. Go to: Desktop > Training Materials > 1-BPM Utilization Level 1

	ProductNo	SerialNo	AvailabilityDate	Warehouse	Extended	Location	QuantityOnHand	DefaultUomCode
1	X-RING-001	(null)	2016-03-02 22:17:05.000	PRD	Production	PRDPTW16	1.0000000000	EA
2	X-STRP-001	(null)	2016-02-12 14:01:12.000	PRD	Production	PRDPTW16	1.0000000000	EA
3	X-STRP-001	SER0001	2016-02-12 14:01:48.000	PRD	Production	PRDPTW16	1.0000000000	EA
4	X-STRP-001	SER0002	2016-02-12 14:02:05.000	PRD	Production	PRDPTW16	1.0000000000	EA
5	X-RING-001	SERMTA01	2016-03-02 22:18:03.000	PRD	Production	PRDPTW16	1.0000000000	EA
6	X-RING-001	SERMTA02	2016-03-02 22:18:23.000	PRD	Production	PRDPTW16	1.0000000000	EA
7	X-RING-001	SERMTA09	2016-03-02 22:19:31.000	PRD	Production	PRDPTW16	1.0000000000	EA
8	X-RING-001	SERMTA03	2016-03-02 22:18:31.000	PRD	Production	PRDPTW16	1.0000000000	EA
9	X-RING-001	SERMTA04	2016-03-02 22:18:40.000	PRD	Production	PRDPTW16	1.0000000000	EA
10	X-RING-001	SERMTA05	2016-03-02 22:18:56.000	PRD	Production	PRDPTW16	1.0000000000	EA
11	X-RING-001	SERMTA06	2016-03-02 22:19:05.000	PRD	Production	PRDPTW16	1.0000000000	EA
12	X-RING-001	SERMTA07	2016-03-02 22:19:14.000	PRD	Production	PRDPTW16	1.0000000000	EA
13	X-RING-001	SERMTA08	2016-03-02 22:19:22.000	PRD	Production	PRDPTW16	1.0000000000	EA