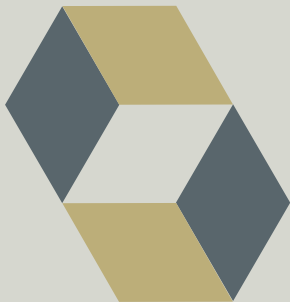


# Hibernate

Hibernate



## Introduction to **Hibernate**

Hibernate

```
Public class PhotoUploader {
```

```
    public void uploadPhoto (Album album, title String, desc String, ... ){
```

```
        Photo uploadedPhoto = new Photo();
```

```
        uploadedPhoto.setTitle(title);
```

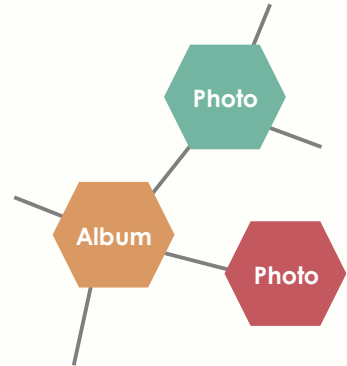
```
        uploadedPhoto.setDescription(desc);
```

```
        ...
```

```
        album.addPhoto(uploadedPhoto);
```

```
    }
```

```
}
```



Language like Java is object-oriented. Such languages represent data as **interconnected graph of objects**.

Hibernate

Album ID	Title	...
1	Memorable	
2	Snapshot	

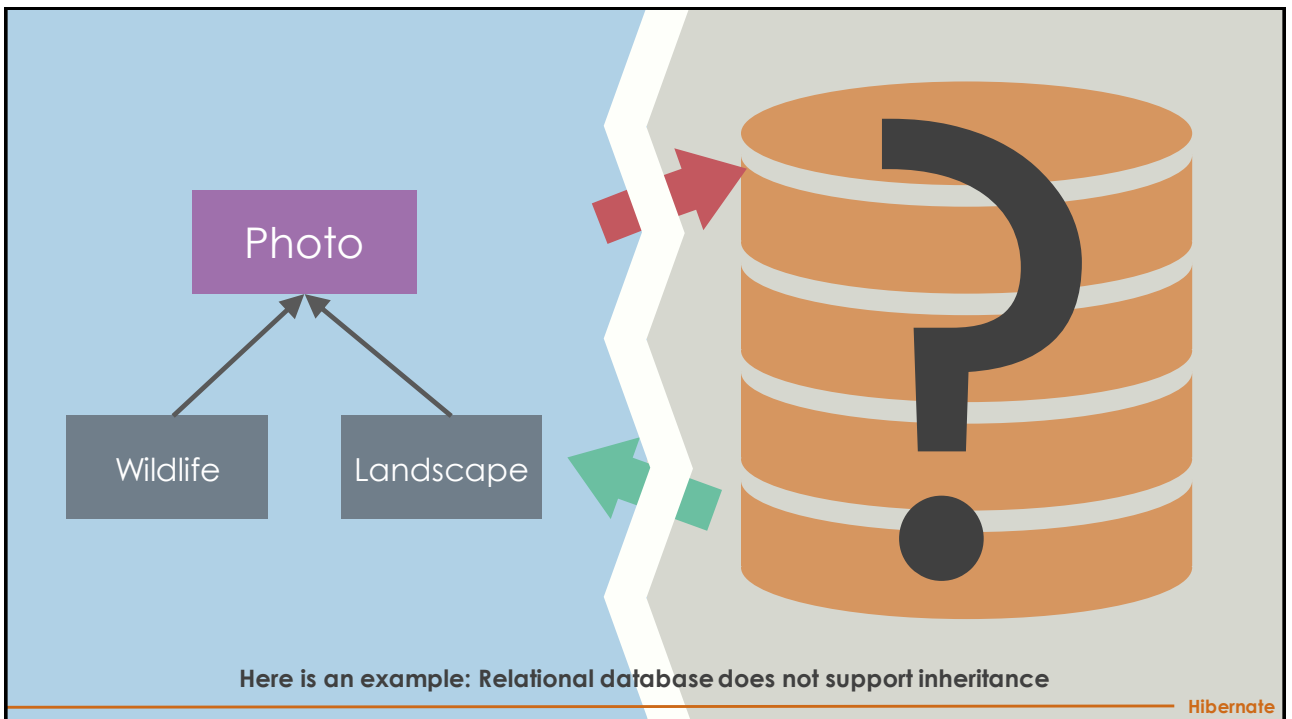
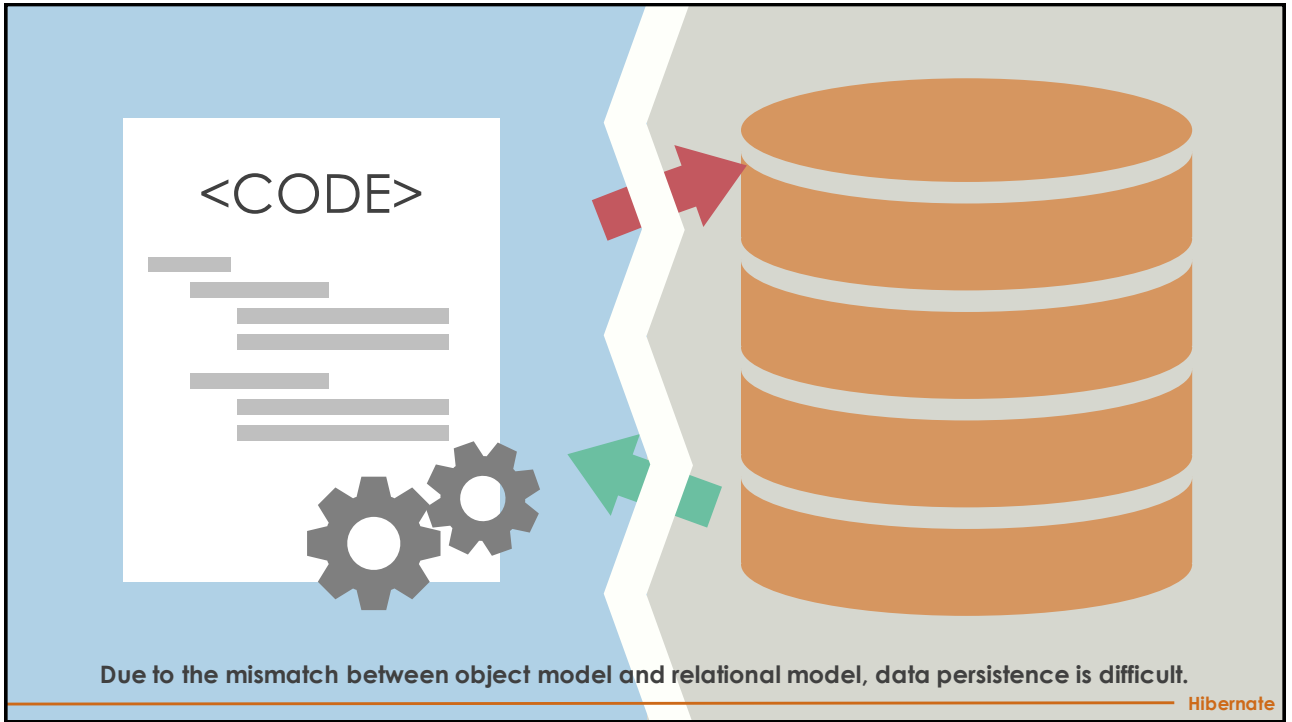
Album

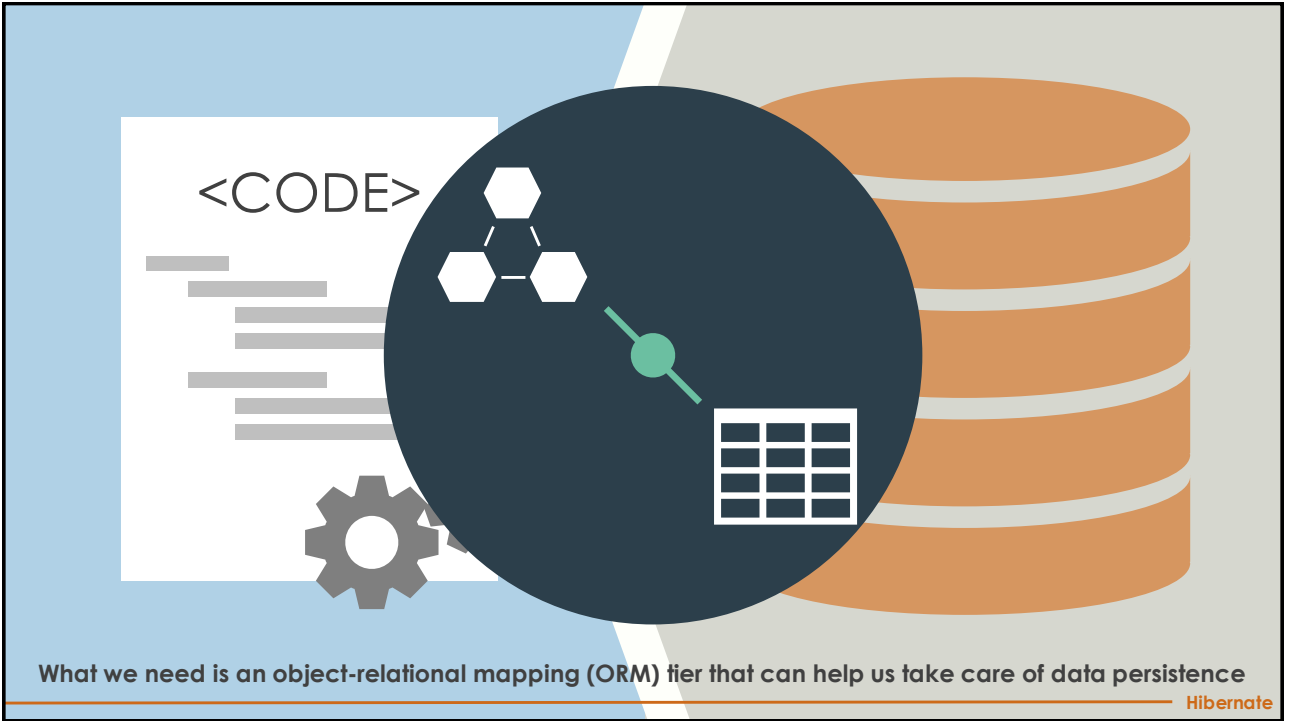
Photo ID	Album ID	Title	...
1	1	The tram	
2	1	Little kitten	
3	1	Chamomile	
4	2	New bus route	
5	2	Ferry pier 9	
6	2	The coast	

Photo

Relational database represents data in **tabular** format (like spreadsheet)

Hibernate





What we need is an object-relational mapping (ORM) tier that can help us take care of data persistence

