2. Perform the following: a. Viewing all databases, Creating a Database, Viewing all Tables in a Database, Creating Tables (With and Without Constraints), Inserting/Updating/Deleting Records in a Table, Saving (Commit) and Undoing (rollback)

Viewing all databases:

SHOW DATABASES;

Result Grid | Filter Rows:

Database
employee
information_schema
mysql
performance_schema
sys

Creating a Database:

CREATE DATABASE Employees;

8	22:34:12	SHOW DATABASES	5 row(s) returned
9	22:42:45	CREATE DATABASE Employees	1 row(s) affected

Viewing all tables in a Database:

SHOW DATABASES;

Result Grid | Filter Rows: | Export:

Database
employee
employee
information_schema
mysql
performance_schema
sys

Result 6

USE Employees;

9	22:42:45	CREATE DATABASE Employees	1 row(s) affected
10	22:44:37	SHOW DATABASES	6 row(s) returned
11	22:45:54	USE Employees	0 row(s) affected

SHOW TABLES;

Result Grid | Filter Rows:

Tables_in_employees

Creating Tables:

<u>Without Constraints:</u>

```
CREATE TABLE employees (
id INT,
name VARCHAR(100),
position VARCHAR(50),
salary DECIMAL(10, 2) );
desc employees;
```

Result Grid | Filter Rows: | Export: | W

Field	Type	Null	Key	Default	Extra
id	int	YES		NULL	
name	varchar(100)	YES		NULL	
position	varchar(50)	YES		NULL	
salary	decimal(10,2)	YES		NULL	

<u>With Constraints (Primary Key, Not Null, Unique):</u>

```
CREATE TABLE employees1 (
id INT PRIMARY KEY,
name VARCHAR(100) NOT NULL,
email VARCHAR(100) UNIQUE,
salary DECIMAL(10, 2),
hire_date DATE );
desc employees1;
```

Result Grid | Filter Rows: | Export: | Wrap C

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
name	varchar(100)	NO		NULL	
email	varchar(100)	YES	UNI	NULL	
salary	decimal(10,2)	YES		NULL	
hire_date	date	YES		NULL	

Inserting Records into a Table:

INSERT INTO employees1 (id, name, position, salary, hire_date)
VALUES (1, 'John Doe', 'Manager', 65000, '2024-01-15');
Or
INSERT INTO employees1 VALUES (2, 'Jane Smith', 'Engineer', 55000, '2023-11-12');
Select *from employees1;

Result Grid | Filter Rows: | Edit:

id	name	email	salary	hire_date
2	Jane Smith	Engineer	55000.00	2023-11-12
NULL	NULL	NULL	NULL	NULL

Updating Records in a Table:

UPDATE employees1 SET salary = 70000 WHERE id = 2;
Select *from employees1;

Result Grid | Filter Rows: | Edit:

id	name	email	salary	hire_date
2	Jane Smith	Engineer	70000.00	2023-11-12
NULL	NULL	NULL	NULL	NULL

Deleting Records in a Table:

SET SQL_SAFE_UPDATES = 0;

INSERT INTO employees1 VALUES
(1, 'Dua', 'Docter', 65000, '2023-11-12');

Select *from employees1;

id	name	email	salary	hire_date
1	Dua	Docter	65000.00	2023-11-12
2	Jane Smith	Engineer	70000.00	2023-11-12
NULL	NULL	NULL	NULL	NULL

DELETE FROM employees1 WHERE id = 1;

Result Grid | Filter Rows: | Edit:

id	name	email	salary	hire_date
2	Jane Smith	Engineer	70000.00	2023-11-12
NULL	NULL	NULL	NULL	NULL

Saving (Commit):

COMMIT;

Undoing Changes (Rollback):

ROLLBACK;

55	23:42:37	COMMIT	0 row(s) affected
56	23:42:43	Select *from employees1 LIMIT 0, 1000	1 row(s) returned
57	23:42:49	ROLLBACK	0 row(s) affected
58	23:42:55	Select *from employees1 LIMIT 0, 1000	1 row(s) returned

3. Perform the following:
a. Altering a Table, Dropping/Truncating/Renaming Tables, Backing up / Restoring a Database.

a. Altering a Table:

You can modify a table structure by adding, removing, or changing columns.

