Week 03 – Day 01 Short Notes

Introduction Terminologies

A Data is just information without any context. A database is a systematic collection of related data. RDBMS is data stored in form of rows and columns.

SQL stands for Structured Query Language. Some types of SQL are: Microsoft SQL Server, MySQL, Oracle etc.

MySQL was founded in 1995 in Sweden. In 2000, MySQL went open source, so it could be accessed and used by all. MySQL is named after co-founder Monty Widenius's daughter, My.

MySQL Workbench provides a visual console to easily administer MySQL environments and gain better visibility into databases. MySQL Shell is an advanced client and code editor for MySQL. A primary key is used to assure the value in the particular column is unique. The foreign key provides the link between the two tables.

Practical Implementation

Opening a Local Server on MySQL Workbench

- Open MySQL Workbench
- Click on the Local MySQL Connection
- Enter your password (entered while installing)
- A new window will be open and you are all set to create the MySQL files/databases.

Creating a new database

- Open a new SQL file
- Type the following query :

```
create database Student;
```

- You can change the name of the database according to your own preferences.

Create a new Table

- Go to the "schemas" and check for your database
- Under the drop-down menu, look for Tables
- Right click on Tables and select "Create Table".
- On the new table screen, you need to fill all the details to create a table. Here, we are going to enter the table name and other attributes.

Understanding some MySQL basic queries

Create Database

create database <name of the database>;

To use the database: use <database name>;

Example –

```
create database employee01;
```

Create a new Table

create table <database name>.<table name> (c1 datatype1 property1, c2 datatype2 property2, c3 datatype3 property3……, Primary Key (c1));

If you don't want to have database.tableName then "use <database name>;" before creating the table.

Example – use employee01;

```
CREATE TABLE employeedata01 (
  `empID` INT NOT NULL,
  `empName` VARCHAR(45) NULL,
  `empLoc` VARCHAR(45) NULL,
  PRIMARY KEY (`empID`));
```

Select Query

*select * from <database>.<tableName>;*

Example – select * from employee01.employeedata01;

Insert Query

insert into <database>.<tableName> values(v1,v2,v3...);

Example – insert into employeedata01 values(1,"Rahut","Delhi");

Delete Query

delete from <database>.<tableName> where <condition>;

Example – delete from employeedata01 where empID=2;

Note: The condition needs to be based on the primary key. If the condition is based on any other column, then it gives an error as it is an unsafe access to the table.

For instance: delete from employeedata01 where empName="Cassie";

This will result in error as empName is not the primary key.

Update Query

Update <database>.<tableName> set <columnName> = <newValue> where <condition>;

Example – update employeedata01 set empName = 'Harsh' where empID=3;

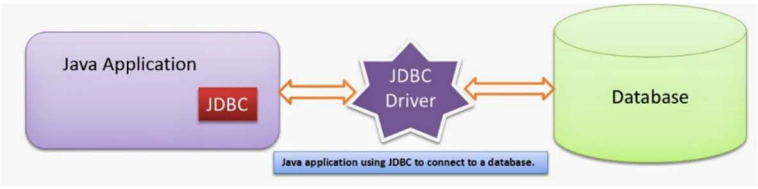
Note: Here as well the condition must be put on the primary key.

Drop Query

Drop database <databaseName>;

Example – drop database employee01;

Introduction to JDBC



Java application only knows Java and Database only knows SQL. In order to connect both together, the developers work on a single API known as JDBC (Java Database Connectivity). JDBC is already present in JDK and is installed automatically.

JDBC makes it possible to do the following things within a Java application:

- Establish a connection with a data source
- Send queries and update statements to the data source

5 steps to connect Database with JDBC

Step Name	Description
Load/Register Driver	**What is a Driver?** A driver provides a standard way to access data using the Java programming language. Each SQL language has its own JDBC driver which can be connected to the Java program. **There are 2 ways to load the driver:** 1. Class.forName(<full className>) Format of full className : *com.organizationName.module.submodule.class* forName is a public static method of the class. Full className of MySQL is **com.mysql.cj.jdbc.Driver** 2. Create object of Driver Use the "new" keyword Two methods: Driver driver = new Driver(); DriverManager.registerDriver(driver); OR DriverManager.registerDriver(new Driver()); DriverManager is present inside the java.sql package.

<table>
<tr><td>Establish the connection</td><td>After loading the register, the RDBMS driver comes into the Java application. But JDBC doesn't know the exact location of the driver. Hence, this step helps to establish the connection between JDBC and the driver.
There are 3 ways to do the same:
1. DriverManager.getconnection(url,username,password);
2. DriverManager.getconnection("query");
3. DriverManager.getconnection(url, .properties file);

.properties file contains the usernames and passwords in form of key-value pairs.
All the methods return the 'connection interface' which is present in java.sql package.</td></tr>
<tr><td>Establish the statement</td><td>Statements are present in the java.sql package.
Two types of statements:
1. Statement
To implement static queries.
connection.createStatement();
2. PreparedStatement
Used to implement dynamic queries
A placeholder(delimeter) is used => ?
connection.preparedStatement('');

<table><tr><th>Statement</th><th>Prepared Statements</th></tr><tr><td>Base Interface</td><td>Child Interface</td></tr><tr><td>Static Query</td><td>Dynamic Query</td></tr><tr><td>No placeholder</td><td>Placeholder used</td></tr><tr><td>Slower</td><td>Faster</td></tr></table></td></tr>
<tr><td>Execute the statement</td><td>There are three ways to execute the statements:
1. execute();
return type is Boolean
statement.execute();
preparedStatement.execute();
2. executeUpdate();
return type is Int
Used for non-select queries
3. executeQuery();
return type is ResultSet
Used for select queries</td></tr>
<tr><td>Close the connection</td><td>connection.close();</td></tr>
</table>

Adding Dependency to the Maven Project

- Go to pom.xml file => add the <dependencies></dependencies> tag
- Go to https://mvnrepository.com/ and search for mysql => Connector/J
- Choose the version acc to usage
- Copy the code and paste in dependencies