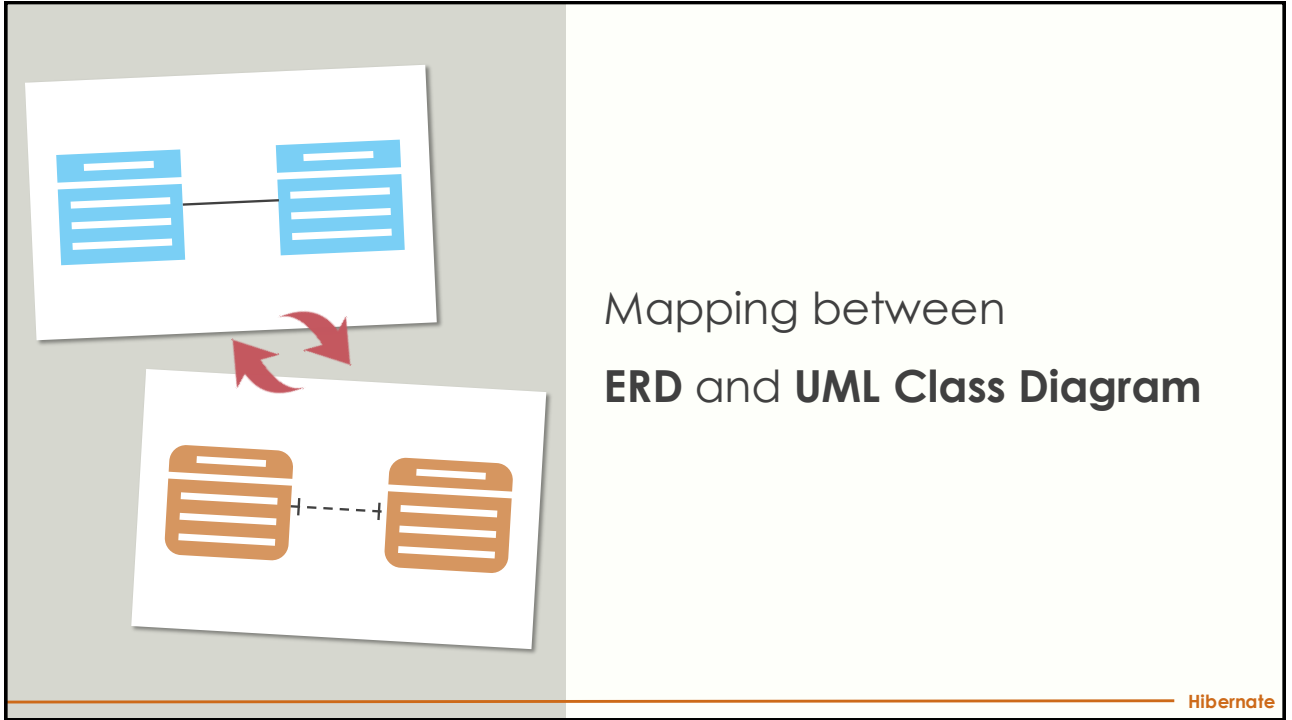
Hibernate



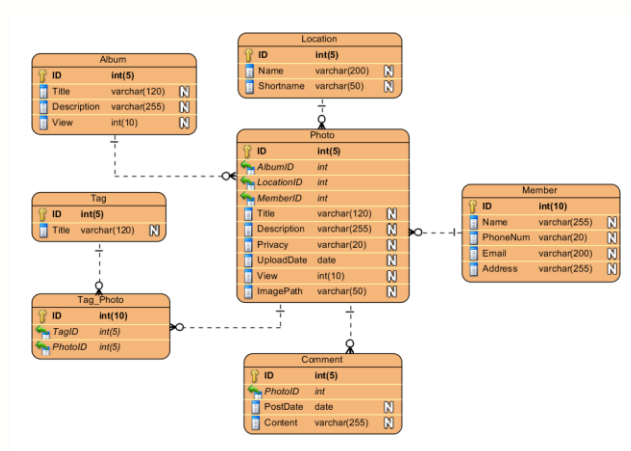
- Part of a community of Red Hat projects -

Hibernate is an object-relational mapping (ORM) library that can serve this purpose.

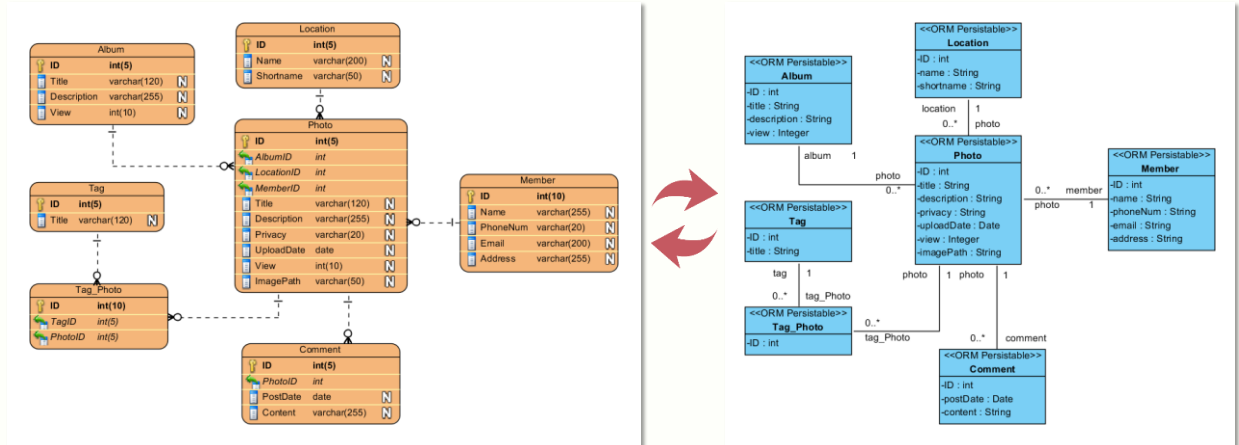
Hibernate



## Example



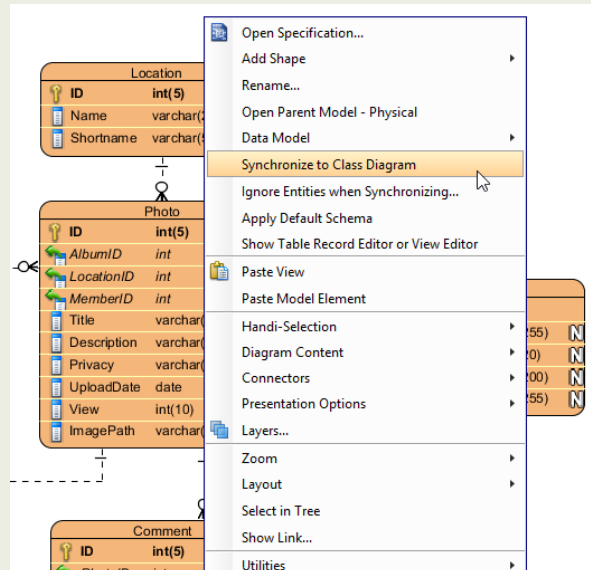
## Example



Hibernate

## Synchronization – The steps

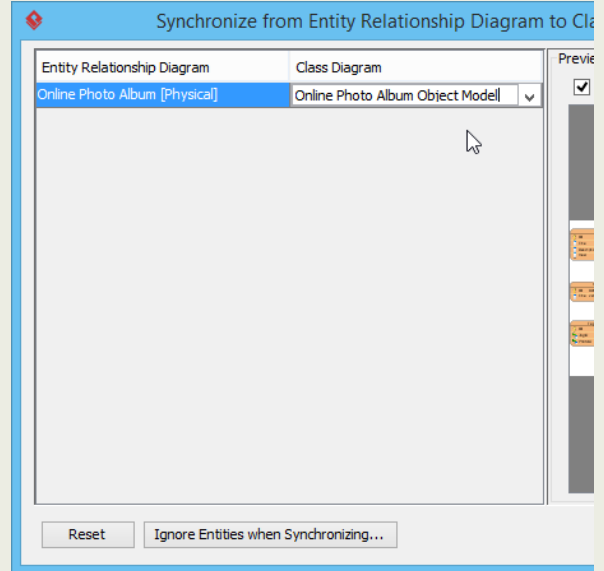
1. Right click on the source ERD and select **Synchronize to Class Diagram...** from the popup menu.



Hibernate

## Synchronization – The steps

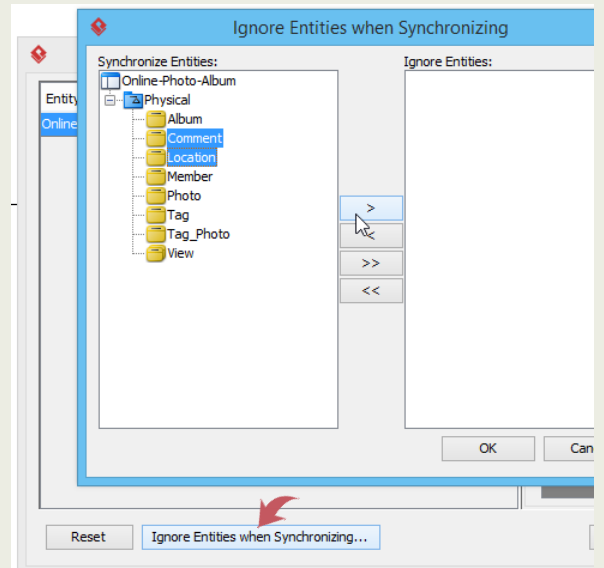
1. Right click on the source ERD and select **Synchronize to Class Diagram...** from the popup menu.
2. Select the class diagram to synchronize to, or enter a new name to form a new diagram.



Hibernate

## Synchronization – The steps

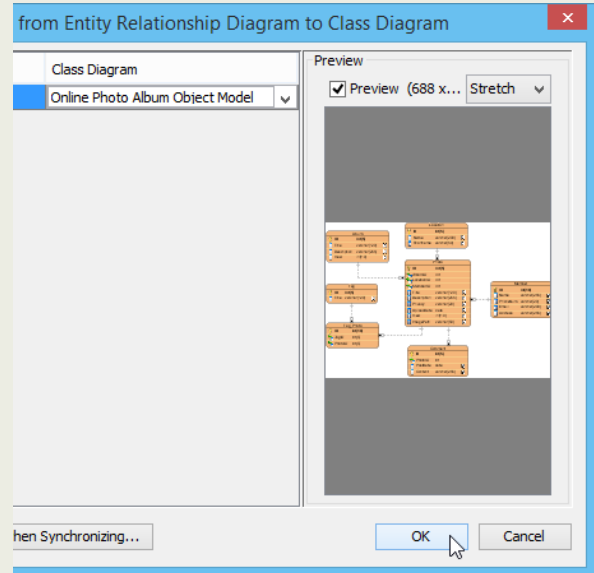
1. Right click on the source ERD and select **Synchronize to Class Diagram...** from the popup menu.
2. Select the class diagram to synchronize to, or enter a new name to form a new diagram.
3. **[Optional]** Select the entities that you want to skip in synchronization.



