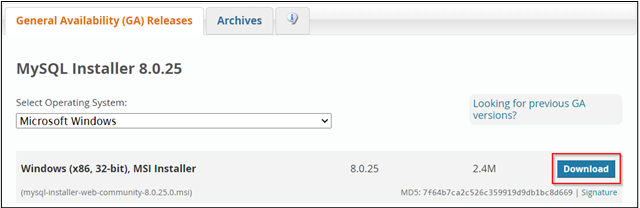
# Lab 1. Download MySQL installer for windows

## Download the MySQL installer

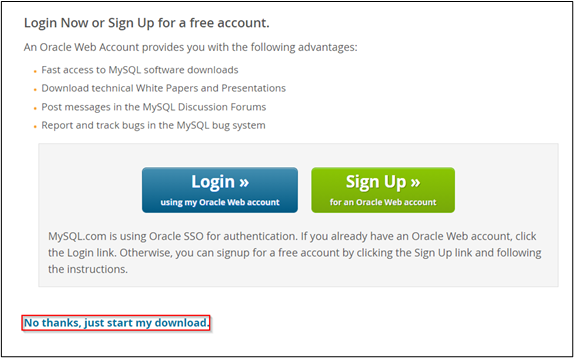
1. Download the MySQL installer from the following

<https://dev.mysql.com/downloads/installer/>

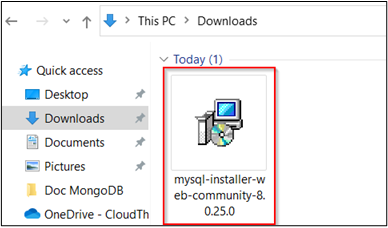
1. Click on download



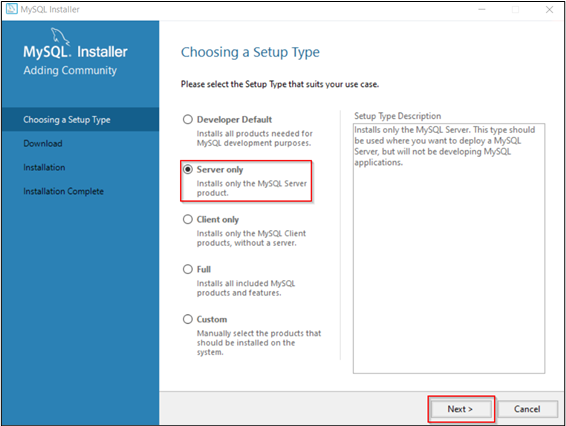
1. Select “No thanks, just start my download”



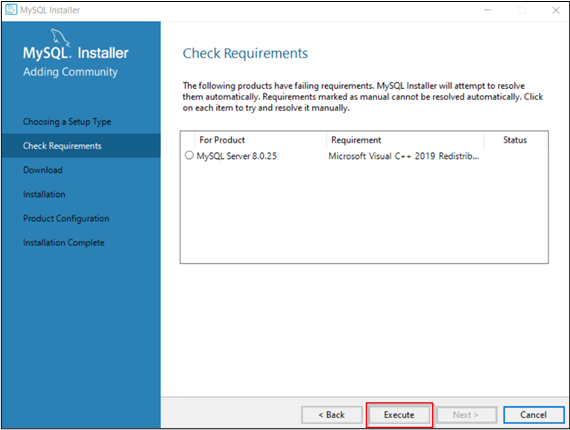
1. Go to Downloads folder and double click on the package you just downloaded



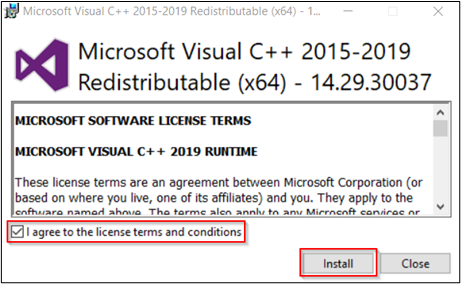
1. Select Server only and click on Next



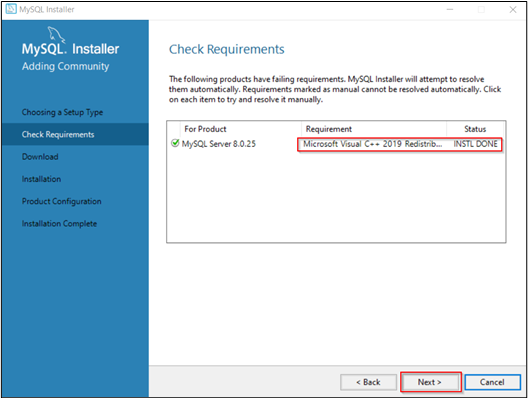
1. Click on Execute



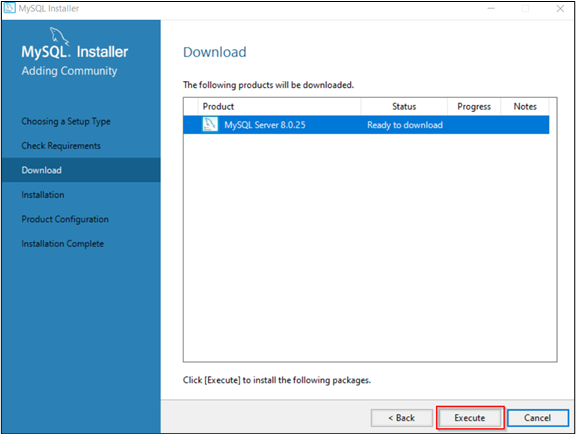
1. If prompted Accept the license agreement and install it or skip to step 9



