Customers

Cus_ID	Name	City
1	Alice	Durban
2	Bob	Cape-town
3	Carol	Durban

order

Orde-ID	Cus_ID	amount
101	1	500
102	2	300
103	4	450

```
SELECT  Customer_ID, name, amount
FROM    Customers AS A
INNER Join   order AS B
~~OM~~ ON A.Customer_ID = B.Customer_ID
```

Customer_ID	Name	amount
1	Alice	500
2	Bob	300

- once we apply the Join, we can ~~apply~~ Select any column between the two tables

Exercise 2

1)
```
SELECT Distinct Department ✓
FROM   Students;
```

Department ✓
IT
HR
Finance

2)
```
SELECT Distinct Department
       AVG(Age) AS avg_age ✓
FROM   Students
Group By  Department;
```

Department	Avg_Age
IT	20,5 ✓
HR	22 ✓
Finance	23 ✓

3) SELECT DISTINCT Department
Count (Department) AS Student_Count
FROM Students
WHERE Department IN ("IT", "HR")

Department	Student-Count
IT	2
HR	2

SELECT department
count(stu_ID) AS Stud
FROM Students
Group By department
Having Student-count > 1;

4) SELECT Student_ID, name, age, department ✓
FROM Student
WHERE AGE Between 21 AND 23 ✓

Students_ID,	Name,	AGE	d Department
2	BoB	22	HR
3	Charlie	21	IT
4	Diana	23	Finance
5	EVE	22	HR

✓

5) SELECT Student_ID, Name, AGE, Department
FROM Students
WHERE Department = IT OR HR AND AGE > 21

IN ("HR", ~~"HR"~~ "IT") AND age > 21;

Student.ID	Name	AGE.	Department
2	BoB	22	HR
5	EVE	22	~~HR~~ HR

6) ~~SEE~~ SELECT Distinct Department
SUM (credits) AS total_credits
FROM courses
WHERE total-credits > 5

Department	total_ credits
IT	11

7) SELECT course_ID, course_name, department, credits
FROM Course
WHERE ~~Course~~ credits < 4

Course_ID	course_name	Department	credits
101	SQL Basics	IT	3
104	Excel	Finance	2
105	Statistics	HR	3

8 SELECT Course_ID, Course_name, credits
FROM Course
~~WHERE~~ order By DESC
Limit 3

COURSE_ID	course_name	credits
102	Python	4
103	Data Science	4
101	SQL Basic	3

9 SELECT SELECT MAX(grade) AS max_grade
MIN(grade) AS min_grade
AVG(grade) AS AVG_grade
FROM Enrollment;

Max-grade	Min-grade	AVG-Grade
90	78	84.6

10 SELECT Course_ID
Count(Course_ID) AS enrollment_count
FROM Enrollments

Course_ID	Enrollment-Count
101	1
102	1
103	1
104	1
105	1

11 SELECT ~~Department~~ Distinct Department
Sum(Salary) AS total_salary
Sum(bonus) AS total-bonus
FROM Salaries;

Department	total-Salary	total_bonus
IT	122000	10500
HR	109000	7500
~~Finance~~ Finance	70000	6000

12 SELECT Distinct Department
AVG(Salary) AS AVG_Salary
FROM Salaries
WHERE AVG_Salary > 55000

Department	AVG_Salary
IT	61000
Finance	70000

13 SELECT Employee_ID, name, Salary, bonus, ~~tota_com~~
Salary + Bonus AS total_compensation
FROM Salaries
WHERE total_Compensation > 60000

Employee_ID	name	Salary	Bonus	total_comp
1	Tom	60000	5000	65000
3	Spike	70000	6000	76000
4	Tyke	62000	5500	67500

14 SELECT ~~Employees~~ Distinct Department
Sum(Budget) AS total_budget
AVG(Budget) AS Avg_budget
FROM Projects
WHERE Avg_budget > 70000

Department	total budget	AVG_budget
IT	270000	135000
Finance	80000	80000

15 SELECT ~~Rest~~ project_id, project_name
department, budget
FROM projects
WHERE Budget Between 50000 and 120000
AND Department != "Marketing"

project_ID	project_Name	Department	Budget
1	AI App	IT	120000
2	Payroll System	Finance	80000
3	Dasboard	IT	150 000
5	HR Portal	HR	50 000

18/09/2025
Left Join → Returns all the rows from the Left table and the matched rows from the right table
→ If ~~there~~ no match, a Null is returned
→ NULL missing value
→ empty space
→ Blank
→ missing information

~~John~~ Joins affects the rows