Hibernate

## Synchronization – The steps

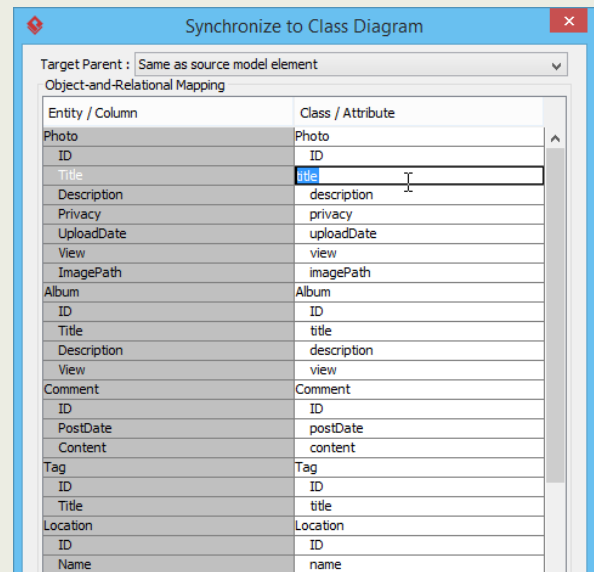
1. Right click on the source ERD and select **Synchronize to Class Diagram...** from the popup menu.
2. Select the class diagram to synchronize to, or enter a new name to form a new diagram.
3. **[Optional]** Select the entities that you want to skip in synchronization.
4. Click **OK**.



Hibernate

## Synchronization – The steps

1. Right click on the source ERD and select **Synchronize to Class Diagram...** from the popup menu.
2. Select the class diagram to synchronize to, or enter a new name to form a new diagram.
3. **[Optional]** Select the entities that you want to skip in synchronization.
4. Click **OK**.
5. **[Optional]** Double click on a class/attribute name to rename it.



Hibernate



## Synchronization – The steps

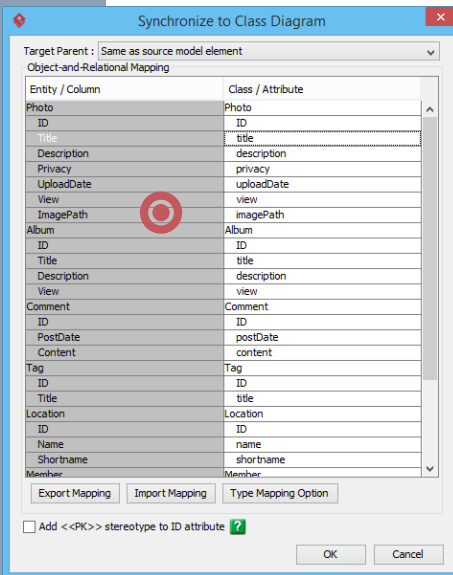
## Details of Mapping Window



- The place to store the class diagram and the generated classes.
  - **Same as source model element** – The parent of the source ERD.
  - **Root** – The project root.
  - **Specify** – Select a model manually.

## Synchronization – The steps

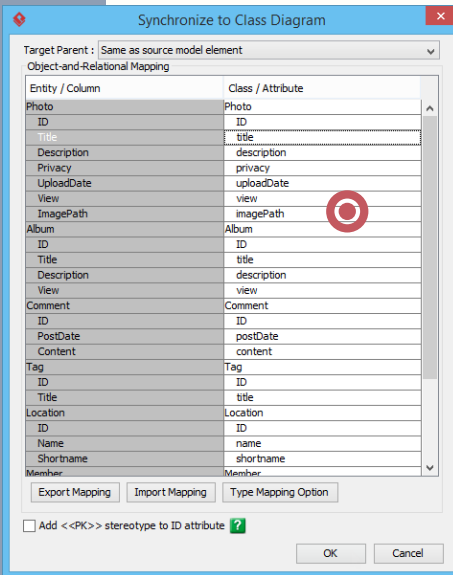
## Details of Mapping Window



- The list of entities and columns to be synchronized to object model.

## Synchronization – The steps

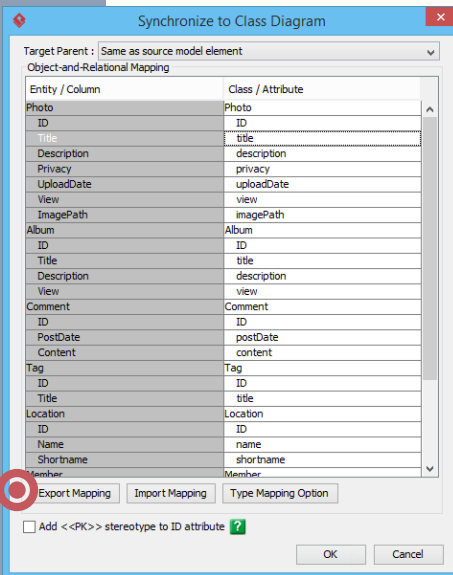
## Details of Mapping Window



- The list of classes and attributes to be created from entities and their columns.
- You can double click on a class or attribute rename it.

## Synchronization – The steps

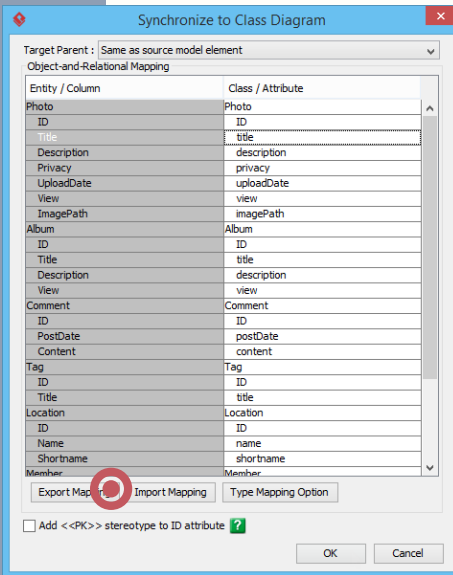
## Details of Mapping Window



- Export the mapping into an Excel file.
- You can rename classes and attributes in Excel, and then import the changes by clicking **Import Mapping**.

## Synchronization – The steps

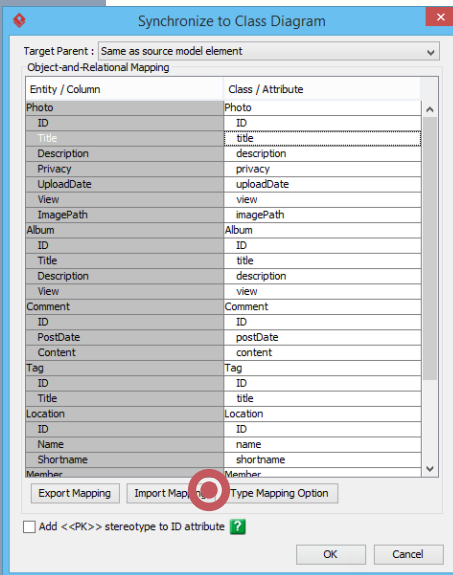
## Details of Mapping Window



- Import an exported Excel to update the mapping.

## Synchronization – The steps

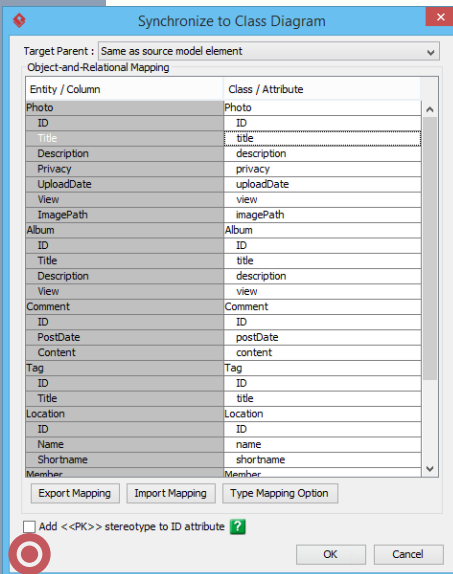
## Details of Mapping Window



- Enforce numeric and decimal typed columns to be synchronized as attributes in Integer, Float, Double or BigDecimal type.

## Synchronization – The steps

## Details of Mapping Window

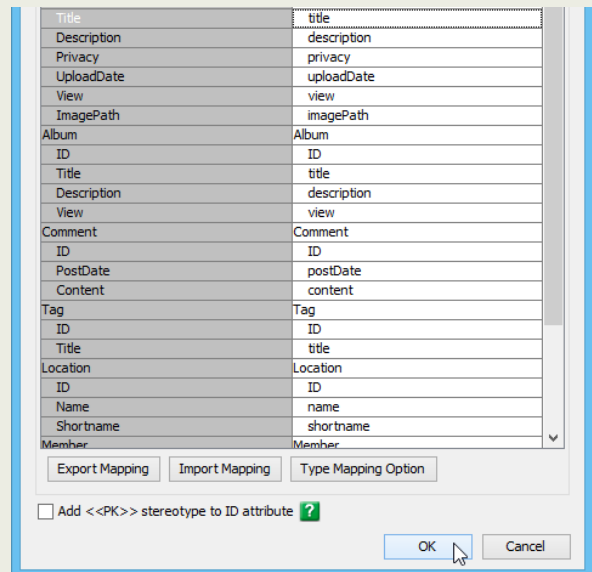


- If you want to indicate in class diagram that an attribute is mapping with a primary key column in ERD, you can check this option to add <<PK>> to those “primary key attributes”.

Hibernate

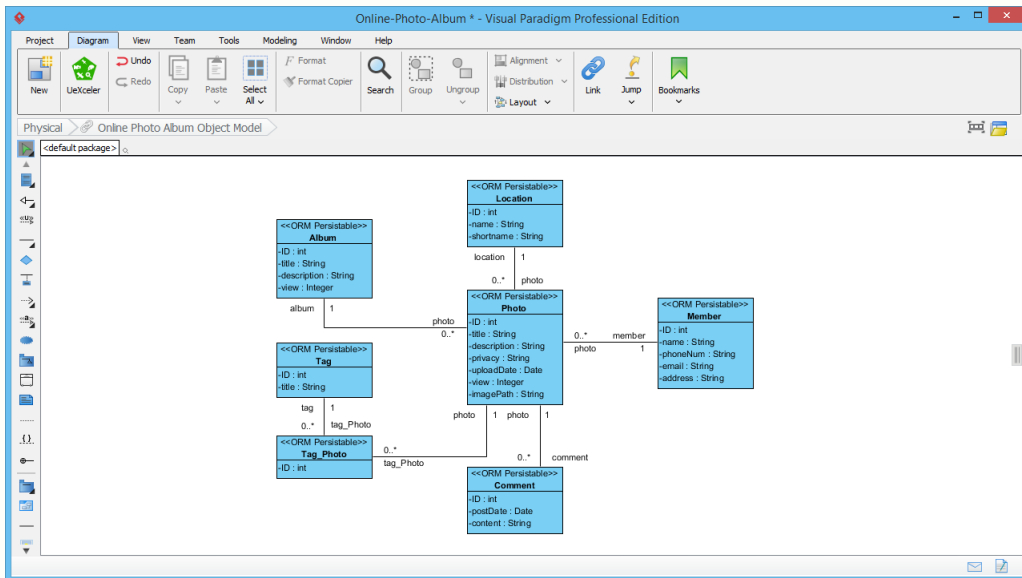
## Synchronization – The steps

1. Right click on the source ERD and select **Synchronize to Class Diagram...** from the popup menu.
2. Select the class diagram to synchronize to, or enter a new name to form a new diagram.
3. **[Optional]** Select the entities that you want to skip in synchronization.
4. Click **OK**.
5. **[Optional]** Double click on a class/attribute name to rename it.
6. Click **OK**.



