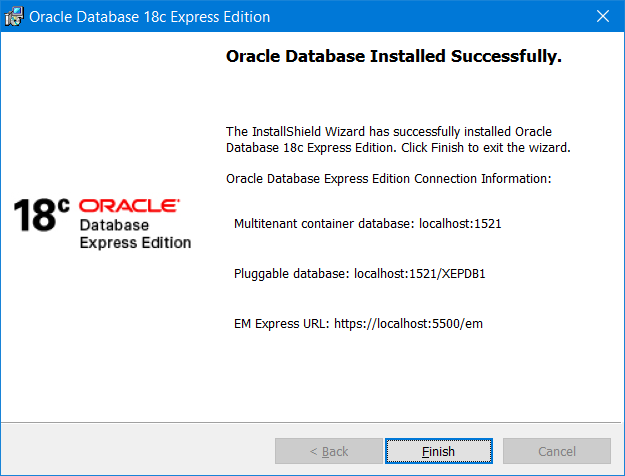
<https://docs.oracle.com/en/database/oracle/oracle-database/index.html> - Oracle 12c Official Documentation

<https://docs.oracle.com/database/121/index.html> - 12c Support URL

<https://docs.oracle.com/en/database/oracle/oracle-database/18/xeinw/installation-guide.html> oracle 18c XE documentation



SELECT PDB\_ID, PDB\_NAME, STATUS FROM DBA\_PDBS ORDER BY PDB\_ID;

select username from dba\_users; -- where ORACLE\_MAINTAINED='N' order by 1;

SELECT USERNAME FROM ALL\_USERS ORDER BY USERNAME;

SELECT USERNAME FROM ALL\_USERS WHERE username NOT IN ('ANONYMOUS', 'APEX\_030200', 'APEX\_040200', 'APEX\_PUBLIC\_USER', 'APPQOSSYS', 'AUDSYS', 'CLOUD\_ENGINE\_USER', 'CLOUD\_SWLIB\_USER', 'CTXSYS',

'DBSNMP', 'DIP', 'DMSYS', 'EXFSYS', 'DVF', 'DVSYS', 'FLOWS\_030000', 'FLOWS\_FILES', 'GSMADMIN\_INTERNAL', 'GSMCATUSER', 'GSMUSER',

'LBACSYS', 'MDDATA', 'MDSYS', 'MGMT\_VIEW', 'OJVMSYS', 'OLAPSYS', 'ORDDATA', 'ORDPLUGINS', 'ORACLE\_OCM', 'ORDSYS', 'OUTLN', 'OWBSYS',

'SCOTT', 'SI\_INFORMTN\_SCHEMA', 'SPATIAL\_CSW\_ADMIN\_USR', 'SPATIAL\_WFS\_ADMIN\_USR', 'SYS', 'SYSBACKUP', 'SYSDG', 'SYSKM', 'SYSMAN', 'SYSMAN\_APM',

'SYSMAN\_MDS', 'SYSMAN\_OPSS', 'SYSMAN\_RO', 'SYSTEM', 'TSMSYS', 'OWBSYS\_AUDIT', 'WKPROXY', 'WKSYS', 'WK\_TEST', 'WMSYS', 'XDB', 'XS$NULL');

ALTER USER HR ACCOUNT UNLOCK IDENTIFIED BY *hr*;

ALTER USER HR Account UNLOCK IDENTIFIED BY hr;

Sqlplus system

Enter Password : Indian123#

SELECT \* FROM DBA\_USERS\_WITH\_DEFPWD;

PASSWORD <USER\_NAME>;

Enter Password :

Retype Password :

User Password is modified.

Different ways of Storing the Data

1. Flat files (.txt, .rtf, .doc, .pdf etc.,) – consumes less space, no complex operations, easy to create
2. Dis Adv – Searching, getting partial data, filtering

DB – Database (It’s a way of storing the data in an organised format) – Logical Organised format

In Database, everything is called as Entity.

