Dt : 9/1/2025(Day-10)

**imp*

2.PreparedStatement:

=>'PreparedStatement' is an interface from java.sql package and which is used to execute normal queries with IN-Parameters.

=>we use prepareStatement()-method from Connection-Interface is used to create implementation object for 'PreparedStatement' Interface,because the prepareStatement()-method internally holding 'Anonymous Local InnerClass as implementation class of PreparedStatement Interface'.

Method Signature of prepareStatement():

```
public abstract java.sql.PreparedStatement prepareStatement(java.lang.String)
throws java.sql.SQLException;
```

syntax:

```
PreparedStatement ps = con.prepareStatement("query-structure");
```

=>The following are two important methods of PreparedStatement:

(i)executeQuery()

(ii)executeUpdate()

(i)executeQuery():

=>executeQuery()-method is used to execute select-queries.

Method Signature:

```
public abstract java.sql.ResultSet executeQuery() throws java.sql.SQLException;
```

syntax:

ResultSet rs = ps.executeQuery();

(ii)executeUpdate():

=>executeUpdate()-method is used to execute NonSelect-queries.

Method Signature:

public abstract int executeUpdate() throws java.sql.SQLException;

syntax:

int k = ps.executeUpdate();

Assignment-2:(Solution with PreparedStatement)

Construct JDBC Application to perform the following operations on Product table

1.AddProduct

2.ViewAllProducts

3.ViewProductByCode

4.UpdateProductByCode(price-qty)

5.DeleteProductByCode

Construct DB table with name Product70

(code,name,price,qty)

Primary Key : code

create table Product70(code varchar2(10),name varchar2(15),price number(10,2),

qty number(10),primary key(code));

Dt : 10/1/2025

Program : DBCon4.java

```
package test;
import java.util.*;
import java.sql.*;
public class DBCon4
{
        public static void main(String[] args)
        {
    Scanner s = new Scanner(System.in);
    try(s;){
        Class.forName("oracle.jdbc.driver.OracleDriver");
        Connection con = DriverManager.getConnection
                        ("jdbc:oracle:thin:@localhost:1521:xe","system","tiger");
        PreparedStatement ps1 = con.prepareStatement
                        ("insert into Product70 values(?,?,?,?)");//Compilation Completed
        PreparedStatement ps2 = con.prepareStatement
                        ("select * from Product70");//Compilation Completed
```

```
PreparedStatement ps3 = con.prepareStatement
                ("select * from Product70 where code=?");//Compilation Completed
PreparedStatement ps4 = con.prepareStatement
                ("update Product70 set price=?,qty=qty+? where code=?");
                 ////Compilation Completed
PreparedStatement ps5 = con.prepareStatement
                ("delete from Product70 where code=?");
while(true) {
        System.out.println("*******Operations Choice******");
        System.out.println("\t1.AddProduct"
                        + "\n\t2.ViewAllProducts"
                        + "\n\t3.ViewProductByCode"
                        + "\n\t4.UpdateProductByCode(Price-Qty)"
                        + "\n\t5.DeleteProductByCode"
                        + "\n\t6.Exit");
        System.out.println("Enter your Choice:");
        int choice = Integer.parseInt(s.nextLine());
        switch(choice)
        {
        case 1:
                 //data loaded to Local variables of main()-method
                System.out.println("======Product Details=====");
                System.out.println("Enter the Prod-Code:");
                String code = s.nextLine();
                System.out.println("Enter the Prod-Name:");
```

```
String name = s.nextLine();
System.out.println("Enter the Prod-Price:");
float price  = Float.parseFloat(s.nextLine());
System.out.println("Enter the Prod-Qty:");
int qty = Integer.parseInt(s.nextLine());

 //Loading data to PreparedStatement Object from Local variables
ps1.setString(1, code);
ps1.setString(2, name);
ps1.setFloat(3, price);
ps1.setInt(4, qty);

int k = ps1.executeUpdate();
if(k>0) {
        System.out.println("Product added Successfully....");
}
break;
case 2:
ResultSet rs1 = ps2.executeQuery();
System.out.println("------Product Details-----");
while(rs1.next())
{
        System.out.println(rs1.getString(1)+"\t"
                        +rs1.getString(2)+"\t"
                        +rs1.getFloat(3)+"\t"
```

```
                    +rs1.getInt(4));
            }//end of loop
            break;
case 3:
            System.out.println("Enter the Prod-Code to retrieve details:");
            String pc1 = s.nextLine();
            ps3.setString(1, pc1);
            ResultSet rs2 = ps3.executeQuery();
            if(rs2.next()) {
                    System.out.println(rs2.getString(1)+"\t"
                            +rs2.getString(2)+"\t"
                            +rs2.getFloat(3)+"\t"
                            +rs2.getInt(4));
            }else {
                    System.out.println("Invalid Product code....");
            }
            break;
case 4:
            System.out.println("Enter the Prod-Code to update Product details:");
            String pc2 = s.nextLine();
            ps3.setString(1, pc2);
            ResultSet rs3 = ps3.executeQuery();
            if(rs3.next()) {
                    System.out.println("Existing product price:"+rs3.getFloat(3));
                    System.out.println("Enter the new price:");
```

```
                float nPrice = Float.parseFloat(s.nextLine());
                System.out.println("Existing product qty:"+rs3.getInt(4));
                System.out.println("Enter the new qty:");
                int nQty = Integer.parseInt(s.nextLine());
                ps4.setFloat(1,nPrice);
                ps4.setInt(2, nQty);
                ps4.setString(3, pc2);
                int k1 = ps4.executeUpdate();
                if(k1>0) {
                        System.out.println("Product Updated Successfully....");
                }
        }else {
                System.out.println("Invalid Prod-Code....");
        }
        break;
case 5:
        System.out.println("Enter the Prod-Code to delete product details:");
        String pc3 = s.nextLine();
        ps3.setString(1, pc3);
        ResultSet rs4 = ps3.executeQuery();
        if(rs4.next()) {
                ps5.setString(1, pc3);
                int k3 = ps5.executeUpdate();
                if(k3>0) {
                        System.out.println("Product details deleted Successfully..");
```

```
                    }
                }else {
                    System.out.println("Invalid Prod-Code...");
                }
                break;
            case 6:
                System.out.println("Operations Stopped...");
                System.exit(0);
            default:
                System.out.println("Invalid Choice..");
            }//end of switch
        }//end of loop
    }catch(Exception e) {
        e.printStackTrace();
    }
        }
}
```

