

SQL INDIVIDUAL PROJECT

INFO8075-22W-Sec1-SQL and Data Analysis

Conestoga College, Doon, Kitchener

Name- Amritjot Kaur

Student Id-8798254

Date-22/02/2022

Introduction:

The goal of this project is to get familiar with the SQL basic functions. This will include 6 tables in all, on which I will be performing different SQL functions like: Filters, Joins, Sub-queries, CTEs. Forthcoming, will be the description of all Data-Sets that were made.

Data-sets:

- 1-Associates: containing Associates' numbers, associates name, location name, commission of each associate, phone number and email
- 2-Consumers: containing consumers' numbers, consumers' names, city, rating to each associate, associates' numbers
- 3-Insurances: includes Insurance numbers, premium, issue dates, insurance type, associates' number who were able to get the consumers an insurance policy and consumers numbers.
- 4-ConstructionCodes: contains construction code type, construction codes and descriptions.
- 5-OccupancyCodes: contains occupancy code type, occupancy codes and descriptions.
- 6-Property_Types: includes location number, consumers' number, address, countryISO codes, city, construction and occupancy codes, dollars of damage that might have happen due to different catastrophes and reasons of damage.

Objectives and Goals:

In this assignment is will be performing several SQL queries and will be providing the answers to each of them.

- 1- Which associates were able to sell the insurance policy to different consumers and how much commission did each associate secure?
- 2- Return all the information from Consumers and Insurances tables.
- 3-Using the ConstructionCodes, OccupancyCodes and Property_Types, find out the respective code description of the property's construction and occupancy, next to their code columns, where Damage in dollars is 889853,664123,12547.
- 4-Return only first 6 rows of consumers' property along with the type of insurance they opted for, sort the data by insurance number in decreasing order.
- 5- Find out the maximum and minimum loss that happened due to Earthquake.

6- Return distinct associates name, email, phone number, with the property damage in dollars.

7- Create a view for end users in the team but not all can have access to your data. Write a query to create a view with all the associates, who sold the policy and consumers who bought only property insurance with their property details.

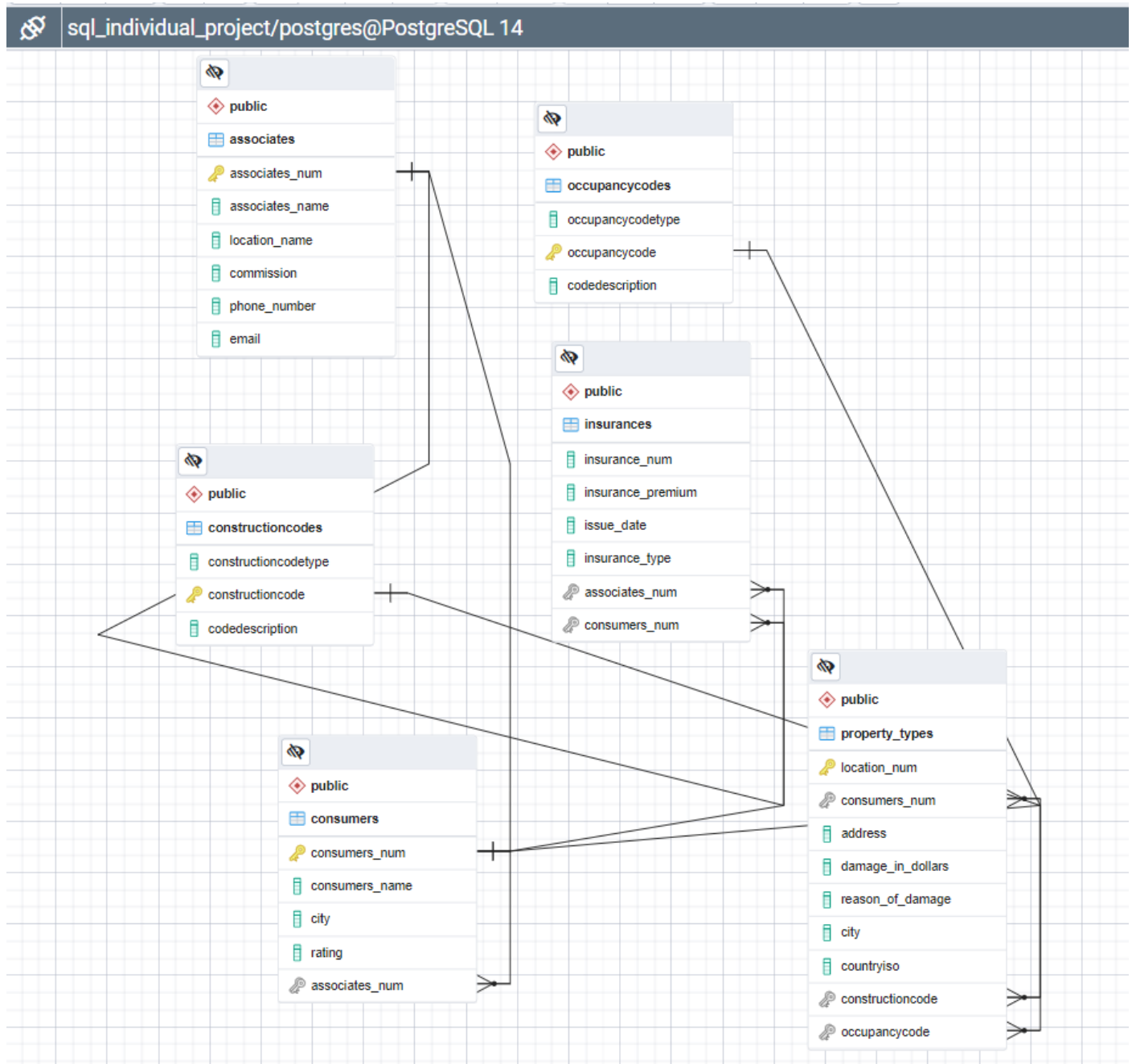
8- Write a query to find the vulnerabilities of the property based on construction and occupancy types, when a catastrophe hits, along with consumer and its property information. Add a new column called "Catastrophe_Prone_Areas" and group into the following Vulnerabilities: Highly Vulnerable, Medium Vulnerable, Less Vulnerable, not applicable. The range is as follows:

Catastrophe Prone Areas	Damage to property
Highly Vulnerable	>140K
Medium Vulnerable	12K-140K
Less Vulnerable	<12K
Not Applicable	0

9- Count number of consumers have property at different locations and only return the value if the person has 2 or more properties.

10-Change the names of associates with associate numbers "2008","2017","2025","2011" by "Dario"," Kim","Oleg","Deepa",in such a way that if by chance we make any mistake we can undo the mistake.






Relational Schema:



Analysis:

1- Which associates were able to sell the insurance policy to different consumers and how much commission did each associate secure?

```
Code- SELECT a.associates_num, a.associates_name, c.consumers_num, c.consumers_name
        FROM associates a
        INNER JOIN consumers c
        ON a.associates_num=c.associates_num
```

Data Output		Explain	Messages	Notifications				
	associates_num integer		associates_name character varying (100)		consumers_num integer		consumers_name character varying (100)	
1		2008	Salem		3001		Josh	
2		2002	Peter		3002		Gio	
3		2006	Amy		3003		Leo	
4		2001	Paul		3004		Green	
5		2012	Alex		3005		Chloe	
6		2001	Paul		3006		Clinton	
7		2012	Alex		3007		Bill	
8		2014	Vee		3008		Helen	
9		2011	John		3009		Ellen	
10		2010	Nichole		3010		Steel	
11		2007	North		3011		Catherine	
12		2012	Alex		3012		Austin	
13		2013	Wanda		3013		Milly	
14		2010	Nichole		3014		Rock	
15		2008	Salem		3015		Selena	

Insights- Here the question demanded to show the name and number of all those associates who were able to secure their commission by selling the policy to certain consumers. We can see there are 15 associates out of 30 who were able to sell the policy, which we gathered from the insurances table by joining insurances with associates' table.

2- Return all the information from Consumers and Insurances tables.

