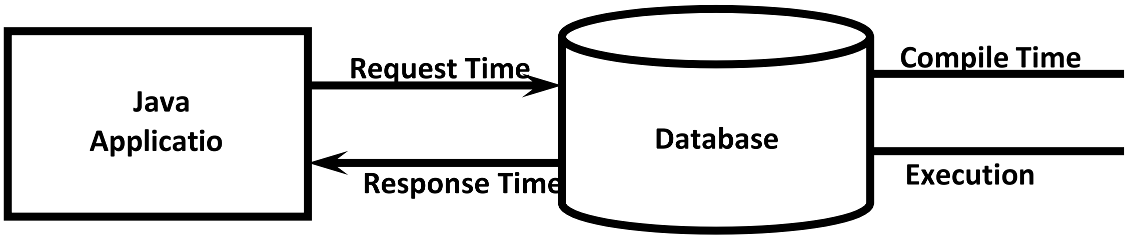
PreparedStatement (I)

Need of PreparedStatement:

In the case of normal Statement, whenever we are executing SQL Query, every time compilation and execution will be happened at database side.

```
Statement st = con.createStatement();
ResultSet rs = st.executeQuery ("select * from employees");
```



Total Time per Query = Req.T+C.T+E.T+Resp.T
= 1 ms + 1 ms + 1 ms + 1 ms = 4ms

per 1000 Queries = 4 * 1000ms = 4000ms

Sometimes in our application,we required to execute same query multiple times with same or different input values.

Eg1:

In IRCTC application,it is common requirement to list out all possible trains between 2 places

```
select * from trains where source='XXX' and destination='YYY';
```

Query is same but source and destination places may be different. This query is required to execute lakhs of times per day.

Eg2:

In BookMyShow application, it is very common requirement to display theatre names where a particular movie running/playing in a particular city

```
select * from theatres where city='XXX' and movie='YYY';
```

In this case this query is required to execute lakhs of times per day. May be with different movie names and different locations.

For the above requirements if we use Statement object, then the query is required to compile and execute every time, which creates performance problems.

To overcome this problem, we should go for PreparedStatement.

The main advantage of PreparedStatement is the query will be compiled only once even though we are executing multiple times, so that overall performance of the application will be improved.

We can create PreparedStatement by using prepareStatement() method of Connection interface.

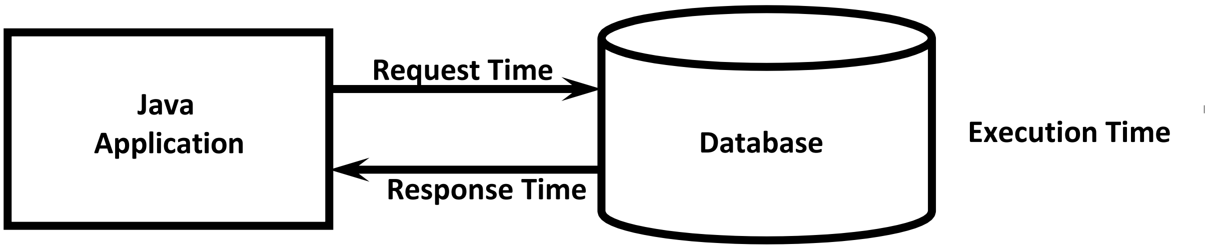
public PreparedStatement prepareStatement(String sqlQuery) throws SQLException

Eg: PreparedStatment pst=con.prepareStatement(sqlQuery);

At this line,sqlQuery will send to the database. Database engine will compile that query and stores in the database.
That pre compiled query will be returned to the java application in the form of PreparedStatement object.

Hence PreparedStatement represents "pre compiled sql query".

Whenever we call execute methods,database engine won't compile query once again and it will directly execute that query,so that overall performance will be improved.



Total Time per Query = Req.T+Resp.T+E.T
= 1 ms + 1 ms + 1 ms = 3ms

1000 Queries = 3000ms

Steps to develop JDBC Application by using PreparedStatement

1. Prepare SQLQuery either with parameters or without parameters.

Eg: insert into employees values(100,'ramya',1000,'hyd');

insert into employees values(?, ?, ?, ?);

Positional Parameter OR Place Holder OR IN Parameter

2. Create PreparedStatement object with our sql query.

```
PreparedStatement pst = con.prepareStatement(sqlQuery);
```

At this line only query will be compiled.

3. If the query is parameterized query then we have to set input values to these parameters by using corresponding setter methods.

We have to consider these positional parameters from left to right and these are 1 index based. i.e index of first positional parameter is 1 but not zero.

```
pst.setInt(1,100);
pst.setString(2,"ramya")
;pst.setDouble(3,1000);
pst.setString(4,"Hyd");
```

Note:
Before executing the query, for every positional parameter we have to provide input values otherwise we will get SQLException

4. Execute SQL Query:

PreparedStatement is the child interface of Statement and hence all methods of Statement interface are bydefault available to the PreparedStatement.Hence we can use same methods to execute sql query.

```
executeQuery()
executeUpdate()
execute()
```

Note:

We can execute same parameterized query multiple times with different sets of input values. In this case query will be compiled only once and we can execute multiple times.