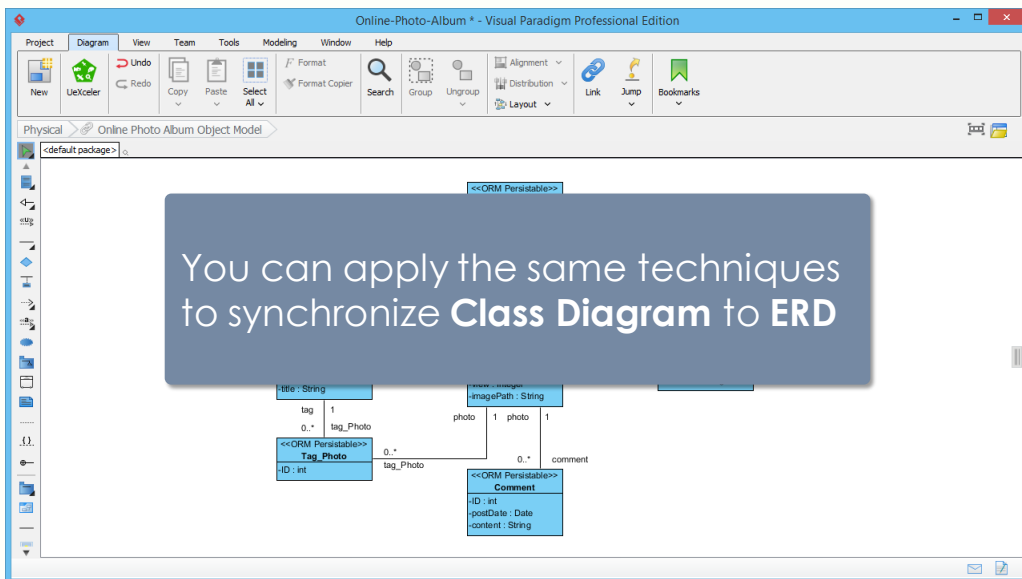
Hibernate

## Synchronization – The steps

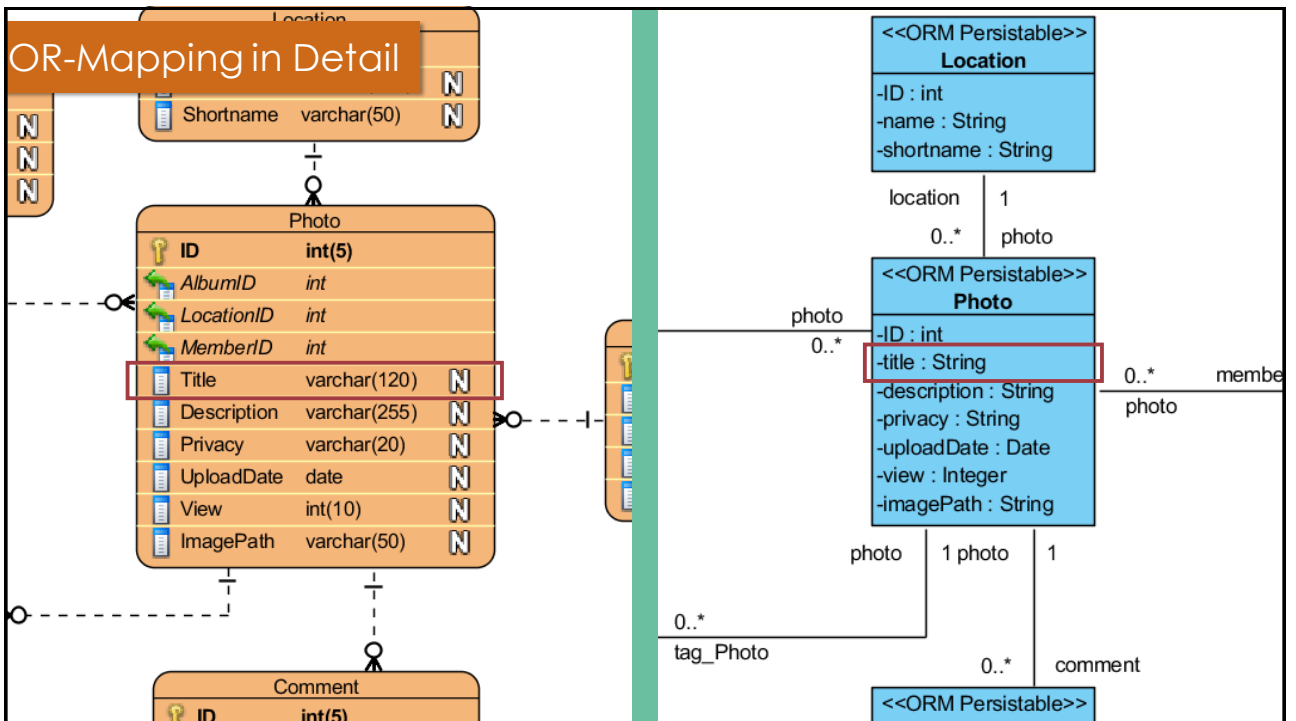
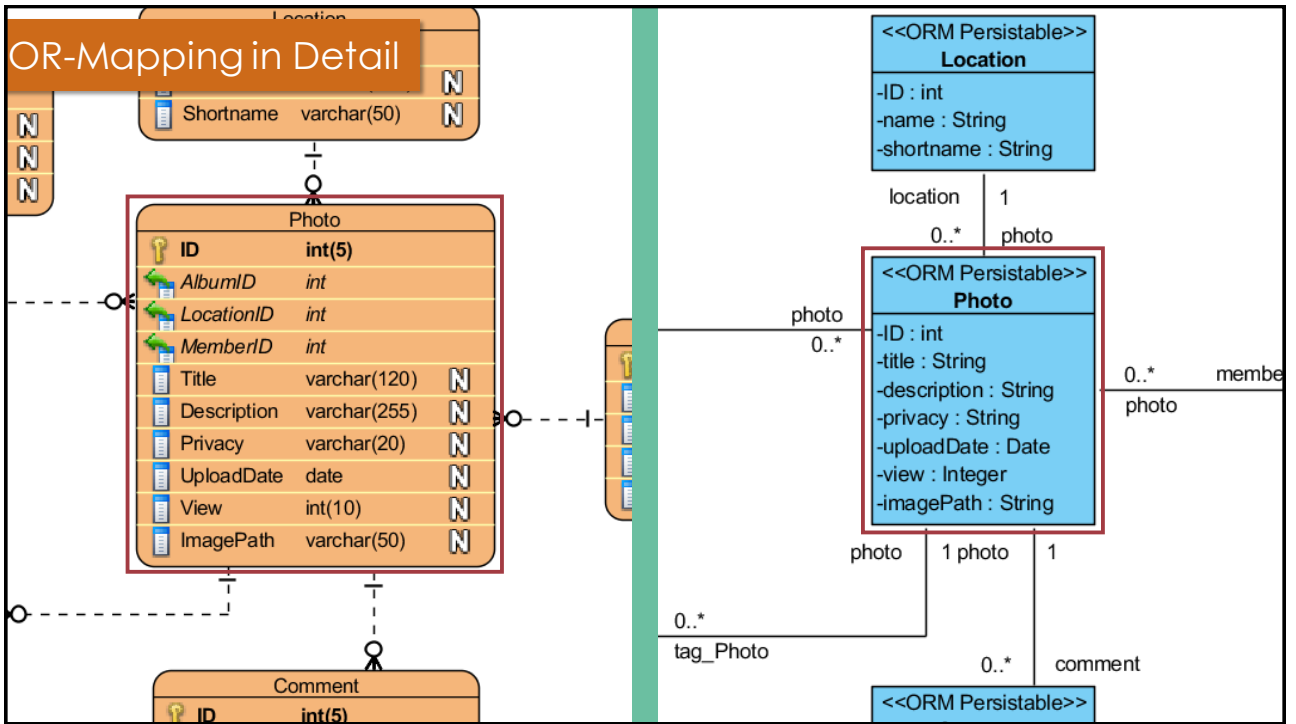


Hibernate

## Synchronization – The steps

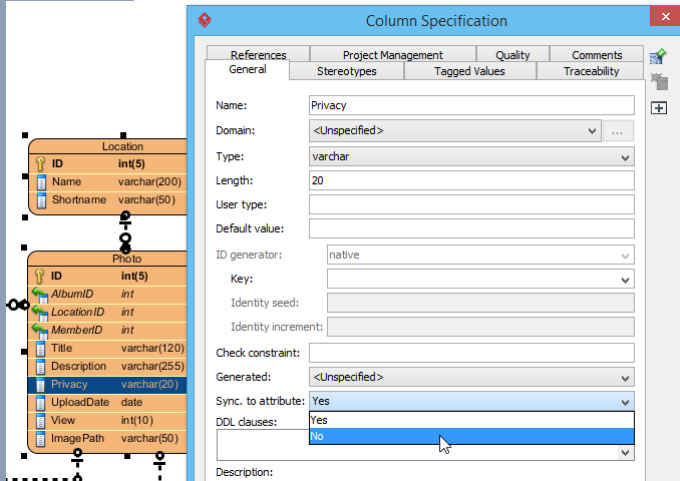


Hibernate



## OR-Mapping in Detail

To make an entity column **NOT** synchronized with class attribute:

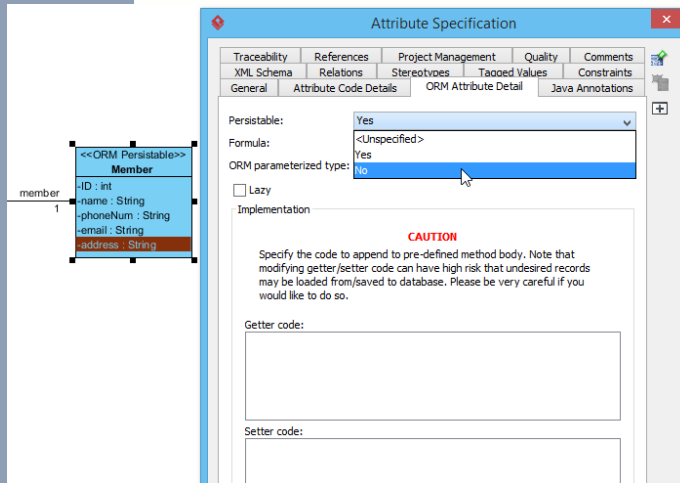


In the **Column Specification**, choose **No** for **Sync. to attribute**. In the next synchronization, this column will be ignored.

