

Top level elements

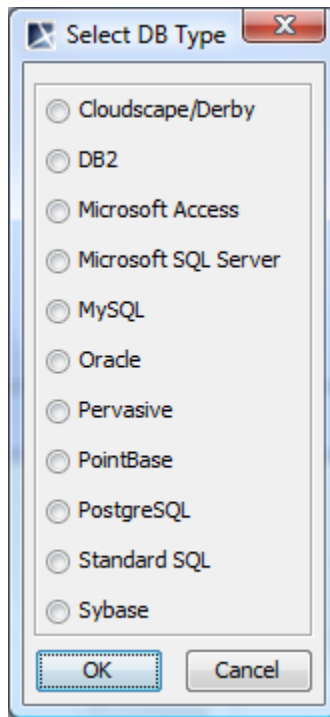
On this page

- [Database](#)
- [Schema](#)
- [Catalog](#)
- [GLOBALS](#)

There are several top-level model elements, that serve as the containers for other model elements of the database model: Database, Schema, and Catalog.

Top level elements are not strictly necessary to begin database modeling. You can start modeling database elements (like tables) in the standard UML package (even directly under root 'Data' model element). But top level elements help to provide context for those other elements and their naming and positioning in the database. So, at least one top level element should be present - either Schema element or Database element. Optimally both Database and Schema element should be present in the model (Schema package inside the Database package). Catalog modeling is less important, it can be skipped. Not all databases have support for catalogs.

When top-level element is created (either on the diagram or in the containment tree), a special dialog is shown for selecting database flavor.



Database flavor selection dialog.

When DB flavor is chosen, the necessary profile for that DB flavor is attached to the project (providing standard data types for that DBMS and / or additional stereotypes for modeling extensions of that DB flavor). Then profile application relationship is created from the package that is being created (Database, Schema) to the attached DB profile. This marks the top level element as belonging to this DB flavor, Other DB elements, created under that top level element will be automatically considered as belonging to this DB flavor.

If you would like to switch database flavor after creating a top level element, you can do this in the following way.

To switch database flavor after creating a top level element

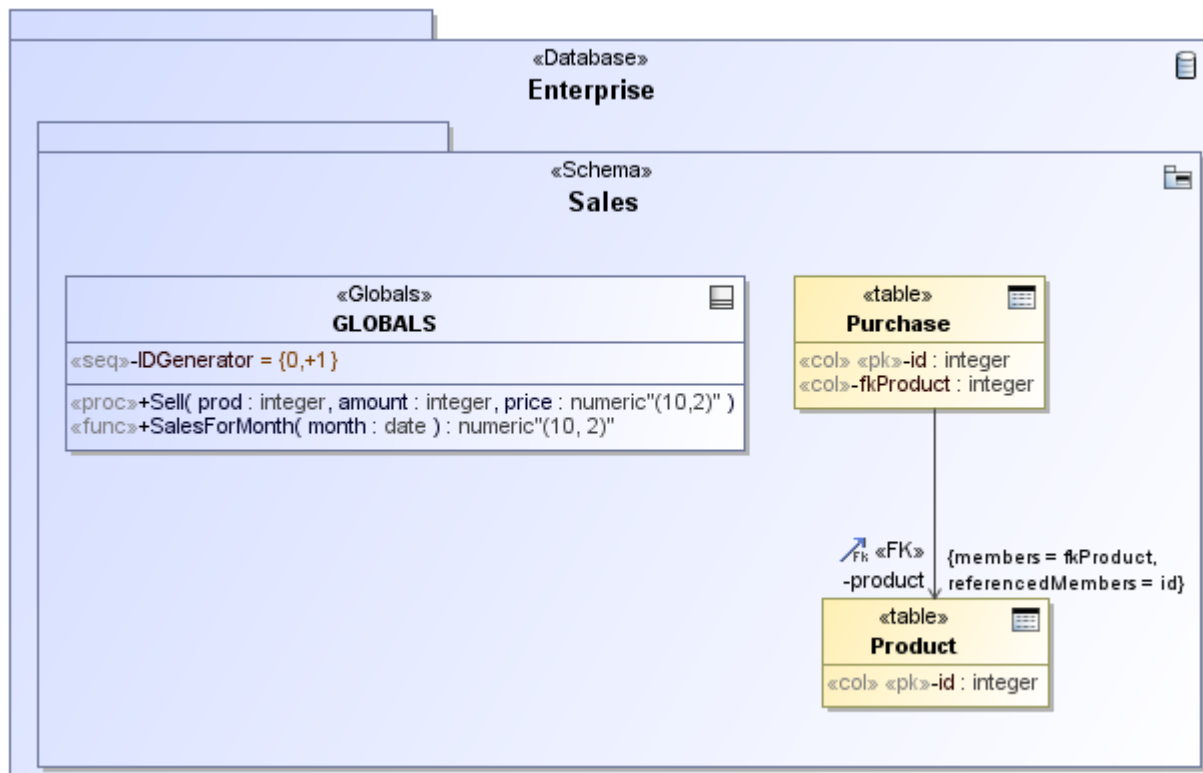


Important

You must have the necessary module attached to your project (Use **File>Use Module** menu then choose the necessary module from your **<install.root>profiles** predefined location.).

