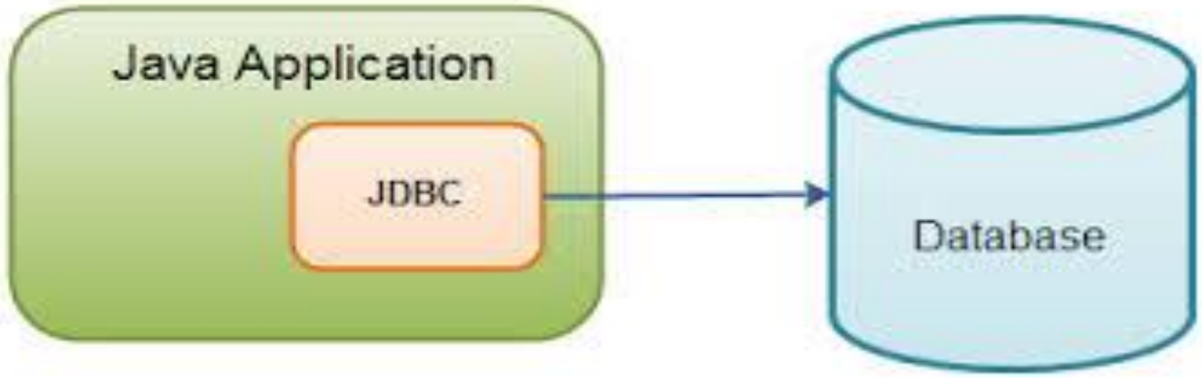
Object Oriented Programming

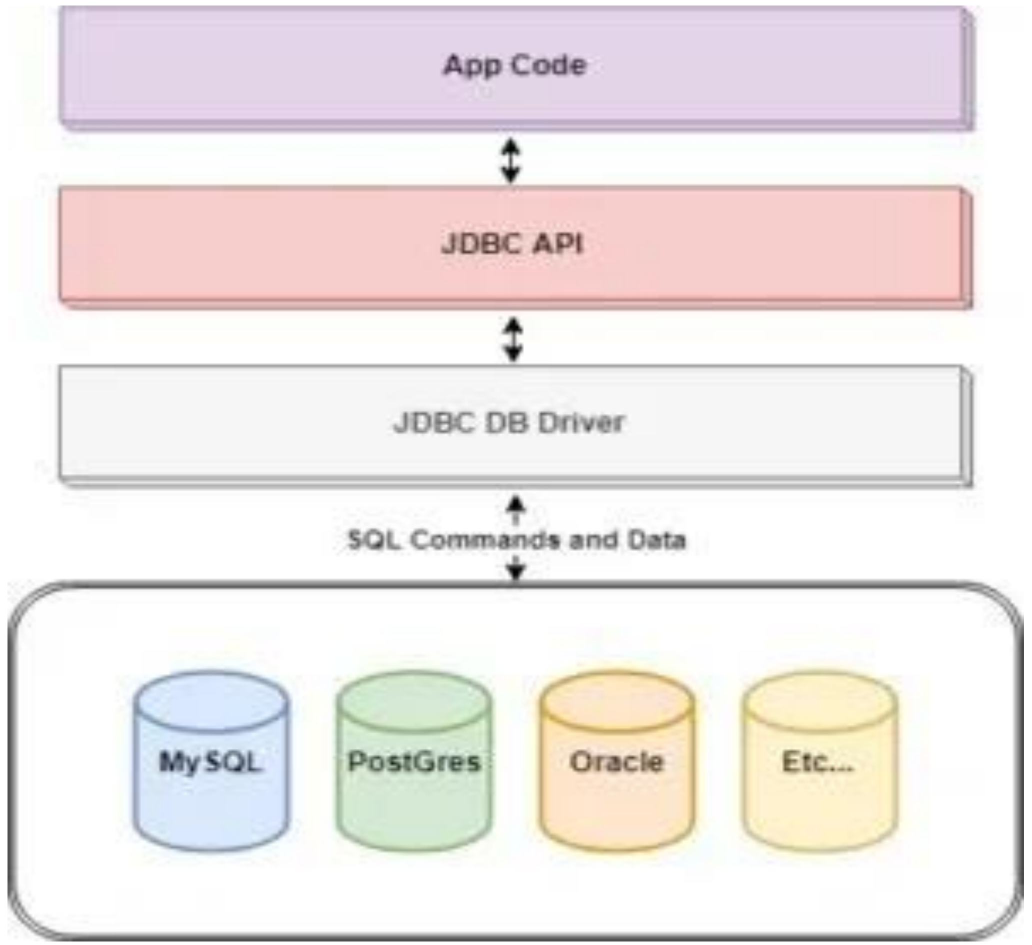
Database Connectivity

Sadaf Anwar
sanwar@numl.edu.pk
Department of Software Engineering

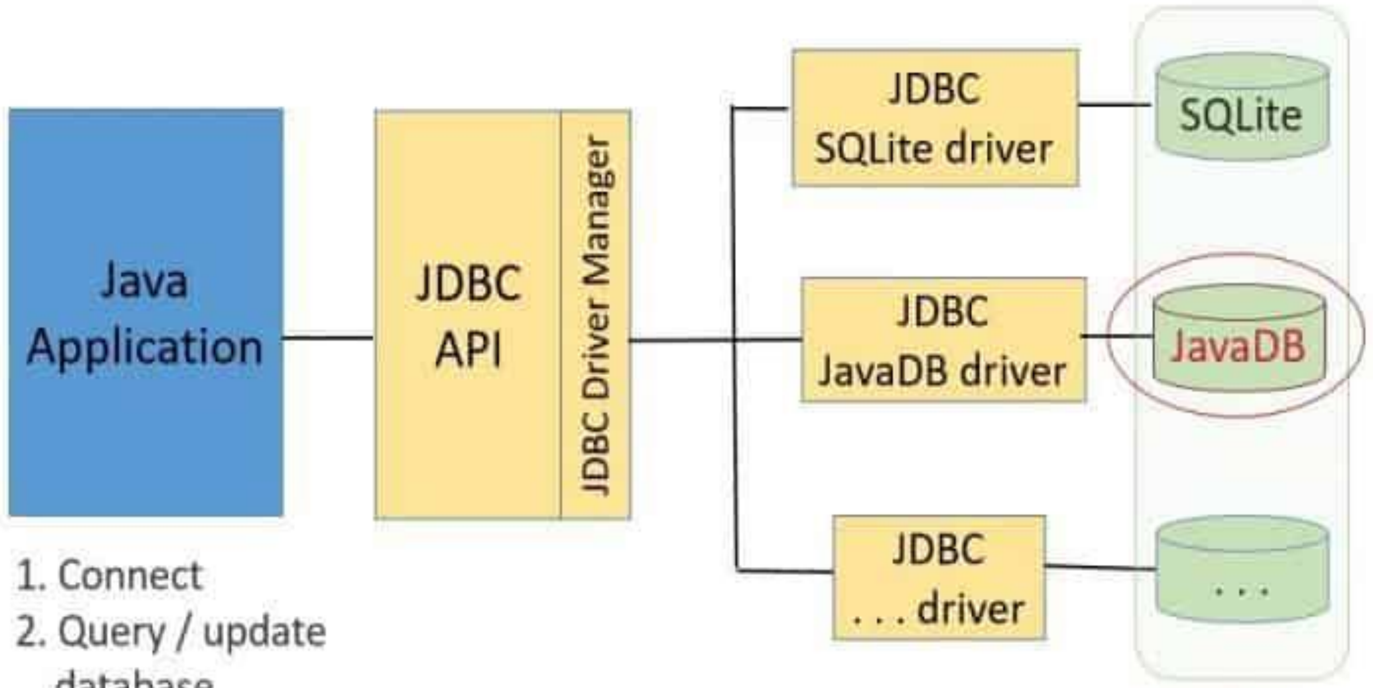
Introduction to JDBC

- **Java Database Connectivity(JDBC)** is an **Application Programming Interface(API)** used to connect Java application with Database.
- JDBC is used to interact with various type of Database such as Oracle, MS Access, My SQL and SQL Server.
- JDBC can also be defined as the platform-independent interface between a relational database and Java programming.
- It allows java program to execute SQL statement and retrieve result from database.


