Date: Roll No.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 4 | B | 1 | 1 | A | I | 2 | 7 | 2 |

# AIM: 1.1 Familiarization with installation of any DBMS.

## Description:

### Introduction to MySQL on Windows and Installation

**a. Overview of MySQL Edition:**

MySQL is a database engine that provides fast, secure and scalable solutions for both small and large applications. MySQL is known for its reliability, scalability and ease of use, making it a popular choice for web developers and organizations worldwide. By installing MySQL on your Windows machine, you can:

- Store and manage data for various types of applications (websites, software, business systems).
- Use advanced SQL features for querying, filtering, and modifying data.
- Integrate with popular programming languages such as PHP, Python, Java, and Node.js.
- Scale your applications seamlessly, handling thousands of transactions and concurrent users.

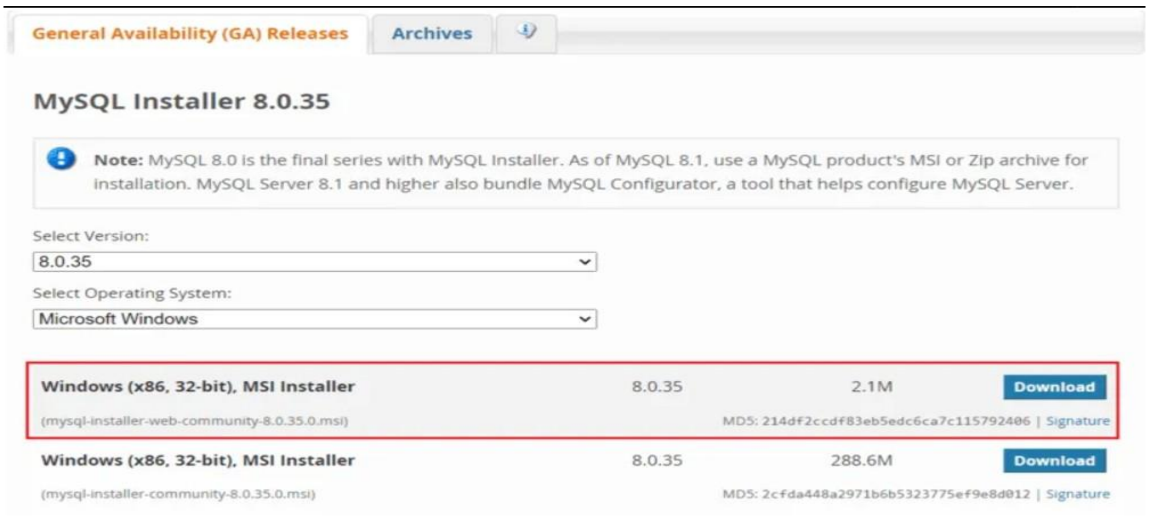
**SQL Command Line**: A simple command-line interface called "SQL Command Line" was included for direct SQL execution and basic administration tasks.

**b.** Installation Steps:

Now, Let's break down MySQL software downloading steps for a better understanding and see install MySQL on Windows 10 step by step.

### Step 1: Visit the Official MySQL Website

Open your preferred web browser and navigate to the official MySQL website. Now, Simple click on first download button.



MySQL Installer 8.0.35

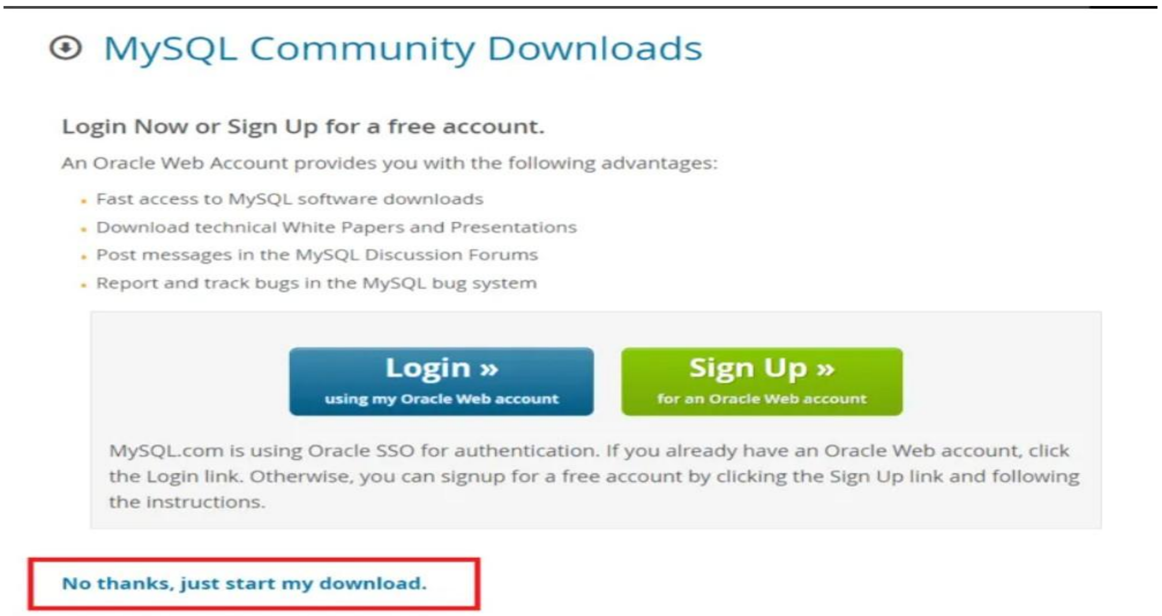
Note: MySQL 8.0 is the final series with MySQL Installer. As of MySQL 8.1, use a MySQL product's MSI or Zip archive for installation. MySQL Server 8.1 and higher also bundle MySQL Configurator, a tool that helps configure MySQL Server.

Select Version: 8.0.35

Select Operating System: Microsoft Windows

| | | | |
|---|---|---|---|
| Windows (x86, 32-bit), MSI Installer (mysql-installer-web-community-8.0.35.0.msi) | 8.0.35 | 2.1M | Download<br>MD5: 214df2ccdf83eb5edc6ca7c115792406 \| Signature |
| Windows (x86, 32-bit), MSI Installer (mysql-installer-community-8.0.35.0.msi) | 8.0.35 | 288.6M | Download<br>MD5: 2cfda448a2971b6b5323775ef9e8d012 \| Signature |

**Step 2:**
**Go to the Downloads Section** On the MySQL homepage, Click on the " **No thanks, just start my download**" link to proceed MySql downloading.


MySQL Community Downloads

Login Now or Sign Up for a free account.

An Oracle Web Account provides you with the following advantages:

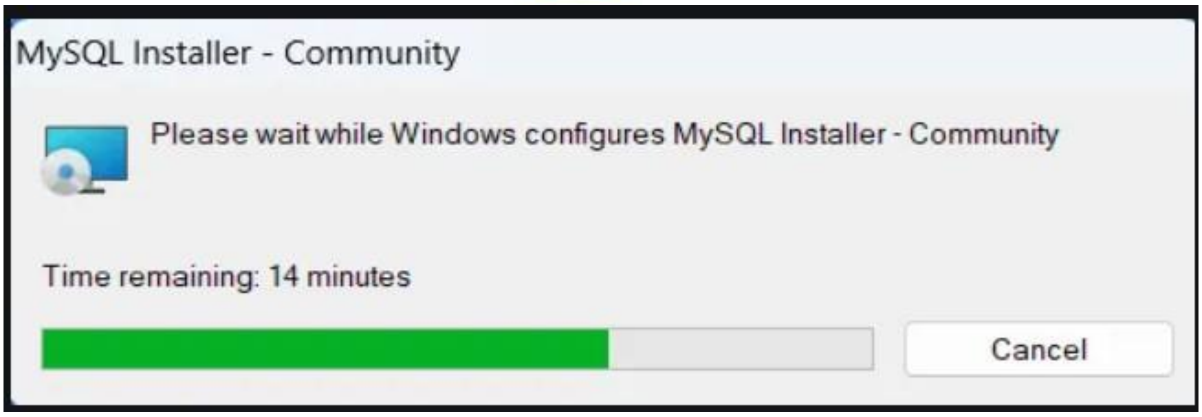
- Fast access to MySQL software downloads
- Download technical White Papers and Presentations
- Post messages in the MySQL Discussion Forums
- Report and track bugs in the MySQL bug system

Login » using my Oracle Web account
Sign Up » for an Oracle Web account

MySQL.com is using Oracle SSO for authentication. If you already have an Oracle Web account, click the Login link. Otherwise, you can signup for a free account by clicking the Sign Up link and following the instructions.