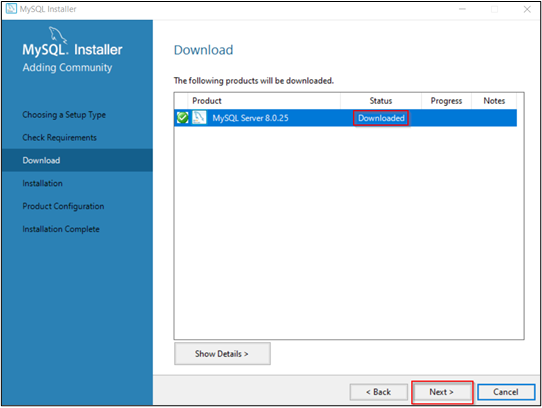
1. Click on Next and Make sure the status of Visual C++ should be Installed and click on Next



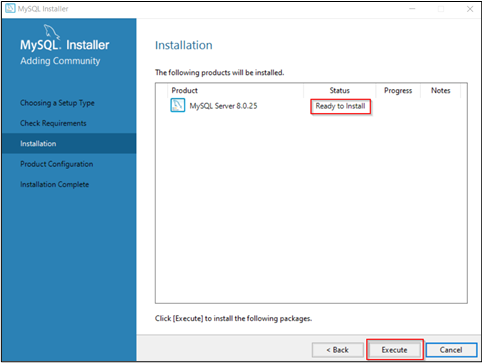
1. Click on Execute



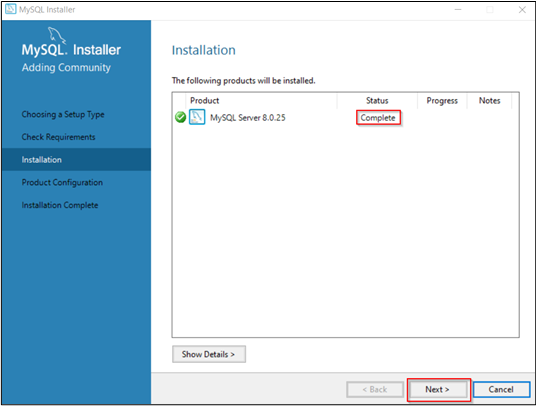
1. Once the status of downloading is labeled as Downloaded, click on Next



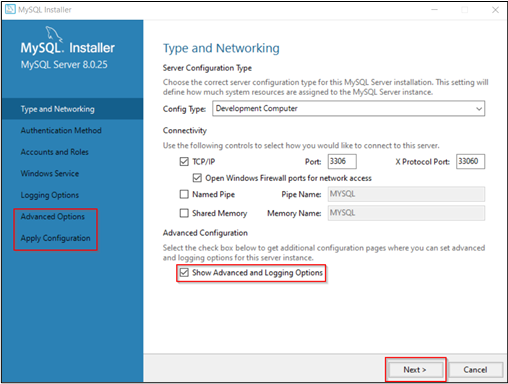
1. Click on Execute



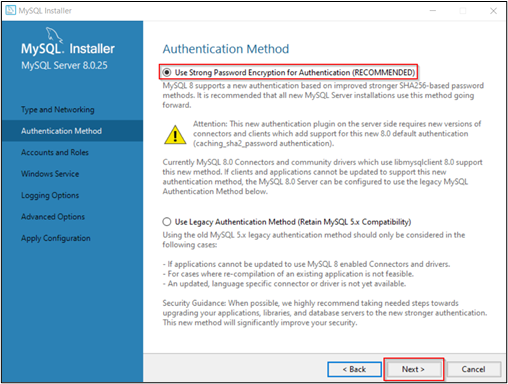
1. Click on Next



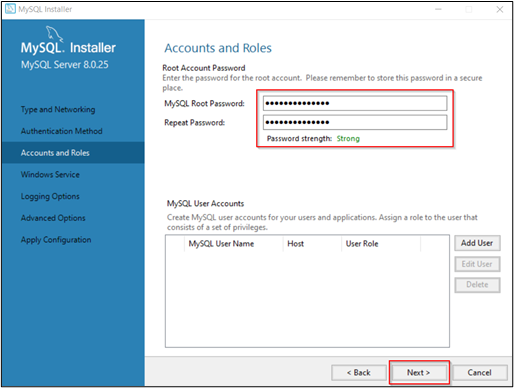
1. Click on “Show Advanced and Logging Options” and click on Next



