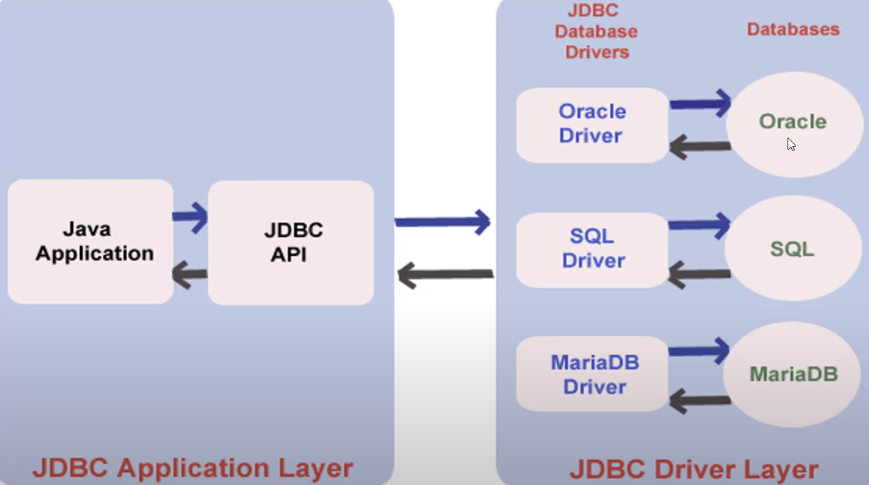
**JDBC**

****

**How JDBC works**

Java me apn code likhte h using JDBC API (Application programming Interface) ye API ka use krke apn code likhte h to code jaata h JDBC driver ke paas and Driver saari chize database tk pahucha deta h and CRUDE perform hota h.

As shown in Diagram, saare databases ka apna apna Driver hota h and JDBC API sabse conversate kr leta h.

Watch this video till 10:14 for basic setup needed to be done in Intelij to use JDBC driver

[**https://www.youtube.com/watch?v=KipCKAuQ5Fw**](https://www.youtube.com/watch?v=KipCKAuQ5Fw)

**Connecting a Java application with Database using JDBC API.**

**It is mainly 5 steps**

1. **Load the JDBC driver:** Registering the driver class in JDBC is like telling Java which database driver to use (After MySQL 8 ye part krne ke ki koi jarurat nhi).
2. **Create Connection**: In JDBC, creating a connection is like establishing a bridge between your Java program and a database. Agar connection nhi hoga to deta receive and send nhi hoga.
3. **Create Statement**: Creating a statement in JDBC is like preparing a message to send to the database. It allows you to write SQL queries or commands in your Java code. Without a statement, you can't tell the database what to do, like retrieving data or updating records.
4. **Execute Query.** Khuch Query execute krna h to karo.
5. **Close Connection/Resources**: Failing to close resources can lead to memory leaks and database connection issues, causing performance problems and potential crashes in your application.

Trick to remember this steps: 1) kaha jaana h 2) raasta kya h 3),4) Jaake krna kya h 5) Jaake aane ke baad raasta tod do.

**Make sure to import java.sql.\*** (JDBC API ka sabkhuch isse me h)

import java.sql.\*;  
public class Main {  
  
 public static void main(String[] args) {  
 // JDBC connection parameters  
  
 try {  
 // Step 1: Load the JDBC driver (Not neccessary after MySQL version 8)  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
  
 // Step 2: Establish a connection  
 String jdbcUrl = "jdbc:mysql://localhost:3306/dharam"; // dharam is the name of database. We can enter your desired database present in MySQL

String username = "root";  
 String password = "anappleaday.?@20";  
 Connection connection = DriverManager.*getConnection*(jdbcUrl, username, password);  
  
 // Step 3: Create a statement/preparedStatement ya something which is needed.  
 Statement statement = connection.createStatement();  
  
 // Step 4: Execute a query. Execution part me CRUDE operation perform kr sakte h.   
  
 // Step 5: Close resources

resultSet.close();  
 statement.close();  
 connection.close();  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**Creating a Table using Java**

Connection con = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/dharam","root", "anappleaday.?@20");  
  
Statement st = con.createStatement();  
String q = "create table tabThroughJava(id int primary key,name varchar(50), salary int not null);";  
st.executeUpdate(q); // This method passes the command to the database using the Statement   
con.close(); // At the end always close the connection

As usual we created the connection. Using Connection, we created the Statement. Using executeUpdate() member and Statement we passed the query to the Database and the table is created.  **Note:** All this is written inside try block.

**Inserting Into a table using PreparedStatement**

//As usual written inside try block

Connection con = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/dharam","root", "anappleaday.?@20");

String q = "Insert into tabThroughJava (id, name, salary) values (?,?,?);";  
PreparedStatement prSt = con.prepareStatement(q);  
// Tuple 1;  
prSt.setInt(1,1); // para -> 1. for first “?” 2. Value   
prSt.setString(2, "Dhano"); // para -> 1. for second “?” 2. Value   
prSt.setInt(3, 50000); // para -> 1. for third “?” 2. Value

prSt.executeUpdate(); // Make sure not to pass ‘q’ here

con.close(); // At the end always close the connection

**Note:** This way we can only insert 1 tuple at a time. For, eg

// Assuming Tuple 1;  
prSt.setInt(1, 1);

prSt.setString(2, "Dhano");  
prSt.setInt(3, 50000);  
// Above code won’t be inserted into the table  
// Assuming tuple 2  
prSt.setInt(1,2);  
prSt.setString(2,"Bano");  
prSt.setInt(3, 25000);

Is tarah karega and then run karega code, to bss 1 hi tuple insert hoga table me. But 1 konsa? last waala tuple.