No thanks, just start my download.

**Step 3: Run the Installer**
After MySQL downloading **MySQL.exe** file , go to your Downloads folder, find the file, and double-click to run the installer.

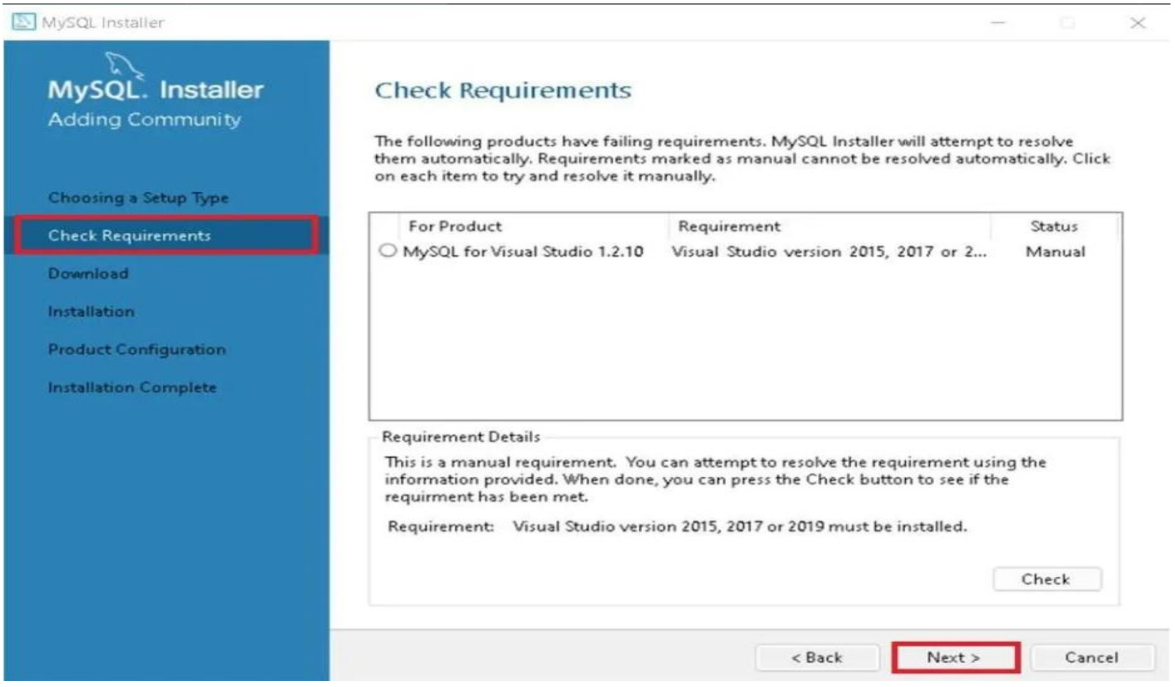

MySQL Installer - Community

Please wait while Windows configures MySQL Installer - Community

Time remaining: 14 minutes

**Step 4: Choose Setup Type**
The installer will instruct you to choose the setup type. For most users, the "**Developer Default**" is suitable. Click "Next" to proceed.

**Step 5: Check Requirements**
You might be prompted to install necessary MySQL software, typically **Visual Code**. The installer can auto-resolve some issues, but not in this case.


MySQL Installer

MySQL. Installer
Adding Community

Choosing a Setup Type
Check Requirements
Download
Installation
Product Configuration
Installation Complete

Check Requirements

The following products have failing requirements. MySQL Installer will attempt to resolve them automatically. Requirements marked as manual cannot be resolved automatically. Click on each item to try and resolve it manually.

| For Product | Requirement | Status |
|---|---|---|
| MySQL for Visual Studio 1.2.10 | Visual Studio version 2015, 2017 or 2... | Manual |

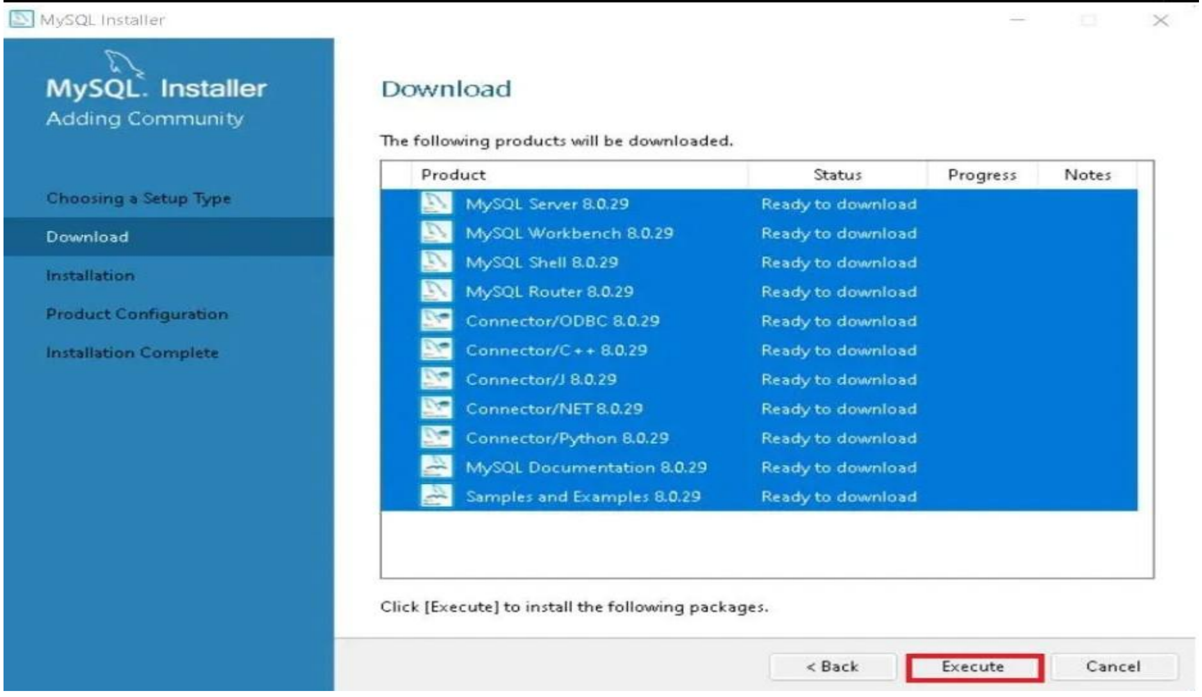
Requirement Details

This is a manual requirement. You can attempt to resolve the requirement using the information provided. When done, you can press the Check button to see if the requirment has been met.

Requirement: Visual Studio version 2015, 2017 or 2019 must be installed.

## Step 6: MySQL Downloading

Now that you're in the download section, click "**Execute**" to start downloading the components you selected. Wait a few minutes until all items show tick marks, indicating completion, before moving forward.