Hibernate

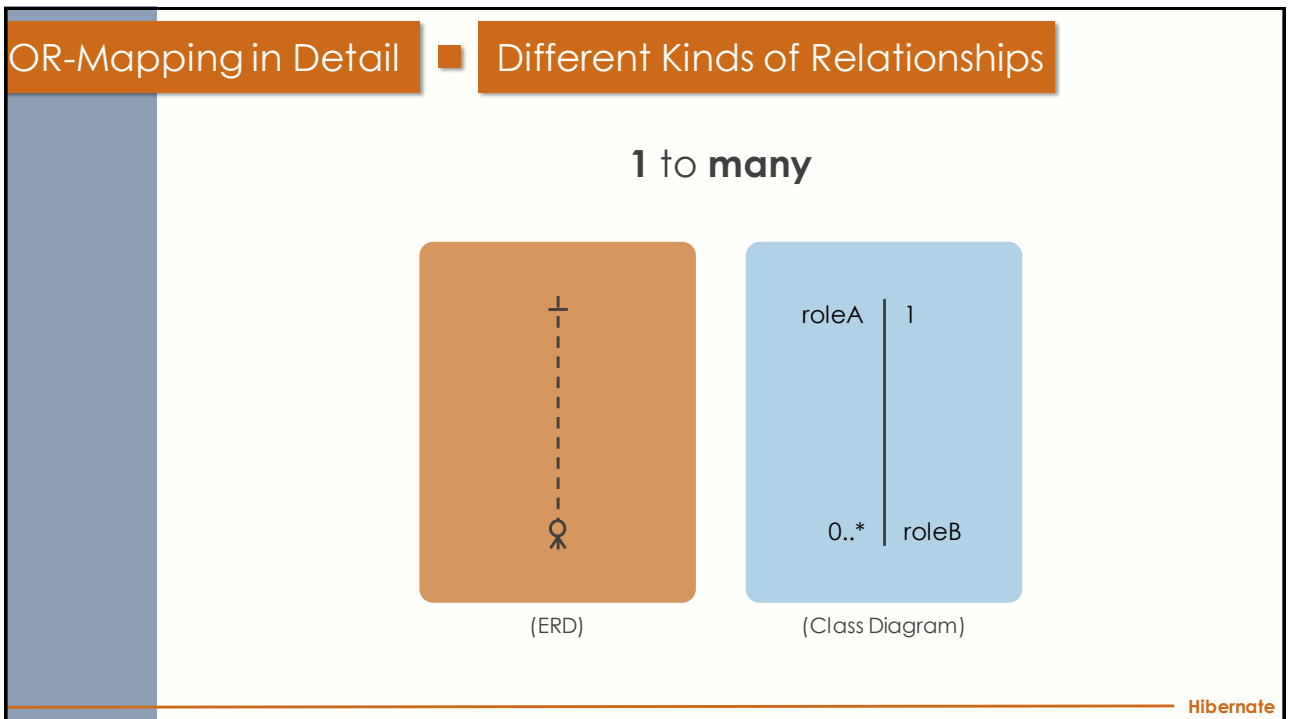
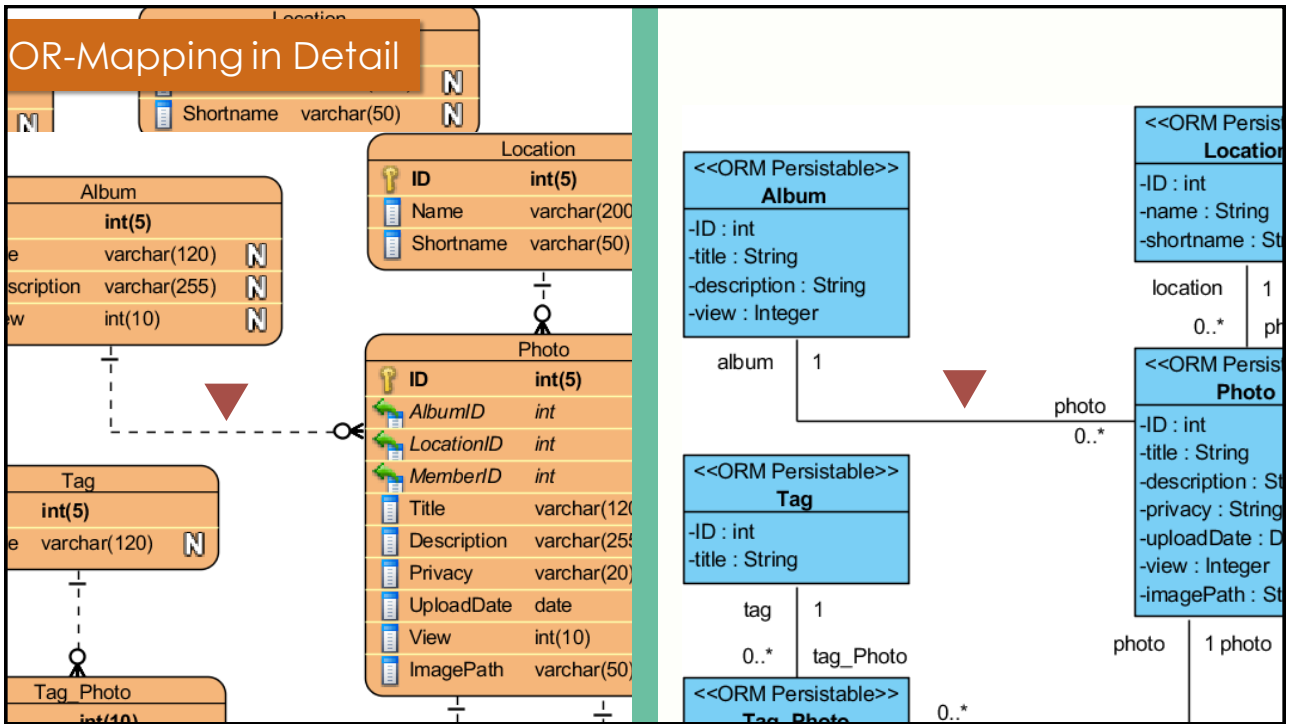
## OR-Mapping in Detail

To make a class attribute **NOT** synchronized with entity column:



In the **Attribute Specification**, open the **ORM Attribute Detail** tab and choose **No** for **Persistable**. In the next synchronization, this attribute will be ignored.

Hibernate

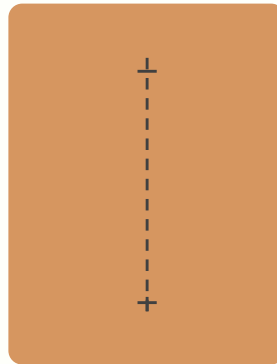




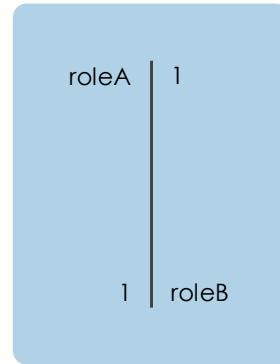
## OR-Mapping in Detail

### Different Kinds of Relationships

#### 1 to 1



(ERD)

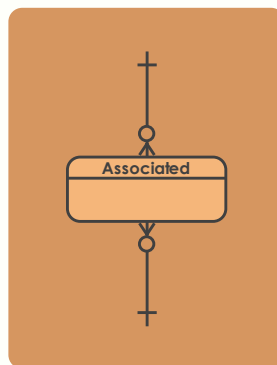


(Class Diagram)

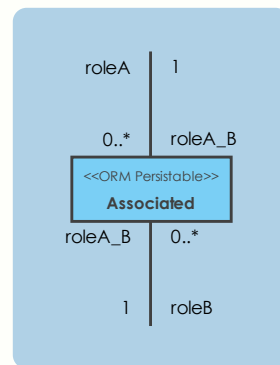
## OR-Mapping in Detail

### Different Kinds of Relationships

#### Many to Many



(ERD)



(Class Diagram)

## OR-Mapping in Detail ■ Inheritance

So, how is inheritance in object model  
(i.e. **generalization** in UML) being mapped?

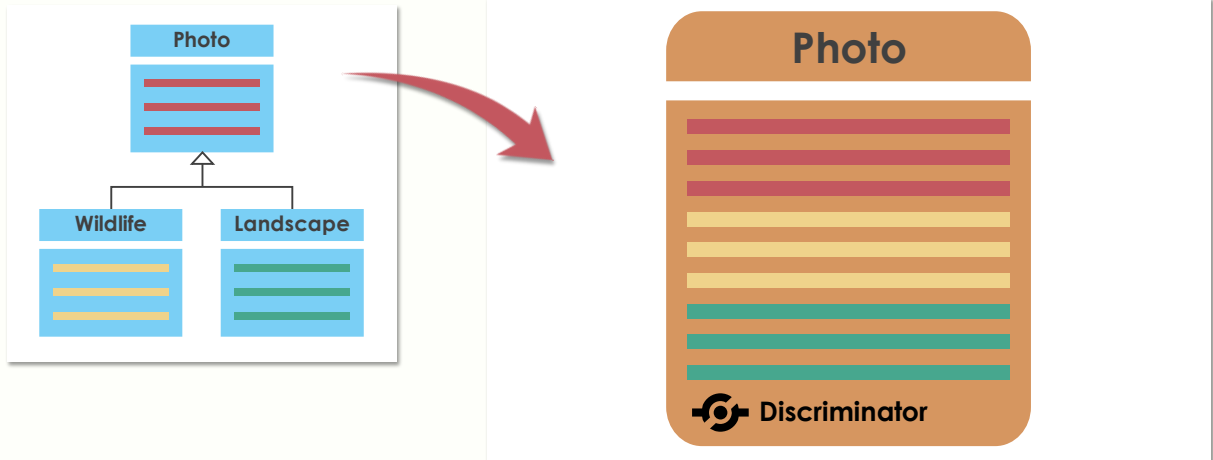
Hibernate

## OR-Mapping in Detail ■ Inheritance

Strategy 1 - Table per class hierarchy

Hibernate

## OR-Mapping in Detail ■ Inheritance



Note: A **discriminator column** is used for identifying the kind of photo a photo record belongs to.

