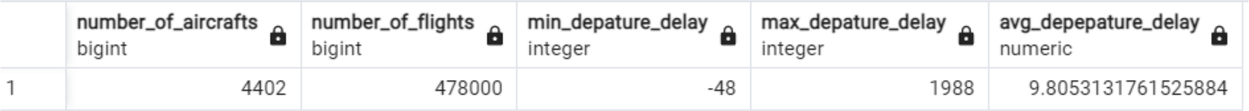
Part A: SQL using PostgreSQL Server (50 points)

(a) Provide the key summary statistics of the data contained in the table by retrieving the number of distinct aircrafts, total number of flights as well as a few statistics about flights departure delays (e.g., min, max & avg departure delays):

```
 SELECT

     COUNT(DISTINCT TAIL_NUMBER) AS number_of_aircrafts,

     COUNT(*) AS number_of_flights,

     MIN(DEPARTURE_DELAY) AS min_depature_delay,

     MAX(DEPARTURE_DELAY) AS max_depature_delay,

     AVG(DEPARTURE_DELAY) AS avg_depepature_delay

FROM
```

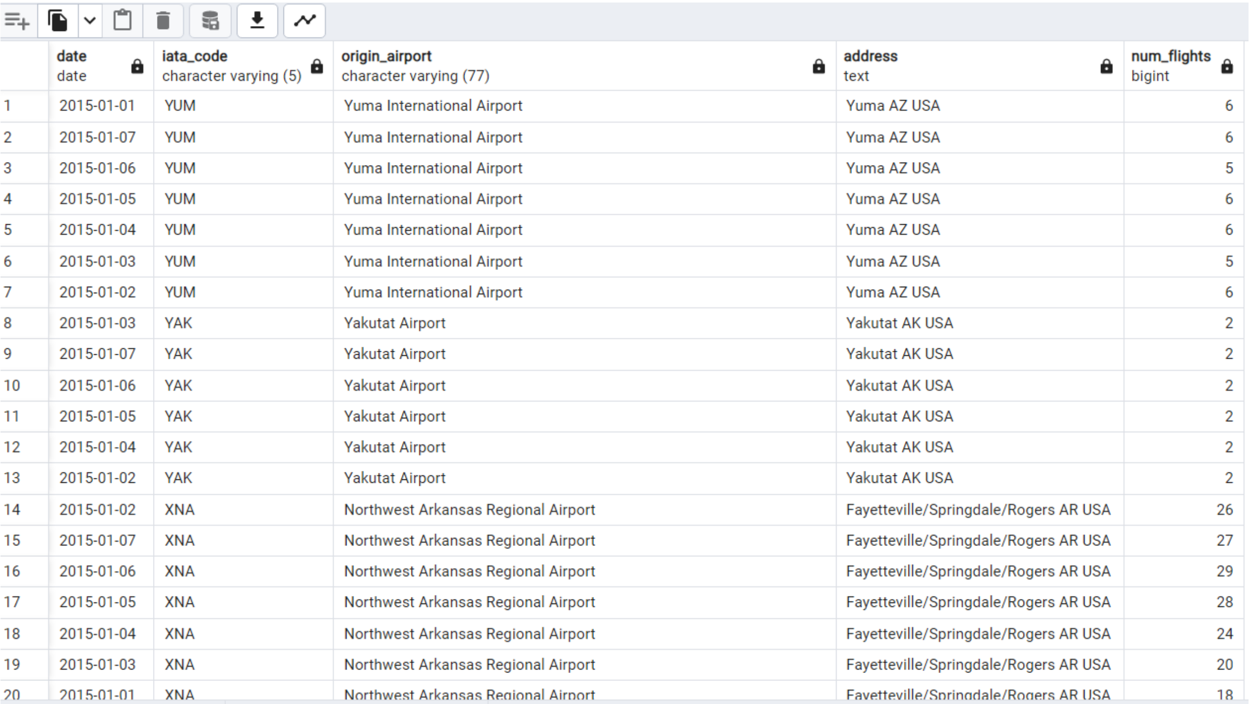


	number_of_aircrafts	number_of_flights	min_depature_delay	max_depature_delay	avg_depepature_delay
	bigint	bigint	integer	integer	numeric
1	4402	478000	-48	1988	9.8053131761525884