MySQL Installer

MySQL. Installer
Adding Community

Choosing a Setup Type
Download
Installation
Product Configuration
Installation Complete

Download

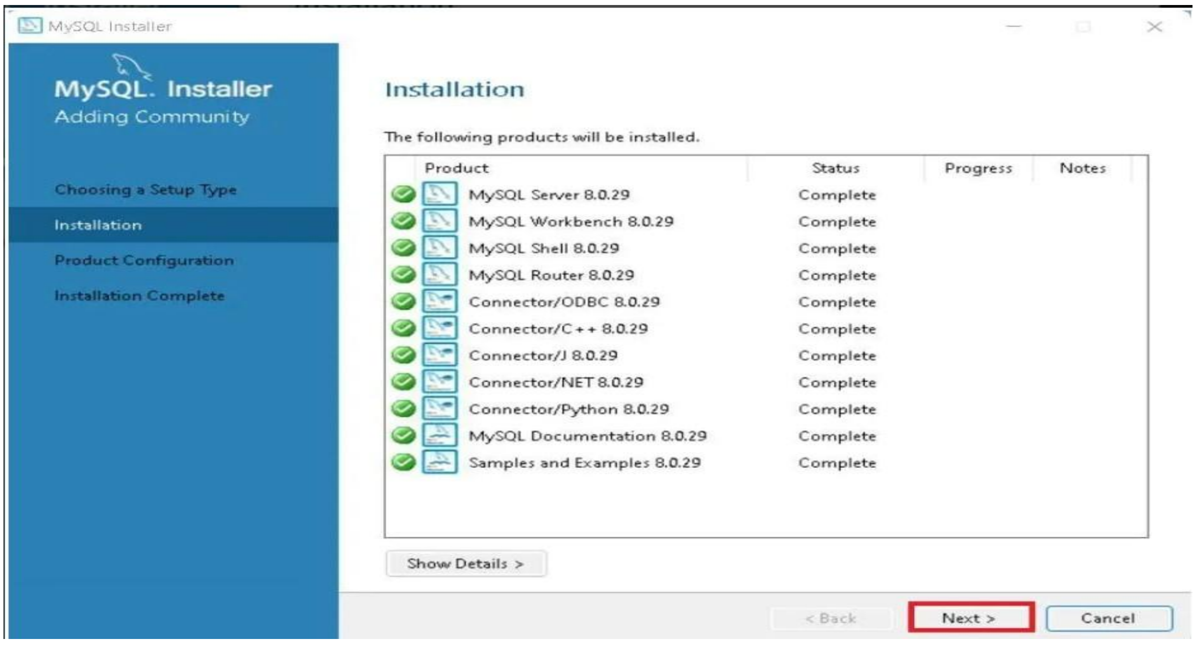
The following products will be downloaded.

| Product | Status | Progress | Notes |
|---|---|---|---|
| MySQL Server 8.0.29 | Ready to download | | |
| MySQL Workbench 8.0.29 | Ready to download | | |
| MySQL Shell 8.0.29 | Ready to download | | |
| MySQL Router 8.0.29 | Ready to download | | |
| Connector/ODBC 8.0.29 | Ready to download | | |
| Connector/C++ 8.0.29 | Ready to download | | |
| Connector/J 8.0.29 | Ready to download | | |
| Connector/NET 8.0.29 | Ready to download | | |
| Connector/Python 8.0.29 | Ready to download | | |
| MySQL Documentation 8.0.29 | Ready to download | | |
| Samples and Examples 8.0.29 | Ready to download | | |

Click [Execute] to install the following packages.

## Step 7: MySqL Installation

Now the downloaded components will be installed. Click "**Execute**" to start the installation process. MySQL will be installed on your Windows system. Then click **Next** to proceed.



MySQL Installer

MySQL. Installer
Adding Community

Choosing a Setup Type
Installation
Product Configuration
Installation Complete

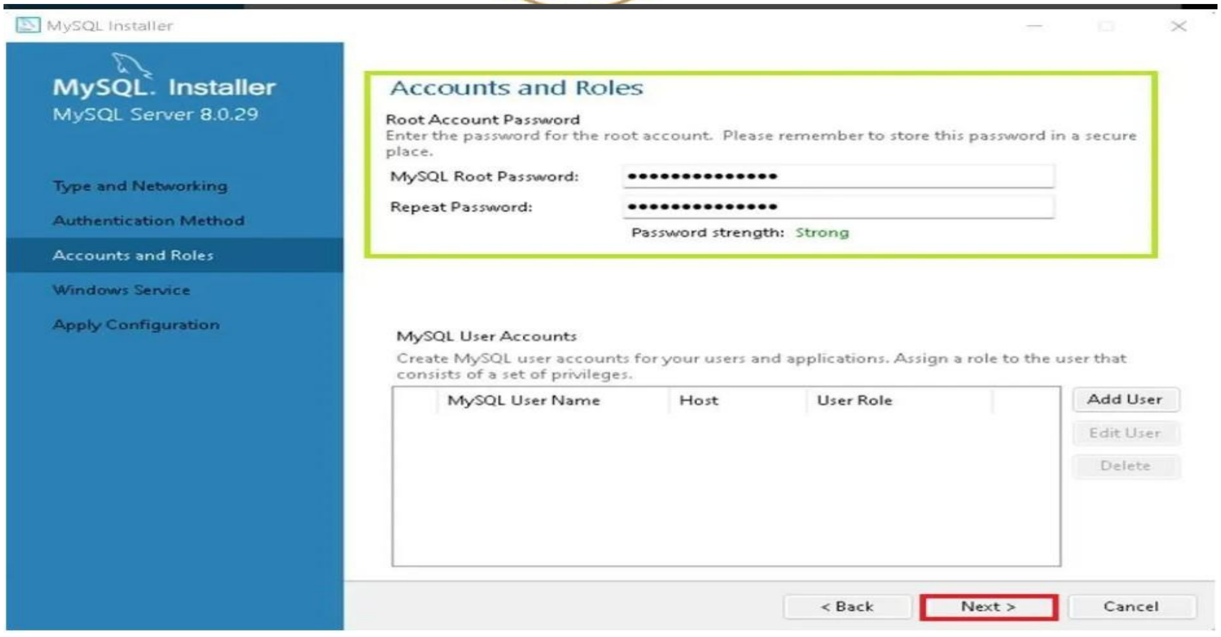
Installation

The following products will be installed.

| Product | Status | Progress | Notes |
|---|---|---|---|
| MySQL Server 8.0.29 | Complete | | |
| MySQL Workbench 8.0.29 | Complete | | |
| MySQL Shell 8.0.29 | Complete | | |
| MySQL Router 8.0.29 | Complete | | |
| Connector/ODBC 8.0.29 | Complete | | |
| Connector/C++ 8.0.29 | Complete | | |
| Connector/J 8.0.29 | Complete | | |
| Connector/NET 8.0.29 | Complete | | |
| Connector/Python 8.0.29 | Complete | | |
| MySQL Documentation 8.0.29 | Complete | | |
| Samples and Examples 8.0.29 | Complete | | |

**Step 8: Navigate to Few Configuration Pages**
Proceed to "**Product Configuration**" > "**Type and Networking**" > "**Authentication Method**" Pages by clicking the "Next" button.

**Step 9: Create MySQL Accounts**
Create a password for the MySQL root user. Ensure it's strong and memorable. Click "**Next**" to proceed.



MySQL Installer

MySQL. Installer
MySQL Server 8.0.29

Type and Networking
Authentication Method
Accounts and Roles
Windows Service
Apply Configuration

Accounts and Roles

Root Account Password
Enter the password for the root account. Please remember to store this password in a secure place.

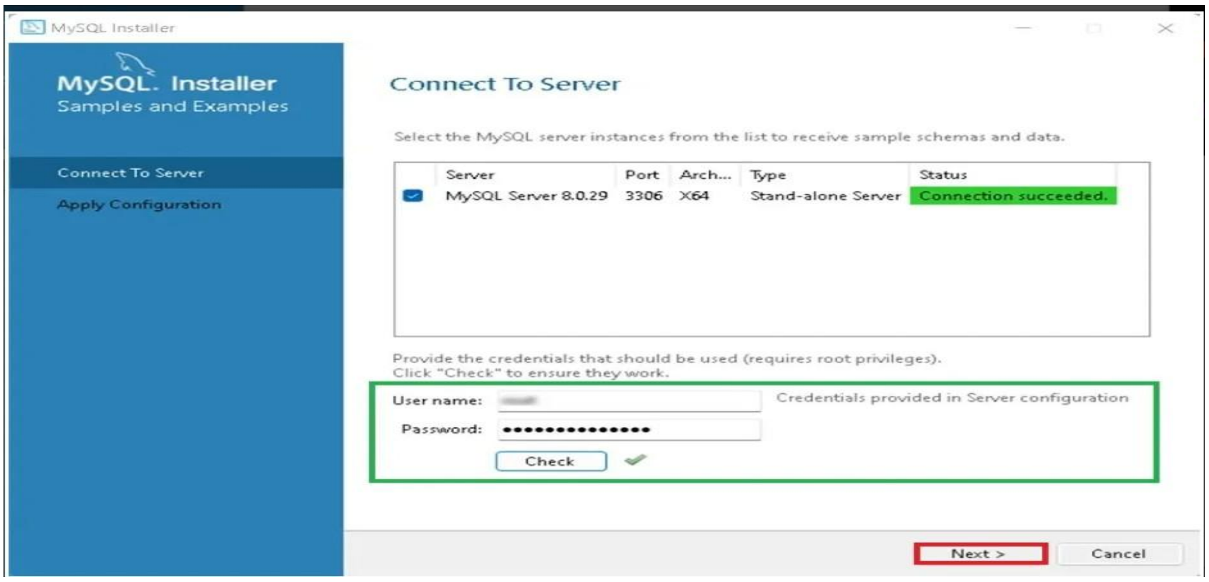
MySQL Root Password: •••••••••••••••
Repeat Password: •••••••••••••••
Password strength: Strong

MySQL User Accounts
Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

| MySQL User Name | Host | User Role |
|---|---|---|

**Step 10: Connect To Server**

Enter the root password, click Check. If it says "**Connection succeed,**" you have successfully connected to the server.


