1.Spring Data JPA:

Spring Data JPA is used to develop Persistence layer in the application.

=> Spring Data JPA providing ready made methods to perform CRUD operation in DB tables.

=> Data JPA providing ready made methods using interfaces like below

1) CrudRepository ( I )

2) JpaRepository ( I )

Note : JpaRepository = CrudRepository + Pagination Methods + Sorting Methods

Spring Data JPA Terminology:

\*Data JSA support Embedded DB Supports

\* JPA Support No SQL Db Support

Note: Data Source properties we can configure in "application.properties" or "application.yml" file

2) Entity Class : The class which is mapped with database table

@Entity

@Table

@Id

@Column

3) Repository interface : For every Table we will create one repository interface to perform Crud Operations

public interface StudentRepository extends CrudRepository<Student, Integer>{

}

1) save ( Entity )

2) saveAll (Iterable<Entity> i )

Note: Above two methods are called as "UPSERT" methods ( UPDATE + INSERT )

3) findById (ID id)

4) findAllById (Iterable<ID> ids)

5) findAll ( )

6) count ( )

7) existById (ID id)

8) deleteById (ID id)

9) deleteAllById (Iterable<ID> ids)

10) deleteAll ( )

1) auto\_ddl : Dynamic Schema Generation

2) show\_sql : Display generated queries on the console

=> By using findByXXX ( ) methods we can retrieve the data based on non - primary key columns also

=> When we write findByXXX method , JPA will construct query based on method name

Note: Method Naming convention is very important for findByXXX methods

=> Using findBy methods we can perform select operations only (retrieval). INSERT / UPDATE / DELETE operations we can't do using findBy methods.

Note : In findBy method syntax we will use entity variable names.

Custom Queries:

=> We can execute Custom Queries also in JPA (our own queries)

=> To execute custom queries we will use @Query annotation

=> @Query will support for executing both HQL queries & Native SQL queries also.

HQL : Hibernate Query Language ( Database Independent Queries )

=> In HQL, we will use Entity class name & Entity class variables to write query

=> HQL queries will converted to SQL queries by Dialect class for execution

=> If we change app from one DB to another DB then no need to change any query

because Dialect class will take care of query conversion

=> HQL queries will give poor performance because of conversion ( HQL -> SQL )

### SQL : Structured Query Language ( Database Dependent Queries

=> In SQL, we will use table name & column names to write the query

=> SQL queries will directley execute in database

=> If we change app from one DB to another DB then all queries may not execute

=> SQL queries will give better performance than HQL