(b) Create a view called FlightSummaryView to display the date (e.g., 2015-01-01), iata_code, origin_airport, concatenated city, state and country renamed as Address, and the total number of flights departing from each airport for the first week of 2015. Use the JOIN ON syntax and order by the iata_code in descending order (Make sure to add space between the address if required):

```
CREATE VIEW FlightSummaryView AS

SELECT

    TO_DATE(year || '-' || month || '-' || day, 'YYYY-MM-DD') AS date,

    f.ORIGIN_AIRPORT AS IATA_CODE,

    a.AIRPORT AS ORIGIN_AIRPORT,

    CONCAT_WS(' ', a.CITY, a.STATE, a.COUNTRY) AS Address,

    COUNT(f.*) AS num_flights



FROM

    FLIGHTS f

JOIN

    AIRPORTS a ON f.ORIGIN_AIRPORT = a.IATA_CODE

WHERE
```

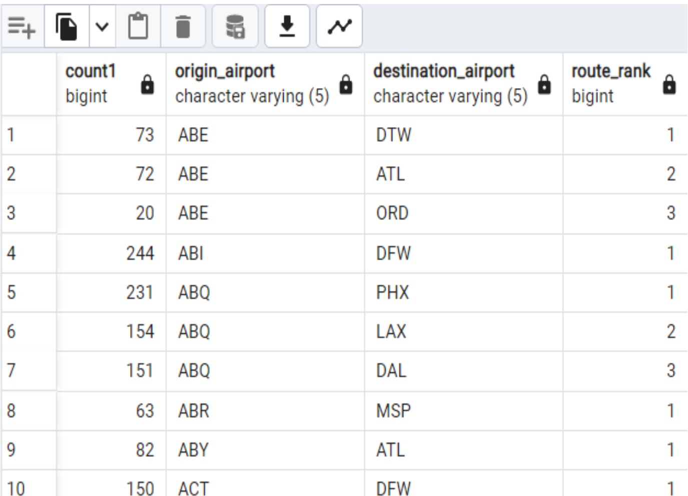
```
    year = 2015 AND month = 1 AND day BETWEEN 1 AND 7
GROUP BY
   ORIGIN_AIRPORT, date, IATA_CODE,    Address
ORDER BY
    IATA_CODE DESC;
```



	date date	iata_code character varying (5)	origin_airport character varying (77)	address text	num_flights bigint
1	2015-01-01	YUM	Yuma International Airport	Yuma AZ USA	6
2	2015-01-07	YUM	Yuma International Airport	Yuma AZ USA	6
3	2015-01-06	YUM	Yuma International Airport	Yuma AZ USA	5
4	2015-01-05	YUM	Yuma International Airport	Yuma AZ USA	6
5	2015-01-04	YUM	Yuma International Airport	Yuma AZ USA	6
6	2015-01-03	YUM	Yuma International Airport	Yuma AZ USA	5
7	2015-01-02	YUM	Yuma International Airport	Yuma AZ USA	6
8	2015-01-03	YAK	Yakutat Airport	Yakutat AK USA	2
9	2015-01-07	YAK	Yakutat Airport	Yakutat AK USA	2
10	2015-01-06	YAK	Yakutat Airport	Yakutat AK USA	2
11	2015-01-05	YAK	Yakutat Airport	Yakutat AK USA	2
12	2015-01-04	YAK	Yakutat Airport	Yakutat AK USA	2
13	2015-01-02	YAK	Yakutat Airport	Yakutat AK USA	2
14	2015-01-02	XNA	Northwest Arkansas Regional Airport	Fayetteville/Springdale/Rogers AR USA	26
15	2015-01-07	XNA	Northwest Arkansas Regional Airport	Fayetteville/Springdale/Rogers AR USA	27
16	2015-01-06	XNA	Northwest Arkansas Regional Airport	Fayetteville/Springdale/Rogers AR USA	29
17	2015-01-05	XNA	Northwest Arkansas Regional Airport	Fayetteville/Springdale/Rogers AR USA	28
18	2015-01-04	XNA	Northwest Arkansas Regional Airport	Fayetteville/Springdale/Rogers AR USA	24
19	2015-01-03	XNA	Northwest Arkansas Regional Airport	Fayetteville/Springdale/Rogers AR USA	20
20	2015-01-01	XNA	Northwest Arkansas Regional Airport	Fayetteville/Springdale/Rogers AR USA	18

(c) Display the origin_airport, destination_airport, and the rank for the top 3 routes departing from each airpor:

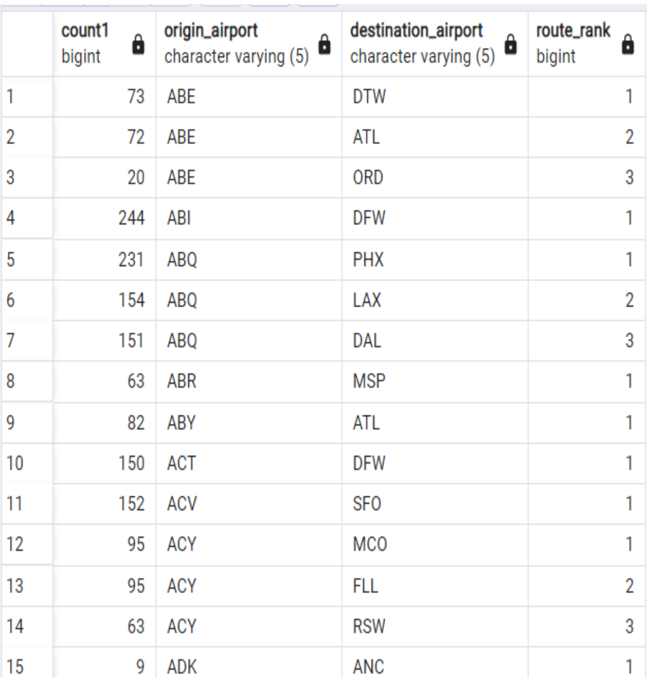
```
Select f.count1,f.origin_airport, f.destination_airport, f.route_rank
from
(
Select    count(*)as count1,origin_airport, destination_airport,
Row_number () over (partition by origin_airport order by count(*) DESC ) as route_rank
from flights
Group by origin_airport, destination_airport
) as f
WHERE route_rank <= 3;
```



	count1 bigint	origin_airport character varying (5)	destination_airport character varying (5)	route_rank bigint
1	73	ABE	DTW	1
2	72	ABE	ATL	2
3	20	ABE	ORD	3
4	244	ABI	DFW	1
5	231	ABQ	PHX	1
6	154	ABQ	LAX	2
7	151	ABQ	DAL	3
8	63	ABR	MSP	1
9	82	ABY	ATL	1
10	150	ACT	DFW	1

(d) Display the airport iata_code, airport name, airline iata_code, airline name, flight_number, tail_number, origin_airport, destination_airport, departure_time, and arrival_time for all flights that fly on weekends (Saturdays and Sundays) and landed between 4 and 5 am

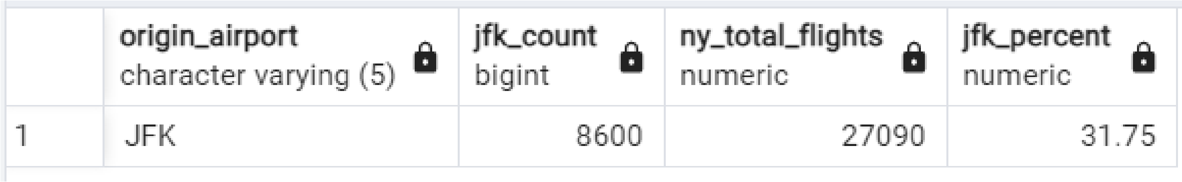
```
 SELECT
        a.iata_code AS airport_iata_code,
        a.airport AS air_name,
        f.airline AS airline_iata_code,
        an.airline AS airline_name,
        f.flight_number AS flight_number,
        f.tail_number AS tail_number,
        f.origin_airport AS origin_airport,
        f.destination_airport AS destination_airport,
        f.departure_time AS departure_time,
        f.arrival_time AS arrival_time
FROM
        flights f
JOIN
        airports a ON f.origin_airport = a.iata_code
JOIN
        airlines an ON f.airline = an.iata_code
WHERE
        f.day_of_week IN (1, 7)
        AND f.arrival_time >= '0400' AND f.arrival_time <= '0500'
ORDER BY
        f.arrival_time;
```



	count1 bigint	origin_airport character varying (5)	destination_airport character varying (5)	route_rank bigint
1	73	ABE	DTW	1
2	72	ABE	ATL	2
3	20	ABE	ORD	3
4	244	ABI	DFW	1
5	231	ABQ	PHX	1
6	154	ABQ	LAX	2
7	151	ABQ	DAL	3
8	63	ABR	MSP	1
9	82	ABY	ATL	1
10	150	ACT	DFW	1
11	152	ACV	SFO	1
12	95	ACY	MCO	1
13	95	ACY	FLL	2
14	63	ACY	RSW	3
15	9	ADK	ANC	1