MySQL Installer

MySQL. Installer
Samples and Examples

Connect To Server
Apply Configuration

Connect To Server

Select the MySQL server instances from the list to receive sample schemas and data.

| | Server | Port | Arch... | Type | Status |
|---|---|---|---|---|---|
| ✓ | MySQL Server 8.0.29 | 3306 | X64 | Stand-alone Server | Connection succeeded. |

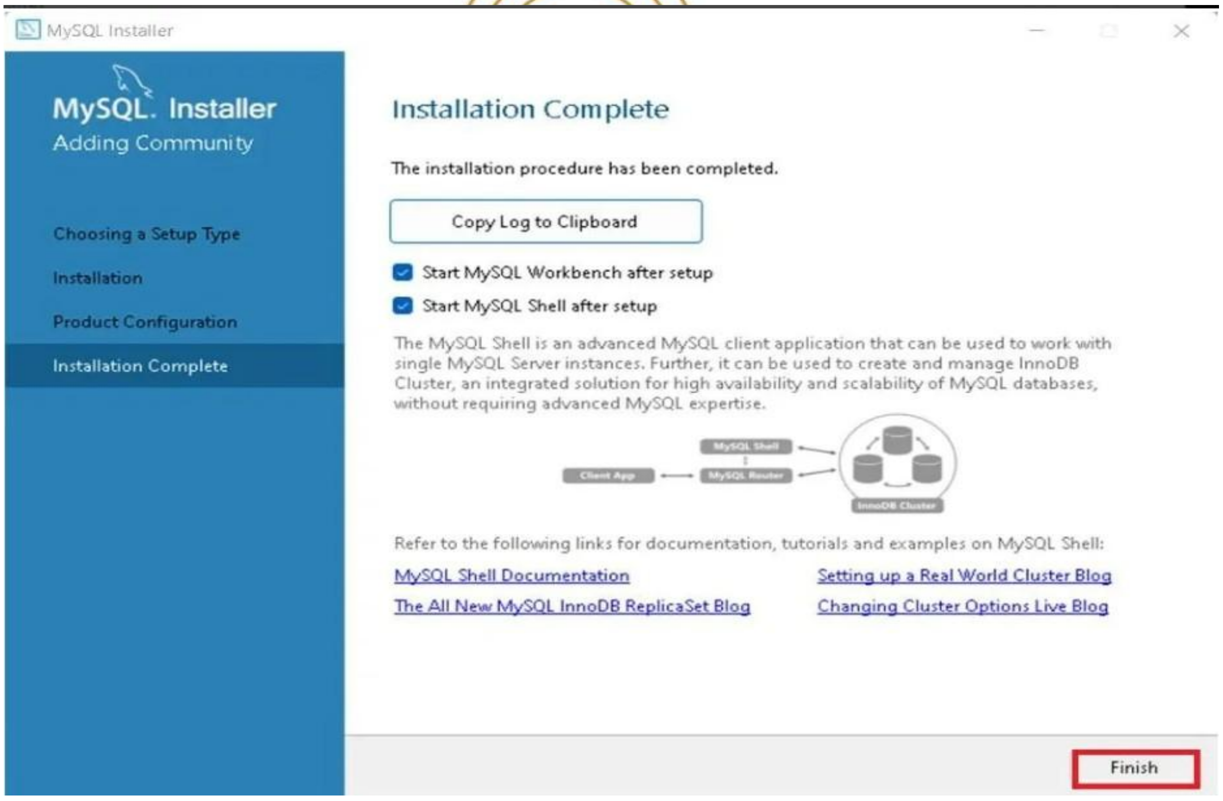
Provide the credentials that should be used (requires root privileges).
Click "Check" to ensure they work.

User name: [redacted]
Password: ••••••••••••••••
Check ✓

Credentials provided in Server configuration

Next > Cancel

**Step 11: Complete Installation**
Once the installation is complete, click "**Finish**." Congratulations! MySQL is now installed on your Windows system.


MySQL Installer

MySQL. Installer
Adding Community

Choosing a Setup Type
Installation
Product Configuration
Installation Complete

Installation Complete

The installation procedure has been completed.

Copy Log to Clipboard

✓ Start MySQL Workbench after setup
✓ Start MySQL Shell after setup

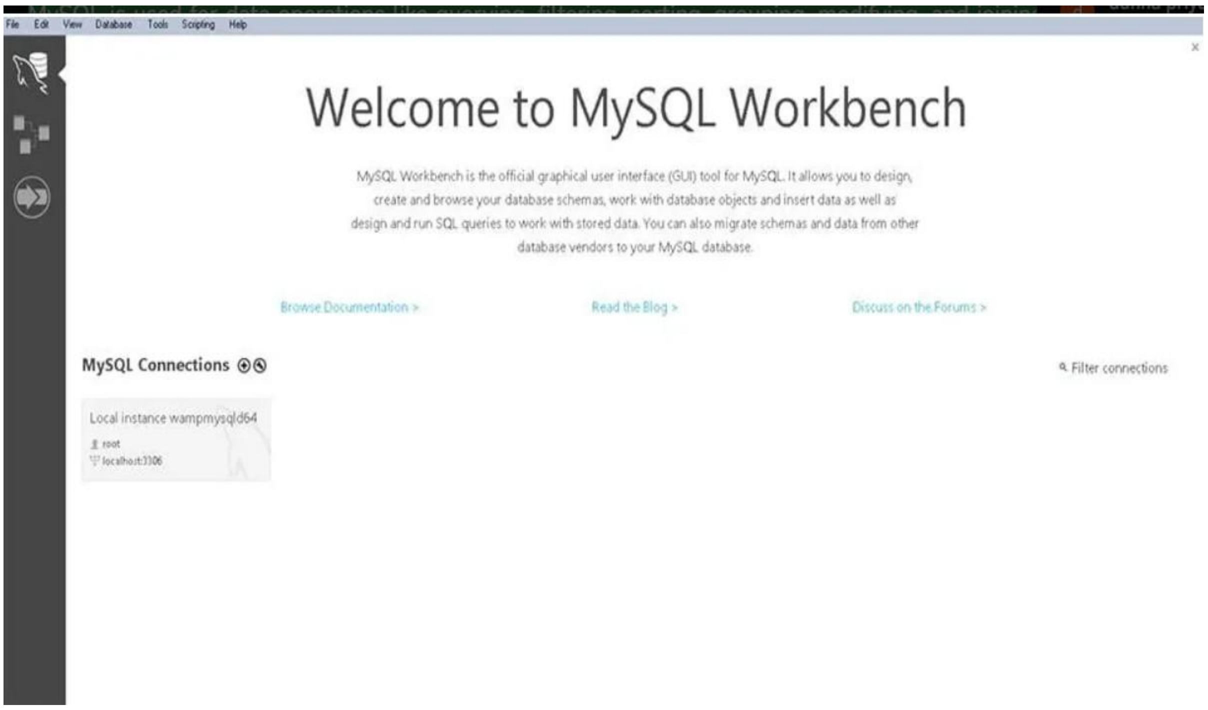
The MySQL Shell is an advanced MySQL client application that can be used to work with single MySQL Server instances. Further, it can be used to create and manage InnoDB Cluster, an integrated solution for high availability and scalability of MySQL databases, without requiring advanced MySQL expertise.

Refer to the following links for documentation, tutorials and examples on MySQL Shell:
MySQL Shell Documentation
Setting up a Real World Cluster Blog
The All New MySQL InnoDB ReplicaSet Blog
Changing Cluster Options Live Blog

Finish

**Step 12: Verify Installation**

To ensure a successful installation of MySQL, open the MySQL Command Line Client or MySQL Workbench, both available in your Start Menu. Log in using the root user credentials you set during installation.



