Test 3

1) DB ? Why is it used ?

A DB or Database is used to store large amounts of data. It is used to increase the efficient storing of data as a database has a structured storing structure.
All entries into a database are sorted based on a certain attribute or column. This by default is done using an ID column, if it exists. Else the first column of the db is used to sort the data in an ascending order by default.

2) DBMS and RDMS ? Difference between them.

DBMS - Database Management System.
RDMS - Relational Data Management System.
RDBMS - Relational Database Management System.

Difference between DBMS and RDBMS :

- DBMS - uses files to store data
  RDBMS - uses tables to store data

- DBMS - data does not contain relationships between them
  RDBMS - data contains relationships between them

- DBMS - cannot handle large data
  RDBMS - can handle large amounts data

- DBMS - eg : file systems
  RDBMS - eg : SQL server

3) SQL ? Why is it used ?

SQL - Structured Query Language. It is an RDBMS.
SQL is used to create queries to :

- Access data stored in some database.
- Manipulate the data stored in a database, i.e., perform CRUD operations - Create, Read, Update, Delete.
- Create databases to perform these CRUD operations.

4) JDBC ? Steps ?

JDBC - Java Database Connectivity.

It is an interface that connects Java and SQL databases using an SQL connector.
JDBC performs SQL queries and then executes or calls these queries through a Java Application such as Eclipse, or any other Java based IDE.
It acts as a server that queries the SQL database and performs SQL operations.

Steps for JDBC :

- Load and register the driver.
    - Class.forName("com.mysql.jdbc.cj.Driver")
        - This may be used to call the Driver class
    - This must be in a try catch block to handle ClassNotFoundException.
    - The Driver class may be imported from SQL or can be a user-defined class.
- Establish a connection with database server
    - Connection con = DriverManager.getConnection(//url)
        - url - "jdbc:mysql://localhost:3306/db_name?user=user_name&password=user_defined_password.
    - This must be in a try catch block to handle SQLException incase a connection cannot be established due to some error.
- Create a statement
    - Statement statement = con.createStatement();
    - PreparedStatement ps = con.prepareStatement("sql query");
    - Either of the above may be used, however PreparedStatement is more commonly used and SQLException must be handled.
- Execute Query
    - The PreparedStatement SQL query must be executed :
        - executeUpdate() func called for INSERT, UPDATE or DELETE queries.
        - execute() func called for table manipulation - CREATE, ALTER, DROP queries.
        - executeQuery() func called for SELECT queries. This returns 1 or more reows from the table and must therefore be initialized to a ResultSet.
            - ResultSet rs = ps.executeQuery();
              while(rs.next()) { } //used to display results collected.
- Close connections
    - Costly resources such as the connection and PreparedStatement must be closed by calling the close() function.
        - ps.close() and con.close().

5) OOPS ? Fundamentals of OOPS w real-time egs.

OOPS - Object-Oriented Programming System.
OOPS uses the concept of creating objects where an object is an instance of a class.
The object can be used to access private or public instance members of a class.

Fundamentals of OOPS :

- Abstraction - Hiding of data so that a user or client sees or has access to only public information or data that is required by a user.
    - Eg : Parts of a vehicle used to manufacture different types of vehicles.
- Encapsulation - Wrapping of data so that it can be accessed only by certain methods of a class.
    - Eg : Packing clothes in a bag for a trip or vacation.
- Inheritance - A class gaining the properties or behaviors of another class which is its parent class so that a method of a child class can access both parent and child class instance members.
    - Eg : A child inheriting certain characteristics of their parents.
- Polymorphism - The concept of performing a certain method in different ways
    - Eg : A woman is a Mother, Teacher, Daughter, Cook all at the same time.

6) Diff between compile time Polymorphism and run time Polymorphism

Compile time Polymorphism - Polymorphism occurs during compile time, i.e., when a program is compiled and constructors or methods exist that have different method signatures but perform a similar yet different operation.

