1. Create Database, use database, create table , insert table

```sql
DROP DATABASE IF EXISTS `Parks_and_Recreation`;
CREATE DATABASE `Parks_and_Recreation`;
USE `Parks_and_Recreation`;

CREATE TABLE employee_salary (
  employee_id INT NOT NULL,
  first_name VARCHAR(50) NOT NULL,
  last_name VARCHAR(50) NOT NULL,
  occupation VARCHAR(50),
  salary INT,
  dept_id INT
);

INSERT INTO employee_salary (employee_id, first_name, last_name, occupation, salary, dept_id)
VALUES
(1, 'Leslie', 'Knope', 'Deputy Director of Parks and Recreation', 75000,1),
(2, 'Ron', 'Swanson', 'Director of Parks and Recreation', 70000,1);
```

2. Select ____ from , select distinct

```sql
SELECT first_name,
age,
(age+10)*100
 FROM parks_and_recreation.employee_demographics;

select distinct gender from parks_and_recreation.employee_demographics;
```

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

gender
Female
Male

3. Where , Logical Operators(AND OR NOT), like statement

```sql
-- where statement
select * from employee_salary where salary <= 50000;
select * from employee_demographics where gender = 'male';
select * from employee_demographics where birth_date > '1985-01-01';
select * from employee_demographics where birth_date > 1985-01-01; -- not work
-- AND OR NOT -- Logical Operators
select * from employee_demographics where (first_name = 'Leslie' AND age=44) OR age >55;
-- LIKE statement
-- % and _
select * from employee_demographics where first_name like 'a__%';
select * from employee_demographics where birth_date like '1980%';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

employee_id	first_name	last_name	age	gender	birth_date
8	Chris	Traeger	43	Male	1980-11-11
NULL	NULL	NULL	NULL	NULL	NULL

4. Group By + Order By in Mysql

```
SELECT
gender,avg(age), max(age), min(age), count(age)
FROM parks_and_recreation.employee_demographics
group by gender;
```

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

gender	avg(age)	max(age)	min(age)	count(age)
Female	38.5000	46	29	4
Male	41.2857	61	34	7

```
SELECT
  gender,
  avg(age),   -- This finds the **average age** for each gender
  max(age),   -- This finds the **oldest age** for each gender
  min(age),   -- This finds the **youngest age** for each gender
  count(age)  -- This counts **how many people** are in each gender group
FROM parks_and_recreation.employee_demographics
GROUP BY gender;
```

Order by

```
SELECT * FROM parks_and_recreation.employee_demographics order by age, gender;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content

employee_id	first_name	last_name	age	gender	birth_date
1	Leslie	Knope	30	Female	1979-09-25
4	April	Ludgate	30	Female	1994-03-27
3	Tom	Haverford	30	Male	1987-03-04
10	Andy	Dwyer	34	Male	1989-03-25
7	Ann	Perkins	35	Female	1988-12-01
12	Craig	Middlebrooks	37	Male	1986-07-27

5. Having Vs Where

```
SELECT
occupation, avg(salary)
FROM parks_and_recreation.employee_salary
where occupation like '%manager%'
group by occupation
having avg(salary)>75000;
```

Result Grid | Filter Rows: | Export:

occupation	avg(salary)
City Manager	90000.0000

So why not use WHERE avg(salary) > 75000?
Because:

- avg(salary) is **not available yet** during the WHERE phase.
- You can only use **aggregate functions** (like AVG(), SUM(), COUNT()) in the **HAVING clause**, not in WHERE.

6. **Limit + Aliasing in MySQL**

```
2 •  SELECT *
3    FROM parks_and_recreation.employee_demographics
4    order by age desc
5    limit 2,1;
```

Result Grid | Filter Rows: | Edit:

employee_id	first_name	last_name	age	gender	birth_date
8	Chris	Traeger	43	Male	1980-11-11
NULL	NULL	NULL	NULL	NULL	NULL

- ORDER BY age DESC puts the **oldest first**.
- LIMIT 2, 1 skips the **1st and 2nd oldest**, and shows the **3rd**.

Aliasing (as)

```
2 •  select gender,avg(age) as avg_age
3    from parks_and_recreation.employee_demographics
4    group by gender
5    having avg_age >40;
```

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

gender	avg_age
Male	40.4286