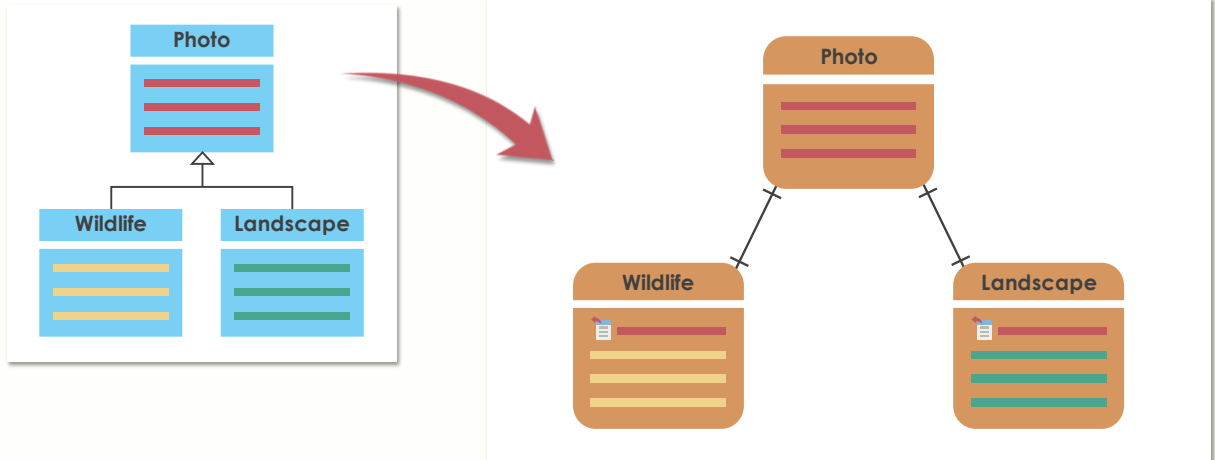
Hibernate

## OR-Mapping in Detail ■ Inheritance

Strategy 2 - Table per subclass

Hibernate

## OR-Mapping in Detail ■ Inheritance



Note: Foreign key is used to links the “sub-entities” with the “parent entity”

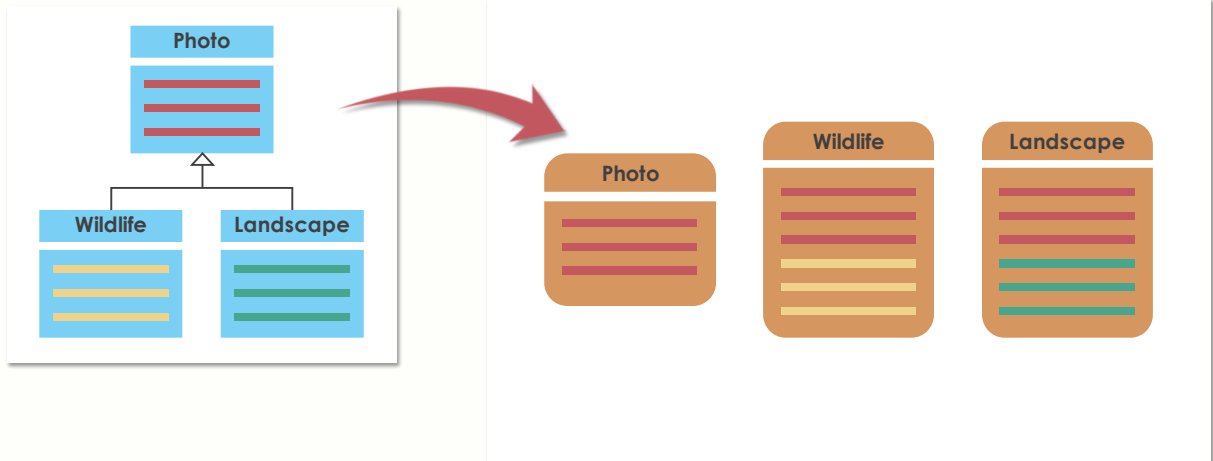
Hibernate

## OR-Mapping in Detail ■ Inheritance

Strategy 3 - Table per concrete class

Hibernate

## OR-Mapping in Detail ■ Inheritance

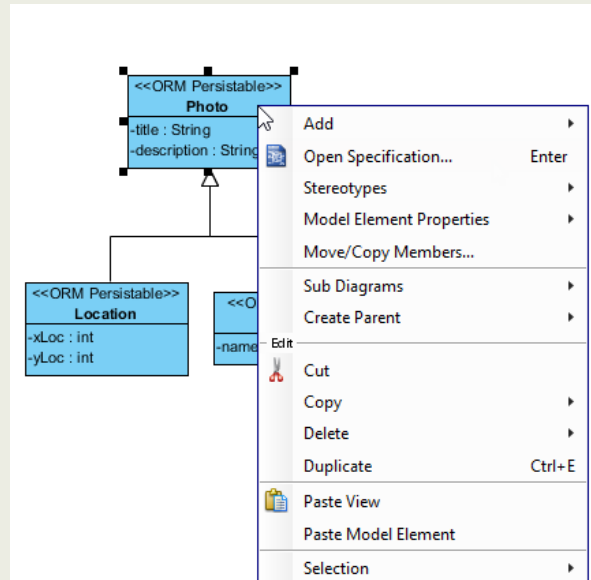


**Note:** Entities are not linked with each other. Columns from “parent entity” are copied to “sub-entities”.

Hibernate

## Configuring Inheritance Strategy

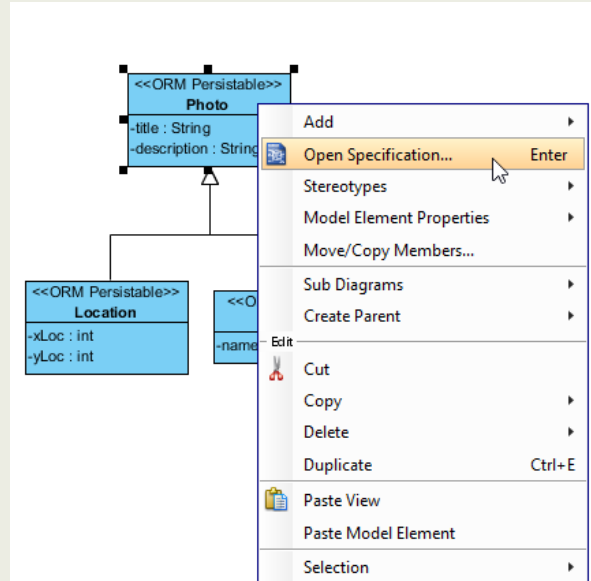
1. Right click on the super class within a generalization hierarchy.



Hibernate

## Configuring Inheritance Strategy

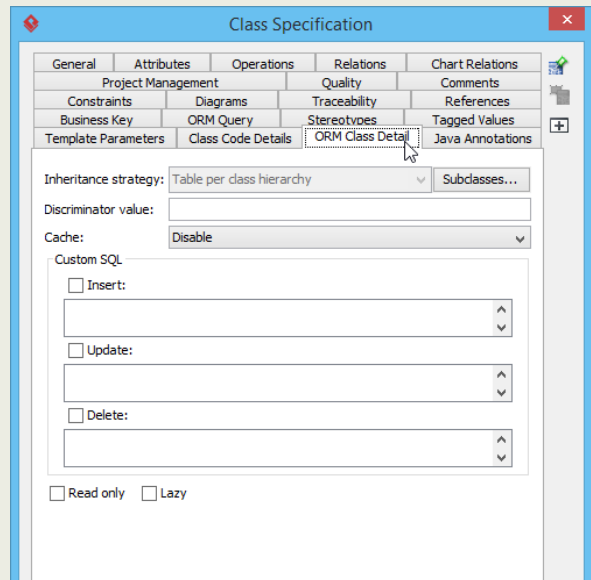
1. Right click on the super class within a generalization hierarchy.
2. Select **Open Specification...** from the popup menu.



Hibernate

## Configuring Inheritance Strategy

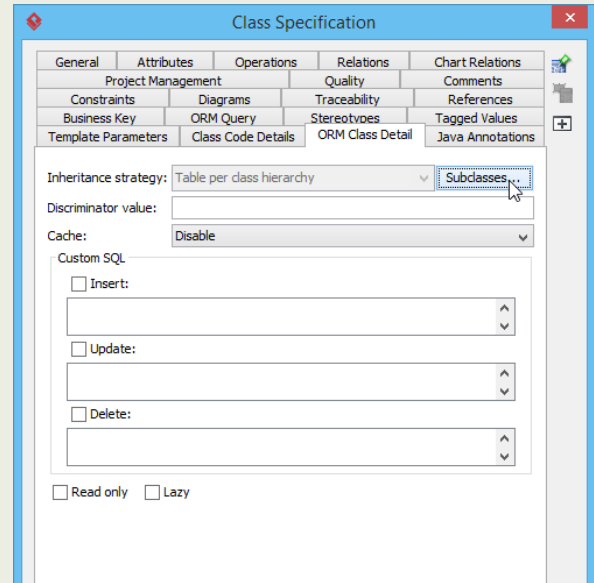
1. Right click on the super class within a generalization hierarchy.
2. Select **Open Specification...** from the popup menu.
3. Open the **ORM Class Detail** tab.



Hibernate

## Configuring Inheritance Strategy

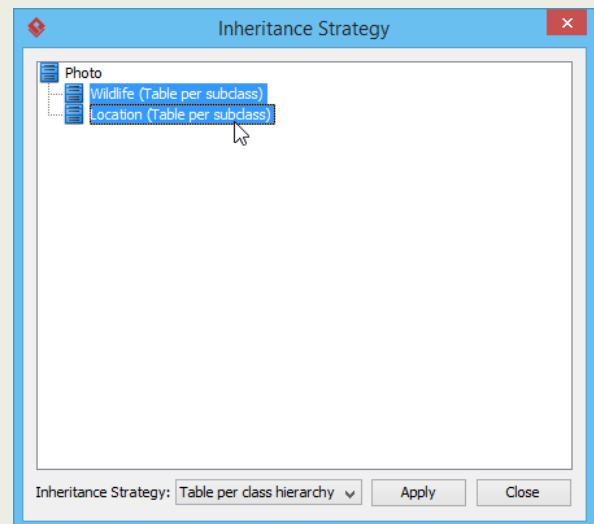
1. Right click on the super class within a generalization hierarchy.
2. Select **Open Specification...** from the popup menu.
3. Open the **ORM Class Detail** tab.
4. Click **Subclasses**.



Hibernate

## Configuring Inheritance Strategy

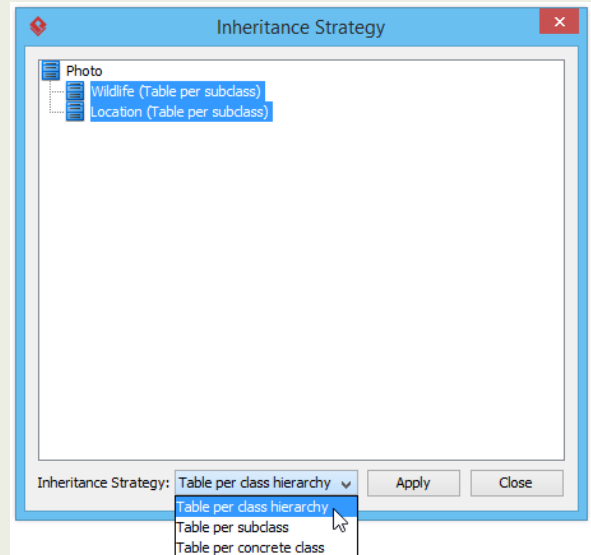
1. Right click on the super class within a generalization hierarchy.
2. Select **Open Specification...** from the popup menu.
3. Open the **ORM Class Detail** tab.
4. Click **Subclasses**.
5. Select the sub-classes. Press **Shift/Ctrl** to perform a multi selection.