Q. Which of the following are valid sql statements?

1. delete from employees where ename=?
2. delete from employees ? ename=?
3. delete from ? where ename=?
4. delete ? employees where ename=?

Note:

We can use ? only in the place of input values and we cannot use in the place of sql keywords,table names and column names.

Static Query vs Dynamic Query:

The sql query without positional parameter(?) is called static query.

Eg: delete from employees where ename='ramya'

The sql query with positional parameter(?) is called dynamic query.

Eg: select * from employees where esal>?

Program-1 to Demonstrate PreparedStatement:

```
import java.sql.*;
public class PreparedStatementDemo1
{
    public static void main(String[] args) throws Exception
    {
        String driver="oracle.jdbc.OracleDriver";
        Class.forName(driver);
        String jdbc_url="jdbc:oracle:thin:@localhost:1521:XE";
        String user="scott";
        String pwd="tiger";
        Connection con = DriverManager.getConnection(jdbc_url,user,pwd);
        String sqlQuery ="delete from employees where ename=?";

        PreparedStatement pst = con.prepareStatement(sqlQuery);
        pst.setString(1,"Mallika");
        int updateCount=pst.executeUpdate();
        System.out.println("The number of rows deleted :"+updateCount);

        System.out.println("Reusing PreparedStatement to delete one more record...");
        pst.setString(1,"Durga");
        int updateCount1=pst.executeUpdate();
        System.out.println("The number of rows deleted :"+updateCount1);
        con.close();
    }
}
```

Program-2 to Demonstrate PreparedStatement:

```
import java.sql.*;
import java.util.*;
public class PreparedStatementDemo2
{
    public static void main(String[] args) throws Exception
    {
        String driver="oracle.jdbc.OracleDriver";
        String jdbc_url="jdbc:oracle:thin:@localhost:1521:XE";
        String user="scott";
    String pwd="tiger";
    Class.forName(driver);
    Connection con = DriverManager.getConnection(jdbc_url,user,pwd);
    String sqlQuery="insert into employees values(?,?,?,?)";
    PreparedStatement pst = con.prepareStatement(sqlQuery);

    Scanner sc = new Scanner(System.in);
    while(true)
    {
      System.out.println("Employee Number:");
      int eno=sc.nextInt();
      System.out.println("Employee Name:");
      String ename=sc.next();
      System.out.println("Employee Sal:");
      double esal=sc.nextDouble();
      System.out.println("Employee Address:");
      String eaddr=sc.next();
      pst.setInt(1,eno);
      pst.setString(2,ename);
      pst.setDouble(3,esal);
      pst.setString(4,eaddr);
      pst.executeUpdate();
      System.out.println("Record Inserted Successfully");
      System.out.println("Do U want to Insert one more record[Yes/No]:");
      String option = sc.next();
      if(option.equalsIgnoreCase("No"))
      {
        break;
      }
    }
    con.close();
  }
}
```

Advantages of PreparedStatement:

1. Performance will be improved when compared with simple Statement b'z query will be compiled only once.

2. Network traffic will be reduced between java application and database b'z we are not required to send query every time to the database.

3. We are not required to provide input values at the beginning and we can provide dynamically so that we can execute same query multiple times with different sets of values.

4. It allows to provide input values in java style and we are not required to convert into database specific format.

5. Best suitable to insert Date values

6. Best Sutitable to insert Large Objects (CLOB,BLOB)

7. It prevents SQL Injection Attack.

Limitation of PreparedStatement:

We can use PreparedStatement for only one sql query (Like CDMA Phone), but we can use simpleStatement to work with any number of queries (Like GSM Phone).

Eg:

```
Statement st =
con.createStatement();
st.executeUpdate("insert into ...");
st.executeUpdate("update
employees...");
st.executeUpdate("delete...");
```

Here We Are Using One Statement Object To Execute 3 Queries

PreparedStatement pst = con.prepareStatement("insert into

employees..");Here PreparedStatement object is associated with

only insert query.

Note: Simple Statement can be used only for static queries where as PreparedStatement canused for both static and dynamic queries.

Differences between Statement And PreparedStatement

Statement	PreparedStatement
1) At the time of creating Statement Object, we are not required to provide any Query. Statement st = con.createStatement(); Hence Statement Object is not associated with any Query and we can use for multiple Queries.	1) At the time of creating PreparedStatement, we have to provide SQL Query compulsory and will send to the Database and will be compiled. PS pst = con.prepareStatement(query); Hence PS is associated with only one Query.
2) Whenever we are using execute Method, every time Query will be compiled and executed.	2) Whenever we are using execute Method, Query won't be compiled just will be executed.
3) Statement Object can work only for Static Queries.	3) PS Object can work for both Static and Dynamic Queries.
4) Relatively Performance is Low.	4) Relatively Performance is High.
5) Best choice if we want to work with multiple Queries.	5) Best choice if we want to work with only one Query but required to execute multiple times.

6) There may be a chance of SQL Injection Attack.	**6) There is no chance of SQL Injection Attack.**
7) Inserting Date and Large Objects (CLOB and BLOB) is difficult.	**7) Inserting Date and Large Objects (CLOB and BLOB) is easy.**