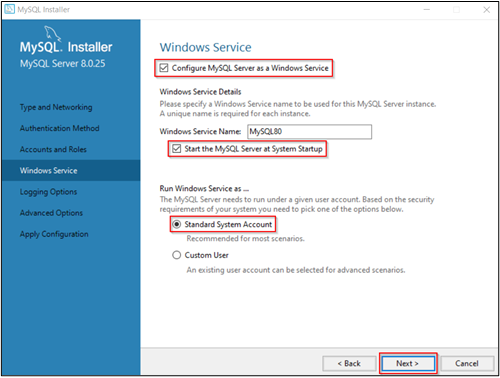
1. In Authentication method, select “Use Strong Password Encryption for Authentication” and click on Next



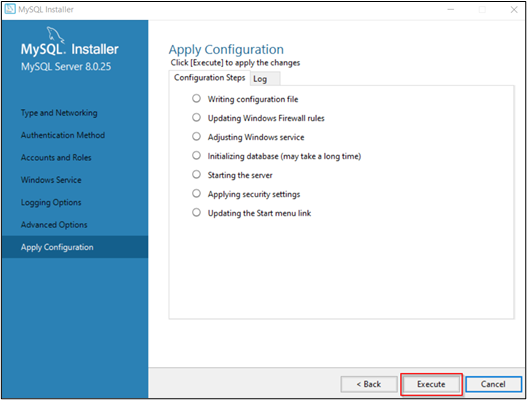
1. Enter a strong password for MySQL root user and Click on Next



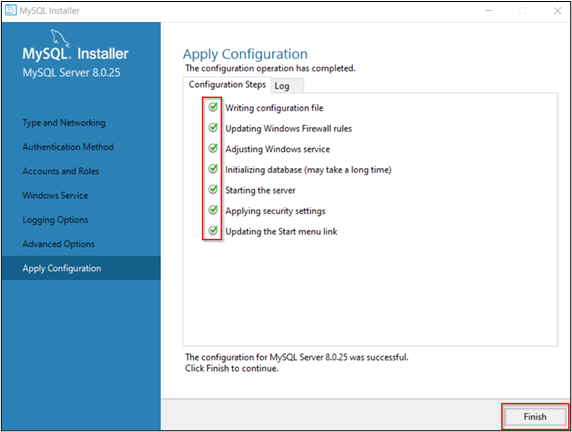
1. Select the following setting and click on Next



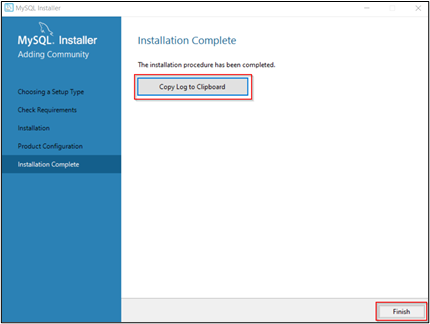
1. Click on Next, Next and Execute the configuration



1. Once configuration is successfully completed, click on Finish



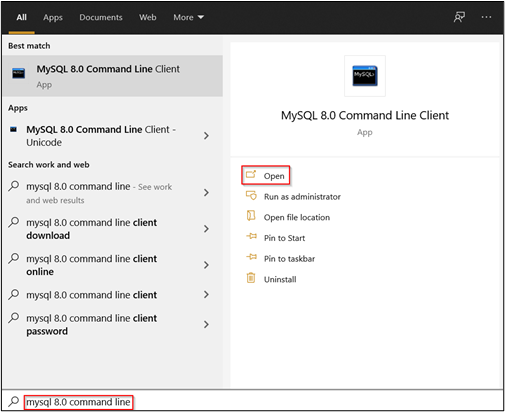
1. Product configuration is completed then click on next
2. Copy the logs to clipboard and save it in a file, Click on Finish



## Lab 2. Structured Query Language

### 1. Open MySQL Command Line

1. Search “mysql 8.0 command line” and click on open



1. Enter the MySQL root password that you are set in Lab-1

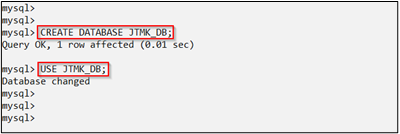
### 2. Creating Database and Tables in MySQL

### 

### Create a database and named it as JTMK\_DB

CREATE DATABASE JTMK\_DB;

USE JTMK\_DB;



1. Create two tables as shown below

#First\_Table

CREATE TABLE DEPARTMENT

(DEPT\_NO CHAR(3) NOT NULL,

DEPT\_NAME VARCHAR(36),

DEPT\_LOCATION CHAR(16),

 PRIMARY KEY (DEPT\_NO));

#Second\_Table

CREATE TABLE EMPLOYEE

(EMP\_NO VARCHAR(10)NOT NULL,

EMP\_FIRSTNAME VARCHAR(15),

 EMP\_LASTNAME VARCHAR(15),

EMP\_PHONENO CHAR(4),

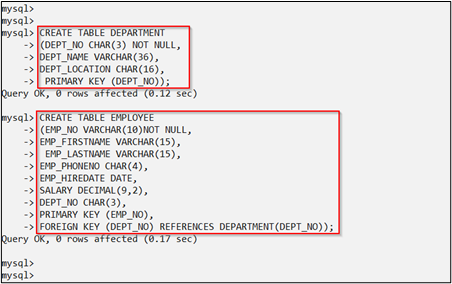
EMP\_HIREDATE DATE,

SALARY DECIMAL(9,2),

DEPT\_NO CHAR(3),

PRIMARY KEY (EMP\_NO),

FOREIGN KEY (DEPT\_NO) REFERENCES DEPARTMENT(DEPT\_NO));

