JDBC

1.What is JDBC?

JDBC is an acronym for Java Database Connectivity, it is an API which helps to connect a java program or application to the database.

2.Who are Database vendors? Name any 5 database vendors?

A database vendor is an entity that offers one or more databases to customers for license or sale.
Some of the top database vendors are
MySQL, Oracle, PostgreSQL, MongoDB, Microsoft SQL Server, and Redis etc...

3.What is Database driver software and who provides it?

A database driver is a computer program that implements a protocol (ODBC or JDBC) for a database connection.
The driver works like an adaptor which connects a generic interface to a specific database vendor implementation.
Database driver is provided by the vendor of the particular database.

4.Explain Class.forName()

The forName() method of java.lang.Class class is used to get the instance of this Class with the specified class name. This class name is specified as the string parameter.
Syntax:
public static Class<T> forName(String className) throws ClassNotFoundException
Parameter:
This method accepts the parameter className which is the Class for which its instance is required.
Return Value:
This method returns the instance of this Class with the specified class name.

5.Explain registerDriver() method

The registerDriver() method of the Driver Manager class accepts an object of the diver class as a parameter and, registers it with the JDBC driver manager.

6.What are the different ways of loading and registering the drivers in JDBC.

To connect with a database using JDBC we need to select, get the driver for the respective database and register the driver.
You can register a database driver in two ways –
1.Using Class.forName() method –
The forName() method of the class named Class accepts a class name as a String parameter and loads it into the memory, soon it is loaded into the memory it gets registered automatically.
2. Using the registerDriver() method –
The registerDriver() method of the DriverManager class accepts an object of the diver class as a parameter and, registers it with the JDBC driver manager.

7.What are the different ways of getting a connection object in JDBC.

JDBC application connects to a target data source using one of two classes:

DriverManager:

This fully implemented class connects an application to a data source, which is specified by a database URL. Connecting to your DBMS with the DriverManager class involves calling the method DriverManager.getConnection() which returns connection object.

When this class first attempts to establish a connection, it automatically loads any JDBC 4.0 drivers found.

Data Source:

Objects instantiated by classes that implement the Data Source represent a particular DBMS or some other data source, such as a file.

8.What is properties file?

The properties object contains key and value pair both as a string.

The java.util.Properties class is the subclass of Hashtable.

It can be used to get property value based on the property key. The Properties class provides methods to get data from the properties file and store data into the properties file. Moreover, it can be used to get the properties of a system.

9.Explain with an example how to read the data from properties file?

To get information from the properties file, create the properties file first.

example,

db.properties

```
user=system
password=oracle
```

Now, let's create the java class to read the data from the properties file.

Test.java

```
import java.util.*;
import java.io.*;
public class Test {
public static void main(String[] args)throws Exception{
  FileReader reader=new FileReader("db.properties");

  Properties p=new Properties();
  p.load(reader);

  System.out.println(p.getProperty("user"));
  System.out.println(p.getProperty("password"));
}
}
```

10.What is Connection in JDBC?

A Connection is a session between a Java application and a database.

It helps to establish a connection with the database.

The Connection interface is a factory of Statement, PreparedStatement, and DatabaseMetaData, i.e., an object of Connection can be used to get the object of Statement and DatabaseMetaData.

The Connection interface provide many methods for transaction management like commit(), rollback(), setAutoCommit(), setTransactionIsolation(), etc.

11.What is Result Set?

The result set is an object that represents a set of data returned from a data source, usually as the result of a query.
It is an Interface.

12.Explain few methods of Result Set.

public boolean next():
Used to move the cursor to the one row next from the current position.
public boolean previous():
Used to move the cursor to the one row previous from the current position.
public boolean first():
Used to move the cursor to the first row in result set object.
public boolean last():
Used to move the cursor to the last row in result set object.
public boolean absolute(int row):
Used to move the cursor to the specified row number in the ResultSet object.
public boolean relative(int row):
Used to move the cursor to the relative row number in the ResultSet object, it may be positive or negative.
public int getInt(int columnIndex):
Used to return the data of specified column index of the current row as int.
public int getInt(String columnName):
Used to return the data of specified column name of the current row as int.
public String getString(int columnIndex):
Used to return the data of specified column index of the current row as String.
public String getString(String columnName):
Used to return the data of specified column name of the current row as String.

13.What is Statement?

