

Validate Credit Worthiness of a Customer

- Query to create a table and columns in the Table

```
Create Table credit_card(  
u_id integer,  
LIMIT_BAL varchar(50),  
SEX varchar(2),  
EDUCATION varchar(2),  
MARRIAGE varchar(2),  
AGE integer,  
PAY_0 integer,  
PAY_2 integer,  
PAY_3 integer,  
PAY_4 integer,  
PAY_5 numeric,  
PAY_6 numeric,  
BILL_AMT1 numeric,  
BILL_AMT2 numeric,  
BILL_AMT3 numeric,  
BILL_AMT4 numeric,  
BILL_AMT5 numeric,  
BILL_AMT6 numeric,  
PAY_AMT1 numeric,  
PAY_AMT2 numeric,  
PAY_AMT3 numeric,  
PAY_AMT4 numeric,  
PAY_AMT5 numeric,  
PAY_AMT6 numeric,  
default_payment_next_month integer  
)
```

Validate Credit Worthiness of a Customer

- Query to fetch all the details of columns in the Table

select * from credit_card

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays the database structure: Servers (1) > PostgreSQL 10 > Databases (2) > credit_card_customer > credit_card. The main pane shows the 'Query' tab with the SQL query: `select * from credit_card`. Below the query, the 'Data Output' tab displays the results of the query. The table has 12 columns: u_id (integer), limit_bal (numeric), sex (character varying (2)), education (character varying (2)), marriage (character varying (2)), age (integer), pay_0 (integer), pay_2 (integer), pay_3 (integer), pay_4 (integer), pay_5 (numeric), and pay_6 (numeric). The results show 14 rows of data.

u_id	limit_bal	sex	education	marriage	age	pay_0	pay_2	pay_3	pay_4	pay_5	pay_6
1	20000	2	2	1	24	2	2	-1	-1	-2	
2	120000	2	2	2	26	-1	2	0	0	0	
3	90000	2	2	2	34	0	0	0	0	0	
4	50000	2	2	1	37	0	0	0	0	0	
5	50000	1	2	1	57	-1	0	-1	0	0	
6	50000	1	1	2	37	0	0	0	0	0	
7	500000	1	1	2	29	0	0	0	0	0	
8	100000	2	2	2	23	0	-1	-1	0	0	
9	140000	2	3	1	28	0	0	2	0	0	
10	20000	1	3	2	35	-2	-2	-2	-2	-1	
11	200000	2	3	2	34	0	0	2	0	0	
12	260000	2	1	2	51	-1	-1	-1	-1	-1	
13	630000	2	2	2	41	-1	0	-1	-1	-1	
14	70000	1	2	2	30	1	2	2	0	0	

Validate Credit Worthiness of a Customer

- Overall outstanding amount trends

```
select sum(bill_amt1) as "outstanding in Sept",sum(bill_amt2) as "outstanding in aug",sum(bill_amt3) as "outstanding in july",  
sum(bill_amt4) as "outstanding in june",sum(bill_amt5) as "outstanding in may",sum(bill_amt6) as "outstanding in april"  
from credit_card
```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Servers' tree is expanded to show 'PostgreSQL 10' and 'credit_card_customer'. The main pane displays a SQL query in the 'Query' tab. The query is:

```
1 select sum(bill_amt1) as "outstanding in Sept",sum(bill_amt2) as "outstanding in aug",sum(bill_amt3) as "outstanding in july",  
2 sum(bill_amt4) as "outstanding in june",sum(bill_amt5) as "outstanding in may",sum(bill_amt6) as "outstanding in april"  
3 from credit_card
```

Below the query, the 'Data Output' tab shows the results of the query. The results are displayed in a table with 6 columns: 'outstanding in Sept', 'outstanding in aug', 'outstanding in july', 'outstanding in june', 'outstanding in may', and 'outstanding in april'. The data is as follows:

	outstanding in Sept numeric	outstanding in aug numeric	outstanding in july numeric	outstanding in june numeric	outstanding in may numeric	outstanding in april numeric
1	1536699927	1475372255	1410394644	1297888469	1209342029	1166152812

The Windows taskbar at the bottom shows the time as 4:40 PM on 4/10/2018.

