

SQL QUERIES- group A

Main Business Problems

1. To discover if the annual salaries of customers lead to a better credit ID.

Query:

```
SELECT creditid, round(avg(annualincome))
FROM customer
GROUP BY creditid
order by creditid
```

Result:

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
creditid	round(avg(annualincome))		
1	29500		
2	48684		
3	58607		
4	59438		
5	63494		
6	59458		
7	NULL		

2. To discover if there is any correlation between credit rating and number of encounters

Query:

```
SELECT CUSTOMER.creditid, count(purchase) as Total_Number_of_Encounters, COUNT(CASE
WHEN PURCHASE ='TRUE' THEN 1 END) AS Number_of_TRUE_PURCHASE, COUNT(CASE
WHEN purchase='FALSE' THEN 1 END) AS Number_of_FALSE_PURCHASE
FROM customer
JOIN encounter
ON CUSTOMER.customerid = encounter.customerid
GROUP BY creditid
order by creditid
```

Result:

Result Grid Filter Rows: <input type="text"/> Export: Wrap Cell Content:				
	creditid	Total_Number_of_Encounters	Number_of_TRUE_PURCHASE	Number_of_FALSE_PURCHASE
▶	1	8	3	5
	2	23	6	17
	3	33	8	25
	4	42	8	34
	5	48	14	34
	6	54	18	36
	7	8	2	6

3. To discover any connection between experienced salesperson and their number of successful encounters sales.

Query:

```
select sfirstname as salesperson , salesperson.salesid, shiredate, count(encounter.purchase) AS
Number_of_Encounters
FROM salesperson
JOIN encounter
On salesperson.salesid = encounter.salesid
group by sfirstname
order by shiredate
```

Result:

Result Grid Filter Rows: <input type="text"/> Export: Wrap Cell Content:				
	salesperson	salesid	shiredate	Number_of_Encounters
▶	Ron	3	1989-05-02	39
	Hermione	2	1989-05-02	45
	Dudley	4	1996-04-27	38
	Katie	5	1996-05-15	22
	Ginny	6	2001-06-01	35
	Neville	7	2004-11-09	36
	Colin	8	2006-04-26	1

4. Weekly report with respect to encounters.

Query:

```
Select count(*) AS positiveSaleinWeek1 from encounter WHERE encounter.purchase = 'TRUE'
AND encounter.encdate BETWEEN '2015-07-01' AND '2015-07-07'; Select count(*) AS
positiveSaleinWeek2 from encounter WHERE encounter.purchase = 'TRUE' AND encounter.encdate
```

BETWEEN '2015-07-08' AND '2015-07-14'; Select count(*) AS positiveSaleinWeek3 from encounter WHERE encounter.purchase = 'TRUE' AND encounter.encdate BETWEEN '2015-07-15' AND '2015-07-22'; Select count(*) AS positiveSaleinWeek4 from encounter WHERE encounter.purchase = 'TRUE' AND encounter.encdate BETWEEN '2015-07-23' AND '2015-07-31';

Result:

Result Grid		Filter Rows
	positiveSaleinWeek1	
▶	10	

Result Grid		Filter Rows:
	positiveSaleinWeek2	
▶	14	

Result Grid		Filter Rows:
	positiveSaleinWeek3	
▶	16	

Result Grid		Filter Rows:
	positiveSaleinWeek4	
▶	19	

5. To identify the loyal customers for Gaskins

Query:

```
SELECT c.firstname, count(purchase = 'TRUE')
FROM customer
JOIN encounter
ON customer.CustomerID = encounter.CustomerID
where purchase = 'True'
```

GROUP BY CFirstName
order by count(purchase = 'TRUE') DESC
LIMIT 10

Result:

	cfirstname	count(purchase = 'TRUE')
▶	Katrina	2
	Owen	1
	Danny	1
	George	1
	Stephen	1
	Mitchell	1
	Dan	1
	Bob	1
	Vicki	1
	Paul	1

