

Managing & Programming Database

Assignment 1

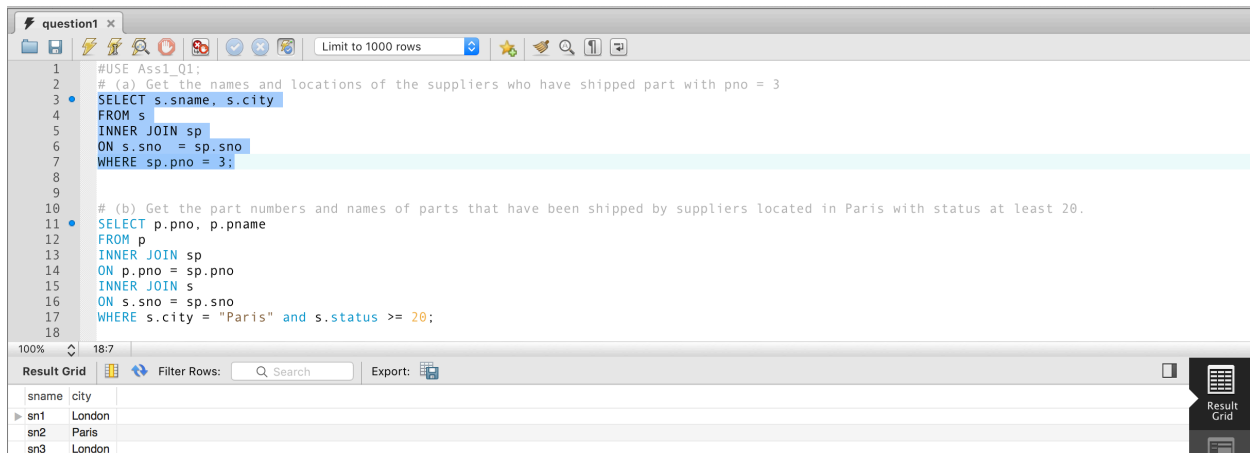
Submitted By : Caner Adil Irfanoglu

Student No: A00425840

Course Coordinator: Dr. Hai Wang

Instructor: Trishla Shah

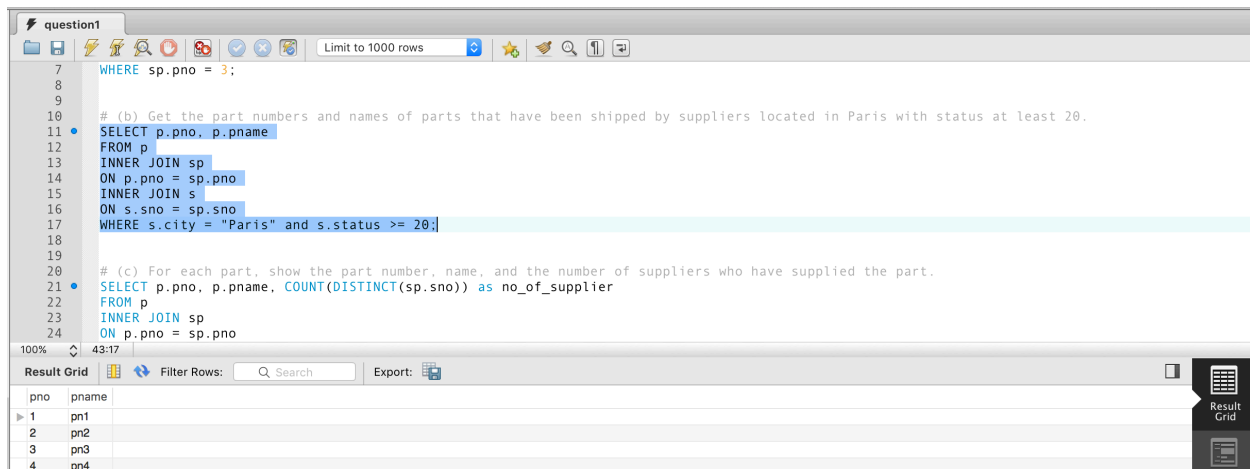
Question 1 – Queries & Results



```
1 #USE Ass1_Q1;
2 # (a) Get the names and locations of the suppliers who have shipped part with pno = 3
3 • SELECT s.sname, s.city
4 FROM s
5 INNER JOIN sp
6 ON s.sno = sp.sno
7 WHERE sp.pno = 3;
8
9
10 # (b) Get the part numbers and names of parts that have been shipped by suppliers located in Paris with status at least 20.
11 • SELECT p.pno, p.pname
12 FROM p
13 INNER JOIN sp
14 ON p.pno = sp.pno
15 INNER JOIN s
16 ON s.sno = sp.sno
17 WHERE s.city = "Paris" and s.status >= 20;
18
```