Welcome to MySQL Workbench

MySQL Workbench is the official graphical user interface (GUI) tool for MySQL. It allows you to design, create and browse your database schemas, work with database objects and insert data as well as design and run SQL queries to work with stored data. You can also migrate schemas and data from other database vendors to your MySQL database.

**MySQL Workbench Is Ready To Use**

MySQL is an open-source relational database management system that is based on SQL queries. MySQL is used for data operations like querying, filtering, sorting, grouping, modifying, and joining the tables present in the database.

Date: Roll No. | 2 | 4 | B | 1 | 1 | A | I | 2 | 7 | 2 |

# AIM: 1.2 Implementing a University Database System.

## Description:

### a. Schema

It is overall design of the database. It tells how many tables are there, what are their attributes, and how they are related. It includes table names, column names(attributes), datatypes, and keys.

Syntax of Schema:

Table Name (column1 : datatype, column2: datatype, column3: datatype, ……., columnN: datype)

### b. Attributes per Table

- **Students Table**

Attributes: StudentID, StudentName, Major

- Courses Table

Attributes: CourseID, CourseName, Credits

- Enrollments Table

Attributes: StudentID, CourseID, EnrollmentDate

- Instructors Table

Attributes: InstructorID, InstructorName, Phone

- Course_Instructors Table

Attributes: CourseID, InstructorID

### c. Schema for University Database:

Students (StudentID:string, StudentName:string, Major:string) ;

Courses (CourseID:string, CourseName:string, Credits:integer) ;

Enrollments (StudentID:string, CourseID:string, EnrollmentDate:date);

Instructors (InstructorID:integer, InstructorName:string, Phone:integer) ;

Course_Instructors (CourseID:string, InstructorID:integer);

# AIM:2.1 Querying and modifying the database using Data Manipulation Language commands -select, insert, update, delete

**Description:**

DML COMMANDS are INSERT, UPDATE, DELETE and SELECT.

INSERT COMMAND:

This command is used to create data into the table which is already defined through DDL commands. The data can be entered in the form of rows and columns.

**Syntax:**

INSERT INTO <Table name> (column1, [column2, ,columnN]) values (column1value, column2value,.............................,columnNvalue);
OR
INSERT INTO <Table name> Values (column1value,column2value,… ,columnNvalue);

UPDATE COMMAND:

This command is used to modify or change or replace the existingdata of a table.

**Syntax:**

UPDATE <Table_name> Set <column1> = <column1value> [,<column2>=<column2value> ,.......... ,<columnN>=<columnNvalue> ] [where<condition>];

DELETE COMMAND:

This command is used to remove a single row or multiple rows of a table.

**Syntax:** DELETE FROM<Table_name> [where<condition>];

SELECT COMMAND:

This command is used to view a single row or multiple rows or single column or multiple columns of a table.

Syntax:

Select [distinct] column1,column2 as newname from table1,table2 Where condition Group by column_name Having condition Order by column_name asc|desc;

**Creating Students Table:**

**Syntax:**
create table students ( rollno varchar2(30), name varchar2(30));
Table Created.

Inserting Data into the table:

insert into students values('24B11CS234','Bala'); 1 row(s) inserted.

insert into students values('24B11CS381','Kiran'); 1 row(s) inserted.

Displaying Data from the table

select * from students;

| ROLLNO | NAME |
|---|---|
| 24B11CS234 | Bala |
| 24B11CS381 | Kiran |

select name from students;

| NAME |
|---|
| Bala |
| Kiran |

select * from students9 where rollno='24B11CS234'

| ROLLNO | NAME |
|---|---|
| 24B11CS234 | Bala |

Deleting a row from the table

delete from students where rollno='24B11CS381'; 1 row(s) deleted.

Updating a row in the table

update students set name='Bala Raju' where rollno='24B11CS381'; 1 row(s) updated.

Date: Roll No.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 4 | B | 1 | 1 | A | I | 2 | 7 | 2 |

# AIM: 2.2 Implementation of Aggregate Functions – sum, avg, min, max, count. Use group-by and having clause.

**Description:**

Aggregate Functions

**AVG function:** It can be used on numeric data or character data that contains only numeric's.

**MAX function:** It is used to find the maximum value of x. It can be used on any type of data.

**MIN function:** It is used to find the minimum value of x

**SUM function:** It sums the values and can be used on numeric data also.

**COUNT(*):** It counts the number of rows in the table or the number of row in the group including NULL.

**Group by:** The attribute or attributes given in the clauses are used to form groups. Tuples with the same value on all attributes in the group by clause are placed in one group.

**Having:** SQL applies predicates (conditions) in the having clause after groups have been formed, so aggregate function be used.

**Source Table**

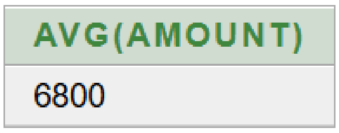
select * from company;

| company | amount |
|---|---|
| wipro | 5000 |
| ibm | 8000 |
| dell | 9000 |
| wipro | 2000 |
| dell | 10000 |

**Queries**

Find the average salary of company

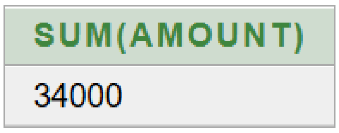
Select AVG (amount) from company;



| AVG(AMOUNT) |
|---|
| 6800 |

Find the Sum of salaries of company

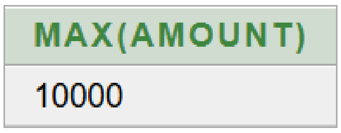
Select SUM(amount) from company;



| SUM(AMOUNT) |
|---|
| 34000 |

Find the Maximum amount of company

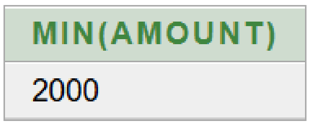
Select Max(amount) from company;



| MAX(AMOUNT) |
|---|
| 10000 |

Find the Minimum amount of company

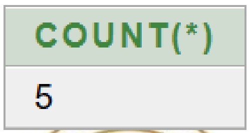
Select Min(amount) from company;



| MIN(AMOUNT) |
|---|
| 2000 |

Find the number of rows in a company

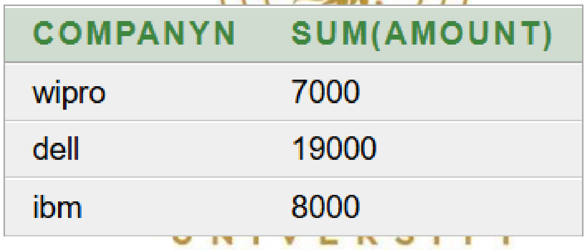
Select Count(*) from company;



| COUNT(*) |
|---|
| 5 |

Find the sum of amount of each company.

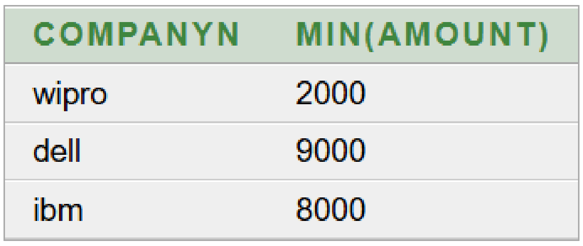
select companyn, sum(amount) from company group by company;



| COMPANYN | SUM(AMOUNT) |
|---|---|
| wipro | 7000 |
| dell | 19000 |
| ibm | 8000 |

Find the minimum amount of each company.

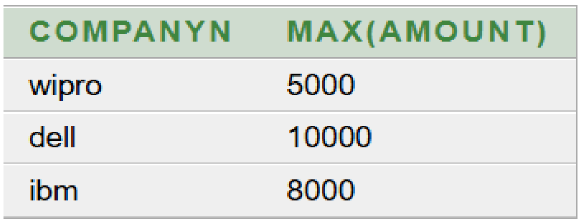
select companyn,min(amount) from company group by company;



| COMPANYN | MIN(AMOUNT) |
|---|---|
| wipro | 2000 |
| dell | 9000 |
| ibm | 8000 |

Find the maximum amount of each company.