Potential Business Problems

1. How many customers paid cash without looking into financing options?

Query:

```
SELECT count(customer.customerid)
Customers_paid_cash_without_looking_into_financing_options
FROM customer
INNER JOIN creditrating
ON customer.creditid = creditrating.creditid
WHERE customer.creditid = '7'
```

AS

Result:

1	•	SELECT count(customer.customerid) AS Customers_paid_cash_without_looking_into_financing_options
2		FROM customer
3		INNER JOIN creditrating
4		ON customer.creditid = creditrating.creditid
5		WHERE customer.creditid = '7'
6		

	Customers_paid_cash_without_looking_into_finan
▶	6

2. Get the total number of customers on each date that purchased.

Query:

```
SELECT encdate AS Date, count(customerid) AS Total_number_of_Customers
FROM encounter
WHERE purchase = 'TRUE'
GROUP BY (encdate)
```

Result:

```
1 • SELECT encdate AS Date, count(customerid) AS Total_number_of_Customers FROM encounter
2 WHERE purchase = 'TRUE' GROUP BY (encdate)
```

Date	Total_number_of_Customers
2015-07-02	2
2015-07-03	1
2015-07-04	2
2015-07-05	1
2015-07-06	1
2015-07-07	1
2015-07-08	1
2015-07-09	1
2015-07-10	1
2015-07-11	4
2015-07-12	2
2015-07-13	2
2015-07-14	3
2015-07-15	1
2015-07-17	4
2015-07-18	2
2015-07-19	2
2015-07-20	2
2015-07-21	4
2015-07-22	1
2015-07-23	3
2015-07-24	2
2015-07-26	1
2015-07-27	3
2015-07-28	2
2015-07-29	3

3. Which Salesperson have encountered most?

Query:

```
SELECT sfirstname AS Salesperson_encountered_most
FROM salesperson
INNER JOIN encounter
ON salesperson.salesid = encounter.salesid
GROUP BY sfirstname
order by count(purchase) DESC
limit 1
```

Result:

```

1 • SELECT sfirstname AS Salesperson_encountered_most
2 FROM salesperson
3 INNER JOIN encounter
4 ON salesperson.salesid = encounter.salesid
5 GROUP BY sfirstname
6 order by count(purchase) DESC
7 limit 1
8
9

```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
	Salesperson_encountered_most				
▶	Hermione				

4. Identify the salesperson who sold the most.

Query:

```

SELECT sfirstname AS Salesperson_sold_most
FROM salesperson
INNER JOIN encounter
ON salesperson.salesid = encounter.salesid
where purchase = 'True'
GROUP BY sfirstname
order by count(purchase = 'TRUE') DESC
LIMIT 1

```

Result:

```

1 • SELECT sfirstname AS Salesperson_sold_most
2 FROM salesperson
3 INNER JOIN encounter
4 ON salesperson.salesid = encounter.salesid
5 where purchase = 'True'
6 GROUP BY sfirstname
7 order by count(purchase = 'TRUE') DESC
8 LIMIT 1

```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
	Salesperson_solda_most				
▶	Hermione				

5. Get details of customers that purchased most.

Query:

```

SELECT *

```

```

FROM customer
INNER JOIN encounter
ON customer.CustomerID = encounter.CustomerID
where purchase = 'True'
GROUP BY CFirstName
order by count(purchase = 'TRUE') DESC
LIMIT 1

```

Result:

```

1 • SELECT *
2 FROM customer
3 INNER JOIN encounter
4 ON customer.CustomerID = encounter.CustomerID
5 where purchase = 'True'
6 GROUP BY CFirstName
7 order by count(purchase = 'TRUE') DESC
8 LIMIT 1

```

customerid	cfirstname	clastname	cphone	annualincome	crediti	encid	salesid	customerid	encdate	purchase
65	Katrina	Newsom	(801) 763-7396	54500	5	91	6	65	2015-07-14	true

6. What is the percentage of poor and good credit ratings in customers?

Query:

```
SELECT COUNT(customerid) FROM project.`customer (1)`;
```

```
SELECT COUNT(customerid) FROM project.`customer (1)`
```

```
WHERE credited>=4
```

```
WHERE credited<4
```