sname	city
sn1	London
sn2	Paris
sn3	London

Figure 1: Answer to Q1/a



```
7 WHERE sp.pno = 3;
8
9
10 # (b) Get the part numbers and names of parts that have been shipped by suppliers located in Paris with status at least 20.
11 • SELECT p.pno, p.pname
12 FROM p
13 INNER JOIN sp
14 ON p.pno = sp.pno
15 INNER JOIN s
16 ON s.sno = sp.sno
17 WHERE s.city = "Paris" and s.status >= 20;
18
19
20 # (c) For each part, show the part number, name, and the number of suppliers who have supplied the part.
21 • SELECT p.pno, p.pname, COUNT(DISTINCT(sp.sno)) as no_of_supplier
22 FROM p
23 INNER JOIN sp
24 ON p.pno = sp.pno
```

pno	pname
1	pn1
2	pn2
3	pn3
4	pn4

Figure 2: Answer to Q1/b

question1

Limit to 1000 rows

```

14 ON p.pno = sp.pno
15 INNER JOIN s
16 ON s.sno = sp.sno
17 WHERE s.city = "Paris" and s.status >= 20;
18
19
20 # (c) For each part, show the part number, name, and the number of suppliers who have supplied the part.
21 • SELECT p.pno, p.pname, COUNT(DISTINCT(sp.sno)) as no_of_supplier
22 FROM p
23 INNER JOIN sp
24 ON p.pno = sp.pno
25 GROUP BY p.pno;
26
27
28 # (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 shi
29 • SELECT c.sname, c.total_shipped
30 FROM
31 (SELECT s.sname, s.city, SUM(sp.qty) as total_shipped

```

100% 16:25

Result Grid Filter Rows: Search Export:

pno	pname	no_of_supplier
1	pn1	4
2	pn2	3
3	pn3	3
4	pn4	2
5	pn5	1

Result Grid Form Editor

Figure 3: Answer to Q1/c

```

27
28 # (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/s
29 • SELECT c.sname, c.total_shipped
30 FROM
31 (SELECT s.sname, s.city, SUM(sp.qty) as total_shipped
32 FROM s
33 INNER JOIN sp
34 ON s.sno = sp.sno
35 GROUP BY s.sno
36 HAVING s.city = "London" and total_shipped >= 1000) as c;

```

100% 58:36

Result Grid Filter Rows: Search Export:

sname	total_shipped
sn1	1500

Figure 4: Answer to Q1/d

```

39 # (e) Get the names and cities of the suppliers who have supplied all parts that weigh less than 4 grams.
40 • SELECT sname, city
41 FROM(
42 SELECT s.sname, s.city, p.weight, "invalid" =
43 CASE WHEN p.weight > 4 THEN 0 ELSE 1 END as invalid_weight
44 FROM s
45 INNER JOIN sp
46 ON s.sno = sp.sno
47 INNER JOIN p
48 ON p.pno = sp.pno) a
49 GROUP BY a.sname, a.city
50 HAVING SUM(invalid_weight) = 0;
51
52
53