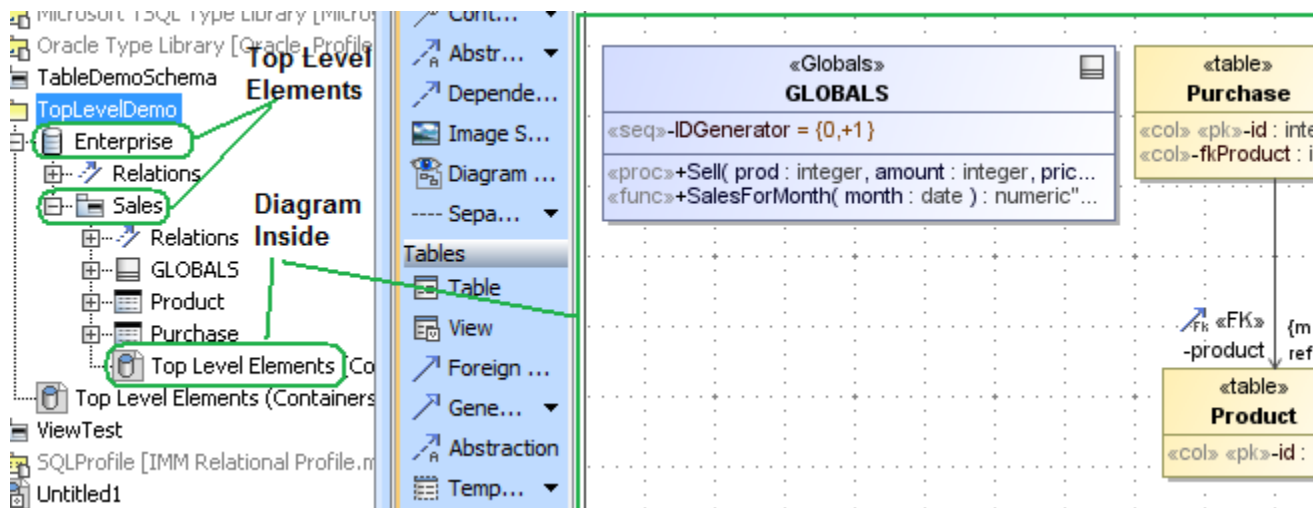
1. Right-click the top level element.
2. From the shortcut menu, select **Apply Profiles**.
3. Select the check box near the needed profile and clear the check box near the old profile.
4. Click **Apply**.

Top level elements can be explicitly drawn on the diagram.



Database top level containers (database and schema) on the diagram pane.

However, showing top level elements on the diagram, and nesting their contents inside them is often clumsy, and consumes valuable diagram space. Showing them on the diagram pane is not necessary; it is enough to create them in the Containment tree (using the **New Element** command on the shortcut menu). Then, place your diagram inside the created containers, and the elements that you will be creating in your diagram, will go into the necessary container. See the following figure (logically equivalent to the previous one), showing a top level element just in the Containment tree and not displayed on the diagram pane.



Database top level containers (database and schema) in Containment tree, but not on diagram pane.

There is also one additional complication, stemming from the limitations of UML. UML does not allow placing UML properties (which are used for SQL sequence modeling), or operations (which are used for SQL stored procedure & function modeling) directly into packages. Properties and operations can only occur in classes. A special model element was introduced to work around this limitation - GLOBALS element (based on UML class). This intermediate element can be placed directly inside the top level element (usually Schema, but can also be placed under Database) and then the necessary database elements - sequences, stored procedures can be placed inside it.

Database

 **Note**
Database is modeled as UML Package with Database stereotype applied.

Database is a top level element, representing entire database within DBMS. Besides the standard SQL element properties, database has the following properties available in the Specification window.

Property name	Description
Vendor	Specifies the vendor and the version of the database software. These fields are used for information purposes only. They do not affect the generation or further modeling.
Version	

Schema

 **Note**
SQL Schema is modeled as UML Package with Schema stereotype applied.


Schema element represents a collection of database elements - tables, indexes, stored procedures, etc. - grouped for particular purpose (such as data structures for some particular application).

Catalog

 **Note**
SQL Catalog is modeled as UML Package with Catalog stereotype applied.

Catalog element represents intermediate grouping level between database and schema. Catalogs are also reused for Oracle DB modeling - to implement Oracle packages.

GLOBALS

 **Note**
GLOBALS is modeled as UML Class with the «Globals» stereotype applied.

GLOBALS element is a special intermediate element to work around limitation of UML. UML does not allow placing UML properties (which are used for SQL sequence modeling), or UML operations (which are used for SQL stored procedure & function modeling) directly into packages. Properties and operations can only occur in classes.

To work around this limitation, GLOBALS element (based on UML class) was introduced. This intermediate element can be placed directly inside the top level element (usually Schema, but can also be placed under Database) and then the necessary database elements - sequences, stored procedures and functions can be placed inside it.

Name of GLOBALS model element is not important, but for the sake of tidiness it should be named "GLOBALS". There should be at most one such element per the container (Schema, Database, Package). This model element does not carry any additional useful properties; it serves just as a carrier of inner elements - sequences and routines.