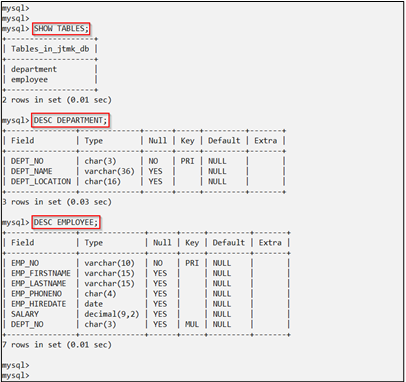


1. Check the tables description that you just created

SHOW TABLES;

DESC DEPARTMENT;

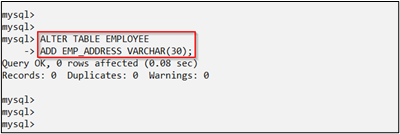
DESC EMPLOYEE;



1. Alter the Employee table

ALTER TABLE EMPLOYEE

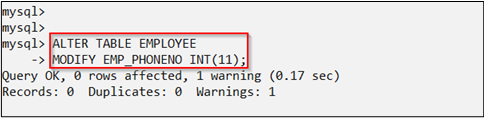
ADD EMP\_ADDRESS VARCHAR(30);



1. Alter the EMPLOYEE table by changing the data type of the column named EMP\_PHONENO CHAR(4) to EMP\_PHONENO INT(11)

ALTER TABLE EMPLOYEE

MODIFY EMP\_PHONENO INT(11);

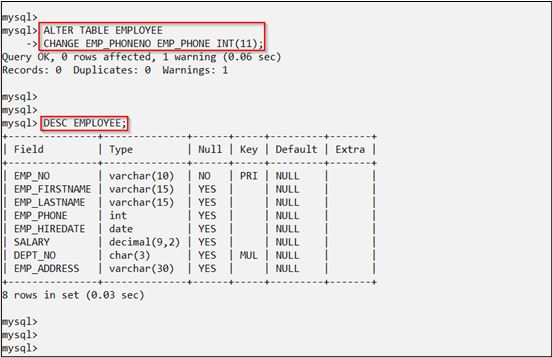


1. Alter the EMPLOYEE table by changing the column name to the new name of column EMP\_PHONENO INT to EMP\_PHONE INT and check the description of the altered table

ALTER TABLE EMPLOYEE

CHANGE EMP\_PHONENO EMP\_PHONE INT(11);

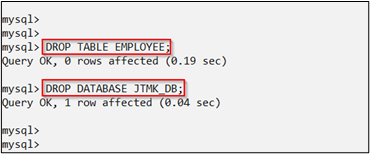
DESC EMPLOYEE;



1. Drop table EMPLOYEE and database JTMK\_DB

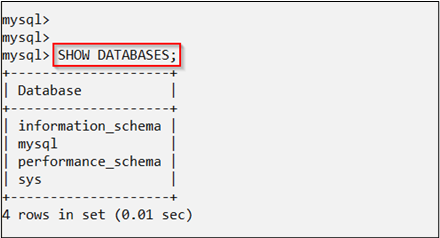
DROP TABLE EMPLOYEE;

DROP DATABASE JTMK\_DB;



1. Check the list of databases available

SHOW DATABASES;



### EXERCISES

A database named <your\_name>\_DB will be developed by an application software house company. There are 6 tables in the database. Relationship scheme for the tables is as below:

Student(**StudID**, StudName,StudAddress,StudBirthDate,CourseID); Course(**CourseID**,CourseName,LectID);

Lecturer(**LectID**, LectName,LectTelNo,DepartID);

 Subject(**SubID**, SubName); Department**(DepartID**, DepartName);

Stud\_Sub (**StudSubID**,StudID,SubID, Mark,Grade)

*(Primary Key showed by underlined and bold while in italic and dotted lined is Foreign Key)*

Data Dictionary for the tables is as below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tables Name** | **Attributes Name** | **Data Types** | **PK/FK** | **FK refer to** |
| **STUDENT** | StudID | VARCHAR(5) | PK |  |
| StudName | VARCHAR(20) |  |  |
| StudAddress | VARCHAR(30) |  |  |
| StudBirthDate | DATE |  |  |
| CourseID | VARCHAR(5) | FK | COURSE |
| **COURSE** | CourseID | VARCHAR(5) | PK |  |
| CourseName | VARCHAR(10) |  |  |
| LectID | VARCHAR(5) | FK | LECTURER |
| **LECTURER** | LectID | VARCHAR(5) | PK |  |
| LectName | VARCHAR(20) |  |  |
| LectTelNo | INT(10) |  |  |
| DepartID | VARCHAR(5) | FK | DEPARTMENT |
| **DEPARTMENT** | DepartID | VARCHAR(5) | PK |  |
| DepartName | VARCHAR(30) |  |  |
| **SUBJECT** | SubID | VARCHAR(5) | PK |  |
| SubName | VARCHAR(20) |  |  |
| **STUD\_SUB** | StudSubID | VARCHAR(5) | PK |  |
| StudID | VARCHAR(5) | FK | STUDENT |
| SubID | VARCHAR(5) | FK | SUBJECT |
| Mark | INT(3) |  |  |
| Grade | CHAR(2) |  |  |

## Lab 3. Structured Query Language

Activity Outcome: Able to apply SQL commands to manipulate data.