o/p:

```
*******Operations Choice******
        1.AddProduct
        2.ViewAllProducts
        3.ViewProductByCode
        4.UpdateProductByCode(Price-Qty)
        5.DeleteProductByCode
        6.Exit
```

Enter your Choice:

2

------Product Details-----

A21 Mous 1200.0 12

C12 CDR 700.0 10

********Operations Choice*******

1.AddProduct

2.ViewAllProducts

3.ViewProductByCode

4.UpdateProductByCode(Price-Qty)

5.DeleteProductByCode

6.Exit

Enter your Choice:

5

Enter the Prod-Code to delete product details:

C12

Product details deleted Successfully..

********Operations Choice*******

1.AddProduct

2.ViewAllProducts

3.ViewProductByCode

4.UpdateProductByCode(Price-Qty)

5.DeleteProductByCode

6.Exit

Enter your Choice:

2

------Product Details-----

A21 Mous 1200.0 12

********Operations Choice*******

1.AddProduct

2.ViewAllProducts

3.ViewProductByCode

4.UpdateProductByCode(Price-Qty)

5.DeleteProductByCode

6.Exit

Enter your Choice:

6

Operations Stopped...

Assignment:

step-1 : Create table with name Student70

(rollNo,name,branch,mid,phno,totMarks,per,Result)

step-2 : Construct JDBC Application to perform the following Operations

1.AddStudent

2.ViewAllStudents

3.ViewStudentByRollNo

4.UpdateStudentByRollNo(mid-phno)

5.DeleteStudentByRollNo

Calculations:

=>totMarks based on 6 Subject Marks.

(Exception Condition : The marks must be in b/w 0 to 100,else 'Exception')

==========

Dt : 11/1/2025

**imp*

'ResultSet' in JDBC:

=>'ResultSet' is an interface from java.sql package and which is instatiated to hold the data retrieved using select-queries.

(ResultSet-Objects will hold data retrived using select-queries)

=>ResultSet-Objects are categorized into two types:

1.NonScrollable ResultSet Objects

2.Scrollable ResultSet Objects

1.NonScrollable ResultSet Objects:

=>The ResultSet-Objects in which,the cursor can be moved only in one direction are known as NonScrollable ResultSet Objects.

(In NonScrollable ResultSet Objects,the cursor can be moved from top-of-table data to bottom-of-table data)

=>We use the following syntaxes to create NonScrollable ResultSet Objects:

syntax-1 : Using 'Statement'

```
Statement stm = con.createStatement();
ResultSet rs = stm.executeQuery("select-query");
```

syntax-2 : Using 'PreparedStatement'

```
PreparedStatement ps = con.prepareStatement("select-query-structure");
ResultSet rs = ps.executeQuery();
```

2.Scrollable ResultSet Objects:

=>The ResultSet-Objects in which,the cursor can be moved in both directions (forward-backward) are known as Scrollable ResultSet Objects.

=>we use the following syntaxes to create Scrollable ResultSet Objects:

syntax-1 : Using 'Statement'

```
Statement stm = con.createStatement(type,mode);
ResultSet rs = stm.executeQuery("select-query");
```

syntax-2 : Using 'PreparedStatement'

```
PreparedStatement ps = con.prepareStatement("select-query-structure",type,mode);
ResultSet rs = ps.executeQuery();
```

define 'type'?

=>'type' specifies the direction of cursor on ResultSet-Objects.

=>The following fields from 'ResultSet' represent "type":

```
public static final int TYPE_FORWARD_ONLY;
public static final int TYPE_SCROLL_INSENSITIVE;
public static final int TYPE_SCROLL_SENSITIVE;
```

define 'mode'?

=>'mode' specifies the action performed on ResultSet-Objects.

=>The following fields from 'ResultSet' represent "mode":

public static final int CONCUR_READ_ONLY;

public static final int CONCUR_UPDATABLE;

=>we use the following some important methods to control cursor on ResultSet-Objects:

1.afterLast()

2.beforeFirst()

3.previous()

4.next()

5.first()

6.last()

7.absolute(int)

8.relative(int)

1.afterLast():

=>afterLast()-method will move the cursor pointing after the last-row.

2.beforeFirst():

=>beforeFirst()-method will move the cursor pointing before the first-row

3.previous():

=>previous()-method will move the cursor in backward direction.

4.next():

=>next()-method will move the cursor in forward direction.

5.first():

=>first()-method will make the cursor point to the first-row

6.last():

=>last()-method will make the cursor point to the last-row.

7.absolute(int):

=>absolute(row_no) method will make the cursor point to the specified row_no.

8.relative(int):

=>relative(int) method will move the cursor forward or backward from the current position.