We can enter multiple tuples by using Iterators, For eg,

//As usual written inside try block

Connection con = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/dharam", "root","anappleaday.?@20");  
  
String q = "insert into tabThroughJava (id,name,salary) values(?,?,?);";  
PreparedStatement pSt = con.prepareStatement(q);  
  
Scanner sc = new Scanner(System.*in*);  
int i,id,salary;  
String name;  
while(true) {  
 System.*out*.println("Press: 1. Insert, 2. No more Insertion");  
 i = sc.nextInt();  
 if (i != 1) break;  
 else if (i == 1) {  
 System.*out*.print("Enter id: ");  
 id = sc.nextInt();  
 sc.nextLine(); // why written this? Explaination below  
 // Assume we enter sc.nextInt = 1 and tap enter mtlb 1\n hamne input diya and 1 chala jayega id ke paas but \n reh jayega (in the buffer meomory). ab agar name = sc.nexInt direct kiya to \n jo bacha hua tha vo chala jayega name ko and name user se le hi nhi payenge. Therefore to avoid this we wrote this taaki \n ye khaa jaye.  
  
 System.*out*.print("Enter name: ");  
 name = sc.nextLine();  
  
 System.*out*.print("Enter Salary: ");  
 salary = sc.nextInt();  
  
 pSt.setInt(1, id);  
 pSt.setString(2,name);  
 pSt.setInt(3, salary);  
  
 pSt.executeUpdate();  
 }  
}  
con.close();

**Inserting Image to Database**

We first create the table, in MySQL (could have done using Java too). (Blob is datatype used to store images upto 65kb Size more than this can’t be stored inside blob).

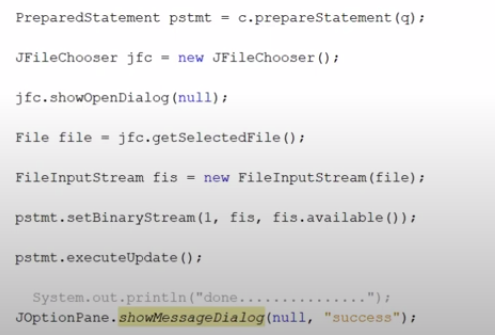
//As usual written inside try block

Connection con = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/dharam", "root","anappleaday.?@20");  
  
String q = "insert into images(pic) values (?)";  
PreparedStatement pSt = con.prepareStatement(q);  
  
FileInputStream fis = new FileInputStream("logo.jpg"); //para -> image path (if in same folder just the name. If not in same folder specify the complete path)

pSt.setBinaryStream(1,fis,fis.available()); // 1st para -> column no, 2nd -> FileInputStream object name, 3rd -> fis me kitne byte ka memory available h ye bata do

pSt.executeUpdate();  
con.close();

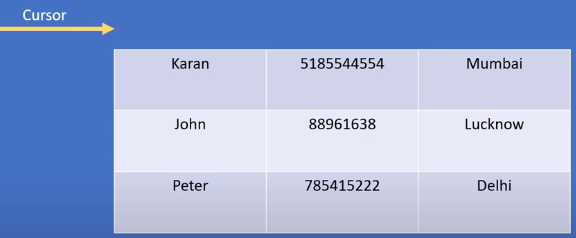
there also exists longblob which can store images of large size.



**Update Table**

// As usual inside try block

Connection cn = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/dharam", "root", "anappleaday.?@20");  
String q = "update tabThroughJava set name = ?, salary = ? where id = ?";  
PreparedStatement pSt = cn.prepareStatement(q);  
  
Scanner sc = new Scanner(System.*in*);  
System.*out*.print("Enter which Id Info you want to change : ");  
int id = sc.nextInt();  
sc.nextLine();  
System.*out*.print("Enter new name: ");  
String name = sc.nextLine();  
System.*out*.print("Enter new salary: ");  
int salary = sc.nextInt();  
  
  
pSt.setString(1,name);  
pSt.setInt(2,salary);  
pSt.setInt(3,id);  
pSt.executeUpdate();  
cn.close();

**ResultSet**

It is an interface provided by JDBC API to store the table of a database in it.

The object created using ResultSet points at nowhere in the table.

We can go through the rows one by one, like flipping through pages in a book, using method **next()** to move to the next row.

We can read the values in each column for the current row using methods like getInt(), getString(), etc.

**Note:** The pointer/cursor can move forward only and the table can’t be updated using it.

**Using Select to print Data in the terminal**

// As usual inside try block

Connection con = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/dharam", "root","anappleaday.?@20");  
  
Statement st = con.createStatement();

String sqlQuery = "SELECT \* FROM department";  
ResultSet resultSet = statement.executeQuery(sqlQuery); // It stores the mentioned table..   
  
while (resultSet.next()) {//.next() -> used to move to the next row of data in a result set  
 int dNo = resultSet.getInt("Dno"); // here we can either pass column name or column no  
 String dName = resultSet.getString("Dname");// here we can either pass column name or column no  
 System.out.println("Dpt No: " + dNo + ", Dpt Name: " + dName);