Code- *SELECT **

FROM Consumers C

JOIN Insurances I

ON C.Consumers_Num=I.Consumers_Num












Data Output	Explain	Messages	Notifications								
	consumers_num integer	consumers_name character varying (100)	city character varying (100)	rating integer	associates_num integer	insurance_num integer	insurance_premium integer	issue_date date	insurance_type character varying (100)	associates_num integer	consumers_num integer
1	3002	Gio	London	2	2002	4001	1852	2021-03-12	Life and PropertyInsurance	2008	3002
2	3015	Selena	Canberra	5	2008	4002	1555	2021-09-15	Property Insurance	2019	3015
3	3010	Steel	California	5	2010	4003	1259	2020-04-10	Property Insurance	2023	3010
4	3012	Austin	London	3	2012	4004	8956	2021-10-09	Life and PropertyInsurance	2005	3012
5	3007	Bill	Canberra	4	2012	4005	2658	2021-05-04	Property Insurance	2006	3007
6	3001	Josh	San Jose	2	2008	4006	1445	2021-03-19	Property Insurance	2022	3001
7	3008	Helen	Berlin	4	2014	4007	4570	2021-06-27	Third Party liability	2002	3008
8	3011	Catherine	Berlin	3	2007	4008	1676	2021-01-31	Property Insurance	2001	3011
9	3004	Green	New York	5	2001	4009	2322	2022-05-28	Third Party liability	2012	3004
10	3003	Leo	Barcelona	5	2006	4010	9858	2021-11-09	Life and PropertyInsurance	2006	3003
11	3005	Chloe	Sydney	3	2012	[null]	[null]	[null]	[null]	[null]	[null]
12	3014	Rock	Barcelona	4	2010	[null]	[null]	[null]	[null]	[null]	[null]
13	3009	Ellen	California	3	2011	[null]	[null]	[null]	[null]	[null]	[null]
14	3006	Clinton	Paris	3	2001	[null]	[null]	[null]	[null]	[null]	[null]
15	3013	Milly	Sydney	5	2013	[null]	[null]	[null]	[null]	[null]	[null]

Insights- Here, the question demanded to show all the entries from both consumers and insurances table.

So, Left join was used to show every input of these 2 tables.

3-Using the ConstructionCodes, OccupancyCodes and Property_Types, find out the respective code description of the property's construction and occupancy, next to their code columns, where Damage in dollars is 889853,664123,12547.









```
Code- SELECT PT.location_num, PT.consumers_num, PT.address, PT.city, PT.countryISO,
        PT.Constructioncode, CC.Codedescription, PT.Occupancycode, OC.codedescription,
        PT.damage_in_dollars
FROM Property_Types PT, OccupancyCodes OC, ConstructionCodes CC
WHERE PT.OccupancyCode=OC.OccupancyCode AND
        PT.ConstructionCode=CC.ConstructionCode AND
        PT.Damage_in_Dollars IN (889853,664123,12547)
```

Data Output										Explain	Messages	Notifications							
 location_num integer		consumers_num integer		address character varying (100)		city character varying (100)		countryiso character varying (10)		constructioncode integer		codedescription character varying (100)		occupancycode integer		codedescription character varying (100)		damage_in_dollars integer	
1		1		3002 63 New Road		London		GB		111		Frame		300		Unknown		12547	
2		2		3015 6 Chester Street		Sydney		AU		101		Wood		301		General Commercial		664123	
3		15		3007 Celso Emilio Ferreiro 58		Barcelona		ES		151		Masonry		315		Offices		889853	

*Insights- Here the question asked to show the construction codes and occupancy code along with the respective code description, so rather than using *(all) in the select query I must type all the column names just to place them by the order of codes and their descriptions. Moreover, to join 3 tables I used subquery here, using WHERE clause, in which I also defined the condition of damage in dollars.*

4-Return only first 6 rows of consumers' property along with the type of insurance they opted for, sort the data by insurance number in decreasing order.

```
Code- SELECT I.insurance_type, I.Insurance_num, C.*
        From consumers C
        INNER JOIN insurances I
        ON C.consumerS_num=I.consumerS_num
        ORDER BY I.Insurance_Num DESC
        LIMIT 6
```

Data Output		Explain	Messages	Notifications				
	insurance_type character varying (100) 	insurance_num integer 	consumers_num integer 	consumers_name character varying (100) 	city character varying (100) 	rating integer 	associates_num integer 	
1	Life and PropertyInsurance	4010	3003	Leo	Barcelona	5	2006	
2	Third Party liability	4009	3004	Green	New York	5	2001	
3	Property Insurance	4008	3011	Catherine	Berlin	3	2007	
4	Third Party liability	4007	3008	Helen	Berlin	4	2014	
5	Property Insurance	4006	3001	Josh	San Jose	2	2008	
6	Property Insurance	4005	3007	Bill	Canberra	4	2012	

Insights- This question asked to show the consumers with the type of insurances they buy. So, I joined the consumers and insurances table. Further, it asked to sort the data in descending order of insurance number, which was achieved by "ORDER BY" function with DESC (meaning descending order). Lastly, to retrieve only top 6 rows "LIMIT" function was used.