Eg : Method Overloading.

```
class Calculate {

public void calculate() {
        int a = 5;
        int b = 10;
        System.out.println(a+b);
}
public void calculate(int a) {
        int b = 4;
        System.out.println(a+b);
}
public void calculate(int a, int b) {
        System.out.println(a+b);
}
public static void main(String[] args) {
        Calculate c = new Calculate();
        Calculate c1 = new Calculate(3);
        Calculate c2 = new Calculate (3,7);
}
}
```

Runtime Polymorphism - Polymorphism occurs during runtime, i.e., more than one class can have a method with the same name and signature. The difference is that here usually a class that extends another class is able to do this. @Override notation must be used when trying to achieve this type of polymorphism.

Eg : Method Overriding.

```
class Display {
        public void display() {
                System.out.println(“Hello”);
}
}

class Display1 extends Display {
        @Override
        public void display() {
                System.out.println(“World”);
}
        public static void main(String[] args) {
                Display disp = new Display1();
                disp.display();
}
}
```

7) Constructor ? Types of Construction ? Constructor Chaining ?

A constructor is a method that has the same name as the class.
There are 2 types of constructors :

- Default constructor - This does not have any parameters and is called by default when an object is created. In the above example for class Calculate, when object c is created, the first/default constructor is called and those operations are performed.
- Parameterized constructor - This may have 1 or more parameters as arguments that are passed to the constructor on creating the object. In the above example for class Calculate, when objects c1 and c2 are created the respective methods are called based on the number of parameters passed.

Constructor chaining is the method of calling one constructor inside another constructor. For example, in class Calculate, creating a new object Calculate c3 = new Calculate() from any of the other constructors will call the default constructor.
This is known as Constructor Chaining

8) MySQL ? Why is it used ?

MySQL uses SQL as its base therefore implementing the same features of SQL. It is also a way to create databases and store data in them using SQL queries.
MySQL is secure, quick and efficient to use and is therefore preferred for manipulation of data and CRUD operations.

9) Query to create Movie Tables

```
Create table movie_details (
Id int primary key not null,
Title varchar(40),
Category varchar(20),
Language varchar(30)
);

Create table movie_member (
ID int primary key not null,
First_name varchar(30),
Last_name varchar (30),
Movie_id int,
Age int,
Foreign key (movie_id) references movie_details(id)
);
```

10) Query to insert into Movie Tables

```
Insert into movie_details values
(1, 'KGF', 'Action', 'Kannada'),
(2, 'Safe House(2012)', 'Action', 'English'),
(3, 'Dil Bechara', 'Romance', 'Hindi'),
(4, '3 idiots', 'Comedy Drama', 'Hindi'),
(5, 'Vikram Vedha', 'Action Thriller', 'Tamil'),
(6, 'Chello Divas', 'Comedy', 'Gujarati'),
(7, 'Real Steel (2012)', 'Animations', 'English'),
(8, 'Safe (2012)', 'Action', 'English');

Insert into movie_member values
(1,'Adam','Smith',1,20),
(2, 'Ravi', 'Kumar', 2, 19),
(3, 'Susan', 'Davidson', 5, 17),
(4, 'Jenny', 'Adrianna', 8, 21),
(6, 'Lee', 'Pong', 4, 18),
(7, 'Rakesh', 'Khanna', 3, 25),
(9, 'Pravin', 'Rathod', 5, 22),
(8, 'Vishal', 'Hatti', 6, 25),
(10, 'Rohan', 'Patil', 1, 23);
```

11) Query for left outer join

```
select * from movie_member
```

```
left outer join movie_details
on movie_member.movie_id = movie_details.id;
```

12) Query for right join

```
select * from movie_member
right outer join movie_details
on movie_member.movie_id = movie_details.id;
```

13) Query to get movie_member for movie ‘3 idiots’

```
select distinct movie_member.ID, movie_member.first_name, movie_member.last_name
from movie_member
inner join movie_details
where movie_member.movie_id = (select movie_details.id
from movie_details where movie_details.Title = '3 idiots');
```

14) Query to get matching rows from both tables

```
select * from movie_member
inner join movie_details
on movie_member.movie_id = movie_details.id;
```

15) Query to get all rows from right table

```
select * from movie_member
right join movie_details
on movie_member.movie_id = movie_details.id;
```

16) Query to get unique rows from both tables

```
select distinct * from movie_member
join movie_details
on movie_member.movie_id = movie_details.id;
```