Java Application
JDBC
Database


App Code
JDBC API
JDBC DB Driver
SQL Commands and Data
MySQL
PostGres
Oracle
Etc...

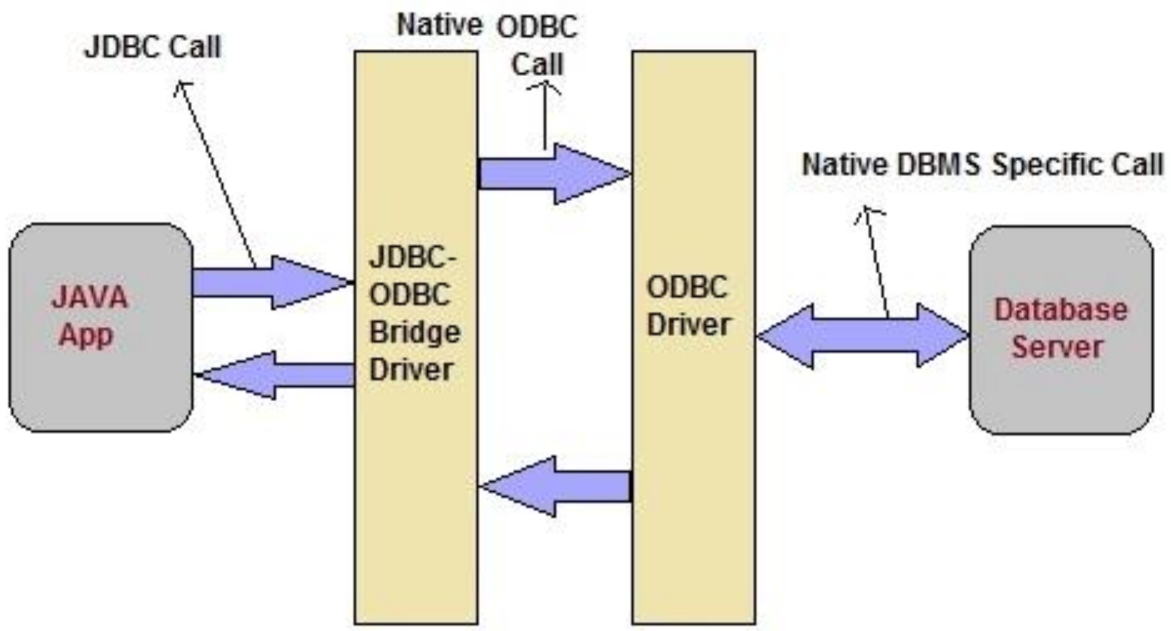
JDBC - Java Database Connectivity



1. Connect
2. Query / update database
3. Receive results

DBMS
Database Management Syste

Driver or JDBC-ODBC bridge



JDBC 4.0 API

JDBC 4.0 API is mainly divided into two package

- java.sql
- javax.sql

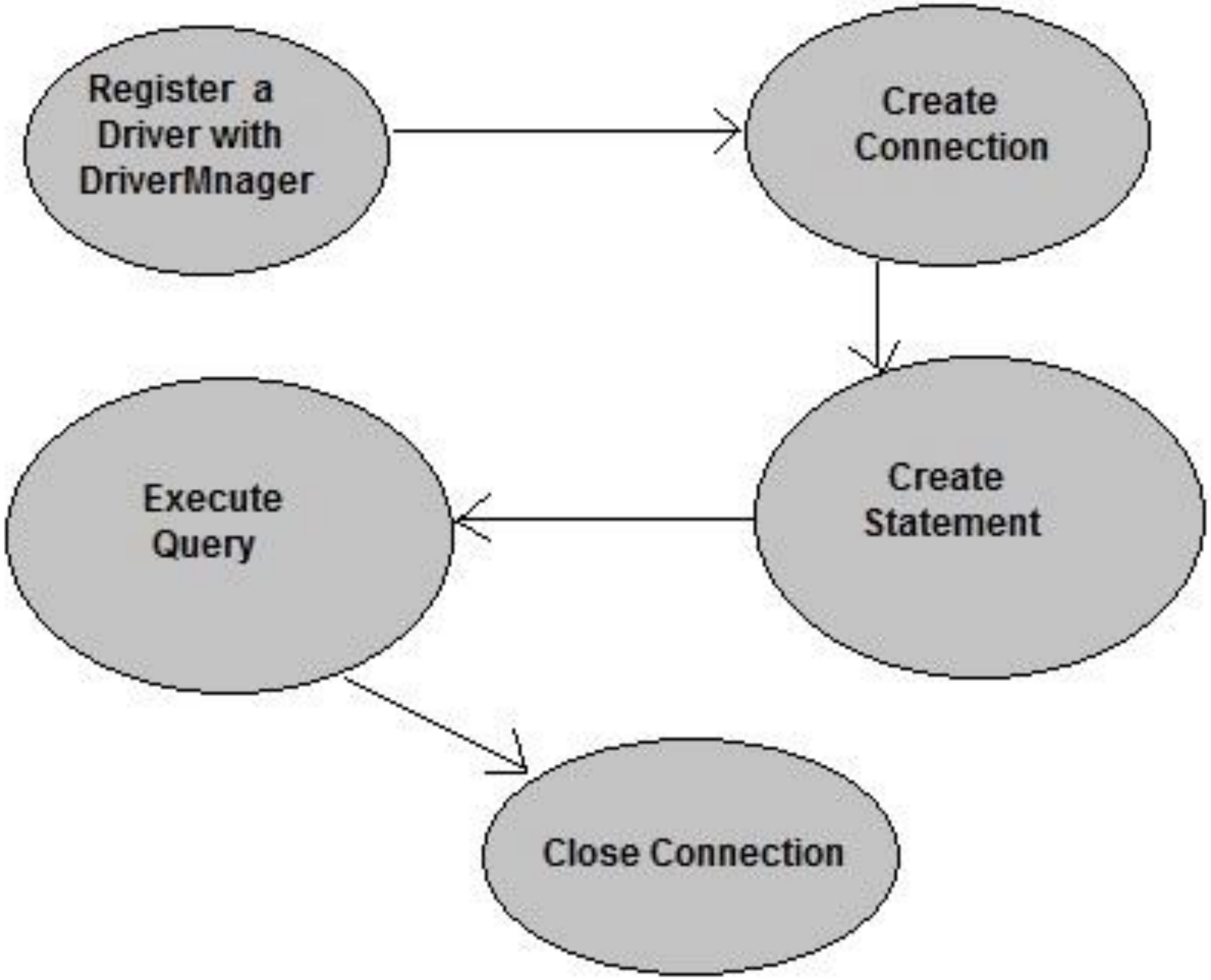
Important classes and interface of java.sql package

classes/interface	Description
java.sql.Connection	creates a connection with specific database
java.sql.Date	Provide support for Date SQL type.
java.sql.Driver	create an instance of a driver with the DriverManager.
java.sql.DriverManager	This class manages database drivers.
java.sql.PreparedStatement	Used to create and execute parameterized query.
java.sql.ResultSet	It is an interface that provide methods to access the result row-by-row.
java.sql.Savepoint	Specify savepoint in transaction.
java.sql.SQLException	Encapsulate all JDBC related exception.
java.sql.Statement	This interface is used to execute SQL statements.

Steps to Connect a Java Application to Database

The following 5 steps are the basic steps involve in connecting a Java application with Database using JDBC.