Result:

```

1 • SELECT COUNT(customerid) FROM project.`customer (1)`
2 ;

```

COUNT(customerid)
177

GOOD:

```
1 • SELECT count(customerid) FROM project.`customer (1)`  
2 WHERE creditid>=4  
3 ;
```

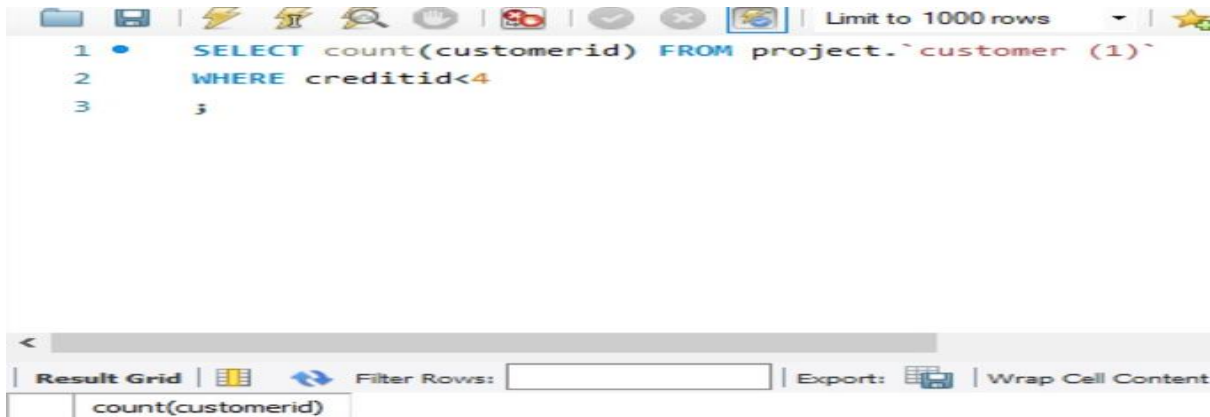


The screenshot shows a SQL query result grid. The header row is labeled 'count(customerid)'. The first data row contains the value '123'. The interface includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' link.

count(customerid)
123

$(123/177)*100=69.5\%$

BAD:



The screenshot shows a SQL query result grid. The header row is labeled 'count(customerid)'. The first data row contains the value '54'. The interface includes a toolbar with various icons, a 'Limit to 1000 rows' dropdown, and a 'Filter Rows' input field.

