Exercise 2 . SQL Aggregate Functions + SQL Operators

Table. students

Student_id	name	Age	Department
1	Alice	20	IT
2	Bob	22	HR
3	Charlie	21	IT
4	Diana	23	Finance
5	Eve	22	HR

SELECT DISTINCT department
FROM students;

Department
IT
HR
Finance

① List all distinct departments in the students table

Expected columns:
department

② Get the average age of students per department

```
SELECT   department
         AVG(age) AS avg_age
FROM student
GROUP BY department;
```

department	Avg-age
IT	20.5
HR	22
Finance	23

③ Show departments with more than 1 student

```
SELECT department,
       COUNT(student_id) AS student_count
  GROUP BY department
  HAVING statement > 1;
```

Department	Student_count
IT	2
HR	2

④ Get students who age is between 21 & 23

```
SELECT student_id,
       name,
       age,
       department
FROM students
WHERE age BETWEEN 21 AND 23;
```

student_id	name	age	Department
~~Bob~~ 2	Bob	22	HR
3	Charlie	21	IT
4	Diana	23	Finance
5	Eve	22	HR

⑤ List all students in the IT or HR department who are older than 21

```
SELECT student_id,
       name,
       age,
       department
FROM students
WHERE department = 'IT' OR department = 'HR' AND 721;
```

student_id	name	age	Department
2	Bob	22	HR
~~4~~ 5	~~Diyaqie~~ Charlie	~~22~~ 21	HR
~~1~~ 3	~~Alice~~ Charlie	~~20~~ 21	~~IT~~ IT

MS2

⑥ Table courses

course_id	course_name	Department	Credits
101	SQL Basics	IT	3
102	Python	IT	4
103	Data Science	IT	4
104	Excel	Finance	~~3~~ 2
105	Statistics	HR	3

Show total credits per department, only for departments with more than 5 total credits

```
SELECT department,
  SUM(credits) AS Total_Credits
FROM courses
GROUP BY DEPARTMENT
HAVING SUM(credits) > 5;
```

→

Department	Total_Credits
IT	11

⑦ List all courses that do not have 4 credits

```
SELECT course_id,
       name,
       department,
       credits
FROM courses
WHERE credits <> 4; /
 WHERE credits =! 4;
```

course id	name	Department	Credits
101	SQLbasics	IT	3
104	Excel	Finance	2
105	Statistics	HR	3

⑧ Show top 3 courses by credits in descending order

```
SELECT course_id,
       course_name,
       credits
FROM courses
ORDER BY credits DESC
LIMIT 3;
```

→

course_id	course_name	credits
102	Python	4
103	Data Science	4
105	Statistics	3
101	SQL Basics	3

⑨ Enrolments

enrolment_id	student_id	course_id	grade
1	1	101	85
2	2	102	78
3	3	103	90
4	4	104	88
5	5	105	82

⑨ Get the maximum, minimum & average grade across all enrolments

```
SELECT max_grade
       min_grade
       avg_grade
FROM enrolments;
```

or

```
SELECT MAX(grade) AS max_grade
FROM enrolments;

SELECT MIN(grade) AS min_grade
FROM enrolments;

SELECT AVG(grade) AS avg_grade
FROM enrolments;
```

max_grade

max_grade	min_grade	avg_grade
90	78	84.6

⑩ Count how many enrolments exist per course

course_id	enrolment_count
101	1
102	1
103	1
104	1
105	1

```
SELECT course_id,
       COUNT(*) AS enrolment_count
FROM enrolments
GROUP BY course_id;
```

salaries

employee-id	name	Department	Salary	Bonus
1	Tom	IT	60000	5000
2	Jerry	HR	55000	4000
3	Spike	Finance	70000	6000
4	Tyke	IT	62000	5500
5	Butch	HR	54000	3500

(11) Find total salary & total bonus per department

```
SELECT department,
       SUM(salary) AS total_salary,
       SUM(bonus) AS total_bonus
FROM salaries
GROUP BY department;
```

MS2 MS2

Total_salary	Total_bonus	Department
122000	10500	IT
109000	7500	HR
70000	6000	Finance

(12) Show departments where avg salary is above 55000

```
SELECT department,
       AVG(salary) AS Avg_salary
FROM salaries
GROUP BY department
HAVING AVG(salary) > 55000;
```

MS2

Department	Avg salary
IT	61000
~~HR~~	~~54500~~
Finance	70000

MS2

(13) List employees whose salary bonus is greater than 60000

```
SELECT employee_id,
       name,
       salary,
       bonus
       (salary + bonus) AS total_compensation
FROM salaries
WHERE (salary + bonus) > 60000
```

employee_id	name	salary	Bonus	Total_compensation
1	Tom	60000	5000	65000
3	Spike	70000	6000	76000
4	Tyke	62000	5500	67500

(14) Projects

Project-id	Project-name	Department	Budget
1	AI App	IT	12000
2	Paywall system	Finance	8000
3	Dashboard	IT	15000
4	Website	Marketing	6000
5	HR Portal	HR	5000

Show total & avg budget per department. Only include department with avg budget above 70000

```
SELECT department,
    SUM(budget) AS total_budget
    AVG(budget) AS avg_budget
FROM projects
GROUP BY department
HAVING AVG(budget) > 70000;
```

Department	total budget	Avg-budget
IT	27000	13500
Finance	8000	8000

(15) List all projects with budgets between 5000 and 120000, excluding the Marketing department

```
SELECT project-id,
    project-name,
    department,
    budget
FROM projects
WHERE budget BETWEEN 5000 AND 120000
AND department != 'Marketing';
```

Project-id	Project-name	Department	Budget
5	HR Portal	HR	5000
2	Paywall system	Finance	8000
1	AI App	IT	12000