MSCDA5540: Assignment 1

Submitted by: A00431152 (Bhagya Shree)

Tools Used:

* MySQL Server and Workbench 8.0
* Operating System: Windows 10 Pro

# Question1

**Query :**

1. (a) Get the names and locations of the suppliers who have shipped part with pno = 3.

SELECT

sname, city

FROM

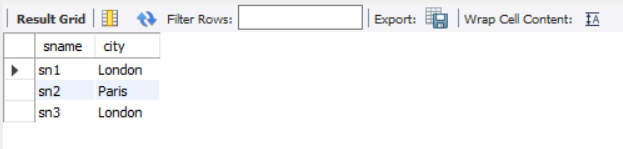
S,

SP

WHERE

S.sno = SP.sno AND SP.pno = 3 ;

Output:



**Query**

1. (b) Get the part numbers and names of parts that have been shipped by suppliers located in Paris with status at least 20.

SELECT

P.pno, pname

FROM

P,

S,

SP

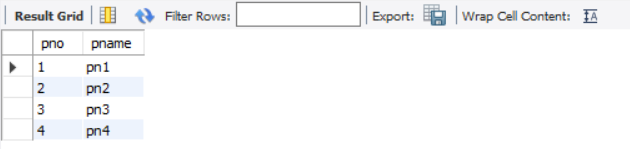
WHERE

P.pno = SP.pno AND S.sno = SP.sno

AND S.city = 'Paris'

AND S.status >= 20;

**Output:**



**Query :3**

1. (c) For each part, show the part number, name, and the number of suppliers who have supplied the part.

SELECT

P.pno, pname, COUNT(\*) AS Supplierstotal

FROM

P,

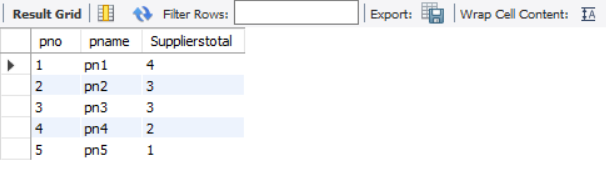
SP

WHERE

P.pno = SP.pno

GROUP BY SP.pno;

**Output:**



**Query :**

1. (d) For each London supplier who has shipped at least 1000 parts, show the name of the supplier and the total number of parts he/she has shipped.

SELECT

sname, SUM(SP.qty) AS Totalquant

FROM

S,

SP

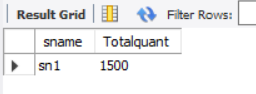
WHERE

S.sno = SP.sno AND S.city = 'London'

GROUP BY SP.sno

HAVING Totalquant >= 1000;

**Output:**



**Query :**

1. (e) Get the names and cities of the suppliers who have supplied all parts that weigh less than 4 grams.

SELECT DISTINCT

s.sname, s.city

FROM

S,

P,

SP

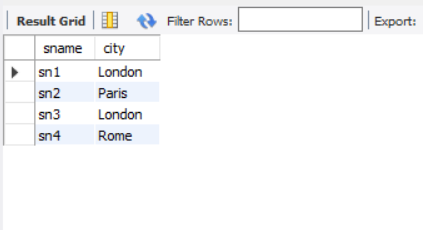
WHERE

P.pno = SP.pno AND S.sno = SP.sno

AND P.weight < 4;

**Note : distinct keyword ,to get the distinct name and city**

**Output:**



# Question 2

(a)Get the names of courses in the CS department

T ← σ(Department=“CS”)(Course)

Result ← π(Course\_name) (T);

(b) Get the names of students who got an 'A' in CS3380.

/\*Join Section and Grade report based on section id and then find the rows based on Grade =’A’) \*/

s1 ← ( σ Grade = ‘A’ ( (Section) section\_identifier = section\_identifier ⋈ (Grade\_report) )

/\* extract rows from result set s1 where course number is CS3380\*/

s2 ← ( σ course\_number = ‘CS3380’ ( S1))

/\* project student name after joining result set s2 and Student table\*/

R← π Name ( (Student) student\_number = student\_number ⋈ (s2) )

1. (c) Get the instructors who have taught CS1310 and CS3380.

π Instructor ( σ course\_number = ‘CS1310’ AND course\_number = ‘CS3380’ (SECTION) )

1. (d) Get the instructors who have taught all courses of the CS department.
2. /\* project course\_number in result set s1 from course table for CS as dept. \*/

s1 ← π course\_number ( σ Department = ‘CS’ ( Course) )

1. /\* Get Instructor and course\_number after joining ,section to result set s1 based on course\_num \*/

s2 ← π Instructor, course\_number ( (Section ) course\_number =course\_number ⋈ (s1))

1. /\* Get the instructor who taught all courses by dividing resultsets2 by s1 \*/

R ← π Instructor ( (s2) (s1) )

# Question 3

**Query :**

1. (a) **Goal:** Summarizing Data in Groups

**Table:** customer

**Query:** Write a query that displays the following statistics for each country:

- Total number of customers

- Total number of male customers

- Total number of female customers

- Percent of all customers that are male (Percent Male).