- **Add a new column:**

ALTER TABLE employees1 ADD COLUMN phone_number VARCHAR(15);
desc employees1;

Result Grid | Filter Rows: | Export: | Wrap Cell

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
name	varchar(100)	NO		NULL	
email	varchar(100)	YES	UNI	NULL	
salary	decimal(10,2)	YES		NULL	
hire_date	date	YES		NULL	
phone_number	varchar(15)	YES		NULL	

Result 1

- **Modify a column (changing data type):**

ALTER TABLE employees1 MODIFY COLUMN salary DECIMAL(12, 2);
desc employees1;

Result Grid | Filter Rows: | Export: | Wrap Cell Co

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
name	varchar(100)	NO		NULL	
email	varchar(100)	YES	UNI	NULL	
salary	decimal(12,2)	YES		NULL	
hire_date	date	YES		NULL	
phone_number	varchar(15)	YES		NULL	

Select *from employees1;

id	name	email	salary	hire_date	phone_number
2	Jane Smith	Engineer	70000.00	2023-11-12	NULL
NULL	NULL	NULL	NULL	NULL	NULL

- **Drop (remove) a column:**

ALTER TABLE employees1 DROP COLUMN phone_number;
Select *from employees1;

Result Grid | Filter Rows: | Edit:

id	name	email	salary	hire_date
2	Jane Smith	Engineer	70000.00	2023-11-12
NULL	NULL	NULL	NULL	NULL

- **Rename a column:**

ALTER TABLE employees1 CHANGE COLUMN name full_name VARCHAR(100);
Select *from employees1;

Result Grid | Filter Rows: | Edit:

id	full_name	email	salary	hire_date
2	Jane Smith	Engineer	70000.00	2023-11-12
NULL	NULL	NULL	NULL	NULL

b. Dropping/Truncating/Renaming Tables:

- **Truncate a table (removes all data but keeps the structure):**

TRUNCATE TABLE employees;
select *from employees1;

Result Grid | Filter Rows: | E

id	full_name	email	salary	hire_date
NULL	NULL	NULL	NULL	NULL

- **Rename a table:**

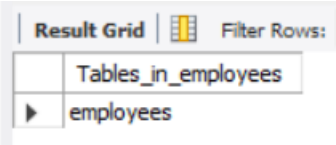
RENAME TABLE employees TO staff;
SHOW TABLES;

Result Grid | Filter Rows:

Tables_in_employees
employees
staff

- **<u>Drop a table (completely removes it):</u>**

DROP TABLE staff;
SHOW TABLES;



	Tables_in_employees
▶	employees

c. **<u>Backing up a Database:</u>**

You can back up a MySQL database using the mysqldump utility.
mysqldump -u your_username -p your_database_name > backup.sql
his will create a backup of the database in a .sql file named backup.sql.

d. **<u>Restoring a Database:</u>**

To restore a database from a backup file: mysql -u your_username -p your_database_name < backup.sql This will restore the data from backup.sql back into the specified database.

4. For a given set of relation schemes, create tables and perform the following Simple Queries, Simple Queries with Aggregate functions, Queries with Aggregate functions (group by and having clause).

Sample Relation Schemes:

1. **Employees** (employee_id, name, department, salary, hire_date)
2. **Departments** (department_id, department_name)
3. **Projects** (project_id, project_name, department_id)
4. **Assignments** (employee_id, project_id, hours_worked)

Step 1: Create Tables

-- Create Employees Table

```
CREATE TABLE Employees444 (
employee_id INT PRIMARY KEY,
name VARCHAR(100),
department VARCHAR(50),
salary DECIMAL(10, 2),
hire_date DATE );
Desc Employees444;
```