Primary Entity in database is Table. Data will be stored in ROWS & COLUMNS.

Each ROW contains different details of a particular person or object.

Each COLUMN contains a particular detail about many persons or Objects.

LOTUS – Father of Excel Sheet

FOXPRO – It is a Programming Language with inbuild database support.

DBMS – DataBase Management System – It’s a software to Manage the Database (Create, Read, Update, Delete – Operations)

RDBMS – Relational Database Management System (Relations between Tables – Primary key, foreign key associations)

RDBMS uses the concept of Relational Theory Mathematics (SET, Venn Diagram, Intersect, Union, Joins, Belongs to, Not Belongs to )

Entities in DBMS

1. Table
2. View
3. Query
4. Materialized View
5. Stored Procedures
6. Functions
7. Triggers
8. Sequences
9. Schema/Database
10. Users

Folders & Files

Popular RDBMS

1. Oracle
2. Postgres – Open Source
3. DB2 (IBM)
4. MySQL
5. In-Memory Database (SQLite, h2)
6. MS-SQL

All the databases work as a Client Server

Client – Server Concepts (Request object & Response Object)

* Clients (It can be a Computer/Mobile Phone/ Smart Device/IoT device) send request to the server
* On receiving the Request from the client, Servers will check the request and requested resource.
* If the Request is valid, it executes/hand-over the request to the proper location and wait for the response.
* Once the response is ready, server sends back the response to the client.
* Server will have all the details of all client’s request

Types of Servers

1. Web Server (Works with Web applications – handles http request & Response) – Tomcat, -Servlets, JSP, JSF
2. Application Server (J2EE Features, Email Config, JMS, EJB, Web Services)
3. Database Server (It handles database request and response)

Types Of Clients

1. CLI clients – (Command Line Interface) – Character User Interface based [Command based]
2. GUI Clients – (Graphical User Interface) – [Stand-alone, web]

Oracle 12c

Oracle XE Download URL -- <https://www.oracle.com/database/technologies/xe-downloads.html>

Git – It’s a Tool (Project Source Code Management Tool) – SCM Tool

Git is a Open Source, SCM tool for maintaining the source code of projects. (developed by linus Torvalds)

Linus Torvalds – Is the creator Linux Operating System Kernel.

<https://git-scm.com/> -- Free, Open Source & Distributed Version Control System (To handle various version of a software/application)

It tracks all the changes made to a repository.

GIT- CLI tool (Command Line Interface)

GIT – GUI Clients –

Git add

Git commit

Git Push

Git Pull

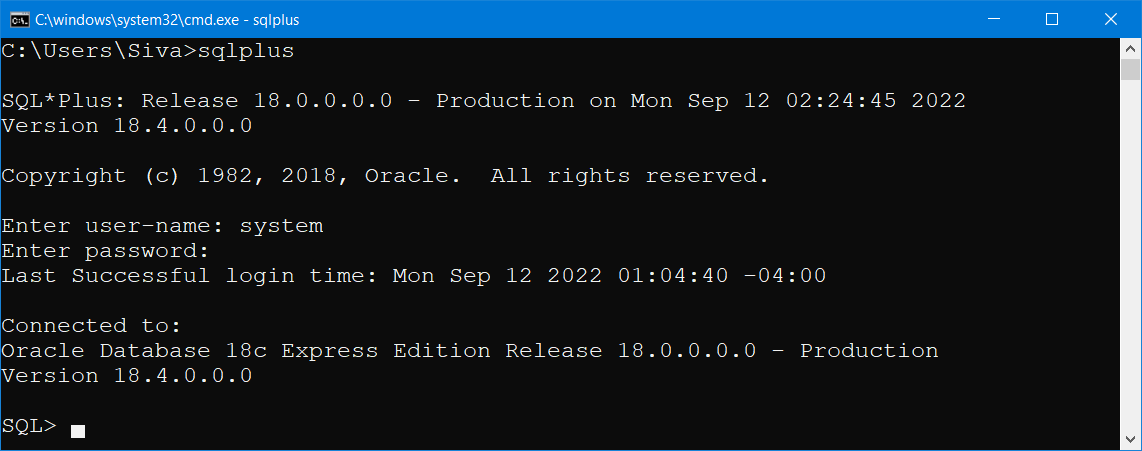
Git branch

Git checkout

Different ways of Connecting to Oracle Database

1. Using SQL PLUS (CLI Client) Command Line Interface
2. Using SQL Developer (Stand-alone GUI Client)
3. Using any Programming Lang (Java/PHP/Python/.Net)