(e) All New York flights originate in one of 3 airports: ‘JFK’ (Kennedy), ‘LGA’ (La Guardia), and ‘EWR’ (Newark in New Jersey). Count how many flights originate at ‘JFK.’ Then show how many flights originate at ‘JFK’ as a percentage of all flights. (hint: use a WITH clause or a FROM subquery)

```
WITH jfk_counting AS (

    SELECT ORIGIN_AIRPORT, COUNT(*) AS jfk_count

    FROM flights

    WHERE ORIGIN_AIRPORT = 'JFK'

    GROUP BY ORIGIN_AIRPORT

),

ny_counting AS (

    SELECT origin_airport, SUM(ny_count) OVER () AS ny_total_flights

    FROM (

        SELECT ORIGIN_AIRPORT, COUNT(*) AS ny_count

        FROM flights

        WHERE ORIGIN_AIRPORT IN ('JFK', 'LGA', 'EWR')

        GROUP BY ORIGIN_AIRPORT

    ) AS f

    GROUP BY origin_airport, ny_count

)


SELECT jfk_counting.ORIGIN_AIRPORT, jfk_count, ny_total_flights, ROUND(((jfk_count / ny_total_flights) * 100), 2) as JFK_Percent

FROM jfk_counting

JOIN ny_counting ON jfk_counting.ORIGIN_AIRPORT = ny_counting.origin_airport;
```

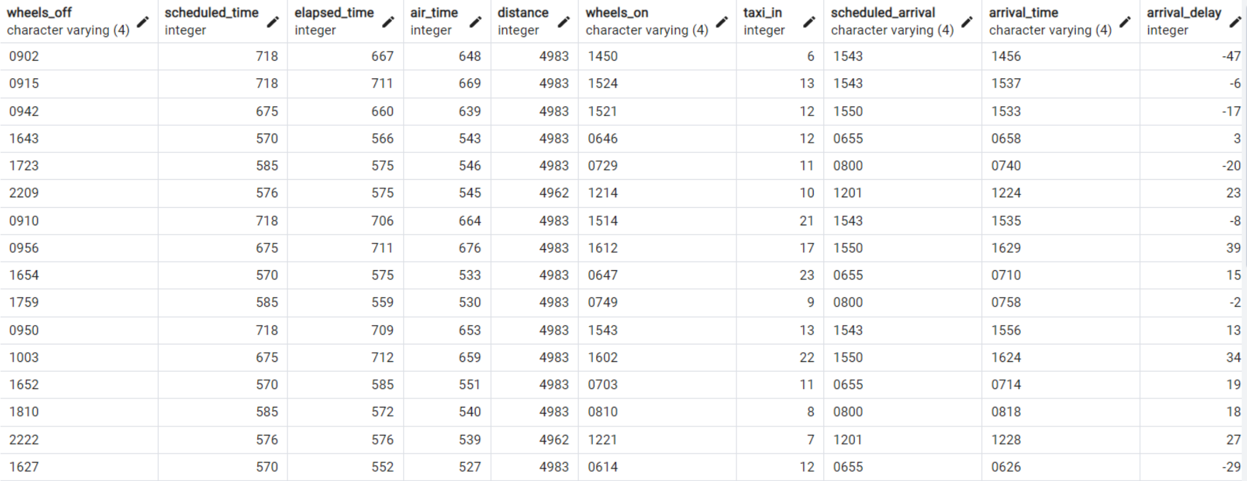


	origin_airport character varying (5)	jfk_count bigint	ny_total_flights numeric	jfk_percent numeric
1	JFK	8600	27090	31.75

(f) All New York flights originate in one of 3 airports: ‘JFK’ (Kennedy), ‘LGA’ (La Guardia), and ‘EWR’ (Newark in New Jersey). Count how many flights originate at ‘JFK.’ Then show how many flights originate at ‘JFK’ as a percentage of all flights. (hint: use a WITH clause or a FROM subquery)

part1

```
where origin_airport in ('Newark');

Select * from flights

        where (origin_airport IN ('JFK', 'LGA')

                OR

                  Destination_airport IN ('JFK', 'LGA', 'EWR'))

                AND

                  Elapsed_time > 500 ;
```