Result Grid | Filter Rows: | Export: | Wrap C

Field	Type	Null	Key	Default	Extra
employee_id	int	NO	PRI	NULL	
name	varchar(100)	YES		NULL	
department	varchar(50)	YES		NULL	
salary	decimal(10,2)	YES		NULL	
hire_date	date	YES		NULL	

-- Create Departments Table

```
CREATE TABLE Departments (
department_id INT PRIMARY KEY,
department_name VARCHAR(100)
);
```

Field	Type	Null	Key	Default	Extra
department_id	int	NO	PRI	NULL	
department_name	varchar(100)	YES		NULL	

-- Create Projects Table

```
CREATE TABLE Projects (
project_id INT PRIMARY KEY,
project_name VARCHAR(100),
department_id INT,
FOREIGN KEY (department_id) REFERENCES
Departments(department_id)
);
Desc Projects;
```

Field	Type	Null	Key	Default	Extra
project_id	int	NO	PRI	NULL	
project_name	varchar(100)	YES		NULL	
department_id	int	YES	MUL	NULL	

-- Create Assignments Table

```
CREATE TABLE Assignments (
employee_id INT,
project_id INT,
hours_worked INT,
PRIMARY KEY (employee_id, project_id),
FOREIGN KEY (employee_id) REFERENCES Employees(employee_id),
FOREIGN KEY (project_id) REFERENCES Projects(project_id)
);
```

Field	Type	Null	Key	Default	Extra
employee_id	int	NO	PRI	NULL	
project_id	int	NO	PRI	NULL	
hours_worked	int	YES		NULL	

<u>Step 2: Insert Sample Data</u>

-- Insert data into Employees table

```
INSERT INTO Employees777 (employee_id, name, department, salary,
hire_date) VALUES
(1, 'Alice', 'IT', 60000, '2021-01-15'),
```

(2, 'Bob', 'HR', 45000, '2020-07-01'),
(3, 'Charlie', 'Finance', 50000, '2019-10-23'),
(4, 'David', 'IT', 55000, '2018-05-16'),
(5, 'Alice', 'IT', 60000, '2021-01-15'),
(6, 'Bob', 'HR', 45000, '2020-07-01'),
(7, 'Charlie', 'Research', 60000, '2019-10-23'),
(8, 'David', 'Accounts', 57000, '2018-05-16'),
(9, 'Alice', ' Accounts ', 80000, '2021-01-15'),
(10, 'Bob', ' Research ', 55000, '2020-07-01'),
(11, 'Charlie', ' Research ', 40000, '2019-10-23'),
(12, 'David', ' Accounts ', 35000, '2018-05-16'),
(13, 'Alice', ' Accounts ', 20000, '2021-01-15'),
(14, 'Bob', ' Research ', 15000, '2020-07-01'),
(15, 'Charlie', ' Research ', 70000, '2019-10-23'),
(16, 'David', ' Accounts ', 58000, '2018-05-16');
Select *from Employees777;

employee_id	name	department	salary	hire_date
1	Alice	IT	60000.00	2021-01-15
2	Bob	HR	45000.00	2020-07-01
3	Charlie	Finance	50000.00	2019-10-23
4	David	IT	55000.00	2018-05-16
5	Alice	IT	60000.00	2021-01-15
6	Bob	HR	45000.00	2020-07-01
7	Charlie	Research	60000.00	2019-10-23
8	David	Accounts	57000.00	2018-05-16
9	Alice	Accounts	80000.00	2021-01-15
10	Bob	Research	55000.00	2020-07-01
11	Charlie	Research	40000.00	2019-10-23
12	David	Accounts	35000.00	2018-05-16
13	Alice	Accounts	20000.00	2021-01-15
14	Bob	Research	15000.00	2020-07-01
15	Charlie	Research	70000.00	2019-10-23
16	David	Accounts	58000.00	2018-05-16
NULL	NULL	NULL	NULL	NULL

-- Insert data into Departments table

INSERT INTO Departments (department_id, department_name)
VALUES
(1, 'IT'),
(2, 'HR'),