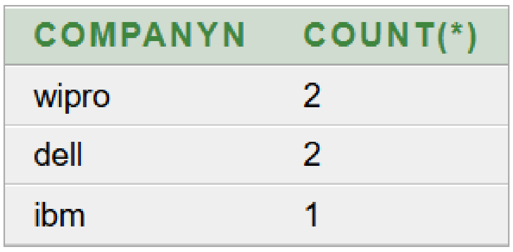
select companyn,max(amount) from company group by companyn;



| COMPANYN | MAX(AMOUNT) |
|---|---|
| wipro | 5000 |
| dell | 10000 |
| ibm | 8000 |

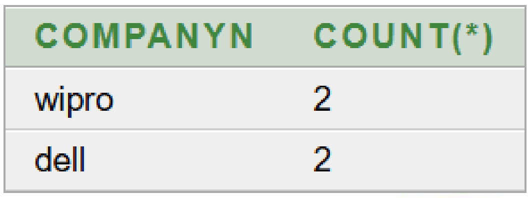
Find the count of all the rows grouped by each company name.

select companyn,count(*) from company group by companyn;



| COMPANYN | COUNT(*) |
|---|---|
| wipro | 2 |
| dell | 2 |
| ibm | 1 |

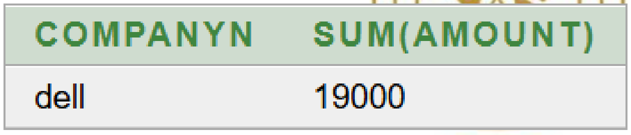
Find the count of all the rows grouped by each company name & having count greater than 1.
select companyn,count(*) from company group by companyn having count(*)>1;



| COMPANYN | COUNT(*) |
|---|---|
| wipro | 2 |
| dell | 2 |

Find the sum of amount of each company and having sum of amount greater than 10000.

select companyn, sum(amount) from company group by companyn having sum(amount)>10000;



| COMPANYN | SUM(AMOUNT) |
|---|---|
| dell | 19000 |

Date: Roll No. | 2 | 4 | B | 1 | 1 | A | I | 2 | 7 | 2 |

# AIM: 3.1 Perform Join Operations-Natural Join, Equi-Join, Outer Join, Left Outer Join, Right Outer Join, Inner Join and assess the impact of query plans on the performance of join heavy queries.

**Description:**

JOIN Keyword is used in SQL queries for joining two or more tables.

## Types of Joins

## Inner Join

The INNER JOIN keyword selects records that have matching values in both tables.
Syntax: select * from tablename1 inner join tablename2 on condition

## Outer Join

Outer Join is based on both matched and unmatched data.
It is divided into

1) Left outer join

The left outer join returns a resultset table with the matched data from the two tables and then the remaining rows of the left table and null from the right table's columns.

Syntax:

SELECT column-name-list FROM table-name1 LEFT OUTER JOIN table-name2 ON table-name1.column-name = table-name2.column-name;

2) Right outer join

The right outer join returns a resultset table with the matched data from the two tables being joined, then the remaining rows of the right table and null for the remaining left table's columns.

Syntax:

SELECT column-name-list FROM table-name1 RIGHT OUTER JOIN table-name2 ON table-name1.column-name = table-name2.column-name;

## Natural join

It is based on column having same name and same datatype present in both the tables to be joined.

Syntax:
SELECT * FROM table-name1 NATURAL JOIN table-name2;

## Cross join

It will return a table which consists of records which combines each row from the first table - with each row of the second table.
Syntax: select * from tablename1,tablename2;

## Self Join

A **self join** is a **join** in which a table is joined with itself.

## EQUI Join

An Equi Join in SQL is a type of join that combines rows from two or more tables based on a common column or set of columns, using only the equality operator (=) to compare the values in those columns.

Syntax

SELECT column-name-list FROM table1, table2.... WHERE table1.column_name =table2.column_name;

**Source Tables**

select *from tb1;

| RNO | NAME | MARKS |
|---|---|---|
| 503 | Suma | 40 |
| 504 | Raju | 70 |
| 505 | Ramu | 45 |
| 501 | Abhi | 50 |
| 502 | Ravi | 60 |

select * from tb2;

| RNO | FEE |
|---|---|
| 501 | 15000 |
| 502 | 5000 |
| 503 | 10000 |
| 504 | 25000 |

## Inner Join

select *from tb1 inner join tb2 on tb1.rno=tb2.rno;

Or use the below query also

select *from tb1 join tb2 on tb1.rno=tb2.rno;

| RNO | NAME | MARKS | RNO | FEE |
|---|---|---|---|---|
| 503 | Suma | 40 | 503 | 10000 |
| 504 | Raju | 70 | 504 | 25000 |
| 501 | Abhi | 50 | 501 | 15000 |
| 502 | Ravi | 60 | 502 | 5000 |

## Left Outer Join

select * from tb1 left outer join tb2 on tb1.rno=tb2.rno;

| RNO | NAME | MARKS | RNO | FEE |
|---|---|---|---|---|
| 501 | Abhi | 50 | 501 | 15000 |
| 502 | Ravi | 60 | 502 | 5000 |
| 503 | Suma | 40 | 503 | 10000 |
| 504 | Raju | 70 | 504 | 25000 |
| 505 | Ramu | 45 | - | - |

## Right outer join

select * from tb1 right outer join tb2 on tb1.rno=tb2.rno;

| RNO | NAME | MARKS | RNO | FEE |
|---|---|---|---|---|
| 503 | Suma | 40 | 503 | 10000 |
| 504 | Raju | 70 | 504 | 25000 |
| 501 | Abhi | 50 | 501 | 15000 |
| 502 | Ravi | 60 | 502 | 5000 |

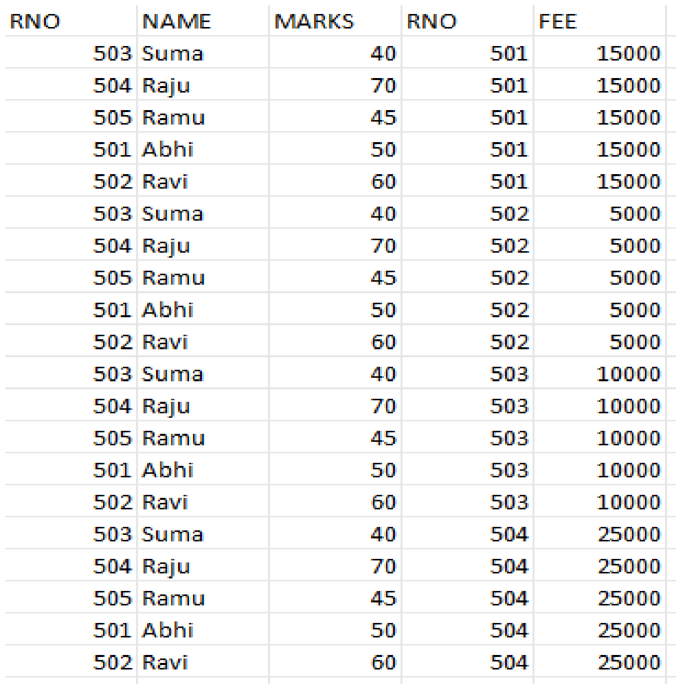
## Natural join

select *from tb1 natural join tb2;

| RNO | NAME | MARKS | FEE |
|---|---|---|---|
| 503 | Suma | 40 | 10000 |
| 504 | Raju | 70 | 25000 |
| 501 | Abhi | 50 | 15000 |
| 502 | Ravi | 60 | 5000 |

## Cross join

select *from tb1 cross join tb2;
or use the below query
select * from tb1,tb2;



| RNO | NAME | MARKS | RNO | FEE |
|---|---|---|---|---|
| 503 | Suma | 40 | 501 | 15000 |
| 504 | Raju | 70 | 501 | 15000 |
| 505 | Ramu | 45 | 501 | 15000 |
| 501 | Abhi | 50 | 501 | 15000 |
| 502 | Ravi | 60 | 501 | 15000 |
| 503 | Suma | 40 | 502 | 5000 |
| 504 | Raju | 70 | 502 | 5000 |
| 505 | Ramu | 45 | 502 | 5000 |
| 501 | Abhi | 50 | 502 | 5000 |
| 502 | Ravi | 60 | 502 | 5000 |
| 503 | Suma | 40 | 503 | 10000 |
| 504 | Raju | 70 | 503 | 10000 |
| 505 | Ramu | 45 | 503 | 10000 |
| 501 | Abhi | 50 | 503 | 10000 |
| 502 | Ravi | 60 | 503 | 10000 |
| 503 | Suma | 40 | 504 | 25000 |
| 504 | Raju | 70 | 504 | 25000 |
| 505 | Ramu | 45 | 504 | 25000 |
| 501 | Abhi | 50 | 504 | 25000 |
| 502 | Ravi | 60 | 504 | 25000 |