The statement interface is used to create SQL basic statements in Java it provides methods to execute queries with the database.
There are different types of statements that are used in JDBC as follows:
1. Create Statement
2. Prepared Statement
3. Callable Statement

14.What is Create statement

It is used for general-purpose access to your database. Useful when you are using static SQL statements at runtime. The Statement interface cannot accept parameters.

15.What is Prepared statement

Prepared Statement represents a recompiled SQL statement, that can be executed many times.
This accepts parameterized SQL queries. In this, "?" is used instead of the parameter, one can pass the parameter dynamically by using the methods of PREPARED STATEMENT at run time.

16.Explain difference between Statement and Prepared Statement

Statement	Prepared Statement
It is used when SQL query is to be executed only once.	It is used when SQL query is to be executed multiple times.
You cannot pass parameters at runtime.	You can pass parameters at runtime.
Used for CREATE, ALTER, DROP statements.	Used for the queries which are to be executed multiple times.
Performance is very low.	Performance is better than Statement.
It is base interface.	It extends statement interface.
Used to execute normal SQL queries.	Used to execute dynamic SQL queries.
We cannot use statement for reading binary data.	We can use Prepared statement for reading binary data.
It is used for DDL statements.	It is used for any SQL Query.
We cannot use statement for writing binary data.	We can use Prepared statement for writing binary data.
No binary protocol is used for communication.	Binary protocol is used for communication.

17.What is the difference between execute(), executeQuery() and executeUpdate()

execute():

This method is used to execute SQL DDL statements, it returns a boolean value specifying weather the ResultSet object can be retrieved.

executeUpdate():

This method is used to execute statements such as insert, update, delete. It returns an integer value representing the number of rows affected.

executeQuery():

This method is used to execute statements that returns tabular data (example select). It returns an object of the class ResultSet.

18.What are placeholders or delimiters

Placeholders in sample code and commands represent values that the user must replace when they use the sample input.

All parameters in JDBC are represented by the "?" symbol, which is known as the parameter marker.

You must supply values for every parameter before executing the SQL statement.

The setXXX() methods bind values to the parameters, where XXX represents the Java data type of the value you wish to bind to the input parameter.

If you forget to supply the values, you will receive an SQLException.

Example:

```
String SQL = SELECT * from STOCKS WHERE TICKER=?
```

```
ps.setString(1, "MSFT");
```

```
ResultSet rs = ps.executeQuery();
```

19.How do we achieve batch execution

Here is a typical sequence of steps to use Batch Processing with Statement Object –
Create a Statement object using createStatement() method.
Set auto-commit to false using setAutoCommit().
Add as many as SQL statements you like into batch using addBatch() method on created statement object.
Execute all the SQL statements using executeBatch() method on created statement object.
Finally, commit all the changes using commit() method.
Here is a typical sequence of steps to use Batch Processing with PrepareStatement Object –
Create SQL statements with placeholders.
Create PrepareStatement object using prepareStatement() method.
Set auto-commit to false using setAutoCommit().
Add as many as SQL statements you like into batch using addBatch() method on created statement object.
Execute all the SQL statements using executeBatch() method on created statement object.
Finally, commit all the changes using commit() method.

20.What is Connection pool, write a program to achieve connection pool.

Establishing JDBC connections is resource-expensive, especially when the JDBC API is used in a middle-tier server environment. In this type of environment, performance can be improved significantly when connection pooling is used.
Connection pooling means that connections are reused rather than created each time a connection is requested. To facilitate connection reuse, a memory cache of database connections, called a connection pool, is maintained by a connection pooling module as a layer on top of any standard JDBC driver product.
Connection pooling is performed in the background and does not affect how an application is coded; however, the application must use a DataSource object (an object implementing the DataSource interface) to obtain a connection instead of using the DriverManager class.
Program:
To Create Connection pool

```
javax.sql.DataSource source = new org.apache.commons.dbcp.BasicDataSource();
source.setDriverClassName("com.mysql.jdbc.Driver");
source.setUsername("username");
source.setPassword("password");
source.setUrl("jdbc:mysql://localhost:3306/myDatabase");
```

To Get connection from the pool we will write,

```
java.sql.Connection connection = source.getConnection();
```

Closing the connection,

```
connection.close();
```

22.What is metadata?

Much of what you have done with JDBC so far requires you to know a lot about the database you are using, including the capabilities of the database engine and the data model against which you are operating. Requiring this level of knowledge may not bother you much, but JDBC does provide the tools to free you from these limitations. These tools come in the form of meta-data.

The term “meta” here means information about your data that does not interest the end users at all, but which you need to know in order to handle the data.