EXAMPLES OF DATA MANIPULATION LANGUAGE (DML) COMMANDS

DEPARTMENT

|  |  |  |
| --- | --- | --- |
| DEPT\_NO | DEPT\_NAME | DEPT\_LOCATION |
| D01 | JTMK | CAMPUS A |
| D02 | JKE | CAMPUS A |
| D03 | JPA | CAMPUS B |
| D04 | JP | CAMPUS B |
| D05 | JKM | CAMPUS A |
| D06 | JKA | CAMPUS A |
| D07 | JKP | CAMPUS A |

EMPLOYEE

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| EMP\_NO | EMP\_FIRSTNAME | EMP\_LASTNAME | EMP\_PHONE | EMP\_HIREDATE | SALARY | DEPT\_NO | EMP\_ADDRESS |
| EMP10 00001 | AHMAD | HAKIMI | 0126747087 | 01/01/2011 | 200000.00 | D01 | TAMAN SRI PUTRA,  BANTING |
| EMP10 00002 | JULIA | JOHAN | 0125432313 | 01/04/2011 | 50000.00 | D01 | TAMAN KEMUNING  ,IPOH |
| EMP10 00003 | SETH | COHAN | 0198767898 | 09/02/2012 | 20000.00 | D03 | TAMAN BUDIMAN,  KANGAR |
| EMP10 00004 | RAYMOND | LIM | 0147839477 | 03/06/2012 | 40000.00 | D05 | TAMAN ANGGERIK  ,CHERAS |
| EMP1000005 | HANITH | EMELIO | 0123457689 | 03/06/2012 | 200000.00 | D05 | TAMAN HILLCRES  T,KL |
| EMP10 00006 | SATHINI | RAJU | 0187992873 | 03/06/2012 | 700000.00 | D02 | TAMAN JAYA,  IPOH |

#### INSERT INTO Statement

1. The first form does not specify the column names where the data will be inserted,

only their values:

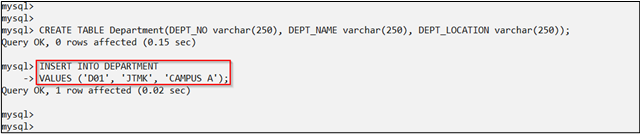
INSERT INTO table\_name

VALUES (value1, value2, value3,);

Example(Refer Lab 2):

INSERT INTO DEPARTMENT

VALUES ('D01', 'JTMK', 'CAMPUS A');



1. The second form specifies both the column names and the values to be inserted:

INSERT INTO table\_name (column1, column2, column3,)

VALUES (value1, value2, value3,);

Example (Refer Lab 2):

INSERT INTO EMPLOYEE (EMP\_NO,EMP\_FIRSTNAME,EMP\_LASTNAME,EMP\_PHONE,EMP\_HIREDATE,SALA RY,DEPT\_NO)VALUES ('EMP1000001', 'AHMAD', 'HAKIMI',0126747087, '2011/01/01',200000.00, 'D01');



#### Insert all the data below by using the pervious exercise on Lab 2

STUDENT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| StudID | StudName | StudAddress | StudBirthDate | CourseID |
| IT101 | Aminah | Perak | 19/10/1990 | C1000 |
| IT102 | Joe | Selangor | 13/08/1990 | C1000 |
| IT103 | Mona | Perak | 23/07/1991 | C1000 |
| IT104 | Husna | Melaka | 09/05/1990 | C1002 |
| IT105 | Lingling | Melaka | 07/07/1989 | C1004 |
| IT106 | Maniam | Penang | 01/12/1989 | C1003 |

COURSE

|  |  |  |
| --- | --- | --- |
| CourseID | CourseName | LectID |
| C1000 | Software Development | L1001 |
| C1001 | Artificial Intelegence | L1003 |
| C1002 | Database | L1002 |
| C1003 | Interactive Multimedia | L1010 |
| C1004 | Networking | L1005 |

DEPARTMENT

|  |  |
| --- | --- |
| DepartID | DepartName |
| D1000 | Software Engineering |
| D1001 | Industrial Computing |
| D1002 | Comp System & Comm |
| D1003 | Interactive Multimedia |
| D1004 | Management |

LECTURER

|  |  |  |  |
| --- | --- | --- | --- |
| LectID | LectName | LectTelNo | DepartID |
| L1001 | Mokhsin | 0178908767 | D1000 |
| L1002 | Sheima | 0192299090 | D1000 |
| L1003 | Raju | 0101989909 | D1001 |
| L1004 | Zulkifli | 0123984902 | D1000 |
| L1005 | Rahmad | 0145667389 | D1002 |
| L1006 | Naina | 0143662738 | D1001 |
| L1007 | Junaida | 0135557728 |  |
| L1008 | Hamid |  | D1004 |
| L1009 | Chong |  |  |
| L1010 | Sam |  | D1003 |