Validate Credit Worthiness of a Customer

- Age of outstanding amount analysis

```
select pay_0, count(u_id)
from credit_card
group by pay_0
```

The screenshot shows the pgAdmin 4 web interface. On the left, the 'Browser' pane displays a tree view of the database structure, with 'credit_card_customer' selected under 'PostgreSQL 10'. The main pane shows a SQL query editor with the following query:

```
1 select pay_0, count(u_id)
2 from credit_card
3 group by pay_0
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format:

	pay_0 integer	count bigint
1	-2	2759
2	6	11
3	7	9
4	0	14737
5	5	26
6	4	76
7	-1	5686
8	3	322
9	1	3688
10	2	2667
11	8	19

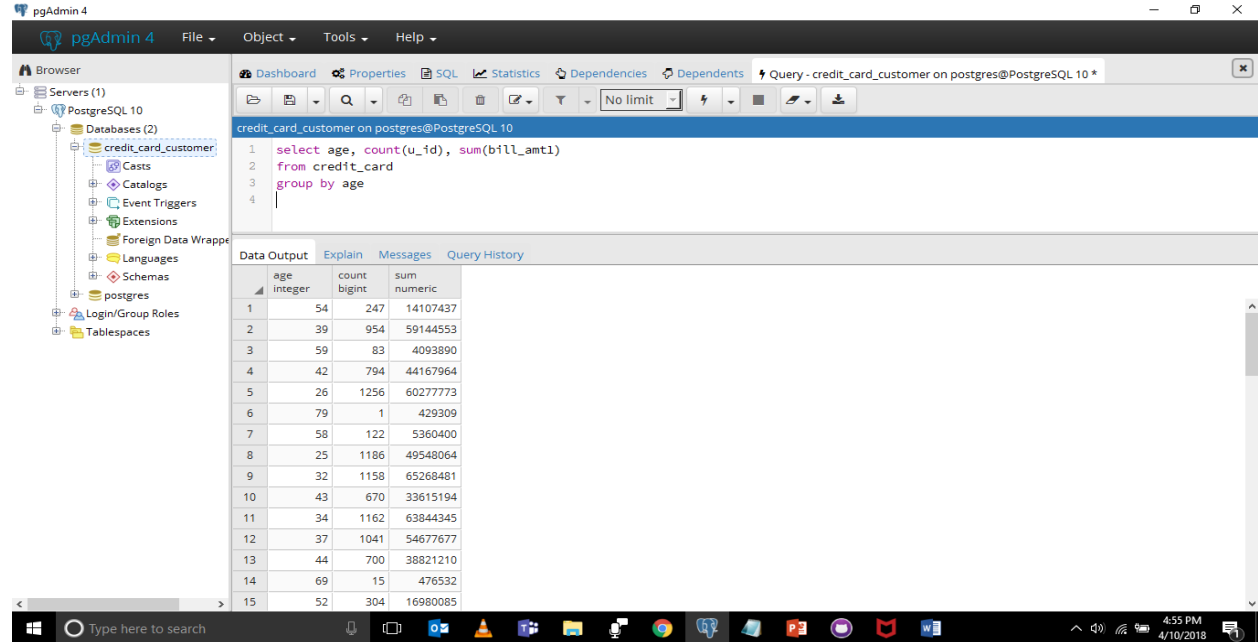
The bottom of the image shows the Windows taskbar with various application icons and the system clock indicating 4:48 PM on 4/10/2018.

Validate Credit Worthiness of a Customer

- Is there any relationship between outstanding amount / trend with respect to age, geo, education, marriage, credit limit

AGE

