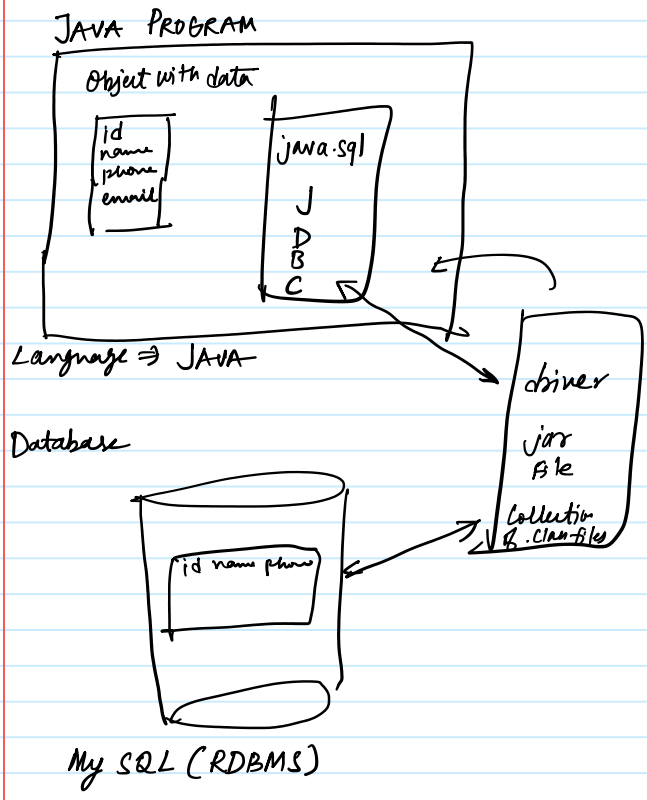
JDBC Connectivity

Friday, March 25, 2022 5:44 PM



In JAVA
- ↳ the data is in Object form
- ↳ In DB, it is in table form

Language ⇒ SQL

* DB connectivity Steps :

1. Link the jar file in projects,
2. Create connection to DB
3. Execute SQL Statements / Instructions.
4. Close the connection.

MySQL DB ⟶ stores DB in tables

1. Create table in MySQL

⇒ customer
{public int cid; ? JAVA

⇒ customer

```
{public int cid;
 public String name;
 public String email;}
```

JAVA

↓

Making Unique

```
Create table customer
{
    cid int primary auto_increment,
    name varchar(256),
    email varchar(256)
```

```
Customer cust = new Customer (1, "John", "99988", "abc@example.com");
```

↑ JAVA Object instantiation (Create customer object)

```
insert into Customers values (1, 'John', '97895', 'abc@example.com');
```

↑ SQL (Insert customer row/records)

↑ String in single quotes

* Install MySQL

Install From Oracle

↳ after installation

↓

```
mysql -u root -p
```

username ⇒ root
password ⇒ Blank (nothing) (Just hit enter)

```
show databases;
```

← To show the DB

```
Create database database_name;
```

↑ To create DB

```
Use estore;
```

↑ Select DB where you want to work.

```
show tables;
```

work.

show tables ;

* Now copy paste your created table

describe tables ;

↑ To see the table.

insert into Customer values (0, 'John', '9876543', 'john@example.com');

↑ insert entry into DB.

select * from Customer

↑ To view the whole customer table.

* JDBC procedure :

1. Load the JDBC MySQL driver
2. Download the Jar file,
3. Link the jar file.
4. Right click on your Java project name and on Build path ⇒ Configure Build Path
   - Click on Libraries
   - Select Module Path
   - Add External JAR's
   - Select JAR
   - Apply & Close.
5. Load the Driver Class from .jar file in JAVA program using class.forName API.
   - In your DB.java (Database)
     - make Constructor

check address in Hierarchy

make constructor

check address in Hierarchy

```
DB () { try {
        Class.forName ("com.mysql.cj.jdbc.Driver");
        System.out.println ("[DB] Driver Loaded");
      }
      catch (Exception e) {
          System.out.println ("Something Went Wrong"
                              + e);
```

(import java.sql.DriveManager)

2. Create Connection to DB.

we need :
url to db , username , password.

Use connection API and driverManager API to create connection to DB.

```
Connection connection ; (import =>
String url = "jdbc:mysql://localhost/estore";
    String User = "root";
    String Password = "";

connection = DriveManger.getConnection(url, user,
                                       password)
```

import java.sql.connection

Connection Created.

3. Execute SQL statement

Create a String sql instruction and use the statement or preparedStatement API to perform execution.

* In constructor at end of try write: statement = connection.

make a method execute SQL statement : (create statement)

```
public int executeSQLstatement (String sql) {
    int result = 0;
```

```
try {
    result = statement.executeUpdate(sql);
    System.out.println("[DB] SQL statemant Executed");
}
catch (Exception e) {
    System.out.print("Error"+e);
}
```

Make a reference outside statement statement

4.

Close Connection

```
public void closeConnection() {
    try {
        connection.close();
        System.out.println("[DB] connection Closed");
    }
    catch(Exception e) {
        System.out.println("Error "+e);
```