Using SQL Plus

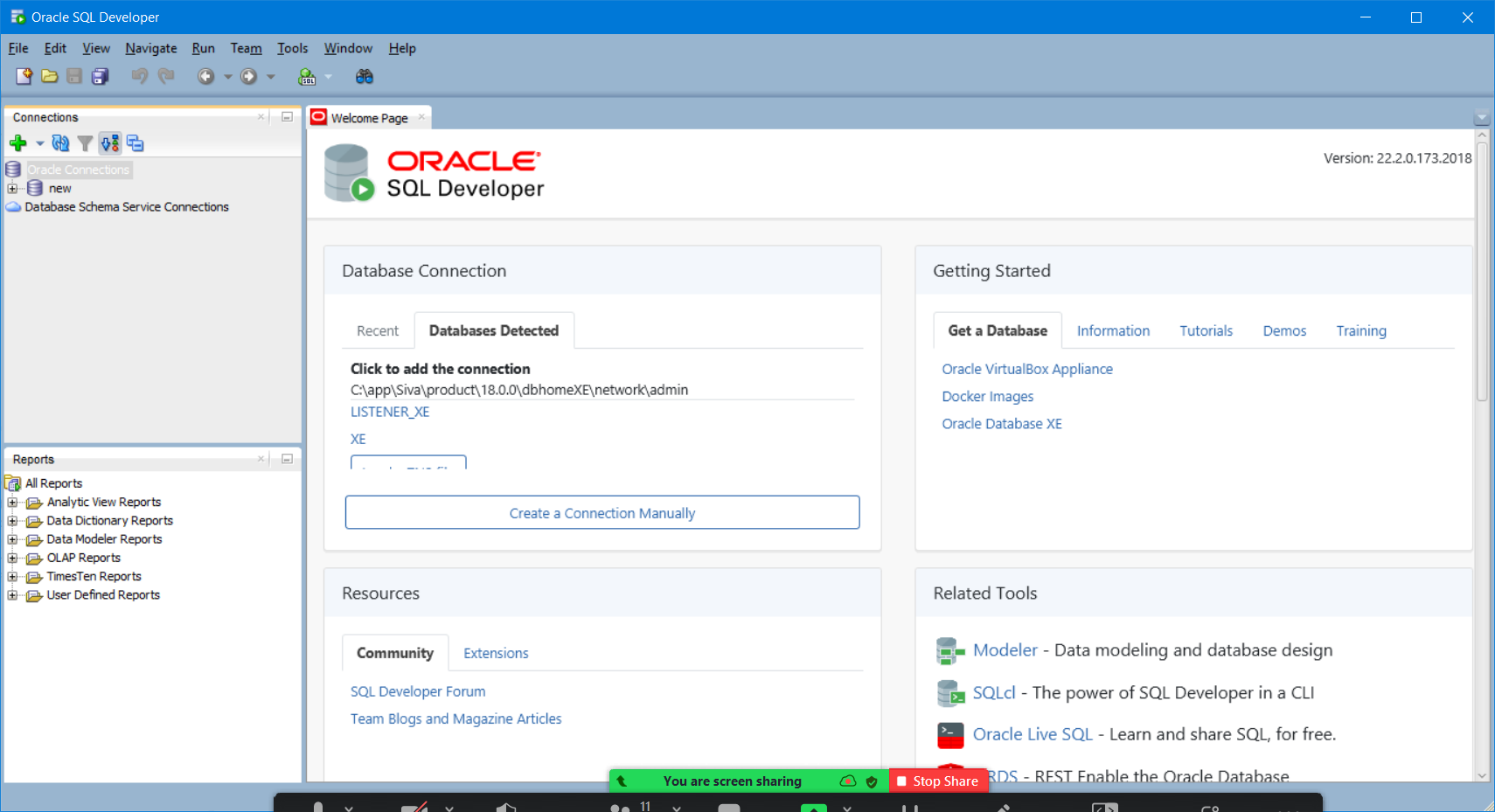


Using SQL Developer

SQL Developer Download URL - <https://www.oracle.com/database/sqldeveloper/technologies/download/>

APEX – Application Express

XE – eXpress Edition



MySQL Installer for Windows - [MySQL :: Download MySQL Installer](https://dev.mysql.com/downloads/installer/)

SQL – Backbone of RDBMS

SQL – Structured Query Language

In Database the primary Operations, CRUD Operations

C – Create /Insert

R – Read (ReadAll(), ReadOne(id))

U – Update

D – Delete

Create <entity\_name> <options> ;

Insert into <entity\_name> (column\_names) values (values1…values n);

Select \* or Column\_names from <Entity\_name>;

Update <entity\_name> SET <column\_name = Value> …;

Delete from <entity\_name> WHERE <condition\_using\_primarykey>;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl No** | **Database Name** | **CLI\_Client Name** | **GUI Client Name** | **Default Port Number** |
| 1 | Oracle XE | Sqlplus | SQL Developer | 8080/5500 |
| 2 | MySQL | mysql | MySQL Workbench | 3306 |
| 3 | Postgres | psql | pgAdmin | 5432 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl No** | **Database Name** | **Default UserName/Password** | **Connection URL** |
| 1 | Oracle | System/manager (scott/tiger) | Jdbc:oracle:thin:@localhost:1521/xe |
| 2 | MySQL | root/root | Jdbc:mysql://localhost:3306/db\_name |
| 3 | PostGres | Postgres/postgres | Jdbc:postgres://localhost:5432/db\_name |

IP Address of Localhost = 127.0.0.1

Physical/Hardware Ports = USB Port, HDMI Port, Audio Ports, VGA Ports, SD Card Slots, Charging Ports