Display the result by value of Percent Male so that the country with the lowest value is listed first, with the remaining countries following in ascending order.

SELECT

Country,

SUM(Gender = 'M') AS Male,

SUM(Gender = 'F') AS Female,

SUM(Gender = 'M') \* 100 / COUNT(\*) AS percentMale

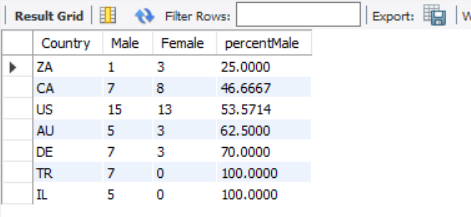
FROM

customers

GROUP BY country

ORDER BY percentMale;

**Output:**



**Query :**

(b) **Goal:** Summarizing Data in Groups

**Table:** product\_dim, order\_fact

**Query:** Create a result by combining two tables.

- Include columns Product\_ID, Product\_Name from product\_dim table.

- Include a column with the label Total Sold. Use a summary function to create this column, which displays the quantity sold for each product.

- Specifies the tables product\_dim, with the alias p and order\_fact with the alias o.

- Join the tables by matching the values of the appropriate columns in each table.

- Groups the results by Product\_ID from product\_dim table and Product\_Name.

- Orders the rows so that products with the highest number sold appear at the top of the report and then by Product\_Name.

Note: DO NOT use nested queries

SELECT

p.product\_id, p.product\_name, SUM(o.Quantity) AS 'totalsold'

FROM

product\_dim p

JOIN

order\_fact o ON p.product\_id = o.product\_id

GROUP BY p.product\_id , product\_name

ORDER BY totalsold DESC , p.product\_name;

**Output:**



**Query :**

1. (c) **Goal:** Create a result with a self-join.

**Table:** employee\_addresses, staff

**Query:** Display result of all trainees and workers at company. For each trainee or temporary worker, the report should include the employee ID, name and job title, and manager ID and name. The report should be ordered by Employee\_ID.

SELECT distinct

e.employee\_ID,

e.employee\_Name,

s.Job\_Title,

s.Manager\_ID,

e1.employee\_Name AS Manager\_Name

FROM

employee\_addresses e,

employee\_addresses e1,

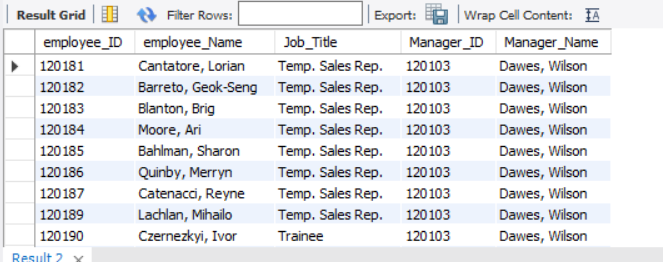
staff s

WHERE e.employee\_ID = s.employee\_ID

AND e1.employee\_ID = s.Manager\_ID

AND s.Job\_Title IN ('Trainee' , 'Temp. Sales Rep.')

ORDER BY e.employee\_ID;

**Output :**

**Query :**

1. (d) **Goal:** LEAD and LAG functions

**Table:** employee\_payroll

**Query:** 1) Calculate the difference between the salary of the current row and the previous row. 2) Calculate the difference between the salary of current row and the following row.

SELECT Employee\_ID, salary as "Salary",

LAG(salary, 1,0) OVER (ORDER BY Employee\_ID) AS "Salary\_prev" ,

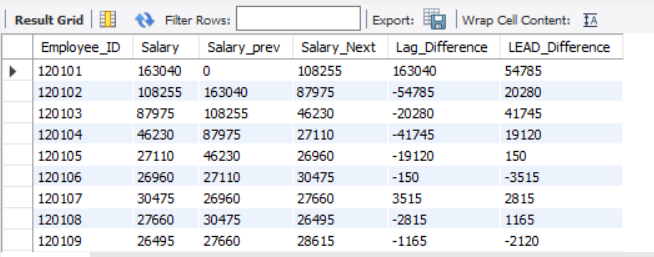
LEAD(salary, 1,0) OVER (ORDER BY Employee\_ID) AS "Salary\_Next" ,

Salary -LAG(Salary,1,0) over (order by Employee\_ID) as "Lag\_Difference",

Salary -LEAD(Salary,1,0) over (order by Employee\_ID) as "LEAD\_Difference"

FROM employee\_payroll;

**Output:**



# References

* <https://www.w3schools.com/sql>
* Lecture Slides on Relational Algebra
* Lecture Slides on Joins
* Lecture Slides on Sql Query