(3, 'Finance');
Select *from Departments;

department_id	department_name
1	IT
2	HR
3	Finance
NULL	NULL

-- Insert data into Projects table

INSERT INTO Projects (project_id, project_name, department_id)
VALUES
(101, 'Website Upgrade', 1),
(102, 'Employee Training', 2),
(103, 'Budget Analysis', 3);
Select *from Projects;

project_id	project_name	department_id
101	Website Upgrade	1
102	Employee Training	2
103	Budget Analysis	3
NULL	NULL	NULL

-- Insert data into Assignments table

INSERT INTO Assignments (employee_id, project_id, hours_worked)
VALUES
(1, 101, 35),
(2, 102, 20),
(3, 103, 40),
(4, 101, 45);
Select *from Assignments;

employee_id	project_id	hours_worked
1	101	35
2	102	20
3	103	40
4	101	45
NULL	NULL	NULL

Step 3: Perform Simple Queries

Get all employees in the IT department:

SELECT * FROM Employees777 WHERE department = 'IT';

employee_id	name	department	salary	hire_date
1	Alice	IT	60000.00	2021-01-15
4	David	IT	55000.00	2018-05-16
5	Alice	IT	60000.00	2021-01-15
NULL	NULL	NULL	NULL	NULL

Get details of employees hired after 2020:

SELECT name, hire_date FROM Employees777 WHERE hire_date > '2020-01-01';

name	hire_date
Alice	2021-01-15
Bob	2020-07-01
Alice	2021-01-15
Bob	2020-07-01
Alice	2021-01-15
Bob	2020-07-01
Alice	2021-01-15
Bob	2020-07-01

Get all projects associated with the Finance department:

SELECT project_name FROM Projects
WHERE department_id = (SELECT department_id FROM Departments
WHERE department_name = 'Finance');

project_name
Budget Analysis

Step 4: Simple Queries with Aggregate Functions

Find the average salary of all employees:

SELECT AVG(salary) AS avg_salary FROM Employees777;

Result Grid

avg_salary
50312.500000

Find the maximum salary in the IT department:
SELECT MAX(salary) AS max_salary FROM Employees777 WHERE department = 'IT';

	max_salary
▶	60000.00

Find the total hours worked by all employees:

SELECT SUM(hours_worked) AS total_hours FROM Assignments;

	total_hours
▶	140

Count the number of employees in each department:

SELECT department, COUNT(*) AS num_employees FROM Employees777 GROUP BY department;

	department	num_employees
▶	IT	3
	HR	2
	Finance	1
	Research	1
	Accounts	1
	Accounts	4
	Research	4

<u>Step 5: Queries with Aggregate Functions (GROUP BY and HAVING Clause)</u>

Find the total hours worked by employees per project:

SELECT project_id, SUM(hours_worked) AS total_hours FROM Assignments GROUP BY project_id;

	project_id	total_hours
▶	101	80
	102	20
	103	40

Find the average salary of employees in each department:

SELECT department, AVG(salary) AS avg_salary FROM Employees777 GROUP BY department;

department	avg_salary
IT	58333.333333
HR	45000.000000
Finance	50000.000000
Research	60000.000000
Accounts	57000.000000
Accounts	48250.000000
Research	45000.000000

Find departments where the average salary is greater than 50,000:

SELECT department, AVG(salary) AS avg_salary FROM Employees777 GROUP BY department HAVING AVG(salary) > 50000;

department	avg_salary
IT	58333.333333
Research	60000.000000
Accounts	57000.000000

Get the projects with more than 30 total hours worked:

SELECT project_id, SUM(hours_worked) AS total_hours FROM Assignments GROUP BY project_id HAVING SUM(hours_worked) > 30;

project_id	total_hours
101	80
103	40

5. Execute the fallowing queries

a. How the resulting salaries if every employee working on the „Research" Departments is given a 10% raise.
b. Find the sum of the salaries of all employees of the „Accounts" department, as well as the maximum salary, the minimum salary, and the average salary in this department

a. **Increase Salaries by 10% for Employees in the "Research" Department**

To update the salaries of employees in the "Research" department, you would execute the following query:

