1. **Difference between SQL and PL/SQL**

|  |  |
| --- | --- |
| SQL | PL/SQL |
| Simple query language | Procedural Language along with Simple Query Language Features |
| Suitable when we execute single update / insert / and delete operations on database | Suitable for executing the multiple inserts / updates / deletes on database |
|  |  |
| It doesn’t not support programming | It supports programming |
| Loop:  It doesn’t loop through the records | Loop:  It can loop through the records and manipulate the single record at a time |
| Exceptions:  Exceptions cannot be handled | Exceptions:  Exceptions can be handled |
| Code Secure:  It doesn’t stored in database and cannot be encrypted | Code Secure:  It can be stored in database and can be encrypted |
| Complex Data Types:  Cannot work with complex data types such as records, objects, collections,cursors,arrays | Complex Data Types:  It can support the complex data types such as records,objects,arrays,collections,cursors |
| It specifies what to do | It specifies what to do and how to do |
| It is used to code the queries, DML and DDL statements | It is used to code the blocks of code ,triggers, stored procedures, functions and packages |
| It can be embedded into Pl/SQL | It cannot be embedded into the SQL |

1. **What is SQLInjection and how to handle it?**

It is a code injection technique, used to attack the data driven applications in which malicious SQL statements are inserted into an entry field for execution. It is an technique which malicious users can inject the SQL commands into the SQL statement via web page input in the web page request. SQL injection is one of the top web page vulnerabilities and it affects the web application security.

**Considers this server code,**

txtUsrId=request.getPameter(“userId”)

txtSqlQuery=”select \* from User where userId=”+txtUsrId

**Scenerio’s when SQLIjection happens for the above server code:**

1. Based on 1=1 is always true

When we input user id field as ,

User ID : 154 or 1=1

The query looks ,

Select \* from User where userId=150 or 1=1

1=1 always true in sql execution , so returns all user details from the table User (including username and password)

1. Based on “”=”” is true

Hackers can easily login to the application by simply inserting the “ or “”=”” in the user anme and password field.

User Id : “ or “”=””

Password : “ or””=””

It will create the query like this ,

Select \* from User where userId=”” or “”=”” and password=”” or “”=””

It will return all the user name and password from the user table

1. Based on Batch SQL statements

User Id : 105 ; Drop Table Suppliers;

It will create query like this,

Select \* from User where userId=105 ; Drop table Suppliers;

Since these are separated by semicolon , both statements are executed and Supplier table will be dropped.

1. **How to handle the above SQL injections in java?**

In java , We have to use parameterized queries and can be done by using the PreparedStatement to execute the query as like,

PreperedStatement ps=con.prepareStatement(“select \* from user where userID=?”);

Ps.setString (1, txtUsrId);

Prepare statement setter methods always truncates the special characters in the input

Use callable statement to execute the stored procedures and functions. This statement itself is not vulnerable to SQLInjection.

1. **How to handle the SQL injections in ORM frameworks such as hibernate ?**

We have to use NamedQuery instead of using the normal query.

**Ex:**

Query namedQuery=session.createNamedQuery(“queryName”);

namedQuery.setString(1,txtUsrId);

List list=namedQuery.list();

1. **What is join ? How many types are available ?**

Join is an SQL technique used to retrieve records from more than one table in efficient manner.

There are four types of joins are available,

1. Inner Join :

Retrieve records from two table based on the conditions met.

Select c.firstName,c.lastName ,o.orderDate,o.orderAmount from Customer c inner join orders o on c.customer\_id=o.cust\_id

1. Left outer join:

Retrieve all records(rows) from left side table and only matched records from right side table based on the conditions specified. The left side table columns for the right side table records are filled with NULL.

Select c.firstName,c.lastName ,o.orderDate,o.orderAmount from Customer c left join orders o on c.customer\_id=o.cust\_id

1. Right outer join:

Retrieve all records from right side table and only matching records from left side table based on certain condition. The right side column for the left side table record are filled with NULL.

Select c.firstName,c.lastName ,o.orderDate,o.orderAmount from Customer c right join orders o on c.customer\_id=o.cust\_id

1. **What are the Database locks are available?**