



MetaModel

About | [Developers guide](#) | [Webcast demo](#) | [Database compliancy](#) | [License](#) | [Download](#)

A common domain model, query-engine and optimizer for different kinds of datastores.

The eobjects.org MetaModel is a project created for maximum reuse of a SQL 99 compliant domain model of the database domain. The [MetaModel](#) is a model that contains classes that represent the **structure** of a database (schemas, tables, column, relationships) and **interaction** with the database (queries). In short a model for modelling (hence the word "metamodel") data in databases and other datastores.

MetaModel is being used in a lot of projects, including the eobjects.org projects [DataCleaner](#), [DataTransformer](#) and [TableAgent](#). Additionally the [QueryParser](#) project is aimed at creating a String-based parser to populate the query-part of [MetaModel](#) by parsing incoming SQL queries.

MetaModel 1.2 is out!

We're proud to present a new major release of [MetaModel!](#) [Go download](#) this innovative new datastore manager, optimizer and metadata component!

How does it work?

Watch the [webcast!](#)

Goals

The goal of [MetaModel](#) is to provide an API for:

- Traversing and building the structure of datastores.
- Executing datastore-neutral queries in a SQL-like manner.
- Provide datastores that do not support queries with a [query engine](#).
- Implement this system for JDBC databases, Comma-separated files, XML files, Excel spreadsheets, MS Access (.mdb), dBase (.dbf) and OpenOffice (.odb) database-files.
- Allow [querying across datastores](#) in a way that is similar to querying a single datastore. This involves transparent client-side joining, filtering, grouping etc.
- [Split single queries into multiple ones](#) that yield the same collective result, enabling powerful performance optimization and grid execution for heavy work loads.

We are designing this API to be:

- Comprehensive - MetaModel should contain all the functionality that is required to model the datastore domain.
- Easy to use - We focus on simple design and fluent interfaces to make the most commonly used functionality easy and fun to use.

- Consistent - With [MetaModel](#) you will not experience all the ambiguities that exist in other frameworks like JDBC. Instead of diversity and hacks we focus on standardization and a common datastore infrastructure - all necessary hacks have been hidden from you, the user.
- Extensible - We provide extension-points to make it possible to use [MetaModel](#) for your own legacy systems, for example to query datastores that may not be supported.
- Light-weight - Deliberate use of POJOs make it easy to understand what is going on inside MetaModel.

Using with Maven

Add this dependency entry to your Maven POM to include [MetaModel](#) in your project:

```
<dependency>
  <groupId>dk.eobjects.metamodel</groupId>
  <artifactId>MetaModel-full</artifactId>
  <version>1.2</version>
</dependency>
```

Contributing

There are several ways you can contribute to [MetaModel](#)...

- Post your ideas to the [mailing list](#).
- Add [new tickets](#) for future development.
- Check out the code and post patches, see [BuildingMetaModel](#).
- [Donating money to eobjects](#).