Hibernate

## Configuring Inheritance Strategy

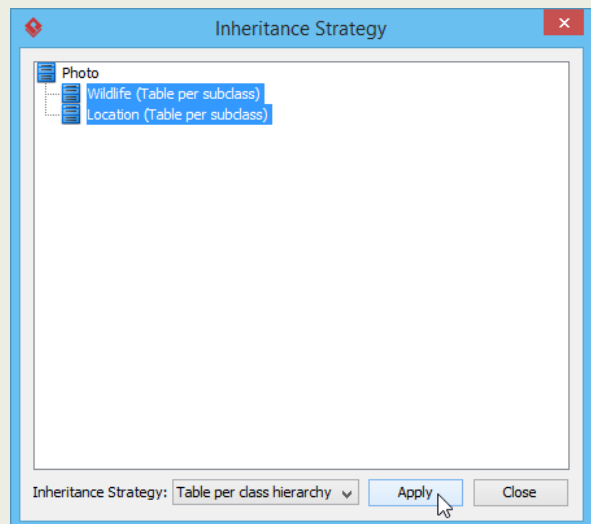
1. Right click on the super class within a generalization hierarchy.
2. Select **Open Specification...** from the popup menu.
3. Open the **ORM Class Detail** tab.
4. Click **Subclasses**.
5. Select the sub-classes. Press **Shift/Ctrl** to perform a multi selection.
6. Select the **Inheritance Strategy**.



Hibernate

## Configuring Inheritance Strategy

1. Right click on the super class within a generalization hierarchy.
2. Select **Open Specification...** from the popup menu.
3. Open the **ORM Class Detail** tab.
4. Click **Subclasses**.
5. Select the sub-classes. Press **Shift/Ctrl** to perform a multi selection.
6. Select the **Inheritance Strategy**.
7. Click **Apply**.

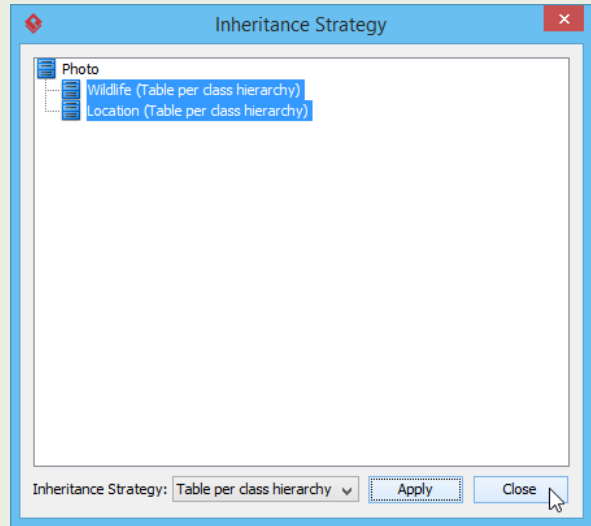


Hibernate



## Configuring Inheritance Strategy

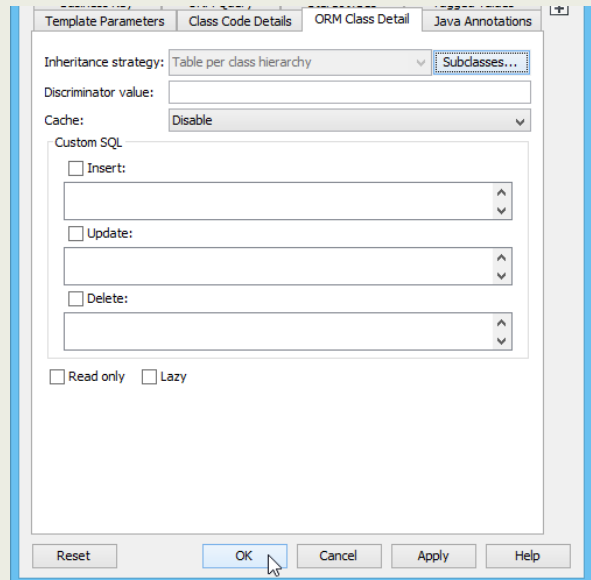
1. Right click on the super class within a generalization hierarchy.
2. Select **Open Specification...** from the popup menu.
3. Open the **ORM Class Detail** tab.
4. Click **Subclasses**.
5. Select the sub-classes. Press **Shift/Ctrl** to perform a multi selection.
6. Select the **Inheritance Strategy**.
7. Click **Apply**.
8. Click **Close**.



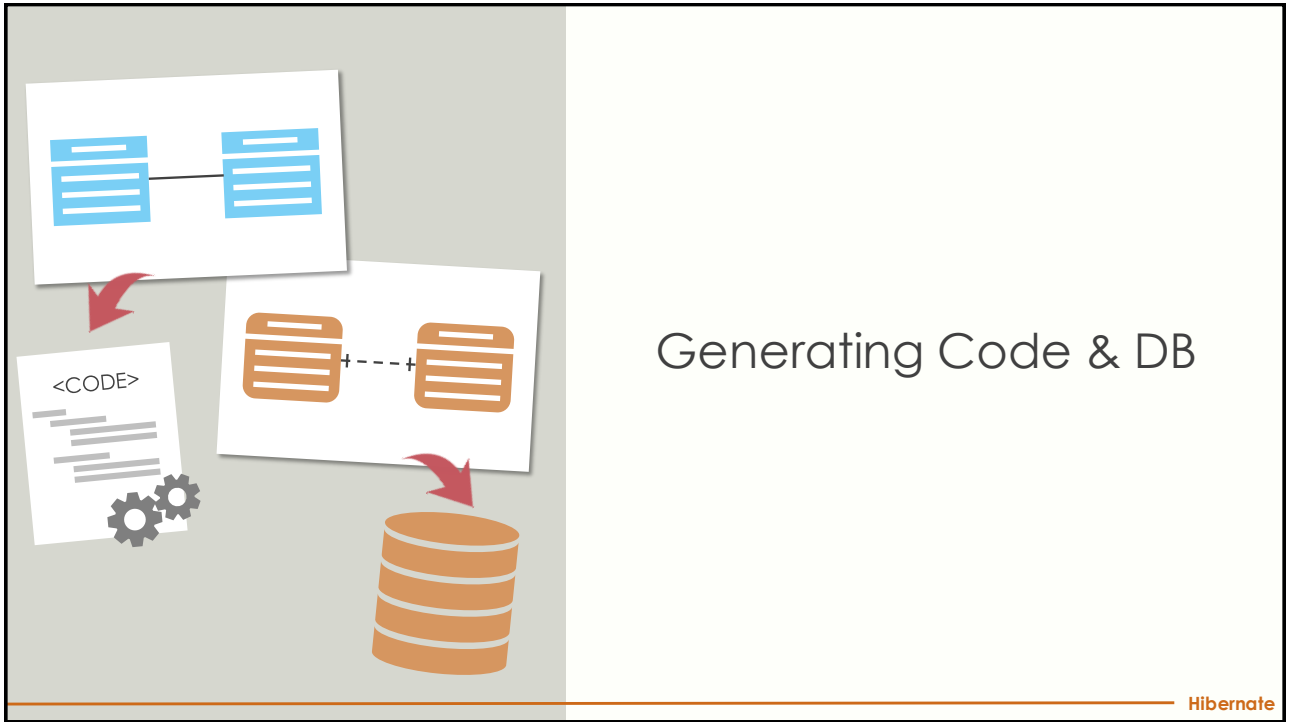
Hibernate

## Configuring Inheritance Strategy

1. Right click on the super class within a generalization hierarchy.
2. Select **Open Specification...** from the popup menu.
3. Open the **ORM Class Detail** tab.
4. Click **Subclasses**.
5. Select the sub-classes. Press **Shift/Ctrl** to perform a multi selection.
6. Select the **Inheritance Strategy**.
7. Click **Apply**.
8. Click **Close**.
9. Click **OK**.

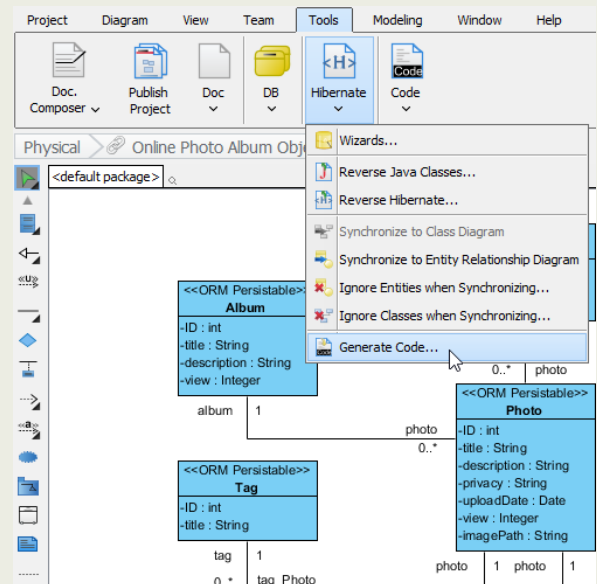


Hibernate



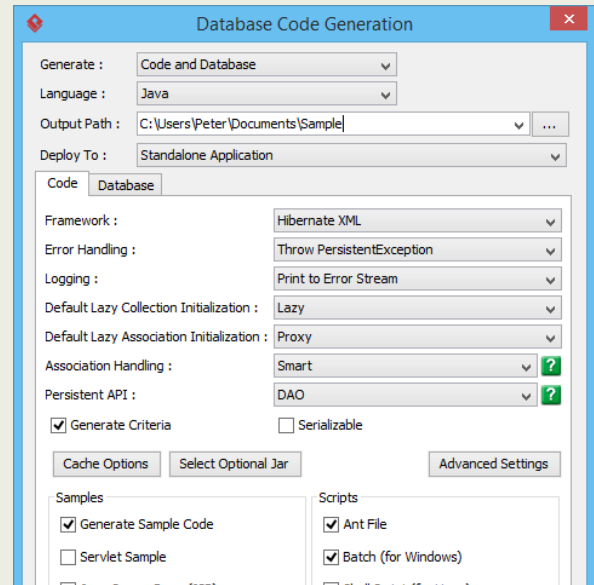
## Generating Code & DB

1. Under the **Tools** tab of Visual Paradigm, select **Hibernate > Generate Code...**



## Generating Code & DB

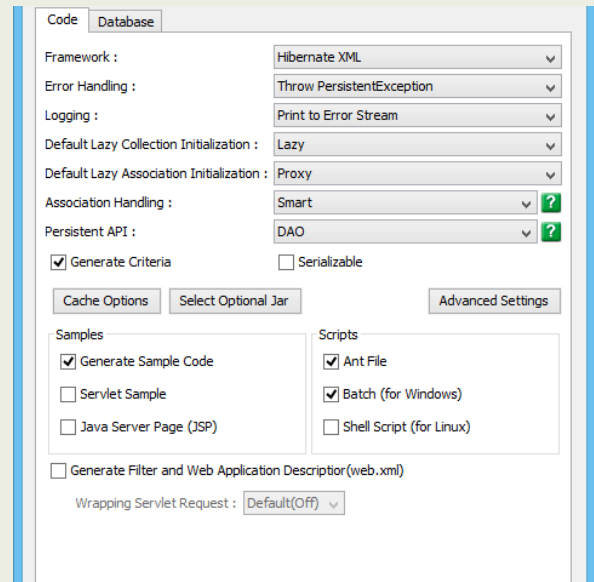
1. Under the **Tools** tab of Visual Paradigm, select **Hibernate > Generate Code...**
2. Fill in the output path for storing the generated file like the DDL file.