Software Ports = Two Digit or 4 digit number (Any applications can use this port for data transfer)

Client is a software (Web Client – Web Browser – Google Chrome, Firefox, Opera, Edge, Safari)

Server is also a Software (Apache Tomcat, GlassFish, WebLogic, IIS.. )

GMAIL

<https://www.gmail.com/>

username & password

GMAIL Inbox

Database Server

1. Database Server URL
2. Username & Password (Credentials)
3. CRUD operations

Using Programming Lang

JAVA – JDBC (Java Database Connectivity)

3 ways of connecting to DATABASE

1. Using Command Line Client (SQL Command Knowledge is important)
2. Using GUI (Graphical User Interface) Client (SQL Knowledge is not necessary)
3. Using any Programming Language

CREATE TABLE `syskan1`.`employee` (

`employee\_id` INT NOT NULL AUTO\_INCREMENT,

`employee\_name` VARCHAR(75) NULL,

`email` VARCHAR(125) NULL,

`mobile` BIGINT(12) NULL,

`dob` DATE NULL,

PRIMARY KEY (`employee\_id`));

DROP TABLE `syskan1`.`employee`; (deletes table with all it’s content)

TRUNCATE `syskan1`.`employee`; (Deletes the contents of the table only)

Create a Table with the name “Account” with the following columns

1. Account\_ID (Auto\_Generated) Primary Key
2. Account\_type (Savings/Current/Deposit/Loan) Not null
3. Opening\_Balance (Float) >0
4. User\_ID (4 digit Number) Not Null
5. Active (Boolean value) Not null
6. Account\_Number (Alphanumeric SA/CA/DA/LA) Unique
7. Interest\_Rate (float) Not Null
8. Opening\_Date (DateTime) Not Null

Insert 5 sample records to the above table.