SUBJECT

|  |  |
| --- | --- |
| SubID | SubName |
| SC100 | Programming Technique |
| SC101 | System Development |
| SC102 | Database |
| SC103 | Object Oriented |
| SC104 | Multimedia System |
| SC105 | Logic Programming |

STUD\_SUB

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| StudSubID | StudID | SubID | Mark | Grade |
| SS100 | IT101 | SC100 | 60 | B |
| SS101 | IT102 | SC100 | 85 | A |
| SS102 | IT103 | SC100 | 90 | A |
| SS103 | IT104 | SC103 | 76 | B |
| SS104 | IT105 | SC103 | 56 | C |
| SS105 | IT106 | SC104 | 65 | B |
| SS106 | IT101 | SC104 | 45 | D |
| SS107 | IT102 | SC104 | 98 | A |
| SS108 | IT103 | SC101 | 67 | B |
| SS109 | IT104 | SC101 | 54 | C |
| SS110 | IT105 | SC101 | 87 | A |
| SS111 | IT106 | SC101 |  | A |

1. UPDATE Statement

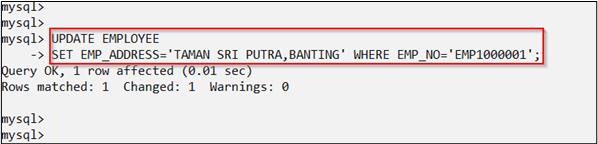
UPDATE table\_name

SET column1=value, column2=value2, WHERE some\_column=some\_value;

Example (REFER LAB 2):

UPDATE EMPLOYEE

SET EMP\_ADDRESS='TAMAN SRI PUTRA,BANTING' WHERE EMP\_NO='EMP1000001';



1. DELETE Statement

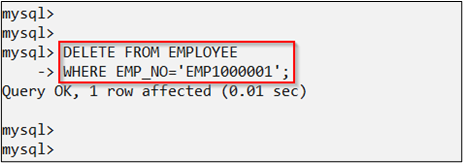
DELETE FROM table\_name

WHERE some\_column=some\_value;

Example (REFER LAB 2):

DELETE FROM EMPLOYEE

WHERE EMP\_NO='EMP1000001';



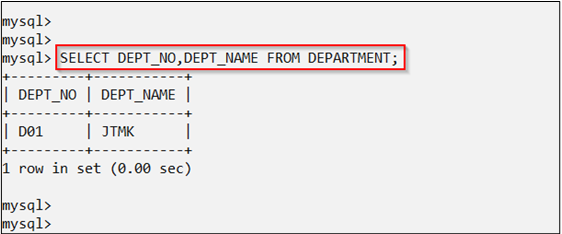
1. SELECTING DATA Statement
2. Select specific column:

SELECT column\_name(s)

FROM table\_name;

Example (REFER LAB 2):

SELECT DEPT\_NO,DEPT\_NAME FROM DEPARTMENT;

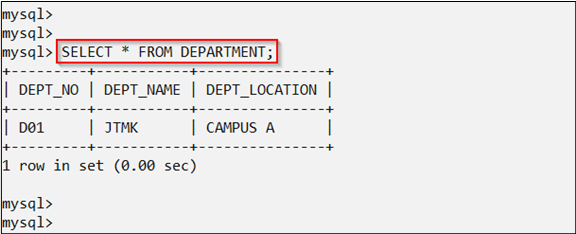


1. Select all columns:

SELECT \* FROM table\_name;

Example (REFER LAB 2):

SELECT \* FROM DEPARTMENT;



1. SELECT-FROM-WHERE Statement

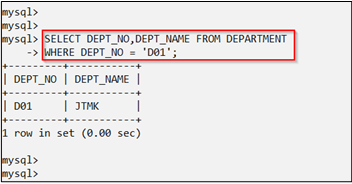
SELECT column\_name(s) FROM table\_name

WHERE column\_name operator value;

Example (REFER LAB 2)

SELECT DEPT\_NO,DEPT\_NAME FROM DEPARTMENT

WHERE DEPT\_NO = 'D01';



Comparison Operators:

|  |  |  |
| --- | --- | --- |
| Operator | Purpose | Example |
| = | Equality test. | SELECT \* FROM EMPLOYEE WHERE EMP\_NO ='EMP1000001'; |
| <>  *!=*  *!>*  *!<* | Not Equal To  *Not Equal To (not ISO standard)*  *Not Greater Than (not ISO standard)*  *Not Less Than (not ISO standard)* | SELECT \* FROM EMPLOYEE WHERE DEPT\_NO <> 'D01'; |
| > | Greater than | SELECT EMP\_NAME,DEPT\_NO FROM EMPLOYEE WHERE EMP\_SALARY > 1000000.00 |
| < | Less than | SELECT EMP\_NAME,DEPT\_NO FROM EMPLOYEE WHERE EMP\_SALARY < 1000000.00 |
| >= | Greater Than or equal to | SELECT EMP\_NAME,DEPT\_NO FROM EMPLOYEE WHERE EMP\_SALARY >= 1000000.00 |
| <= | Less than or equal to | SELECT EMP\_NAME,DEPT\_NO FROM EMPLOYEE WHERE EMP\_SALARY <= 1000000.00 |

