# Views Configuration Reference

HotRod can leverage existing database views by providing DAO Java classes that retrieve data rows from them. It does not support, however, updating, deleting, or inserting of them.

Technically speaking database views can also be updated under some circumstances but these end up being quite varied for different databases. In general, updatable views tend to be based on simple queries and/or a single table, but this is not set in stone.

## The <view> Tag

The <view> tag tells HotRod which database views you want to use. Each <view> tag adds one view to the code generation and produces a DAO. This DAO, however, will be much more limited than a table DAO since it can only select rows using the "by example" strategy.

The <view> tag attributes are:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Description** | **Required** |
| name | The name of the database view. | Yes |
| java-name | The name of the DAO Java class, if different from the default name produced by HotRod. | No |

Most of the time a view definition is quite succinct. It only declares the view name and all the needed metadata is retrieved by HotRod from the database to produce all the DAO properties and methods. In this simple form the <view> tag may look like:

<view name="luxury\_car" />

This informs HotRod there's an LUXURY\_CAR view that is to be included in the code generation. The DAO Java name—and the properties names for its columns—will be generated by default from the database view structure, and the Java types will also be produced using the default rules HotRod has for this specific database. Basically, there is no custom configuration on it.

The name of the DAO Java class is taken from the view name, and prepended and appended with the prefix and suffix as specified in the header configuration section of the configuration file. By default there's no prefix, and by default the suffix is DAO. Therefore, in the example above the LUXURY\_CAR view will produce the DAO Java class LuxuryCarDAO.java.

Now, if you want to use a different name for the DAO Java class, you can specify it in the <view> tag using the java-name attribute. Mind this name must be a valid Java class name, starting with an upper case letter. If, for example, we wanted to use a custom name for the DAO the configuration could look like:

<view name="luxury\_car" java-name="LuxuryVehicle" />

This configuration will produce the DAO Java class LuxuryVehicle.java. Notice that when you openly specify the DAO Java name, HotRod does not prepend or append the prefix and suffix, but it takes it exactly as you declared it.

## The <column> Tag

Each <view> tag can include one or more <column> tags to stipulate specific properties for some or all the columns of the view. The view columns with no <column> definition use the default properties provided by the code generator.

For details on the settings of a <column> tag see the Configuration Reference section for Columns.

## The <sequence> Tag

Each <view> tag can include one or more <sequence> tags. These definitions generate a Java method to directly retrieve a value from the specified sequence. There's no relationship between the named sequence and the <view> tag; the view DAO only serves as a Java class where to place the corresponding Java method to retrieve the value.

For details on the settings of a <sequence> tag see the Configuration Reference section for Sequences.

## The <query> Tag

Each <view> tag can include one or more <query> tags. Each <query> tag contains a SQL statement the developer wants to attach as a method to the DAO. There's no relationship between the SQL statement and the view; the view DAO only serves as a Java class where to place the corresponding Java method that executes the SQL statement. The SQL statement must not necessarily be an SQL update, but can also be an SQL insert, or SQL delete.

For details on how to define an <query> tag see the Configuration Reference section for Updates.