**Primary Key Auto generation by Hibernate**

**Primary key is of two types**

## Surrogate key Natural key

## Surrogate key

A surrogate key is a system generated (could be GUID, sequence, etc.) value with no business meaning that is used to uniquely identify a record in a table.

Example: Serial number is the only example.

## Natural key

A natural key is a column that already exist in the table (e.g. they are attributes of the entity within the data model) and uniquely identify a record in the table.  Since these columns are attributes of the entity they obviously have business meaning.

Example: User ID, Email Address can be the example.

## If V want to generate the surrogate key by our self, then V look into DB what is the last generated sequence of surrogate key then v have to use the next value. This is the head ache for us, so V are telling the HB to create the surrogate key for us.

Marking a field with the [@GeneratedValue](https://www.objectdb.com/api/java/jpa/GeneratedValue) annotation specifies that a value will be automatically generated for that field. This is primarily intended for primary key fields but ObjectDB also supports this annotation for non-key numeric persistent fields as well. Several different value generation strategies can be used as explained below.

* [The Auto Strategy](https://www.objectdb.com/java/jpa/entity/generated#The_Auto_Strategy_)
* [The Identity Strategy](https://www.objectdb.com/java/jpa/entity/generated#The_Identity_Strategy_)
* [The Sequence Strategy](https://www.objectdb.com/java/jpa/entity/generated#The_Sequence_Strategy_)
* [The Table Strategy](https://www.objectdb.com/java/jpa/entity/generated#The_Table_Strategy_)

@Id

@GeneratedValue

**private** Integer id;

* Annotating the primary key field with @GeneratedValue hibernate generate the surrogate key for us.
* @GeneratedValue is equal to the statement @GeneratedValue(strategy=GenerationType.***AUTO***), since If U are not providing any strategy then HB automatically consider it as GenerationType.***AUTO***
* Annotating @GeneratedValue as empty (without specifying the strategy) internally uses GenerationType.***AUTO***
* This Strategy only supported in Oracle not in other DB server like MySQL, DB2, SQL Server, Sybase and Postgres.

@Id

@GeneratedValue(strategy=GenerationType.***IDENTITY***)

**private** Integer id;

* The [IDENTITY](https://www.objectdb.com/api/java/jpa/GenerationType/IDENTITY) strategy also generates an automatic value during the primary key generator. This also same as strategy type **AUTO**.
* The difference between and AUTO and IDENTITY is there is no such concept (IDENTITY) in Oracle. IDENTITY is supported by **MySQL, DB2, SQL Server, Sybase and Postgres**.

@Id

@GeneratedValue(generator="my\_seq")

@SequenceGenerator(name="my\_seq", sequenceName="MY\_SEQ\_CREATED\_in\_DB", allocationSize=1)

**private** Integer id;

* This strategy is used when U need custom sequence generated by USER, so we create a sequence in DB and call it in JAVA entity.
* SequenceName parameter takes the sequence name that U created in DB.
* Allocation size parameter takes the value incremented by how many number. Example If U provide allocationSize as **50** then the increment will be like 50, 100, 150 …

**Note:**

I am ignoring the strategy table, since it is rarely used and very slow.

Since we will maintain the sequence in DB table. While it generates the next upcoming sequence it fetches previous sequence that is stored in our sequence table.