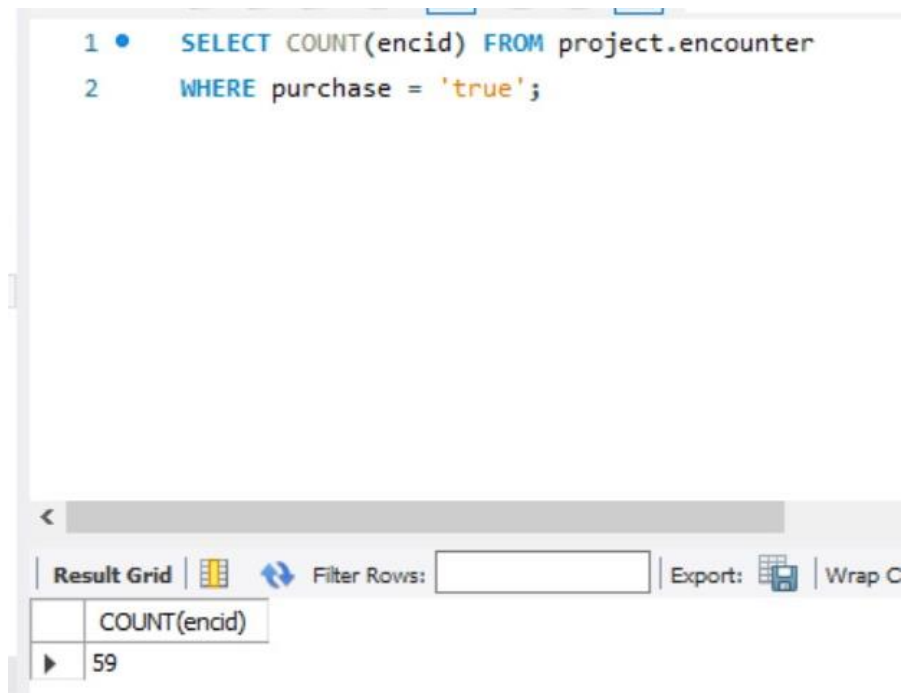
count(customerid)
54

$(54/177)*100=30.5\%$

7. How many sales were there?

Query:

```
SELECT COUNT(encid) FROM project.encounter  
WHERE purchase = 'true';
```

8. Average Credit of Customers

Query:

`SELECT avg(creditrating.creditid) FROM creditrating;`

Result:

14 • `SELECT avg(creditrating.creditid) FROM creditrating;`

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

	avg(creditrating.creditid)
▶	4.0000

9. What number of customers paid with cash

Query:

Highest Annual Income of customers

Select MAX(annualincome) as HighestIncome from customer;

Result:

16 • `Select Max(annualincome) as HighestIncome from customer;`

HighestIncome
124375

10. Lowest Annual Income of customers

Query:

Select Min(annualincome) as LowestIncome from customer;

Result:

16 • `Select Min(annualincome) as LowestIncome from customer;`

LowestIncome
20250

11. Employee having highest salary and their sales

Query:

```
SELECT salesperson.sfirstname, encounter.purchase, salesperson.slastname, salesperson.salary,
salesperson.salesid from salesperson
inner join project.encounter ON salesperson.salesid = encounter.salesid WHERE encounter.purchase =
'true'
group by salesid;
```

Result:

Query 1

```

1 • use project;
2 • SELECT salesperson.sfirstname, encounter.purchase, salesperson.slastname, salesperson.salary,
3   salesperson.salesid from salesperson
4   inner join project.encounter ON salesperson.salesid = encounter.salesid WHERE encounter.purchase = 'true'
5   group by salesid;

```

Result Grid

sfirstname	purchase	slastname	salary	salesid
Ron	true	Weasley	93000	3
Hermione	true	Granger	82000	2
Dudley	true	Dursley	56000	4
Ginny	true	Weasley	75000	6
Neville	true	Longbottom	71000	7
Kate	true	Bell	67000	5

12. IDENTIFY THE RECORDS WHERE THE CUSTOMERS HAVE A BAD CREDIT (id 1,2)

Query:

select customerid, cfirstname, clastname from portfolio.customer where creditid=1 or creditid=2;

Result:

26 ROWS FOUND

MySQL Workbench

Local instance MySQL57

Query 1

```

1 •
2 • select customerid, cfirstname, clastname from portfolio.customer where creditid=1 or creditid=2;

```

Result Grid

customerid	cfirstname	clastname
27	Helen	Fisher
29	Peter	Park
32	Alex	Fletcher
44	Bryant	O'Donnell
45	Carlos	Horatio
56	William	Austin
64	John	Fitzpatrick