```

100% 32:50

Result Grid Filter Rows: Search Export:

sname	city
sn3	London
sn4	Rome

Figure 5: Answer to Q1/e

Question 2 Relational Algebra Expressions

Relational Algebra SQL Group Editor

$\pi \sigma \rho \leftarrow \tau \gamma \wedge \vee \neg = \neq \geq \leq \cap \cup \div - \times \bowtie \ltimes \rtimes \ltimes \rtimes \triangleright = \neg /* \{ \} \boxtimes$

```
1 -- (a) Get the names of courses in the CS department
2
3  $\sigma_{\text{Department} = 'CS'}(\text{COURSE})$ 
```

[▶ execute query](#) [download](#) [history ▼](#)

$\sigma_{\text{Department} = 'CS'}$

COURSE

$\sigma_{\text{Department} = 'CS'}(\text{COURSE})$

COURSE.Course_name	COURSE.Course_number	COURSE.Credit_hours	COURSE.Department
Intro_to_Computer_Science	CS1310	4	CS
Data_Structures	CS3320	4	CS
Database	CS3380	3	CS

Figure1: Q2/a Expression & Output

```
1  $\pi_{\text{Name}}(\text{STUDENT} \bowtie \sigma_{\text{Grade} = 'A'}(\text{GRADE\_REPORT}) \bowtie \sigma_{\text{Course\_number} = 'CS3380'}(\text{SECTION}))$ 
```

[▶ execute query](#) [download](#) [history ▼](#)

relational algebra calculator

π_{Name}

\bowtie

\bowtie

$\sigma_{\text{Course_number} = 'CS3380'}$

SECTION

STUDENT

$\sigma_{\text{Grade} = 'A'}$

GRADE_REPORT

$\pi_{\text{Name}}(\text{STUDENT} \bowtie \sigma_{\text{Grade} = 'A'}(\text{GRADE_REPORT}) \bowtie \sigma_{\text{Course_number} = 'CS3380'}(\text{SECTION}))$

STUDENT.Name
BROWN

Figure2: Q2/b Expression & Output

Question 3 Queries & Results

question1 x question3 x

Limit to 1000 rows

```

1 # USE Ass1_Q3;
2 # 1 - Write a query that displays the following statistics for each country:
3 # Total number of customers
4 # Total number of male customers
5 # Total number of female customers
6 # Percent of all customers that are male (Percent Male).
7 • SELECT Country, Total_Customers, Male, (Total_Customers - Male) as Female, (Male / Total_Customers) * 100 as Percent_Male
8 FROM
9 (SELECT male_count.Country, male_count.male, all_count.Total_Customers
10 FROM
11 (SELECT Country, Gender, COUNT(*) as Male
12 FROM customer
13 GROUP BY Gender, Country
14 HAVING Gender = 'M') male_count
15 INNER JOIN
16 (SELECT Country, COUNT(*) as Total_Customers FROM customer GROUP BY Country) all_count
17 ON male_count.Country = all_count.Country
18 ) combined

```

100% 11:18

Result Grid Filter Rows: Search Export:

Country	Total_Customers	male	Female	Percent_Male
ZA	4	1	3	25.0000
CA	15	7	8	46.6667
US	28	15	13	53.5714
AU	8	5	3	62.5000
DE	10	7	3	70.0000
IL	5	5	0	100.0000
TR	7	7	0	100.0000

Figure1: Q3/1 Expression & Output

```

22 # 2 - Create a result by combining two tables.
23 # Include columns Product_ID, Product_Name from product_dim table.
24 # Include a column with the label Total Sold. Use a summary function to create
25 # this column, which displays the quantity sold for each product.
26 # Specifies the tables product_dim, with the alias p and order_fact with the alias o
27 # Join the tables by matching the values of the appropriate columns in each table.
28 # Groups the results by Product_ID from product_dim table and Product_Name.
29 # Orders the rows so that products with the highest number sold appear at the
30 # top of the report and then by Product_Name.
31
32 • SELECT p.Product_ID, p.Product_NAME, SUM(o.Quantity) as Total_Sold
33 FROM product_dim p
34 INNER JOIN order_fact o
35 ON p.Product_ID = o.Product_ID
36 GROUP BY p.Product_ID, p.Product_Name
37 ORDER BY Total_Sold DESC;
38
39

