Java Database Connectivity (JDBC):

>> mysql will provide the driver class, which will helpful to interact with mysql database by using java

>> There are some prerequisties that we need to follow inorder to use the features

1.Load the driver class

2.Connect to the database

3.Create the statement

4.Execute the statement

5.Process the result

Step : 1 :- Load the driver class

>> We use " Class.forName " in order to load the driver class in mysql driver package

>> we need to mention the path clearly

>> **path follows:**

loader = com.mysql.cj.jdbc.Driver

Class.forName("loader");

>> we are forcefully loading the driver class

>> In current version we don't need to load the class

Step : 2 :- Connect to the database

>> Here we have to make the connection to datatbase

>> In order to make the connection we need the following:

1.Connection interface

2.DriverManager

3.getConnection method

4.url : jdbc:mysql://localhost:3306//database_name

5.username: root

6.passoword:root

Connection connection = DriverManager.getConnection(url,username,passoword);

Step : 3 :- Creating the statement

>> Statement is used to create the sql query or commands and used them for further process

>> There are three types of statements and statement is the father of all statements

1.Statement

>> Statement statement = connection.createStatement();

2.PreparedStatement

>>PreparedStatement pStat = connection.prepareStatement();

// pass the sql query in string format

3.CallableStatement

>>CallableStatement cStat = connection.prepareCall();

// call the store procedure in curly braces and thatin string format

Step : 4 :- Execute the statement

>>Here is the statement will helpful in executing the queries and commands and that will reflect in database

>> There are many ways to execute the queries:

Statement:

1. statement.executeQuery(sql);

>> this is for complete queries and for read operation

2. statement.executeUpdate(sql);

>> this is for update queries

>> it will give the number of rows updated

PreparedStatement:

1. pStat.executeUpdate();

>> this is for incomplete queries

>> It won't need any arguments

>> It will process the sql query and reflect in the database

>> It will return the number of rows updated

2. pStat.setInt(placeholder,data); and so on

>> this is used to set the value to the placeholder in sql query

>> It accepts two arguments

CallableStatement:

1.cStat.setString(1, deptName);

>> This is used to set the value in prepare call statement

2.cStat.registerOutParameter(2,Types.INTEGER);

>> This is used to accept the output from store procedere method from prepare call statement

>> It accepts the two parameters

1. placeholder
2. type of data

3.cStat.execute();

>> This will execute the callable statements

Execute Batch:

1.statement.addBatch(sql);

>> This will accept the number of sql statements and store in it.

2.statement.executeBatch();

>> This will execute all the queries added in the add batch statement

Step : 5 :- Process the result

>> Here we need to process the executed data

>> For that we have Result set inteface

ResultSet result = null;

>> Declaration

result = statement.executeQuery(sql);

>> This will be helpful to process the table data

>> Belongs to statement

int[] arr = statement.executeBatch();

>> This will give the rows updated number in the form of integer array

int i = cStat.getInt();

>> This will give the output of store procedure

>> Belongs to callable statement

result = cStat.getResultSet();

>> This will be helpful to process the table data

>> Belongs to callable statement

these are the five major steps in jdbc in order work with database

==========

There is some extra concepts :

1. ACID Properties
2. Store Procedure
3. File Handling

1. ACID Properties:

>> There the two terminologies we need to understand in order to make the data operations efficiently

1. Transaction
2. ACID Properties

1. Transaction:

>> It is the process of doing operations by the user and the changes made in the database

2. ACID Properties:

>> In order to do efficient transaction we need to meet the reqquirements of these properties

1. Atomitcity:

>> It judges whether the transaction is successfull or failed

>> Transaction need to happen on both sides then it is successfull

>> Transaction does'n happen on both sides then it is unsuccessful

>> It only accepts success or failure not mid way.

2. Consistency:

>> Here we consider how consistency means,

1. comparing the state of data table, before and after transacion
2. Old state of table must be equal to new state of table
3. Otherwise the trasaction is a failure

3. Isolation:

>> When there are two transaction happening and the same attributes are involving

1. The transaction must happen concurrently, means one by one
2. Then trasaction is successful in this case
3. If the one trasaction will involve in other trasaction the data must be inadequate
4. So, the trasaction need to be failed in this case

4. Durability:

>> Whenever we do transaction , the transaction must happens whether there is a system failure or not

>> In order to maintain this, we need to backup the data into non volatile memory

>> In JDBC we need to use the following methods in order to Apply ACID Properties:

1. connection.setAutoCommit(false);

>> this will prevent the changes happens to database by passing the false argument

>> so, we can work on applying the ACID Properties

2. connection.rollback();

>> this will help to revert back the transaction to last successful transaction

>> so, the changes happened in ram won't change the state of database

3. connection.commit();

>> this will make sure the transaction to be successfull

>> the data must be upated in the database

2. Store Procedure:

>> Like in java we can also create methods in mysql,and we call them as store procedure

>> Here we don't have any return types

>> This is how the sql structure looks like

```
CREATE PROCEDURE `dept_count( in name varchar(20))
    BEGIN
        select count(*)
            from employee
            where dept = "Sales";
    END
```


>> This is how we can extract data from storeprocedure

```
set @output = 0;
call jdbc01.dept_count('sales', @output);
select @output;
```

>> Coming to JDBC, we have to use Callable interface in order work with store procedure

CallableStatement cStat = connection.prepareCall("{ call dept_count(?,?)}");

cStat.setString(1,Sales);

>> setting value to input parameter of sql

cStat.registerOutParameter(2,Types.INTEGER);

>> setting up the output parameter

cStat.execute();

>> This will execute the store procedure and it stored the result

int i = cStat.getInt();

>> This will help to get the stored result

3. File handling:

>> We can also store the file type data in mysql, in order to do that we need to make use of filehandling

>> There are two types of handling the data:

1.Binary Large Object (BLOB)

>> This is used for images, gifs, pdf etc

>> We need to use the file input stream

>> Example:

FileInputStream file = new FileInputStream();

// absolute file path in string format

PrepareStatement pStat = connection.prepareStatement(sql);

sql = "update employee set photo = ? where id = ?";

pStat.setBinaryStream(1,file);

pStat.setInt(2,id);

pStat.execute();

2.Character Large Object (CLOB)

>> This is used for text files

>> We need to use the file reader

>> Example:

```
FileReader file = new FileReader();
    // absolute file path in string format
PrepareStatement pStat = connection.prepareStatement(sql);
sql = "update employee set message = ? where id = ?";
pStat.setBinaryStream(1,file);
pStat.setInt(2,id);
pStat.execute();
```

Thus, I conclude the key concepts and fundamentals in JDBC