5- Find out the maximum and minimum loss that happened due to Earthquake.

```
Code- SELECT MAX(damage_in_dollars) as Max_damage_EQ,
        MIN(damage_in_dollars) as Min_damage_EQ
FROM property_types
WHERE reason_of_damage in ('Earthquake')
```

Data Output	Explain	Messages	Notifications
	max_damage_eq integer	min_damage_eq integer	
1	664123	24879	

Insights- Here, to find the maximum and minimum loss caused by Earthquake, we used MAX and MIN functions of SQL, in the table from where we need to extract the data and WHERE clause was used to apply the condition.

6- Return distinct associates name, email, phone number, with the property damage in dollars.

```
Code- SELECT DISTINCT ON (a.associates_num), a.associates_num, a.associates_name, a.email,
        a.phone_number, c.consumers_num, p.damage_in_dollars
FROM associates a
INNER JOIN consumers c on a.associates_num=c.associates_num
inner join property_types p on p.consumers_num=c.consumers_num
group by a.associates_num, c.consumers_num, p.damage_in_dollars
```

Data Output		Explain	Messages	Notifications		
	<div>associates_num</div> <div>integer</div>	<div>associates_name</div> <div>character varying (100)</div>	<div>email</div> <div>character varying (100)</div>	<div>phone_number</div> <div>double precision</div>	<div>consumers_num</div> <div>integer</div>	<div>damage_in_dollars</div> <div>integer</div>
1	2001	Paul	torquent.per.conubia@hotmail.net	3654887650	3004	6131
2	2002	Peter	nec.metus.facilisis@hotmail.ca	4500056789	3002	12547
3	2006	Amy	bibendum@protonmail.org	7896541235	3003	2659
4	2007	North	molestie.orci.tincidunt@aol.org	5896478522	3011	1359
5	2008	Salem	ornare@outlook.couk	4588632110	3001	57876
6	2010	Nichole	pretium.aliquet@aol.edu	4558962125	3010	132354
7	2012	Alex	tellus@google.net	5648562265	3007	54121
8	2014	Vee	nonummy.ipsium@hotmail.net	2256987441	3008	5312

Insights- In order to get the distinct associates name, email, phone number along with the property damage, we combined 3 tables together and applied “DISTINCT” function of SQL.

7- Create a view for end users in the team but not all can have access to your data. Write a query to create a view with all the associates, who sold the policy and consumers who bought only property insurance with their property details.

CODE- *CREATE OR REPLACE VIEW Insurance_Policy_Sales AS*

```
SELECT A.associates_num,A.associates_name,A.commission,A.phone_number, A.email,
C.consumers_num,C.consumers_name,I.insurance_num, I.issue_date, I.insurance_type
FROM Associates A
LEFT JOIN Consumers C
ON A.Associates_num = C.associates_num
LEFT JOIN Insurances I ON I.Consumers_Num=C.consumers_num
WHERE I.insurance_type in ('Property Insurance');
```

	associates_num integer	associates_name character varying (100)	commission character varying (100)	phone_number double precision	email character varying (100)	consumers_num integer	consumers_name character varying (100)	insurance_num integer	issue_date date	insurance_type character varying (100)
1	2008	Salem	0.15	4588632110	ornare@outlook.co.uk	3001	Josh	4006	2021-03-19	Property Insurance
2	2012	Alex	0.25	5648562265	tellus@google.net	3007	Bill	4005	2021-05-04	Property Insurance
3	2010	Nichole	0.28	4558962125	pretium.aliquet@aol.edu	3010	Steel	4003	2020-04-10	Property Insurance
4	2007	North	0.25	5896478522	molestie.orci.tincidunt@...	3011	Catherine	4008	2021-01-31	Property Insurance
5	2008	Salem	0.15	4588632110	ornare@outlook.co.uk	3015	Selena	4002	2021-09-15	Property Insurance

Insights- In this case, we had to make a separate view for end users, in order to prevent any unnecessary updates in the data-set. This way we can keep our original data safe and can give the access of the data that is required. To create the separate view, “VIEW” function is used with “SELECT” query.