UPDATE employees777 SET salary = salary * 1.10 WHERE department = 'Research';

This query multiplies the current salary by 1.10 (which adds a 10% raise) for all employees whose department is "Research".

employee_id	name	department	salary	hire_date
1	Alice	IT	60000.00	2021-01-15
2	Bob	HR	45000.00	2020-07-01
3	Charlie	Finance	50000.00	2019-10-23
4	David	IT	55000.00	2018-05-16
5	Alice	IT	60000.00	2021-01-15
6	Bob	HR	45000.00	2020-07-01
7	Charlie	Research	72600.00	2019-10-23
8	David	Accounts	57000.00	2018-05-16
9	Alice	Accounts	80000.00	2021-01-15
10	Bob	Research	55000.00	2020-07-01
11	Charlie	Research	40000.00	2019-10-23
12	David	Accounts	35000.00	2018-05-16
13	Alice	Accounts	20000.00	2021-01-15
14	Bob	Research	15000.00	2020-07-01
15	Charlie	Research	70000.00	2019-10-23
16	David	Accounts	58000.00	2018-05-16
NULL	NULL	NULL	NULL	NULL

b. **Sum, Max, Min, and Average Salaries in the "Accounts" Department**

To calculate the sum, maximum, minimum, and average salaries of employees in the "Accounts" department, you can use the following query:

```
SELECT
SUM(salary) AS total_salary,
MAX(salary) AS max_salary,
MIN(salary) AS min_salary,
AVG(salary) AS average_salary
FROM employees777
WHERE department = 'Accounts';
```

This query retrieves the total, maximum, minimum, and average salaries for all employees working in the "Accounts" department.

	total_salary	max_salary	min_salary	average_salary
▶	57000.00	57000.00	57000.00	57000.000000

6,7,8,9,10

To execute the queries you've mentioned, we first need to set up the necessary database schema, including tables for employees, departments, and projects. Let's create a simplified version of these tables and then run the queries as specified. Below, I'll outline the steps involved.

Step 1: Create Tables

We will create the following tables:

1. **Employees**: Contains employee details.

2. **Departments**: Contains department details.

3. **Projects**: Contains project details.

4. **Employee_Project**: A junction table to represent the many-to-many relationship between employees and projects.

Create Departments Table

```
CREATE TABLE Departments (
DeptID INT PRIMARY KEY,
DeptName VARCHAR(50)
);
desc Departments;
```

	Field	Type	Null	Key	Default	Extra
▶	DeptID	int	NO	PRI	NULL	
	DeptName	varchar(50)	YES		NULL	

-Create Employees Table

```
CREATE TABLE Employees (
EmpID INT PRIMARY KEY,
EmpName VARCHAR(50),
Salary DECIMAL(10, 2),
DeptID INT, DOB DATE,
```

FOREIGN KEY (DeptID) REFERENCES Departments(DeptID));
Desc Employees;

	Field	Type	Null	Key	Default	Extra
▶	EmpID	int	NO	PRI	NULL	
	EmpName	varchar(50)	YES		NULL	
	Salary	decimal(10,2)	YES		NULL	
	DeptID	int	YES	MUL	NULL	
	DOB	date	YES		NULL	

-- Create Projects Table

CREATE TABLE Projects (
ProjectID INT PRIMARY KEY,
ProjectName VARCHAR(50)
);
desc Projects;

	Field	Type	Null	Key	Default	Extra
▶	ProjectID	int	NO	PRI	NULL	
	ProjectName	varchar(50)	YES		NULL	

-- Create Employee_Project Table

CREATE TABLE Employee_Project (
EmpID INT,
ProjectID INT,
PRIMARY KEY (EmpID, ProjectID),
FOREIGN KEY (EmpID) REFERENCES Employees(EmpID),
FOREIGN KEY (ProjectID) REFERENCES Projects(ProjectID)
);
desc Employee_Project;