TYPES of SQL Commands

DDL – Data Definition Language (Create, Drop, Alter, Truncate)

DML – Data Manipulation Lang ( Insert, Update, Delete)

DCL – Data Control Language (Grant, Revoke – DBA commands)

DQL – Data Query Language (Select)

TCL – Transaction Control Language (Commit, Rollback, Savepoint)

SQL Queries

1. Simple Query
2. Inner Query (A Query within another Query) - Sub Query

List of Constraints (Evaluating Criteria – Check point)

1. Unique (Duplicates not allowed)
2. Not Null (Null/empty value not allowed)
3. Primary Key ( Unique + Not Null) – It is compulsory for each table
4. Check
5. Default
6. Foreign Key ( A primary key column referred in another table)- It allows duplicate but not null.

Operators in RDBMS

When writing queries we can use symbols (Operators)

=, <, >, <=,>=, !=

Keywords

WHERE

GROUP BY

ORDER BY

HAVING

IN

ANY

BETWEEN

Built-in Functions in RDBMS

Built-In Function / System/Pre-Defined Functions

1. Row level Functions (Applicable on each Row) – Return multiple Rows (TO\_CHAR, upper(), LOWER(), )
2. Column Level Functions (Applicable on Each column) – Return a single ROW (max, min, avg, sum )

Using JAVA programming Language – JDBC API

JDBC – Java Data Base Connectivity

API – Application Programming Interface (Set of Specifications)

JDBC contains many Interfaces (Contract)

Car is a general Term – It is a vehicle with 4 wheels, 1 steering, accelerator, brake, Engine, Gear box

Car is generalization – Interface

Car is a specification

Audi Car, is an implementation of Car (Class)

Honda Car, is an implementation of Car

BMW Car

Suzuki Car

JDBC

Database manufactures provided implementation of JDBC.

This is called as JDBC-Drivers (Database Drivers)

MySQL Driver

Oracle Driver

Postgres Driver

Trying to install a new printer with your laptop.

Assume you have unboxed and assembled your printer.

Steps to connect your printer with laptop/desktop

1) Installing the Drivers

2) Connecting printer with Laptop using Wires

3) Test Print

4) Actual Print

5) Power off

JDBC (Java – RDBMS)

Java is a programming Lang (process)

RDBMS – Is a Relational DB mgmt. system software (process)

Driver will be provided by database manufacturers.

1. Loading & Registering the Driver
2. Establishing the connection
3. Writing & Executing Query
4. Storing & Processing the Result Set
5. Closing all the resources

MySQL Driver download URL - <https://dev.mysql.com/downloads/connector/j/>

Types of Joins

Connecting two or more tables to get combined data.

1. Inner Join - Intersection
2. Right Join – Set Difference (B-A)
3. Left Join – Set Difference (A-B)
4. Outer Join - Union
5. Cross Join – Nested For loop

Select table1.col1, table1.col2, table2.col1, table2.col2 from table1 JOIN table2 on common\_col\_of\_table1 = common\_col\_of\_table2;

SELECT count(\*) FROM sakila.film;

select upper(title) TITLE, round(rating) RATING from film where rating >=3 ;

select TITLE from film where title like "air%";

select \* from film where length > 50 and replacement\_cost >25;

select \* from film order by replacement\_cost desc;

select max(rental\_rate) from film;

select min(rental\_rate) from film;

SELECT \* FROM sakila.customer;

select \* from Customer where address\_id = 5;

select concat(c.first\_name,' ',c.last\_name) as full\_name, concat(ad.address, ' ',ad.district,' ', ad.postal\_code) as address from customer c join address ad on c.address\_id = ad.address\_id;

select concat(c.first\_name,' ',c.last\_name) as full\_name, concat(ad.address, ' ',ad.district,' ', ad.postal\_code) as address, ci.city, co.country from customer c join address ad on c.address\_id = ad.address\_id join city ci on ad.city\_id = ci.city\_id join country co on ci.country\_id = co.country\_id;