8- Write a query to find the vulnerabilities of the property based on construction and occupancy types, when a catastrophe hits, along with consumer and its property information. Add a new column called “Catastrophe_Prone_Areas” and group into the following Vulnerabilities: Highly Vulnerable, Medium Vulnerable, Less Vulnerable, not applicable. The range is as follows:

Catastrophe Prone Areas	Damage to property
Highly Vulnerable	>140K
Medium Vulnerable	12K-140K
Less Vulnerable	<12K
Not Applicable	0

Code- *SELECT DISTINCT COUNT(c.consumers_num) AS Con_count,*

```
c.consumers_num,c.consumers_name,P.constructioncode, CC.codedescription,
```


P.occupancycode, OC.codedescription,p.reason_of_damage,

CASE

WHEN damage_in_dollars > 140000 THEN 'Highly_Vulnerable'

WHEN damage_in_dollars BETWEEN 12000 AND 140000 THEN 'Medium_Vulnerable'

WHEN damage_in_dollars < 12000 THEN 'Less_Vulnerable'

WHEN damage_in_dollars = 0 THEN 'not_applicable'

END AS Catastrophe_Prone_Areas

FROM Consumers C, Property_types P, Constructioncodes CC, Occupancycodes OC

where C.consumers_num = P.consumers_num

and P.constructioncode = CC.constructioncode

and P.occupancycode=OC.occupancycode

GROUP BY c.consumers_num, p.damage_in_dollars, p.constructioncode, cc.codedescription,

p.occupancycode,oc.codedescription,p.reason_of_damage;

sql_individual_project/postgres@PostgreSQL 14 ▾										
Data Output										
	con_count bigint	consumers_num integer	consumers_name character varying (100)	constructioncode integer	codedescription character varying (100)	occupancycode integer	codedescription character varying (100)	reason_of_damage character varying (100)	vulnerability text	
1	1	3015	Selena	101	Wood	301	General Commercial	Earthquake	Highly_Vulnerable	
2	1	3003	Leo	100	Unknown	312	Retail Shops	Flood	Less_Vulnerable	
3	1	3012	Austin	103	Non Combustible	345	Schools	Earthquake	Medium_Vulnerable	
4	1	3007	Bill	151	Masonry	315	Offices	Flood	Highly_Vulnerable	
5	1	3008	Helen	151	Masonry	444	Fertilizer Plants	Fire	Less_Vulnerable	
6	1	3002	Gio	111	Frame	300	Unknown	Flood	Medium_Vulnerable	
7	1	3004	Green	100	Unknown	331	Restaurant	Hurricane	Highly_Vulnerable	
8	1	3012	Austin	133	Reinforced Concrete	312	Retail Shops	Fire	Medium_Vulnerable	
9	1	3010	Steel	111	Frame	321	General Industrial	Fire	Highly_Vulnerable	
10	1	3001	Josh	133	Reinforced Concrete	438	Chemical Plants	Hurricane	Medium_Vulnerable	
11	1	3008	Helen	103	Non Combustible	305	Housing	Fire	Medium_Vulnerable	
12	1	3004	Green	101	Wood	361	Utility	Fire	Medium_Vulnerable	
13	1	3004	Green	131	Concrete	312	Retail Shops	Flood	Medium_Vulnerable	
14	1	3010	Steel	102	Steel	316	Heath care	Fire	Medium_Vulnerable	
15	1	3011	Catherine	101	Wood	313	Wholesale	Fire	Less_Vulnerable	
16	1	3002	Gio	131	Concrete	313	Wholesale	Flood	Medium_Vulnerable	
17	1	3007	Bill	131	Concrete	432	Pharmaceutical Plants	Earthquake	Medium_Vulnerable	
18	1	3002	Gio	115	Steel Frame	404	Industrial Machinery	Earthquake	Medium_Vulnerable	
19	1	3011	Catherine	152	Masonry Veneer	482	Gas Processing Plants	Flood	Medium_Vulnerable	
20	1	3004	Green	111	Frame	310	Entertainment	Fire	Less_Vulnerable	

Insights- In this query we must show the how the construction and occupancy of a building affect the damage caused by any natural catastrophe. Also, we want the consumers information, plus the property information, so we combined consumers with property_types table, construction and occupancy also, and extracted the information on the vulnerability curve.

9- Count number of consumers have property at different locations and only return the value if the person has 2 or more properties.

