1)What is ORM?

* ORM stands for Object Relational Mapping.
* It is a technique used in creating a “bridge” between object-oriented programs and relational database.
* It is mainly for Database Operations
* Every Operation need to be in object format(save, update, delete and select)

2) What is JPA ?

* JPA stands for Java Persistance API
* It is one of ORM specification
* It is Java specification used to persist data between the relational database and java objects.
* It acts as a bridge between java class and relational databases.

3) Explain mapping rule in JPA?

* We define entity (POJO) class for one table
* We define one variable for one column
* Then one object is converted into one Row(even reverse)

4) What is the difference between JPA and Hibernate?

* JPA is a specification given by sun microsystem, now part of Oracle.
* Hibernate is a JPA implementation
* Other JPA implementations are :

EclipseLink, Apache Open JPA, TopLink etc..

5)What are some advantages of using JPA?

* Database independent (Dialect)
* Easy to Develop Database operations
* JPQL/HQL support
* Type(Datatypes) and parameter Handling

6)What is the spring data (or) spring boot data jpa repository?

1)CrudRepository

2)PagingAndSortingRepository

3)JpaRepository

Here all these are interfaces,

It is implementation class(proxy class) is generated by spring boot at run time.

7)What is the difference between PagingAndSortiing and JpaRepository?

* JpaRepository works only for relational databases like MySQL, Oracle, Postgres..etc
* PagingAndSoritngRepository and CrudRepository work ford both SQL and NOSQL databases.

8)What is @Query used for?

* @Query annotation is used to write custom query or user/programmer defined query
* By using this query we can write JPQL/HQL query which is written by the class and variable name rather than DB Table name and Column name.
* Here JPQL is the specification and HQL is the implementation.

9)What type of queries can be implemented using @Query annotation

* By using @Query annotation, we can write both select and non-select operations.
* By using @Query, we can write both JPQL and native SQL queries.
* But when we are using @Query with update or delete, we should also specify @Modifying annotation.

10)Give an example of using @Query annotation with JPQL?

* Based on the employee’s name fetch the data from the database
* @Query (“SELECT e FROM Employee e WHERE e.name =?1”)

List<Employee> getEmployeeBasedOnName (String name);

* Must define inside our Repository interface
* Must be abstract method
* Must check JPQL/HQL Syntax

11)What are Collection Mappings supported by JPA?

* JPA supports collections like List, Set and Map.
* We need to add @ElementCollection over variable
* It creates one child table with 2/3 columns

Key columns: Foreign Key Column

Index Column: Position of value in collection

Element Column: Actual Data Of Collection

* For Set collection, table is created with 2 columns (Key, Element)
* For List/Map Collections, table is created with 3 columns

12)What is Platform Transaction Manager?

* PlatformTransactionManager is an interface that extends Transaction Manager.
* It is the central interface in Spring’s transaction infrastructure.
* It enables @Transactional annotation which does commit in case of success and Rollback in case of Exception/Failed

13)How can we enable Spring Data JPA?

* Spring Boot comes with Auto Configuration, if add spring Data JPA dependency in pom.xml

Spring-boot-starter-data-jpa

* We must write some configuration in application. Properties, like Database connection details and JPA details.

Spring.datasource. driver-class-name =

Spring.datasource.url=

Spring.datasource. Username=

Spring.datasource. Password=

14)Difference between findById () and getOne ()?

* findById () method present in CrudRepository whereas getOne() method present in JPaRepository.
* If data is not found in findById () method then it will return null, means it’s return type is Optional<T> class.
* If data is not found in getOne () method, it will throw an exception called EntityNotFoundException.

15) What is the use of Dialect and Give some Examples?

* Dialect is a class defined in JPA
* It generates SQLs at runtime based on our operations (method class)
* Like on calling save (), generates INSERT…SQL.
* If we move from one Database to another Database Dialect property need to be modified in properties file.
* Ex: Oracle10gDialect, MYSQL8Dialect, PostgressDialect, etc.

16)What is FetchTYpe in JPA?

* It is used for SELECT operations used for Association Mappings or Collection Mappings.
* Fetch Types are: EAGER and LAZY
* EAGER: Load parent Entity along with its associate child entities
* From DB table on select operations
* LAZY: Load parent Entity from DB table on select operations. Later, calling child get () method, loads child entities…

17)What is the default FetchType in JPA ?

* There is specific default value for Fetch Type. It means depends on operation we do
* LAZY for: one-to-many and many-to-many also for ElementCollection
* EAGER for: one-to-one and Many-to-one.

18)How can we see generated SQLs at console/log files?

* Data JPA by default generates SQL and it will not be shown.
* To view them add property show-sql =true in properties file
* Spring.jpa.show-sql = true (default is false)

19)What are different types of joins supported by JPA?

* INNER JOIN: Gets commonly connected rows from both tables.
* OUTER JOIN

1)LEFT OUTER JOIN |LEFT JOIN

Gets rows from Left side table and connected rows form right side table

2)RIGHT OUTER JOIN | RIGHT JOIN

Gets all rows from Right side table and connected rows from left side table

3)FULL OUTER JOIN | FULLJOIN

Gets all rows (both connected and non- connected) from both side tables