1. Register the Driver
2. Create a Connection
3. Create SQL Statement
4. Execute SQL Statement
5. Closing the connection


Register a Driver with DriverMnager
Create Connection
Create Statement
Execute Query
Close Connection

Register the Driver

- Class.forName() is used to load the driver class explicitly.
- Example to register with JDBC-ODBC Driver
- Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

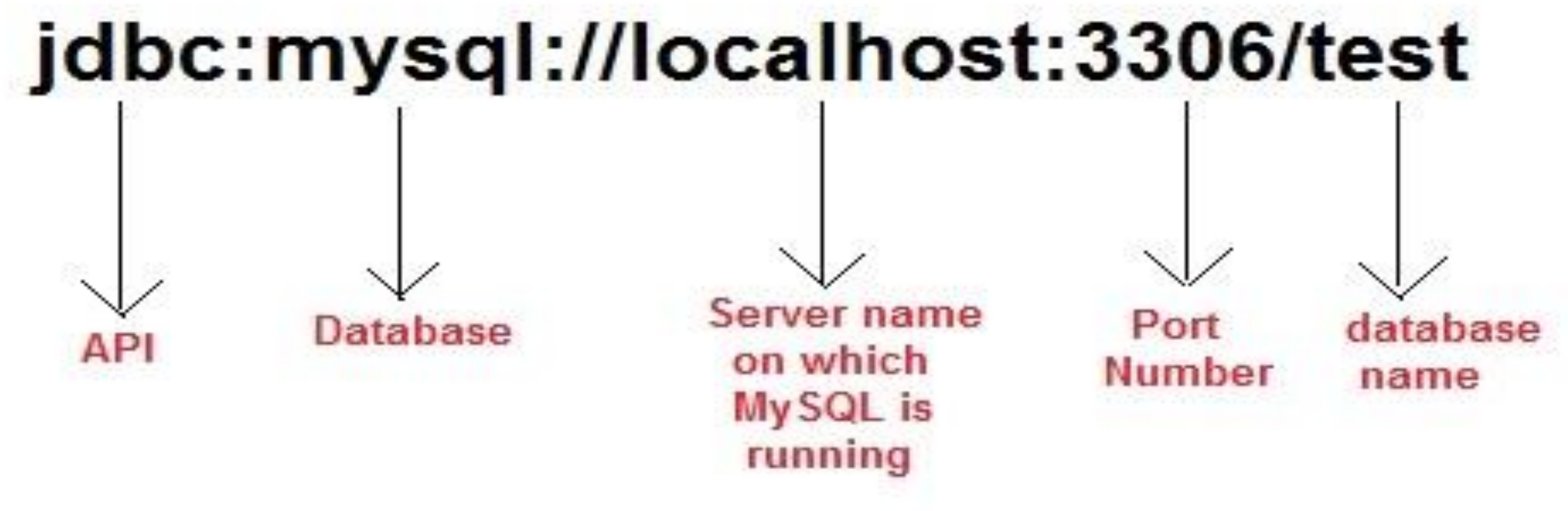
Create a Connection

- getConnection() method of **DriverManager** class is used to create a connection.
- **Syntax**

- getConnection(String url)
- getConnection(String url, String username, String password)
- getConnection(String url, Properties info)

Example establish Connection with Oracle Driver

Connection con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE","username","password");


jdbc:mysql://localhost:3306/test
API
Database
Server name on which MySQL is running
Port Number
database name

Create SQL Statement

- **createStatement()** method is invoked on current **Connection** object to create a SQL Statement

- Statement s=con.createStatement();

Execute SQL Statement

- executeQuery() method of **Statement** interface is used to execute SQL statements.

```
ResultSet rs=s.executeQuery("select * from user");
 while(rs.next())
 {
  System.out.println(rs.getString(1)+" "+rs.getString(2));
}
```

Closing the connection

- After executing SQL statement you need to close the connection and release the session. The close() method of Connection interface is used to close the connection.

- con.close();

Connecting to Access Database using Type-1 Driver

- To connect a Java application with Access database using JDBC-ODBC Bridge(type-1) Driver. You need to follow the following steps

```
import java.sql.*;
class Test
{
  public static void main(String []args)
  {
    try{
      Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
      Connection con = DriverManager.getConnection("jdbc:odbc:Test", "", "");
      Statement s=con.createStatement();      //creating statement

      ResultSet rs=s.executeQuery("select * from student");   //executing statement

      while(rs.next()){
        System.out.println(rs.getInt(1)+" "+rs.getString(2));
      }

      con.close();                    //closing connection

    }
    catch(Exception e)
    {
       e.printStackTrace();
    }
  }
}
```

Questions