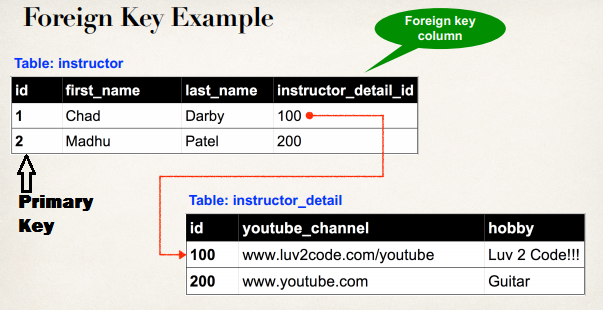
**22.2. Important Database Concepts**

**Primary key**:

A table typically has a column or combination of columns that contain values that uniquely identify each row in the table. This column, or columns, is called the primary key (PK) of the table and enforces the entity integrity of the table. Because primary key constraints guarantee unique data, they are frequently defined on an identity column.

**Foreign Key**:

A foreign key (FK) is a column or combination of columns that is used to establish and enforce a link between the data in two tables to control the data that can be stored in the foreign key table. In a foreign key reference, a link is created between two tables when the column or columns that hold the primary key value for one table are referenced by the column or columns in another table. This column becomes a foreign key in the second table.



**Cascading in hibernate**:

Main concept of hibernate relations is to getting the relation between parent and child class objects.

Cascade attribute is mandatory, whenever we apply relationship between objects, cascade attribute transfers operations done on one object onto its related child objects.

If we write **cascade = “all”** then changes at parent class object will be effected to child class object too, if we write **cascade = “all”** then all operations like **insert, delete, update** at parent object will be effected to child object also

**Example**:

if we apply insert (or update or delete) operation on parent class object, then child class objects will also be stored into the database.

Default value of cascade =”none” means no operations will be transfers to the child class

**Example**:

if we apply insert(or update or delete) operation on parent class object, then child class objects will not be effected, if cascade = “none”

**Cascade having the values**:

1. none (default)
2. save
3. update
4. save-update
5. delete
6. all
7. all-delete-orphan

In hibernate relations, if we load one parent object from the database then child objects related to that parent object will be loaded into one collection right (see one-to-many insert example).

Now if we delete one child object from that collection, then the relationship between the parent object and that child object will be removed, but the record (object) in the database will remains at it is, so if we load the same parent object again then this deleted child will not be loaded [ but it will be available on the database ]

so finally what am saying is all-delete-orphan means, breaking relation between objects not deleting the objects from the database, hope you got what am saying 😉

Note:

what is orphan record ..?

an orphan record means it is a record in child table but it doesn’t have association with its parent in the application.

[ And ]

In an application, if a child record is removed from the collection and if we want to remove that child record immediately from the database, then we need to set the cascade =”all-delete-orphan”

And that’s it about this cascade attribute in hibernate, hope i explained all the values..!!

22.2. Important Database Concepts