## Self Join:

select t1.rno, t2.name from tb1 t1,tb1 t2 where t1.rno=t2.rno;

| RNO | NAME |
|---|---|
| 503 | Suma |
| 504 | Raju |
| 505 | Ramu |
| 501 | Abhi |
| 502 | Ravi |

## EQUI Join:

select * from tb1,tb2 where tb1.rno=tb2.rno;

| RNO | NAME | MARKS | RNO | FEE |
|---|---|---|---|---|
| 503 | Suma | 40 | 503 | 10000 |
| 504 | Raju | 70 | 504 | 25000 |
| 501 | Abhi | 50 | 501 | 15000 |
| 502 | Ravi | 60 | 502 | 5000 |

Date: Roll No. | 2 | 4 | B | 1 | 1 | A | I | 2 | 7 | 2 |

## AIM: 3.2 Perform Set Operations-Union, Intersection, Set Difference Description:

Set Operators

1) UNION is used to combine the results of two or more `SELECT` statements. It will -eliminate duplicate rows from its resultset.

Syntax:-

```
SELECT column_name FROM table1
UNION
SELECT column_name FROM table2;
```

2) UNION ALL This is similar to Union. But it also shows the duplicate rows.

Syntax:-

```
SELECT column_name FROM table1
UNION ALL
SELECT column_name FROM table2;
```

3) Intersect operation is used to combine two `SELECT` statements, but it only retuns the records which are common from both `SELECT` statements.

Syntax:-

```
SELECT  column_name  FROM  table1
INTERSECT
SELECT column_name FROM table2;
```

4) The Minusoperation combines results of two `SELECT` statements and return only those in the final result, which belongs to the first set of the result.

Syntax:-

```
SELECT column_name FROM table1
MINUS
SELECT column_name FROM table2;
```

**INTERSECT and EXCEPT are not natively supported in MySQL but can be implemented using JOIN or NOT EXISTS techniques.**

**Intersect operation in mysql**

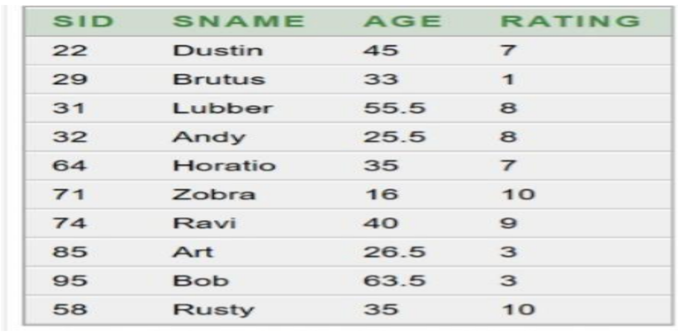
```
SELECT column1, column2 FROM table1 INNER JOIN table2 USING (column1, column2);
```

**Minus operation in mysql**

```
SELECT column1, column2 FROM table1 LEFT JOIN table2 USING (column1, column2)
WHERE table2.column1 IS NULL;
```

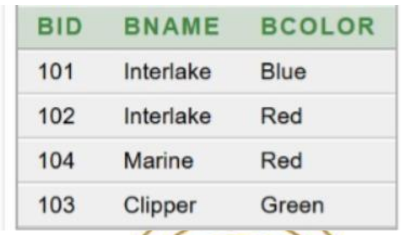
**Source Tables**

**Sailors Table**



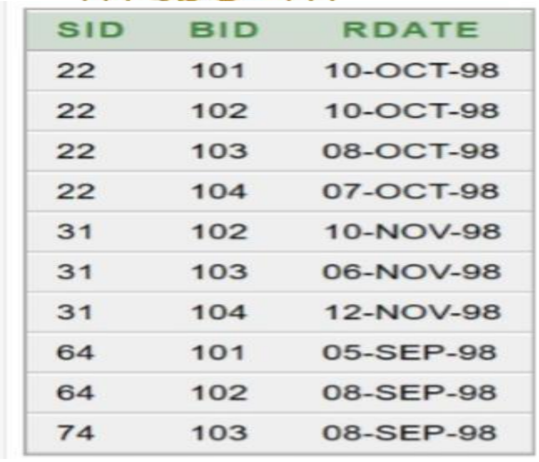
| SID | SNAME | AGE | RATING |
|---|---|---|---|
| 22 | Dustin | 45 | 7 |
| 29 | Brutus | 33 | 1 |
| 31 | Lubber | 55.5 | 8 |
| 32 | Andy | 25.5 | 8 |
| 64 | Horatio | 35 | 7 |
| 71 | Zobra | 16 | 10 |
| 74 | Ravi | 40 | 9 |
| 85 | Art | 26.5 | 3 |
| 95 | Bob | 63.5 | 3 |
| 58 | Rusty | 35 | 10 |

**Boats Table**



| BID | BNAME | BCOLOR |
|---|---|---|
| 101 | Interlake | Blue |
| 102 | Interlake | Red |
| 104 | Marine | Red |
| 103 | Clipper | Green |

**Reserves Table**



| SID | BID | RDATE |
|---|---|---|
| 22 | 101 | 10-OCT-98 |
| 22 | 102 | 10-OCT-98 |
| 22 | 103 | 08-OCT-98 |
| 22 | 104 | 07-OCT-98 |
| 31 | 102 | 10-NOV-98 |
| 31 | 103 | 06-NOV-98 |
| 31 | 104 | 12-NOV-98 |
| 64 | 101 | 05-SEP-98 |
| 64 | 102 | 08-SEP-98 |
| 74 | 103 | 08-SEP-98 |

**Find the names of sailors who have reserved a red or a green boat**

```
select s.sname from sailors s, reserves r,boats b where s.sid=r.sid and b.bid=r.bid and
b.bcolor='Red'
UNION
select s1.sname from sailors s1,reserves r1,boats b1 where s1.sid=r1.sid and r1.bid=b1.bid and
b1.bcolor='Green';
```

| SNAME |
|---|
| Dustin |
| Horatio |
| Lubber |
| Ravi |

Date: Roll No. | 2 | 4 | B | 1 | 1 | A | I | 2 | 7 | 2 |

## Find the names of sailors who have reserved a red or a green boat

```
select s.sname from sailors s, reserves r,boats b where s.sid=r.sid and b.bid=r.bid and b.bcolor='Red'
UNION all
select s1.sname from sailors s1,reserves r1,boats b1 where s1.sid=r1.sid and r1.bid=b1.bid and b1.bcolor='Green';
```

| SNAME |
|---|
| Dustin |
| Lubber |
| Horatio |
| Dustin |
| Lubber |
| Dustin |
| Lubber |
| Ravi |

## Find the names of sailors who have reserved both a red and green boat.

```
select s.sname from sailors s, reserves r,boats b where s.sid=r.sid and b.bid=r.bid and b.bcolor='Red'
INTERSECT
select s1.sname from sailors s1,reserves r1,boats b1 where s1.sid=r1.sid and r1.bid=b1.bid and b1.bcolor='Green';
```

| SNAME |
|---|
| Dustin |
| Lubber |

## Find the names of sailors who have reserved a red boat but not a green boat

```
select s.sname from sailors s, reserves r,boats b where s.sid=r.sid and b.bid=r.bid and b.bcolor='Red'
MINUS
select s1.sname from sailors s1,reserves r1,boats b1 where s1.sid=r1.sid and r1.bid=b1.bid and b1.bcolor='Green';
```

| SNAME |
|---|
| Horatio |

# AIM:3.3 Implementation of Correlated sub-queries and Nested queries

Description:

Nested Queries

A query within another SQL query and embedded within the WHERE clause.In Nested Query, Inner query runs first, and only once. Outer query is executed with result from Inner query.