```

100% 1:39

Result Grid Filter Rows: Search Export:

Product_ID	Product_NAME	Total_Sold
230100600022	Expedition10,Medium,Right,Blue Ribbon	9
230100600016	Expedition Zero,Medium,Right,Charcoal	8
230100700011	Hurricane 4	8
230100500056	Knife	8
230100600030	Outback Sleeping Bag, Large,Left,Blue/Black	8
240700100001	Armour L	8
220101400387	N.d.gear Cap	7
230100600031	Outback Sleeping Bag, Large,Right, Blue/Black	7
240500100017	A-team Sweat Round Neck, Small Logo	7
240700200010	Bat - Home Run S	7
240400300035	Smasher Shorts	7
240200200039	Faale Plain Can	6

Figure2: Q3/2 Expression & Output – Displays 12 of 481

question1 x question3*

</

Figure3: Q3/3 Expression & Output – Displays 12 of 42

question1 x question3*

Limit to 1000 rows

```

53
54
55 # 4- 1) Calculate the difference between the salary of the current row and the previous row
56 • SET @quot=NULL;
57 • SELECT Employee_ID, Salary, lag_salary, (Salary - lag_salary) as Lag_diff
58 FROM
59 (SELECT Employee_ID,Salary,@quot lag_salary, @quot:=salary curr_salary
60 FROM employee_payroll
61 ORDER BY Employee_ID) a;
62
63
64 # 2) Calculate the difference between the salary of current row and the following row.
65 • SELECT t1.Employee_ID, t1.Salary, t2.Salary as Lead_Salary, (t2.Salary - t1.Salary) as Lead_diff
66 FROM (
67 SELECT Employee_ID, Salary, @rownum1 := @rownum1 + 1 AS id
68 FROM
69 employee_payroll , (SELECT @rownum1 := 0) r
70 ORDER BY
71

```

100% 25:61

Result Grid Filter Rows: Search Export:

Employee_ID	Salary	lag_salary	Lag_diff
120101	163040	108255	-54785
120103	87975	108255	-20280
120104	46230	87975	-41745
120105	27110	46230	-19120
120106	26960	27110	-150
120107	30475	26960	3515
120108	27660	30475	-2815
120109	26495	27660	-1165
120110	28615	26495	2120
120111	26895	28615	-1720
120112	26550	26895	-345
120113	26870	26550	320

Figure3: Q3/4-1 Expression & Output – Displays 12 of 424

question1 x question3*

Limit to 1000 rows

```

62
63
64 # 2) Calculate the difference between the salary of current row and the following row.
65 • SELECT t1.Employee_ID, t1.Salary, t2.Salary as Lead_Salary, (t2.Salary - t1.Salary) as Lead_diff
66 FROM (
67 SELECT Employee_ID, Salary, @rownum1 := @rownum1 + 1 AS id
68 FROM
69 employee_payroll , (SELECT @rownum1 := 0) r
70 ORDER BY
71 employee_payroll.Employee_ID ASC
72 ) AS t1
73 LEFT OUTER JOIN (
74 SELECT Employee_ID, Salary, @rownum2 := @rownum2 + 1 AS id
75 FROM
76 employee_payroll, (SELECT @rownum2 := 0) r
77 ORDER BY
78 employee_payroll.Employee_ID ASC
79 ) AS t2
80 ON t1.id +1 = t2.id;
81
82
83

```

100% 1:65

Result Grid Filter Rows: Search Export:

Employee_ID	Salary	Lead_Salary	Lead_diff
120101	163040	108255	-54785
120102	108255	87975	-20280
120103	87975	46230	-41745
120104	46230	27110	-19120
120105	27110	26960	-150
120106	26960	30475	3515
120107	30475	27660	-2815
120108	27660	26495	-1165

Result 13

Figure4: Q3/4-2 Expression & Output – Displays 8 of 424