CREATE TABLE `syskan1`.`course` (

`id` INT NOT NULL AUTO\_INCREMENT,

`name` VARCHAR(150) NULL,

`total\_hours` INT NULL,

`course\_url` VARCHAR(150) NULL,

`employee\_id` INT NULL,

PRIMARY KEY (`id`),

INDEX `cou\_fk\_idx` (`employee\_id` ASC) VISIBLE,

CONSTRAINT `cou\_fk`

FOREIGN KEY (`employee\_id`)

REFERENCES `syskan1`.`employee` (`employee\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

<https://codeshare.io/X8Yw38>

**package** com.deloitte;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.PreparedStatement;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.sql.Statement;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** DBCrud {

**private** String url = "jdbc:mysql://localhost:3306/syskan1";

**private** String username = "root";

**private** String password = "root";

**private** Connection conn = **null**;

**private** Statement stmt = **null**;

**private** PreparedStatement pstmt = **null**;

**private** ResultSet rs = **null**;

**public** **static** **void** main(String[] args) **throws** Exception {

// **TODO** Auto-generated method stub

DBCrud crud = **new** DBCrud();

List<Employee> employees = crud.findAll();

System.***out***.println(employees);

Employee emp = crud.findById(3);

emp.setEmail("updated");

emp.setName("updated");

crud.update(3, emp);

System.***out***.println(crud.findById(3));

Employee employee = **new** Employee();

employee.setId(5);

employee.setName("Test");

employee.setEmail("Test@gmail.com");

employee.setMobile(9089098990l);

crud.save(employee);

System.***out***.println(crud.findAll());

crud.delete(5);

System.***out***.println(crud.findAll());

}

**public** List<Employee> findAll() **throws** SQLException {

List<Employee> employees = **new** ArrayList<Employee>();

DBCrud crud = **new** DBCrud();

conn = crud.getConnection(url, username, password);

String selectAllQuery = "select \* from employee";

stmt = conn.createStatement();

rs = stmt.executeQuery(selectAllQuery);

**while** (rs.next()) {

Employee emp = **new** Employee();

emp.setId(rs.getInt("employee\_id"));

emp.setName(rs.getString(2));

emp.setEmail(rs.getString(3));

emp.setMobile(rs.getLong(4));

emp.setDob(rs.getDate("dob"));

employees.add(emp);

}

**return** employees;

}

**public** Employee findById(**int** id) **throws** SQLException {

Employee emp = **new** Employee();

DBCrud crud = **new** DBCrud();

conn = crud.getConnection(url, username, password);

String selectAllQuery = "select \* from employee where employee\_id=" + id;

stmt = conn.createStatement();

rs = stmt.executeQuery(selectAllQuery);

**if** (rs.next()) {

emp.setId(rs.getInt("employee\_id"));

emp.setName(rs.getString(2));

emp.setEmail(rs.getString(3));

emp.setMobile(rs.getLong(4));

emp.setDob(rs.getDate("dob"));

}

**return** emp;

}

**public** **void** save(Employee emp) **throws** SQLException {

DBCrud crud = **new** DBCrud();

conn = crud.getConnection(url, username, password);

String insertQuery = "insert into employee (employee\_id,employee\_name,email,mobile,dob) values (?,?,?,?,?)";

pstmt = conn.prepareStatement(insertQuery);

pstmt.setInt(1, emp.getId());

pstmt.setString(2, emp.getName());

pstmt.setString(3, emp.getEmail());

pstmt.setLong(4, emp.getMobile());

pstmt.setDate(5, emp.getDob());

**int** insertStatus = 0;

insertStatus = pstmt.executeUpdate();

**if** (insertStatus > 0)

System.***out***.println("1 Record Inserted Successfully!!!");

}

**public** **void** update(**int** id, Employee emp) **throws** SQLException {

DBCrud crud = **new** DBCrud();

conn = crud.getConnection(url, username, password);