JDBC provides two meta-data classes:

java.sql.ResultSetMetaData and java.sql.DatabaseMetaData.

The meta-data described by these classes was included in the original JDBC ResultSet and Connection classes. The team that developed the JDBC specification decided instead that it was better to keep the ResultSet and Connection classes small and simple to serve the most common database requirements. The extra functionality could be served by creating meta-data classes to provide the often-esoteric information required by a minority of developers.

As its name implies, the ResultSetMetaData class provides extra information about ResultSet objects returned from a database query. This class provides you with answers to the following questions:

How many columns are in the result set?

Are column names case-sensitive?

Can you search on a given column?

Is NULL a valid value for a given column?

How many characters is the maximum display size for a given column?

What label should be used in a display header for the column?

What is the name of a given column?

What table did a given column come from?

What is the datatype of a given column?

The DatabaseMetaData class relates to the Connection class (in spite of the naming inconsistency). The DatabaseMetaData class provides methods that tell you about the database for a given Connection object, including:

What tables exist in the database visible to the user?

What username is being used by this connection?

Is this database connection read-only?

What keywords are used by the database that are not SQL2?

Does the database support column aliasing?

Are multiple result sets from a single execute() call supported?

Are outer joins supported?

What are the primary keys for a table? Etc.

23.How to secure data present in the Database?

Physical security:
Whether your database server is on-premise or in a cloud data center, it must be located within a secure, climate-controlled environment.

Administrative and network access controls:
The practical minimum number of users should have access to the database, and their permissions should be restricted to the minimum levels necessary for them to do their jobs.
Likewise, network access should be limited to the minimum level of permissions necessary.

End user account/device security:
Always be aware of who is accessing the database and when and how the data is being used. Data monitoring solutions can alert you if data activities are unusual or appear risky. All user devices connecting to the network housing the database should be physically secure and subject to security controls at all times.

Encryption:
ALL data—including data in the database, and credential data—should be protected with best-in-class encryption while at rest and in transit. All encryption keys should be handled in accordance with best-practice guidelines.

Database software security:
Always use the latest version of your database management software, and apply all patches as soon as they are issued.

Application/web server security:
Any application or web server that interacts with the database can be a channel for attack and should be subject to ongoing security testing and best practice management.

Backup security:
All backups, copies, or images of the database must be subject to the same (or equally stringent) security controls as the database itself.

Auditing:
Record all logins to the database server and operating system, and log all operations performed on sensitive data as well. Database security standard audits should be performed regularly.

24.Name some of the encrypting and decrypting algorithm which are widely used in industries.

1. Triple DES - Data Encryption Standard (DES)- The algorithm uses a 56-bit individual key with the total key length adding up to 168 bits.
2. RSA - Rivest-Shamir-Adleman (RSA)- The keys for the RSA algorithms are generated by multiplying the large number and creating a modulus.
3. Blowfish- The length of the keys can be anywhere from 32 bits to 448 bits, and so far, the encryption has never been defeated.
4. Twofish- It uses block encrypting and splits the data into blocks that are 128 bits long, and the key is applied simultaneously to all blocks. The key for the encryption can be 256 bits long.
5. AES - Advanced Encryption Standard (AES)- It is highly efficient in its basic 128-bit form and uses 192 and 256-bit keys for some robust encryption.

25.How does AES work?

The **AES algorithm** (also known as the **Rijndael algorithm**) is a symmetrical block cipher algorithm that takes plain text in blocks of 128 bits and converts them to ciphertext using keys of 128, 192, and 256 bits. Since the AES algorithm is considered secure, it is in the worldwide standard.

The AES algorithm uses a substitution-permutation, or SP network, with multiple rounds to produce ciphertext. The number of rounds depends on the key size being used. A 128-bit key size dictates ten rounds, a 192-bit key size dictates 12 rounds, and a 256-bit key size has 14 rounds. Each of these rounds requires a round key, but since only one key is inputted into the algorithm, this key needs to be expanded to get keys for each round, including round 0.

Each round in the algorithm consists of four steps.

Substitution of the bytes

In the first step, the bytes of the block text are substituted based on rules dictated by predefined S-boxes

Shifting the rows

Next comes the permutation step. In this step, all rows except the first are shifted by one

Mixing the columns

In the third step, the Hill cipher is used to jumble up the message more by mixing the block's columns.

Adding the round key

In the final step, the message is XORed with the respective round key.

When done repeatedly, these steps ensure that the final ciphertext is secure.