1) IN and NOT IN

It tests whether a value is in a given set of elements

Syntax:-

Select column_names from table_name Where column_name IN/NOT IN (Select column_name from table_name Where condition);

2) ALL and ANY

It is used to compare a value to a list. It is preceded by comparison operator and followed by a list.

Syntax:-

Select column_name from table_name Where column_name comparison operator ALL/ANY(subquery);

**Correlated Sub query**

In Correlated sub query**,** a query is nested inside another query and an inner query uses values from the outer query.

Syntax:-

SELECT *column_names* FROM *table_name* WHERE EXISTS/NOT EXISTS (SELECT *column_name* FROM *table_name* WHERE *condition*);

Find the names of sailors who have reserved boat no 103.

select s.sname from sailors s where s.sid IN( select r.sid from reserves r where r.bid=103);

| SNAME |
|---|
| Dustin |
| Lubber |
| Ravi |

Find the names of sailors who have not reserved boat no 103.

select s.sname from sailors s where s.sid NOT IN( select r.sid from reserves r where r.bid=103);

| SNAME |
|---|
| Brutus |
| Andy |
| Horatio |
| Zobra |
| Art |
| Bob |
| Rusty |

Find the sailor id with the highest rating

select s.sid from sailors s where s.rating>=all(select s1.rating from sailors s1);

| SID |
|---|
| 71 |
| 58 |

Find the sailor id whose rating is better than some sailor called andy.

select s.sid from sailors s where s.rating>ANY(select s1.rating from sailors s1 where s1.sname='Andy');

| SID |
|---|
| 71 |
| 58 |
| 74 |

**Correlated Sub query**

Find the names of sailors who have reserved boat no 103

select s.sname from sailors s where EXISTS(select * from reserves r where s.sid=r.sid and r.bid=103);

| SNAME |
|---|
| Dustin |
| Lubber |
| Ravi |

Find the names of sailors who have not reserved boat no 103

select s.sname from sailors s where NOT EXISTS(select * from reserves r where s.sid=r.sid and r.bid=103);

| SNAME |
|---|
| Brutus |
| Andy |
| Horatio |
| Zobra |
| Art |
| Bob |
| Rusty |

Date: Roll No. 2 4 B 1 1 A I 2 7 2

# AIM: 3.4 Creating and Querying views and Materialized views.

Description:

View

A view is a logical table based on the result set of an SQL Statement. A view contains rows and columns, just like a table. The fields in a view are fields from one or more base tables in the database. We can apply all DDL and DML statements on views.

Syntax:

Create or replace force/noforce view viewname as Select column_list from table_list

Where condition with read only|check option

Dropping a view

syntax:

Drop view viewname;

Source Table

| ROLLNO | NAME | MARKS |
|---|---|---|
| 501 | jyothi | 90 |
| 502 | sai | 95 |
| 504 | yamuna | 70 |
| 505 | padma | 60 |
| 503 | ravi | 80 |

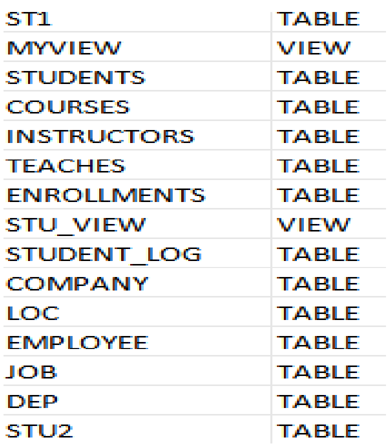
**Creating View**

create view my_view as select rollno, name from source_table;

view created

Display Views and Tables in your login

select * from tab;



```
ST1             TABLE
MYVIEW          VIEW
STUDENTS        TABLE
COURSES         TABLE
INSTRUCTORS     TABLE
TEACHES         TABLE
ENROLLMENTS     TABLE
STU_VIEW        VIEW
STUDENT_LOG     TABLE
COMPANY         TABLE
LOC             TABLE
EMPLOYEE        TABLE
JOB             TABLE
DEP             TABLE
STU2            TABLE
```

Date: Roll No. | 2 | 4 | B | 1 | 1 | A | I | 2 | 7 | 2 |

Inserting a row in view

insert into myview values(506,'prathisha');

1 row(s) inserted

Display view

select * from myview;

| ROLLNO | NAME |
|---|---|
| 501 | jyothi |
| 502 | sai |
| 506 | prathisha |
| 504 | yamuna |
| 505 | padma |
| 503 | ravi |

**Displaying Table**

select * from st1;

| ROLLNO | NAME | MARKS |
|---|---|---|
| 501 | jyothi | 90 |
| 502 | sai | 95 |
| 506 | prathisha | - |
| 504 | yamuna | 70 |
| 505 | padma | 60 |
| 503 | ravi | 80 |

Deleting a row in a view

delete from myview where rollno=506; 1
row(s) deleted

Display view

select * from myview;

| ROLLNO | NAME |
|---|---|
| 501 | jyothi |
| 502 | sai |
| 504 | yamuna |
| 505 | padma |
| 503 | ravi |

Displaying Table

select * from st1;

| ROLLNO | NAME | MARKS |
|---|---|---|
| 501 | jyothi | 90 |
| 502 | sai | 95 |
| 504 | yamuna | 70 |
| 505 | padma | 60 |
| 503 | ravi | 80 |

Change the Structure of the View

create or replace view myview as select * from st1; view created.

Creating View when base table doesn't exist

create or replace force view abc as select * from dummy_table;

Warning: View created with compilation errors.

Creating Read only view

create view myview1 as select * from st1 with read only;

view created

Inserting Data in Read only view

insert into myview1 values(503,'prathisha',80);

MYSQL-01733: virtual column not allowed here

update myview1 set name='suma' where rollno=505; MYSQL-01733: virtual column not allowed here

**Displaying view**

select * from myview1;

| ROLLNO | NAME | MARKS |
|---|---|---|
| 501 | jyothi | 90 |
| 502 | sai | 95 |
| 504 | yamuna | 70 |
| 505 | padma | 60 |
| 503 | ravi | 80 |

**Creating View with check option**

create view myview2 as select * from st1 where marks<101 with check option;

view created.

Inserting a row into view

insert into myview2 values(504,'siri',101);

MYSQL-01402: view WITH CHECK OPTION where-clause violation

Dropping view

Drop View myview1;

View dropped.

**Complex view:**

Creating a view from multiple tables

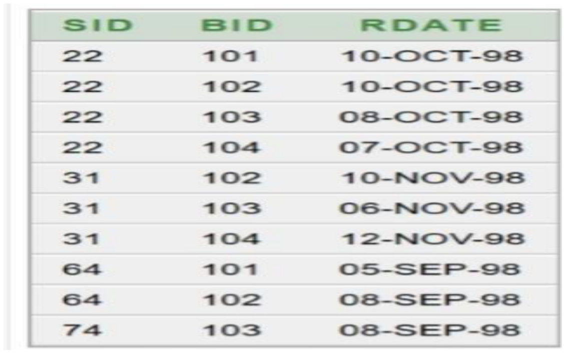
In this example will create a *view named MarksView by taking data from both the table's student details and student marks*

To create a View from multiple tables just simply *include multiple tables in the SELECT statement*

*Table1:*

| ROLLNO | NAME | MARKS |
|---|---|---|
| 501 | jyothi | 90 |
| 502 | sai | 95 |
| 504 | yamuna | 70 |
| 505 | padma | 60 |
| 503 | ravi | 80 |

*Table2:*



| SID | BID | RDATE |
|---|---|---|
| 22 | 101 | 10-OCT-98 |
| 22 | 102 | 10-OCT-98 |
| 22 | 103 | 08-OCT-98 |
| 22 | 104 | 07-OCT-98 |
| 31 | 102 | 10-NOV-98 |
| 31 | 103 | 06-NOV-98 |
| 31 | 104 | 12-NOV-98 |
| 64 | 101 | 05-SEP-98 |
| 64 | 102 | 08-SEP-98 |
| 74 | 103 | 08-SEP-98 |

**Creating Complex View**

create view myview as select rollno, name, sid, bib from Table1, Table2; view created

Display Views and Tables in your login

select * from Table1, Table2;

Change the Structure of the View

create or replace view myview as select * from Table1,Table2;

view created.

Dropping a view
syntax:

Drop view viewname;

Generally, **DML operations (INSERT, UPDATE, DELETE)** are **not allowed** on complex view.