Hibernate

## Generating Code & DB

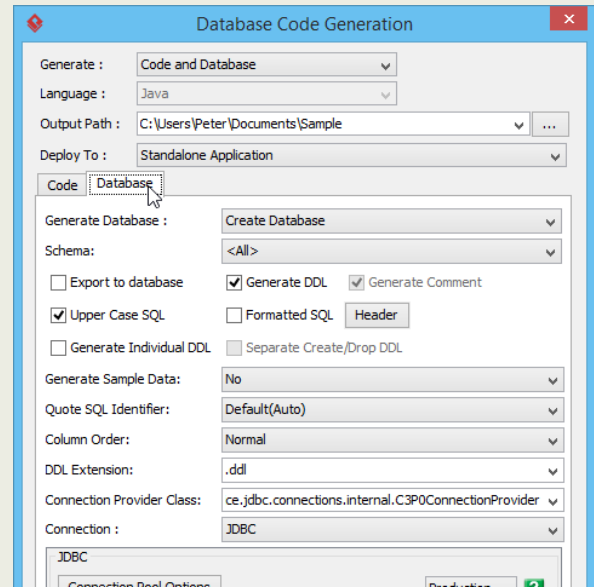
1. Under the **Tools** tab of Visual Paradigm, select **Hibernate > Generate Code...**
2. Fill in the output path for storing the generated file like the DDL file.
3. Configure the generation options.



Hibernate

## Generating Code & DB

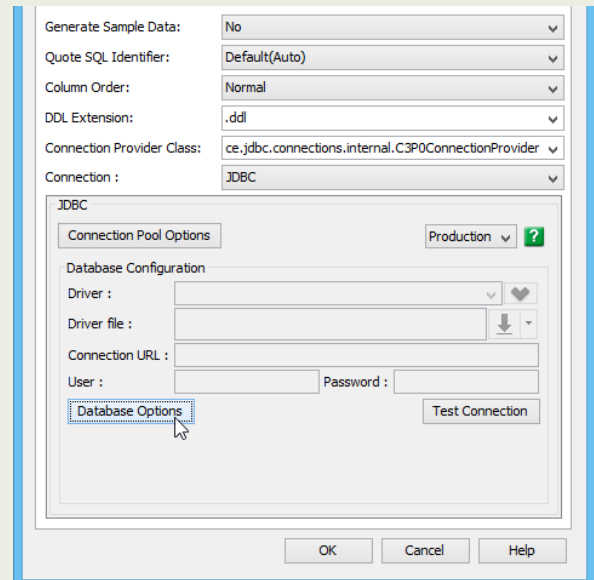
1. Under the **Tools** tab of Visual Paradigm, select **Hibernate > Generate Code...**
2. Fill in the output path for storing the generated file like the DDL file.
3. Configure the generation options.
4. Open the **Database** tab.



Hibernate

## Generating Code & DB

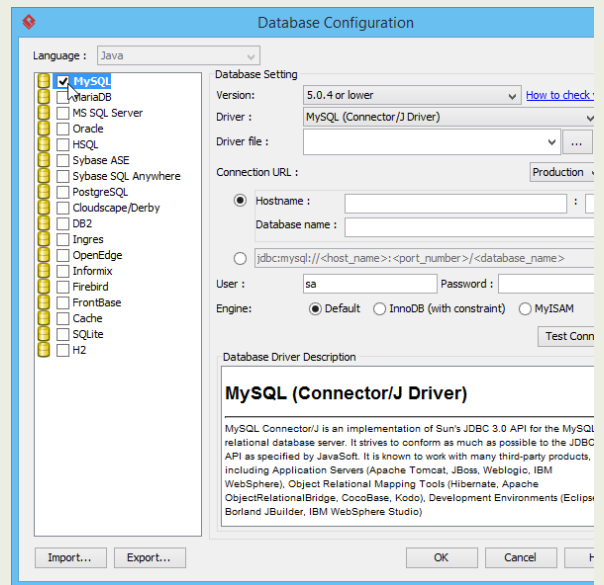
1. Under the **Tools** tab of Visual Paradigm, select **Hibernate > Generate Code...**
2. Fill in the output path for storing the generated file like the DDL file.
3. Configure the generation options.
4. Open the **Database** tab.
5. Click on **Database Options**.



Hibernate

## Generating Code & DB

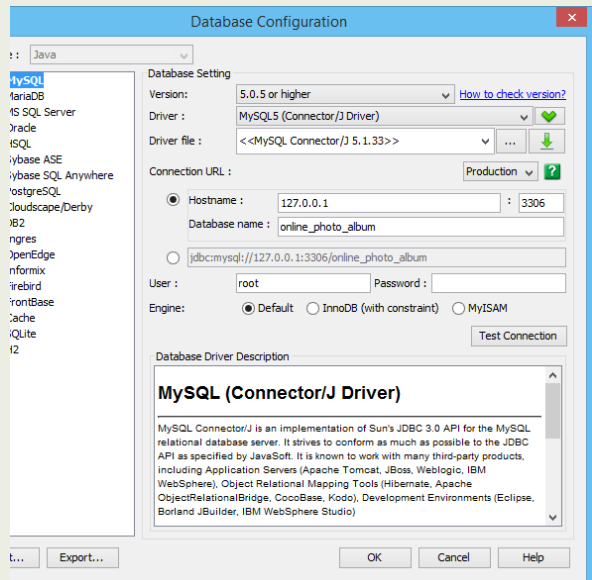
1. Under the **Tools** tab of Visual Paradigm, select **Hibernate > Generate Code...**
2. Fill in the output path for storing the generated file like the DDL file.
3. Configure the generation options.
4. Open the **Database** tab.
5. Click on **Database Options**.
6. Select the DBMS you use.



Hibernate

## Generating Code & DB

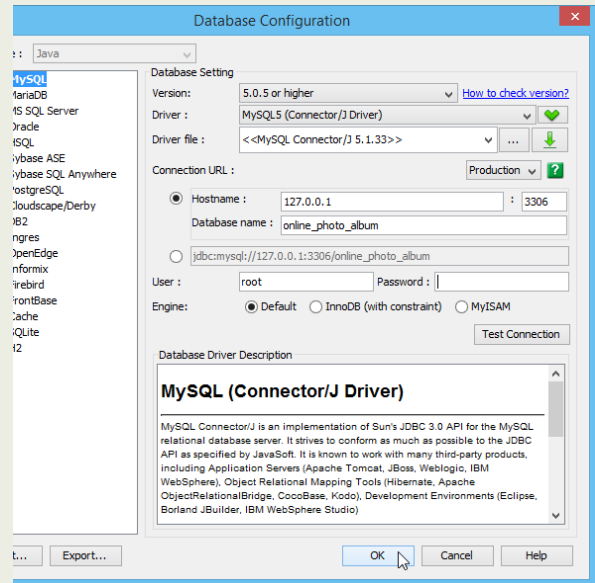
1. Under the **Tools** tab of Visual Paradigm, select **Hibernate > Generate Code...**
2. Fill in the output path for storing the generated file like the DDL file.
3. Configure the generation options.
4. Open the **Database** tab.
5. Click on **Database Options**.
6. Select the DBMS you use.
7. Configure the database connection.



Hibernate

## Generating Code & DB

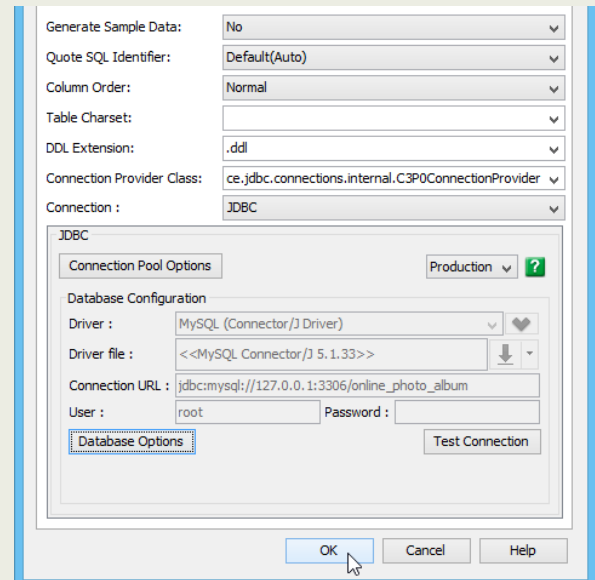
1. Under the **Tools** tab of Visual Paradigm, select **Hibernate > Generate Code...**
2. Fill in the output path for storing the generated file like the DDL file.
3. Configure the generation options.
4. Open the **Database** tab.
5. Click on **Database Options**.
6. Select the DBMS you use.
7. Configure the database connection.
8. Click **OK**.



Hibernate

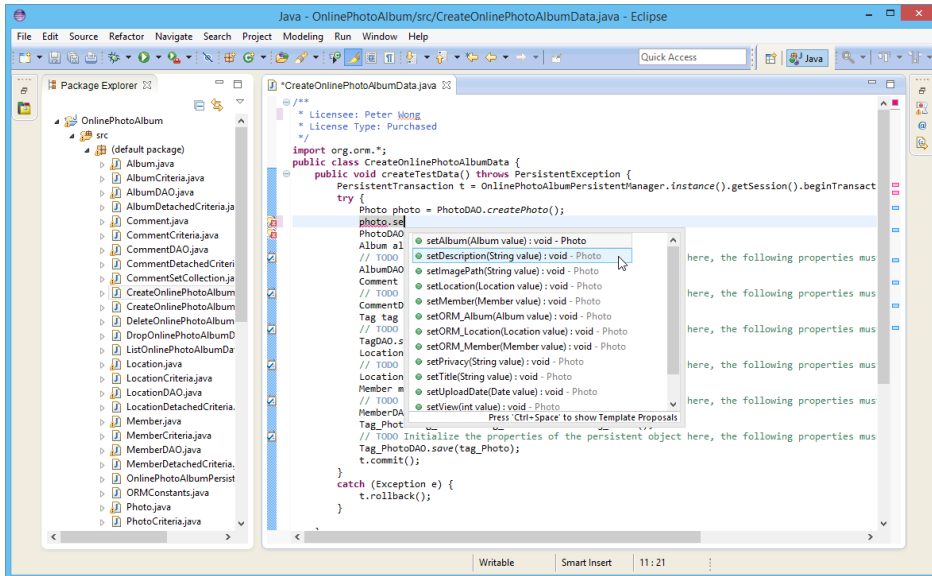
## Generating Code & DB

1. Under the **Tools** tab of Visual Paradigm, select **Hibernate > Generate Code...**
2. Fill in the output path for storing the generated file like the DDL file.
3. Configure the generation options.
4. Open the **Database** tab.
5. Click on **Database Options**.
6. Select the DBMS you use.
7. Configure the database connection.
8. Click **OK**.
9. Click **OK** to generate.



Hibernate

## Generating Code & DB



Hibernate

## Summary

- Introduction to Hibernate
- Mapping ERD and UML Class Diagram
  - Basic idea of mapping
  - How relationships are being handled
  - Different kinds of inheritance strategies
- Generating database and code

Hibernate