Schema: portfolio

Output

#	Time	Action	Message	Duration / Fetch
59	17:51:07	insert into portfolio.creditrating (creditid,creditedescription,minifco,maxifco,comments) values (7,'unknown',null,...	1 row(s) affected	0.000 sec
60	17:51:07	select customerid from customer where creditid=1 AND creditid=2;	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL se...	0.000 sec
61	17:51:17	select from portfolio.customer LIMIT 0, 1000	177 row(s) returned	0.000 sec / 0.000 sec
62	17:51:17	insert into portfolio.creditrating (creditid,creditedescription,minifco,maxifco,comments) values (7,'unknown',null,...	1 row(s) affected	0.000 sec
63	17:51:17	select customerid from customer where creditid=1 AND creditid=2 LIMIT 0, 1000	Error Code: 1146. Table 'mysql.customer' doesn't exist	0.000 sec
64	17:51:23	select from portfolio.customer LIMIT 0, 1000	177 row(s) returned	0.000 sec / 0.000 sec
65	17:51:23	insert into portfolio.creditrating (creditid,creditedescription,minifco,maxifco,comments) values (7,'unknown',null,...	1 row(s) affected	0.000 sec
66	17:51:23	select customerid from customer where creditid=1 AND creditid=2 LIMIT 0, 1000	Error Code: 1146. Table 'mysql.customer' doesn't exist	0.016 sec
67	17:51:50	select customerid from customer where creditid=1 AND creditid=2 LIMIT 0, 1000	Error Code: 1146. Table 'mysql.customer' doesn't exist	0.000 sec
68	17:52:17	select customerid from portfolio.customer where creditid=1 AND creditid=2 LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
69	17:53:01	select customerid, cfirstname, clastname from portfolio.customer where creditid=1 or creditid=2 LIMIT 0, 1000	26 row(s) returned	0.000 sec / 0.000 sec

13. Create a list that shows each salesperson (first and last name) and the number of encounters each has had with customers that have an annual income of \$25,000 or less or have a credit

description that is “Very Poor” or “Extremely Poor.” Order the results alphabetically by the salesperson’s last name

Query:

```
SELECT salesperson.sfirstname, salesperson.slastname, COUNT(encounter.customerid)
FROM portfolio.salesperson
INNER JOIN portfolio.encounter ON salesperson.salesid=encounter.salesid
INNER JOIN portfolio.customer ON encounter.customerid=customer.customerid
INNER JOIN portfolio.creditrating ON customer.creditid=creditrating.creditid
WHERE customer.annualincome <= 25000
OR creditrating.creditdescription IN ('Very Poor', 'Extremely Poor')
GROUP BY salesperson.slastname
ORDER BY salesperson.slastname ASC
```

Result:

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left displays the 'portfolio' schema with tables 'creditrating', 'customer', and 'encounter'. The 'Query Editor' in the center contains the SQL query from the previous block. The 'Result Grid' at the bottom shows the query results for the 'portfolio' schema. The results are as follows:

sfirstname	slastname	COUNT(encounter.customerid)
Katie	Bell	6
Colin	Creevey	1
Dudley	Dursley	13
Hermione	Granger	7
Neville	Longbottom	11
Ron	Weasley	12

The 'Output' pane at the bottom shows the execution log with the following messages:

