


Data Modelling

Menu 

- ▶ [Overview of items.xml file \(http://javainsimpleway.com/overview-of-items-xml-file/\)](http://javainsimpleway.com/overview-of-items-xml-file/)
- ▶ [Defining Item types in hybris \(http://javainsimpleway.com/defining-item-types-in-hybris/\)](http://javainsimpleway.com/defining-item-types-in-hybris/)
- ▶ [Deployment and Typecodes in items.xml \(http://javainsimpleway.com/deployment-and-typecodes-in-items-xml/\)](http://javainsimpleway.com/deployment-and-typecodes-in-items-xml/)
- ▶ [Defining collection in items.xml \(http://javainsimpleway.com/defining-collection-in-items-xml/\)](http://javainsimpleway.com/defining-collection-in-items-xml/)
- ▶ [Defining Enum and Map types in Hybris \(http://javainsimpleway.com/defining-enum-and-map-types-in-hybris/\)](http://javainsimpleway.com/defining-enum-and-map-types-in-hybris/)
- ▶ [Defining Relation in items.xml in Hybris \(http://javainsimpleway.com/defining-relation-in-items-xml-in-hybris/\)](http://javainsimpleway.com/defining-relation-in-items-xml-in-hybris/)
- ▶ [Collection v/s relation in Hybris \(http://javainsimpleway.com/collection-vs-relation-in-hybris/\)](http://javainsimpleway.com/collection-vs-relation-in-hybris/)
- ▶ [Dynamic Attribute \(http://javainsimpleway.com/dynamic-attribute/\)](http://javainsimpleway.com/dynamic-attribute/)
- ▶ [Redeclare in items.xml \(http://javainsimpleway.com/redeclare-in-items-xml/\)](http://javainsimpleway.com/redeclare-in-items-xml/)

Impex

- ▶ [Overview Of Impex \(http://javainsimpleway.com/overview-of-impex/\)](http://javainsimpleway.com/overview-of-impex/)
- ▶ [Impex – Cell Decorator \(http://javainsimpleway.com/impex-cell-decorator/\)](http://javainsimpleway.com/impex-cell-decorator/)
- ▶ [Impex – Translator \(http://javainsimpleway.com/impex-translator/\)](http://javainsimpleway.com/impex-translator/)

Cron job

- ▶ [Cron Jobs Overview \(http://javainsimpleway.com/cron-jobs-overview/\)](http://javainsimpleway.com/cron-jobs-overview/)
- ▶ [CronJob Real time example \(http://javainsimpleway.com/cronjob-example/\)](http://javainsimpleway.com/cronjob-example/)
- ▶ [Cron Jobs Subsidiary Information \(http://javainsimpleway.com/cron-jobs-subsiary-information/\)](http://javainsimpleway.com/cron-jobs-subsiary-information/)
- ▶ [Dynamic Cron Job / Cron job scripting \(http://javainsimpleway.com/dynamic-cron-job-cron-job-scripting/\)](http://javainsimpleway.com/dynamic-cron-job-cron-job-scripting/)

Miscellaneous

- ▶ [Flexible search \(http://javainsimpleway.com/flexible-search/\)](http://javainsimpleway.com/flexible-search/)
- ▶ [Initialization and Update hook up \(http://javainsimpleway.com/initialization-and-update-hook-up/\)](http://javainsimpleway.com/initialization-and-update-hook-up/)
- ▶ [Custom parameters during initialization and update \(http://javainsimpleway.com/adding-custom-parameters-during-initialization-and-update/\)](http://javainsimpleway.com/adding-custom-parameters-during-initialization-and-update/)
- ▶ [Extensions vs Addon \(http://javainsimpleway.com/extensions-vs-addon/\)](http://javainsimpleway.com/extensions-vs-addon/)

Flexible search

Lets understand some basic concepts about flexible search in Hybris with few examples

It's a hybris **built-in query language** based on **SQL syntax**.

It enables us to **search** the **records** from the **database** using **item types**.

In flexible search queries, we **never** use database **table names**.

We always use **item types** which will be **mapped** to a corresponding **tables** by Hybris.

Flexible search query execution has 2 phases

1) Pre-parsing phase

In this phase, Hybris **converts** the **flexible search** query into **SQL** query.

2) Executing the SQL converted query

In this phase, converted **SQL query** will be executed by Hybris.

Examples :

1. `Select * from {Product}`

Copy this code

This query **retrieves** all the **columns** and **rows** of a **Product item type**.

The **item type** specified in the **curly braces** is the **exact item type code** defined in the **items.xml**.

This should have been **mapped** to specific **table** using **deployment tag** in **items.xml**

for more details on how **tables** and **item types** are **related**, click **here** (<http://javainsimpleway.com/defining-item-types-in-hybris/>) and **here** (<http://javainsimpleway.com/deployment-and-typecodes-in-items-xml/>)

1. `SELECT {name[de]}, {name[en]} FROM {Product}`

Copy this code

This query **retrieves name of a Product** in **German** and **English** locale

1. `SELECT {p.code} FROM {Product AS p} ORDER BY {p.code}`

Copy this code

This query **retrieves Product code** from **Product type** using **alias** with the name **“p”** for **Product type**.

We can **specify** the **join** between **2 types** as below

1. `SELECT * FROM {Product JOIN Category}`

Copy this code

This query makes the **Join** between **Product** and **Category** item type.

Note:

We can make different types of Joins like Left outer join,Right outer join and Inner join.

Sub Types

When we search **any type** in flexible search, by default its **subtypes** will also be **retrieved** in the result.

Example:

1. `Select * from {Product}`

Copy this code

This will **retrieve** the **instances** of **Product** and **VariantProduct**.

Since **VariantProduct** is a **sub type** of **Product**, it **retrieves** its **sub types** as well.

If we want to **exclude** the **subtypes** in the result, we must use **exclamation mark (!)** while specifying type as below

1. `Select * from {Product!}`

Copy this code

In this example, we have used **!** for **product type** and hence it retrieves **only Product instances** but **not** its **variants**.

Using conditions

1. `SELECT * FROM {Product} WHERE {code} LIKE '%001'`

Copy this code

Retrieves **all the products** whose **code** has **001** in it.

1. `SELECT * FROM {Product} WHERE {code} LIKE '%001%' AND {code} LIKE '%m%'`

Copy this code

Retrieves **all the products** whose **code** has **001** and **m** in it.

1. `SELECT * FROM {Product} WHERE {code} NOT LIKE '%001%'`

Copy this code

Retrieves **all the products** whose **code does not** contain **001** in it.

1. `SELECT {code},{pk} FROM {Product} ORDER BY {code} DESC`

Copy this code

Retrieves the **result** and **Sorts** the **search results** based on the **code**column in the database in **descending order**

Runtime Parameters

We can **specify** run time parameters in **flexible search** query using **placeholder(?) prefix** to a specific **column**.

1. `SELECT {p:pk} FROM {Product AS p} WHERE {p:code} LIKE ?code`

Copy this code

Here we have specified **?code** which means **run time parameter** is **code** whose **value** will be added atrun time using **query params HashMap**.

Executing flexible search queries using API