	Field	Type	Null	Key	Default	Extra
▶	EmpID	int	NO	PRI	NULL	
	ProjectID	int	NO	PRI	NULL	

Step 2: Insert Sample Data

-- Insert Departments

```
INSERT INTO Departments (DeptID, DeptName) VALUES
(1, 'Research'),
(2, 'Accounts'),
(3, 'HR'),
(4, 'IT'),
(5, 'Sales');
Select *from Departments;
```

DeptID	DeptName
1	Research
2	Accounts
3	HR
4	IT
5	Sales
NULL	NULL

-- Insert Employees

```
INSERT INTO Employees (EmpID, EmpName, Salary, DeptID, DOB)
VALUES
(1, 'Alice', 50000, 1, '1990-01-12'),
(2, 'Bob', 60000, 2, '1999-02-23'),
(3, 'Charlie', 55000, 2, '1990-03-20'),
(4, 'David', 45000, 3, '1999-04-15'),
(5, 'Eve', 40000, 1, '1990-05-16'),
(6, 'Frank', 70000, 5, '1999-06-17'),
(7, 'Grace', 80000, 1, '1998-07-18'),
(8, 'Heidi', 65000, 2, '1997-08-19'),
(9, 'Ivan', 30000, 4, '2000-09-23'),
(10, 'Judy', 90000, 5, '2001-08-2'),
(11, 'Judy', 90000, 5, '2005-09-3'),
(12, 'Judy', 100000, 5, '2004-10-4'),
(13, 'Judy', 200000, 5, '2003-11-5'),
```

(14, 'Judy', 300000, 5. '2002-12-6');
Select *from Employees;

EmpID	EmpName	Salary	DeptID	DOB
1	Alice	50000.00	1	1990-01-12
2	Bob	60000.00	2	1999-02-23
3	Charlie	55000.00	2	1990-03-20
4	David	45000.00	3	1999-04-15
5	Eve	40000.00	1	1990-05-16
6	Frank	70000.00	5	1999-06-17
7	Grace	80000.00	1	1998-07-18
8	Heidi	65000.00	2	1997-08-19
9	Ivan	30000.00	4	2000-09-23
10	Judy	90000.00	5	2001-08-02
11	Judy	90000.00	5	2005-09-03
12	Judy	100000.00	5	2004-10-04
13	Judy	200000.00	5	2003-11-05
14	Judy	300000.00	5	2002-12-06
NULL	NULL	NULL	NULL	NULL

-- Insert Projects

INSERT INTO Projects (ProjectID, ProjectName) VALUES
(1, 'Project A'),
(2, 'Project B'),
(3, 'Project C');
Select *from Projects;

ProjectID	ProjectName
1	Project A
2	Project B
3	Project C
NULL	NULL

-- Insert Employee_Project

INSERT INTO Employee_Project (EmpID, ProjectID) VALUES
(1, 1),
(1, 2),
(2, 1),
(3, 1),
(4, 2),

(5, 3),
(6, 2),
(7, 3),
(8, 1);
Select *from Employee_Project;

EmpID	ProjectID
1	1
2	1
3	1
8	1
1	2
4	2
6	2
5	3
7	3
NULL	NULL

Step 3: Execute the Queries

6. Execute the fallowing queries
a. Retrieve the name of each employee Controlled by department number 5 (use
EXISTS operator).
b. Retrieve the name of each dept and number of employees working in each
department which has at least 2 employees.

a.

```
SELECT e.EmpName
FROM employees e
WHERE EXISTS (
SELECT 1
FROM departments d
WHERE d.DeptID = e.DeptID
AND d.DeptID = 5
);
```

	EmpName
▶	Frank
	Judy
	Judy
	Judy
	Judy
	Judy

b.

```
SELECT d.DeptName, COUNT(e.EmpID) AS employee_count
FROM departments d
JOIN employees e ON d.DeptID = e.DeptID
GROUP BY d.DeptName
HAVING COUNT(e.EmpID) >= 2;
```

	DeptName	employee_count
▶	Research	3
	Accounts	3
	Sales	6