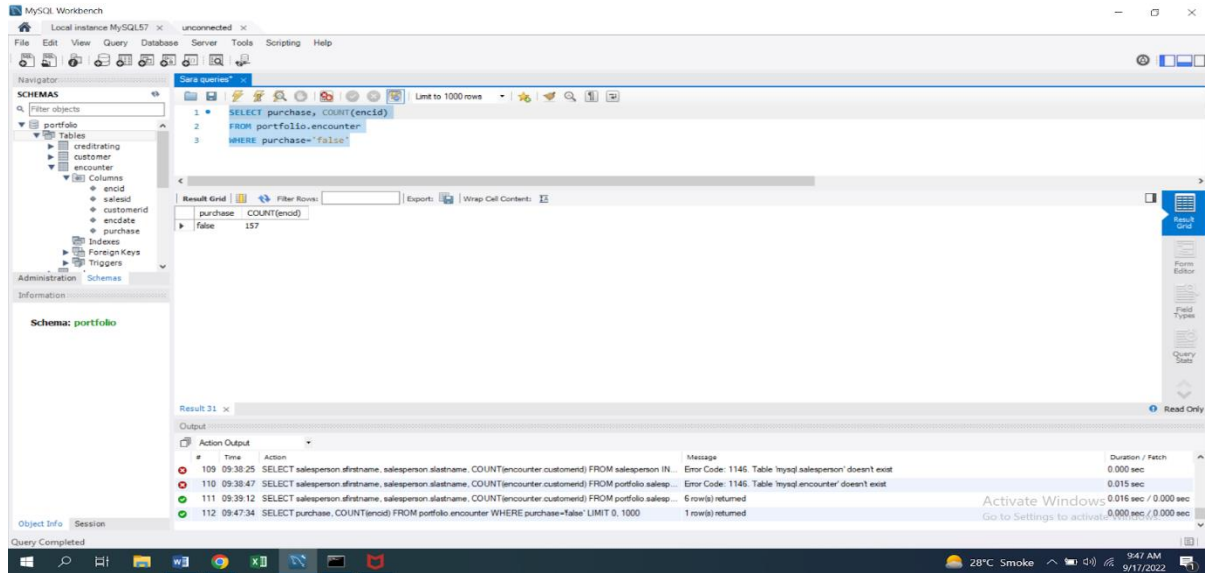
- 108 09:36:54 SELECT AVG(annualincome) FROM portfolio.customer JOIN portfolio.customer ON encounter.customerid=customer.customerid Error Code: 1065. Not unique table/alias: 'customer' 0.000 sec
- 109 09:38:25 SELECT salesperson.sfirstname, salesperson.slastname, COUNT(encounter.customerid) FROM salesperson INNER JOIN portfolio.encounter ON salesperson.salesid=encounter.salesid Error Code: 1146. Table 'mysq.salesperson' doesn't exist 0.000 sec
- 110 09:38:47 SELECT salesperson.sfirstname, salesperson.slastname, COUNT(encounter.customerid) FROM portfolio.salesperson INNER JOIN portfolio.encounter ON salesperson.salesid=encounter.salesid Error Code: 1146. Table 'mysq.encounter' doesn't exist 0.015 sec
- 111 09:39:12 SELECT salesperson.sfirstname, salesperson.slastname, COUNT(encounter.customerid) FROM portfolio.salesperson INNER JOIN portfolio.encounter ON salesperson.salesid=encounter.salesid 6 row(s) returned 0.016 sec / 0.000 sec

14. Count no. of false purchase

Query:

```
SELECT purchase, COUNT(encid)
FROM portfolio.encounter
WHERE purchase='false'
```

Result:



- In which day do they get the maximum order quantity?

Query:

SELECT encdate AS Date, count(customerid) AS Total_number_of_Customers FROM encounter WHERE purchase = 'TRUE' GROUP BY (encdate) order by count(purchase = 'TRUE') DESC

Result:

```
1 SELECT encdate AS Date, count(customerid) AS Total_number_of_Customers FROM encounter WHERE purchase = 'TRUE'
2 GROUP BY (encdate) order by count(purchase = 'TRUE') DESC
3
4
```

Date	Total_number_of_Customers
2015-07-11	4
2015-07-17	4
2015-07-21	4
2015-07-14	3
2015-07-23	3
2015-07-27	3
2015-07-29	3
2015-07-30	3
2015-07-01	2
2015-07-02	2
2015-07-04	2
2015-07-12	2
2015-07-13	2
2015-07-18	2
2015-07-19	2
2015-07-20	2
2015-07-24	2
2015-07-28	2
2015-07-31	2
2015-07-03	1
2015-07-05	1
2015-07-06	1
2015-07-07	1
2015-07-08	1
2015-07-09	1

15. Monthly/yearly salary of an employ

Query:

SELECT sfirstname, salary FROM salesperson

monthly/yearly salary of an employ SELECT sfirstname, salary FROM salesperson

Result:

```
1 • SELECT sfirstname, salary
2 FROM salesperson
3
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	sfirstname	salary		
▶	Harry	140000		
	Hermione	82000		
	Ron	93000		
	Dudley	56000		
	Katie	67000		
	Ginny	75000		
	Neville	71000		
	Colin	62000		

16. Which customer have more income?

Query:

```
SELECT annualincome, cfirstname FROM customer GROUP BY cfirstname order by
max(annualincome) DESC LIMIT 1
```

Result:

```
1 • SELECT annualincome, cfirstname
2 FROM customer
3 GROUP BY cfirstname
4 order by max(annualincome) DESC
5 LIMIT 1
```

Result Grid		Filter Rows:	Export:	W
	annualincome	cfirstname		
▶	124375	Sue		