String updateQuery = "update employee set employee\_name=?,email=?,mobile=?,dob=? where employee\_id= ?";

pstmt = conn.prepareStatement(updateQuery);

pstmt.setInt(5, emp.getId());

pstmt.setString(1, emp.getName());

pstmt.setString(2, emp.getEmail());

pstmt.setLong(3, emp.getMobile());

pstmt.setDate(4, emp.getDob());

**int** updateStatus = 0;

updateStatus = pstmt.executeUpdate();

**if** (updateStatus > 0)

System.***out***.println("1 Record Updated Successfully!!!");

}

**public** **void** delete(**int** id) **throws** SQLException {

DBCrud crud = **new** DBCrud();

conn = crud.getConnection(url, username, password);

String deleteQuery = "delete from employee where employee\_id=" + id;

stmt = conn.createStatement();

**int** deleteStatus = 0;

deleteStatus = stmt.executeUpdate(deleteQuery);

**if** (deleteStatus > 0)

System.***out***.println("1 Record Deleted Successfully!!!");

}

**public** Connection getConnection(String url, String username, String password) {

Connection conn = **null**;

**try** {

conn = DriverManager.*getConnection*(url, username, password);

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**return** conn;

}

**public** **void** closeResource() **throws** SQLException {

**if** (stmt != **null**)

stmt.close();

**if** (pstmt != **null**)

pstmt.close();

**if** (rs != **null**)

rs.close();

**if** (conn != **null**)

conn.close();

}

}

**package** com.deloitte;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.ResultSet;

**import** java.sql.Statement;

**public** **class** ConnectWithRDBMS {

**public** **static** **void** main(String[] args) **throws** Exception {

// **TODO** Auto-generated method stub

// Step 1 : Loading and Registering the Driver

Class.*forName*("com.mysql.cj.jdbc.Driver");

//Class.forName("oracle.jdbc.driver.OracleDriver");

String username = "root";

String password = "root";

String url = "jdbc:mysql://localhost:3306/syskan1";

//String oracleUrl ="jdbc:oracle:thin:@localhost:1521:xe";

// Step 2 : Establishing the connection

Connection conn = DriverManager.*getConnection*(url, username, password);

// Step 3: Creating & Executing the Query

String query = "select \* from employee";

Statement stmt = conn.createStatement();

// Step 4 : Storing & Processing the Result

ResultSet rs = stmt.executeQuery(query);

**while** (rs.next()) {

System.***out***.println(rs.getInt(1) + " " + rs.getString(2) + " " + rs.getString("email") + " "

+ rs.getLong("mobile") + " " + rs.getDate("dob"));

}

//Step 5: Closing Resources

**if** (rs!=**null**)

rs.close();

**if** (stmt!=**null**)

stmt.close();

**if** (conn!=**null**)

conn.close();

}

}

**package** com.deloitte;

**import** java.sql.Date;

//Bean Class - A class with properties, constructor and getters,setters

**public** **class** Employee {

**private** **int** id;

**private** String name;

**private** String email;

**private** **long** mobile;

**private** Date dob;

**public** Employee() {

**super**();

}

**public** Employee(**int** id, String name, String email, **long** mobile, Date dob) {

**super**();

**this**.id = id;

**this**.name = name;

**this**.email = email;

**this**.mobile = mobile;

**this**.dob = dob;

}

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getEmail() {

**return** email;

}

**public** **void** setEmail(String email) {

**this**.email = email;

}

**public** **long** getMobile() {

**return** mobile;

}

**public** **void** setMobile(**long** mobile) {

**this**.mobile = mobile;

}

**public** Date getDob() {

**return** dob;

}

**public** **void** setDob(Date dob) {

**this**.dob = dob;

}

@Override

**public** String toString() {

**return** "Employee [id=" + id + ", name=" + name + ", email=" + email + ", mobile=" + mobile + ", dob=" + dob

+ "]";

}

}