```
select age, count(u_id), sum(bill_amt1)
from credit_card
group by age
```



The screenshot shows the pgAdmin 4 interface. The left pane displays the database structure, including the 'credit_card_customer' database. The main pane shows a SQL query and its results. The query is:

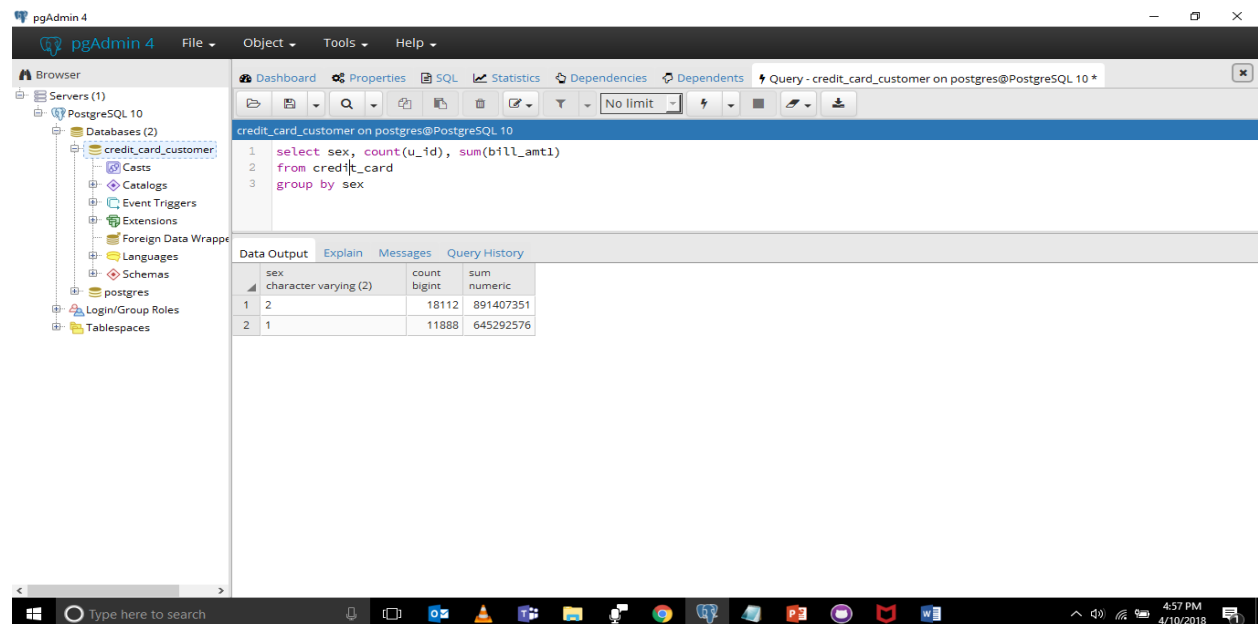
```
1 select age, count(u_id), sum(bill_amt1)
2 from credit_card
3 group by age
4
```

The results are displayed in a table with the following columns: age, count, and sum. The data is as follows:

age	count	sum
54	247	14107437
39	954	59144553
59	83	4093890
42	794	44167964
26	1256	60277773
79	1	429309
58	122	5360400
25	1186	49548064
32	1158	65268481
43	670	33615194
34	1162	63844345
37	1041	54677677
44	700	38821210
69	15	476532
52	304	16980085

GENDER

```
select sex, count(u_id), sum(bill_amt1)
from credit_card
group by sex
```



The screenshot shows the pgAdmin 4 interface. The left pane displays the database structure, including the 'credit_card_customer' database. The main pane shows a SQL query and its results. The query is:

```
1 select sex, count(u_id), sum(bill_amt1)
2 from credit_card
3 group by sex
```

The results are displayed in a table with the following columns: sex, count, and sum. The data is as follows:

sex	count	sum
2	18112	891407351
1	11888	645292576

Validate Credit Worthiness of a Customer

EDUCATION

```
select education, count(u_id), sum(bill_amt1)
from credit_card
group by education
```

A screenshot of the pgAdmin 4 interface. The left sidebar shows a tree view of the database structure, with 'credit_card_customer' selected under 'PostgreSQL 10'. The main pane displays a SQL query and its results. The query is: `select education, count(u_id), sum(bill_amt1) from credit_card group by education`. The results are shown in a table with 7 rows and 3 columns: education, count, and sum.

education	count	sum
0	14	164933
6	51	4190431
4	123	6719724
1	10585	516817256
3	4917	233869956
5	280	22851985
2	14030	752085642

MARRIAGE

```
select marriage, count(u_id), sum(bill_amt1)
from credit_card
group by marriage
```

A screenshot of the pgAdmin 4 interface, similar to the one above. The left sidebar shows the same database structure. The main pane displays a SQL query and its results. The query is: `select marriage, count(u_id), sum(bill_amt1) from credit_card group by marriage`. The results are shown in a table with 4 rows and 3 columns: marriage, count, and sum.

marriage	count	sum
2	15964	794199657
1	13659	727452785
3	323	13945852
0	54	1101633