Code- *select distinct count(p.consumers_num) as con_count,c.consumers_name,c.consumers_num*
from property_types p

```

inner join consumers c
on p.consumers_num=c.consumers_num
group by c.consumers_num
having count (p.consumers_num)>=2

```

Data Output			
	con_count bigint	consumers_name character varying (100)	consumers_num [PK] integer
1	2	Bill	3007
2	2	Catherine	3011
3	4	Green	3004
4	3	Gio	3002
5	2	Helen	3008
6	2	Austin	3012
7	2	Steel	3010

Insights- This query asked to show the values only when a consumers have property 2 or more. In order to apply the condition “HAVING” clause can also be used, just like in this case. I joined 2 tables for information and applied the HAVING condition and retrieved the data.

10-Change the names of associates with associate numbers “2008” ,”2017”,”2025”,”2011” by “Dario”,”Kim”,”Oleg”,”Deepa”,in such a way that if by chance we make any mistake we can undo the mistake.

Code- Begin;

```

update associates set associates_name ='Dario' where associates_num= 2008;
update associates set associates_name ='Kim' where associates_num= 2017;
update associates set associates_name ='Oleg' where associates_num= 2025;
update associates set associates_name ='Deepa' where associates_num= 2011;
select * from associates
Rollback;

```

Snip-1: Message-After executing the UPDATE query

Messages

UPDATE 1

Query returned successfully in 97 msec.

Snip-2: Message-After executing the ROLLBACK query

Messages

ROLLBACK

Query returned successfully in 71 msec.

Snip-3: Data Output- After UPDATE query execution

Data Output

	associates_num [PK] integer	associates_name character varying (100)	location_name character varying (100)	commission character varying (100)	phone_number double precision	email character varying (100)
9	2010	Nichole	San Jose	0.28	4558962125	pretium.aliquet@aol.edu
10	2012	Alex	California	0.25	5648562265	tellus@google.net
11	2013	Wanda	New York	0.18	5578964125	nulla@aol.ca
12	2014	Vee	New York	0.1	2256987441	nonummy.ipsam@hotmail.net
13	2015	Monika	London	0.12	3366987452	quam.quis@protonmail.couk
14	2016	Rachel	Sydney	0.18	6459103137	consequat.enim.diam@aol.org
15	2018	Joe	San Jose	0.15	8694636166	ornare.facilisis@protonmail.couk
16	2019	Chandler	California	0.19	8564210940	venenatis.vel@hotmail.ca
17	2020	Rio	Berlin	0.13	9518233436	mollis.non.cursus@outlook.com
18	2021	Jennifer	London	0.15	5990652651	urna@aol.couk
19	2022	Jordan	California	0.25	8692467497	vehicula.et@yahoo.ca
20	2023	Danielle	Sydney	0.23	6039352650	eu.metus@protonmail.com
21	2024	Melanie	Sydney	0.24	6200828330	pede.nonummy@aol.couk
22	2026	Courtney	Barcelona	0.15	9824560824	adipiscing.elit@icloud.net
23	2027	Joseph	Barcelona	0.22	8473578075	malesuada.ut.sem@aol.edu
24	2028	April	Paris	0.2	7652004778	ante.laculis.nec@icloud.org
25	2029	Teresa	Paris	0.19	8252078502	phasellus.nulla@protonmail.couk
26	2030	Tabatha	Canberra	0.1	9977516170	eros.non@google.com
27	2008	Dario	London	0.15	4588632110	ornare@outlook.couk
28	2017	Kim	Berlin	0.17	7096241497	vitae.diam.proin@yahoo.ca
29	2025	Oleg	San Jose	0.27	7642539816	at@outlook.edu
30	2011	Deepa	California	0.55	4776510014	scelerisque@outlook.couk

Query Editor Query History Data Output Explain Messages Notifications

Insights- In this query we updated the names of the associates as per the questions. The thing to pay attention is, the chance of making mistakes with UPDATE and DELETE queries. Therefore, in order to undo the mistakes in updating and deleting “BEGIN TRANSACTION” along with “ROLLBACK” works as a magic wand. We execute the UPDATE or DELETE queries with “BEGIN” written at the top of the query, then executing the update query with BEGIN, then it will show the message as that of SNIP-1. Then if you want to check the data output after update, just check the SNIP-3, which shows that the respective names were

changes. Furthermore, if we made any mistake just execute the “ROLLBACK” and it will undo every change that we made and gives the message as that of SNIP-2.