1. LIKE

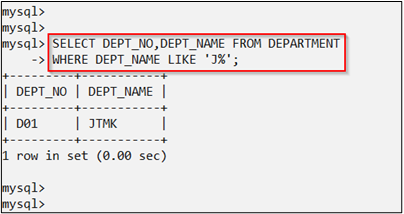
SELECT column\_name(s) FROM table\_name

WHERE column\_name LIKE pattern;

Example (REFER  LAB 2):

SELECT DEPT\_NO,DEPT\_NAME FROM DEPARTMENT

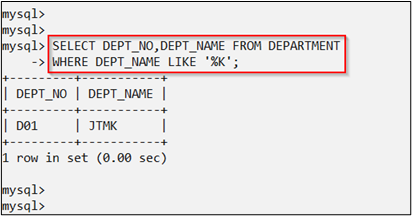
WHERE DEPT\_NAME LIKE 'J%';



This SQL statement will match any department name that start with 'J'. Strings must be in single quotes. Or you can specify,

SELECT DEPT\_NO,DEPT\_NAME FROM DEPARTMENT

WHERE DEPT\_NAME LIKE '%K';



This statement will match any last names that end in a 'K'

1. ORDER BY

SELECT column\_name(s) FROM table\_name

ORDER BY column\_name(s) ASC|DESC;

Example (REFER LAB 2):

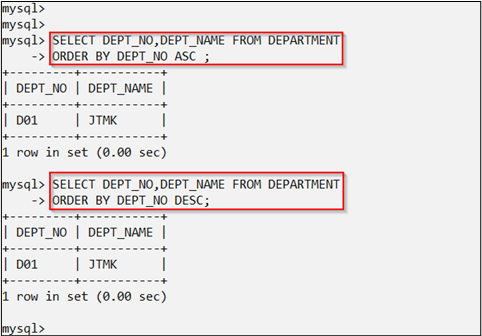
SELECT DEPT\_NO,DEPT\_NAME FROM DEPARTMENT

ORDER BY DEPT\_NO ASC ;

(OR)

SELECT DEPT\_NO,DEPT\_NAME FROM DEPARTMENT

ORDER BY DEPT\_NO DESC;



1. IN

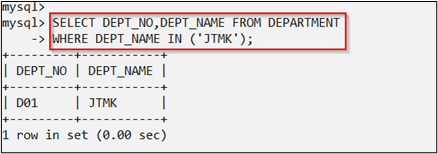
SELECT column\_name(s) FROM table\_name

WHERE column\_name IN (value1,value2,);

Example (REFER LAB 2):

SELECT DEPT\_NO,DEPT\_NAME FROM DEPARTMENT

WHERE DEPT\_NAME IN ('JTMK');



1. DISTINCT

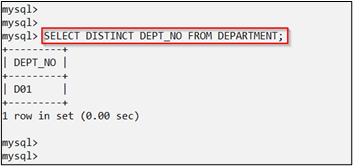
SELECT DISTINCT column\_name(s)

FROM table\_name;

Example (REFER LAB 2):

SELECT DISTINCT DEPT\_NO

FROM DEPARTMENT;



1. AND Operator

The AND operator displays a record if both the first condition and the second condition are true

Example (REFER LAB 2):

SELECT \* FROM EMPLOYEE

WHERE EMP\_FIRSTNAME ='AHMAD' AND EMP\_LASTNAME='HAKIMI';



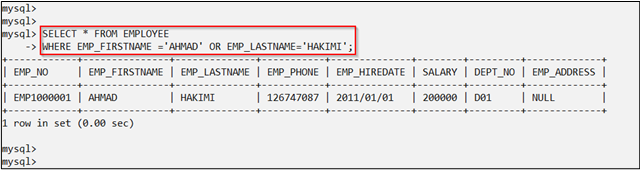
1. OR Operator

The OR operator displays a record if either the first condition or the second condition is true

Example (REFER LAB 2):

SELECT \* FROM EMPLOYEE

WHERE EMP\_FIRSTNAME ='AHMAD' OR EMP\_LASTNAME='HAKIMI';



SQL JOINS

1. JOIN:

Return rows when there is at least one match in both tables

SELECT column\_name(s) FROM table\_name1

INNER JOIN table\_name2

ON table\_name1.column\_name=table\_name2.column\_name;

**PS:**INNER JOIN is the same as JOIN

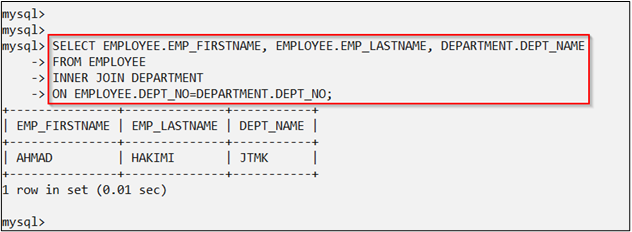
Example (REFER LAB 2):

SELECT EMPLOYEE.EMP\_FIRSTNAME, EMPLOYEE.EMP\_LASTNAME, DEPARTMENT.DEPT\_NAME

FROM EMPLOYEE

INNER JOIN DEPARTMENT

ON EMPLOYEE.DEPT\_NO=DEPARTMENT.DEPT\_NO;



1. LEFT JOIN:

Return all rows from the left table, even if there are no matches in the right table

SELECT column\_name(s) FROM table\_name1

LEFT JOIN table\_name2

ON table\_name1.column\_name=table\_name2.column\_name;

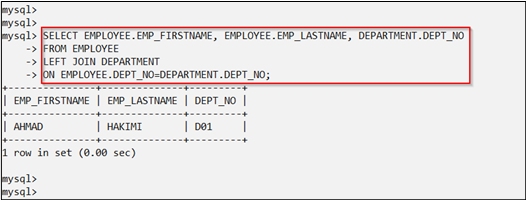
Example (REFER  LAB 2):

SELECT EMPLOYEE.EMP\_FIRSTNAME, EMPLOYEE.EMP\_LASTNAME, DEPARTMENT.DEPT\_NO

FROM EMPLOYEE

LEFT JOIN DEPARTMENT

ON EMPLOYEE.DEPT\_NO=DEPARTMENT.DEPT\_NO;



1. RIGHT JOIN:

 Return all rows from the right table, even if there are no matches in the left table

SELECT column\_name(s) FROM table\_name1

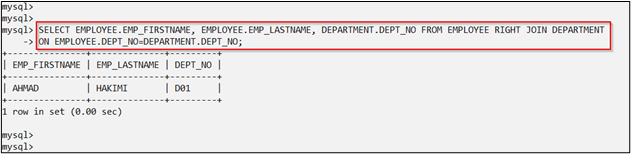
RIGHT JOIN table\_name2

ON table\_name1.column\_name=table\_name2.column\_name;

Example (REFER LAB 2):

SELECT EMPLOYEE.EMP\_FIRSTNAME, EMPLOYEE.EMP\_LASTNAME, DEPARTMENT.DEPT\_NO FROM EMPLOYEE RIGHT JOIN DEPARTMENT

ON EMPLOYEE.DEPT\_NO=DEPARTMENT.DEPT\_NO;



1. FULL JOIN:

Return rows when there is a match in one of the tables

SELECT column\_name(s)

 FROM table\_name1

FULL JOIN table\_name2

ON table\_name1.column\_name=table\_name2.column\_name;

Example (REFER LAB 2):

SELECT EMPLOYEE.EMP\_FIRSTNAME, EMPLOYEE.EMP\_LASTNAME, DEPARTMENT.DEPT\_NO

FROM EMPLOYEE

FULL JOIN DEPARTMENT

ON EMPLOYEE.DEPT\_NO=DEPARTMENT.DEPT\_NO;

Full join does not work in SQL server 5 instead the following query will work

SELECT EMPLOYEE.EMP\_FIRSTNAME, EMPLOYEE.EMP\_LASTNAME, DEPARTMENT.DEPT\_NO

FROM EMPLOYEE

LEFT JOIN DEPARTMENT

ON EMPLOYEE.DEPT\_NO=DEPARTMENT.DEPT\_NO;

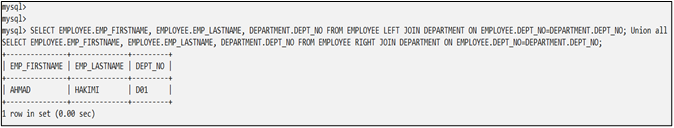
Union all

SELECT EMPLOYEE.EMP\_FIRSTNAME, EMPLOYEE.EMP\_LASTNAME, DEPARTMENT.DEPT\_NO

FROM EMPLOYEE

RIGHT JOIN DEPARTMENT

ON EMPLOYEE.DEPT\_NO=DEPARTMENT.DEPT\_NO;

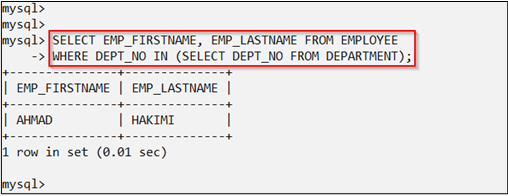


1. NESTED QUERIES:

Return rows when there is a match in one of the tables

SELECT EMP\_FIRSTNAME, EMP\_LASTNAME FROM EMPLOYEE

WHERE DEPT\_NO IN (SELECT DEPT\_NO FROM DEPARTMENT);



Exercise:

Refer to the tables data “**1. INSERT INTO Statement**”, answer all the below questions:-

1. Update the data in Course table from Database-to-Database Management  where CourseId is C1002
2. Delete "Logic Programming" from Subject table
3. Select only the Student from “Perak”
4. Select the Lecturer who phone num start with “014”
5. Show the OUTPUT for SQL syntax below:

Select S.StudName,C.CourseName From Student S, Course C;

Where S.CourseID=C.CourseID;

1. Show the OUTPUT for SQL syntax below:

Select S.StudName,C.CourseName,L.LectName From Student S, Course C ,Lecturer L;

Where S.CourseID=C.CourseID and C.LectID=L.LectID;

1. Select student name, subject name, grade and mark, which mark >=55
2. Write SQL syntax to show a sum of student mark for each subject
3. Write SQL syntax to show lecturer name, department name where the lecturer is from Software Engineering department