wheels_off character varying (4)	scheduled_time integer	elapsed_time integer	air_time integer	distance integer	wheels_on character varying (4)	taxi_in integer	scheduled_arrival character varying (4)	arrival_time character varying (4)	arrival_delay integer
0902	718	667	648	4983	1450	6	1543	1456	-47
0915	718	711	669	4983	1524	13	1543	1537	-6
0942	675	660	639	4983	1521	12	1550	1533	-17
1643	570	566	543	4983	0646	12	0655	0658	3
1723	585	575	546	4983	0729	11	0800	0740	-20
2209	576	575	545	4962	1214	10	1201	1224	23
0910	718	706	664	4983	1514	21	1543	1535	-8
0956	675	711	676	4983	1612	17	1550	1629	39
1654	570	575	533	4983	0647	23	0655	0710	15
1759	585	559	530	4983	0749	9	0800	0758	-2
0950	718	709	653	4983	1543	13	1543	1556	13
1003	675	712	659	4983	1602	22	1550	1624	34
1652	570	585	551	4983	0703	11	0655	0714	19
1810	585	572	540	4983	0810	8	0800	0818	18
2222	576	576	539	4962	1221	7	1201	1228	27
1627	570	552	527	4983	0614	12	0655	0626	-29

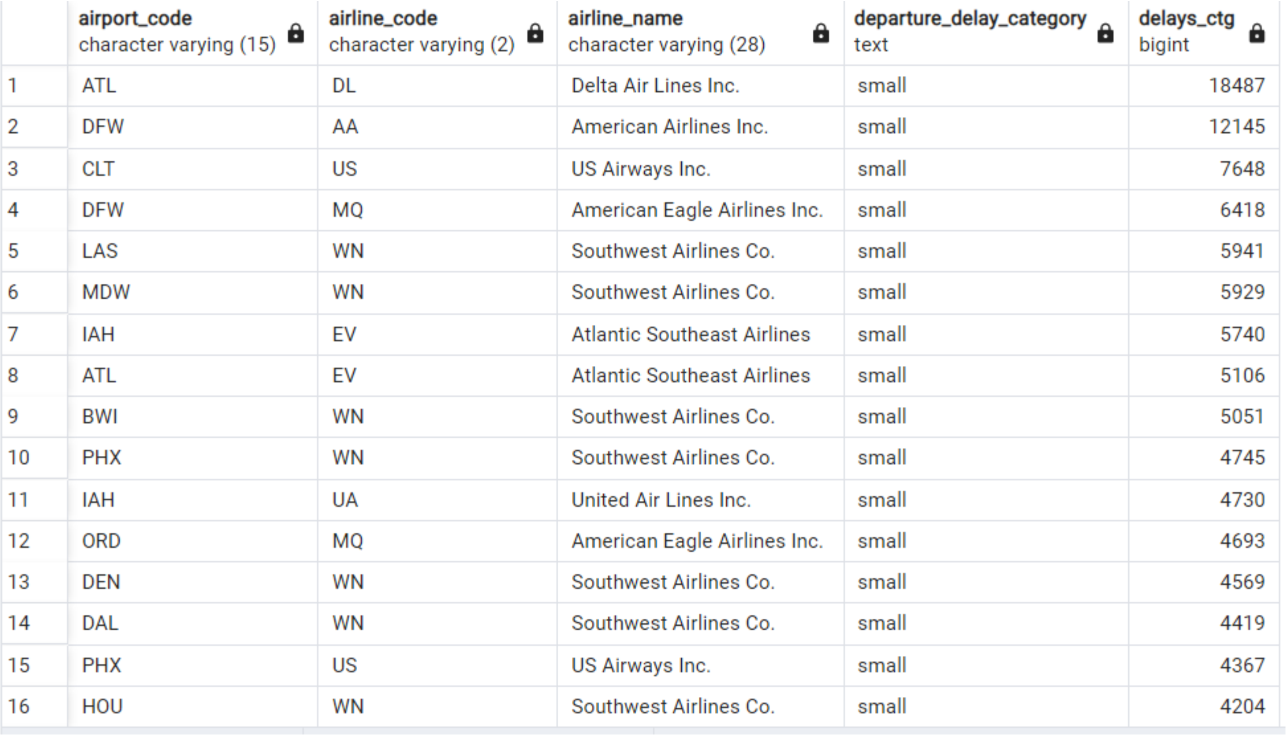
Part2

```
Update flights

        SET cancelled = 1

where (origin_airport IN ('JFK', 'LGA')

                OR

                  Destination_airport IN ('JFK', 'LGA', 'EWR'))

                AND

                  Elapsed_time > 500
```

(g) Build a single temporary table called Departure_Delays that capture the categories of the departure_delays of flights based on how many are ‘big,’ ‘medium,’ and ‘small’ delays. Provide the iata_code, airline, departure delay category, and determine the total number of delays in each category. Order the result based on the total number of delays in descending order:

```
CREATE TEMPORARY TABLE Departure_Delays AS
```

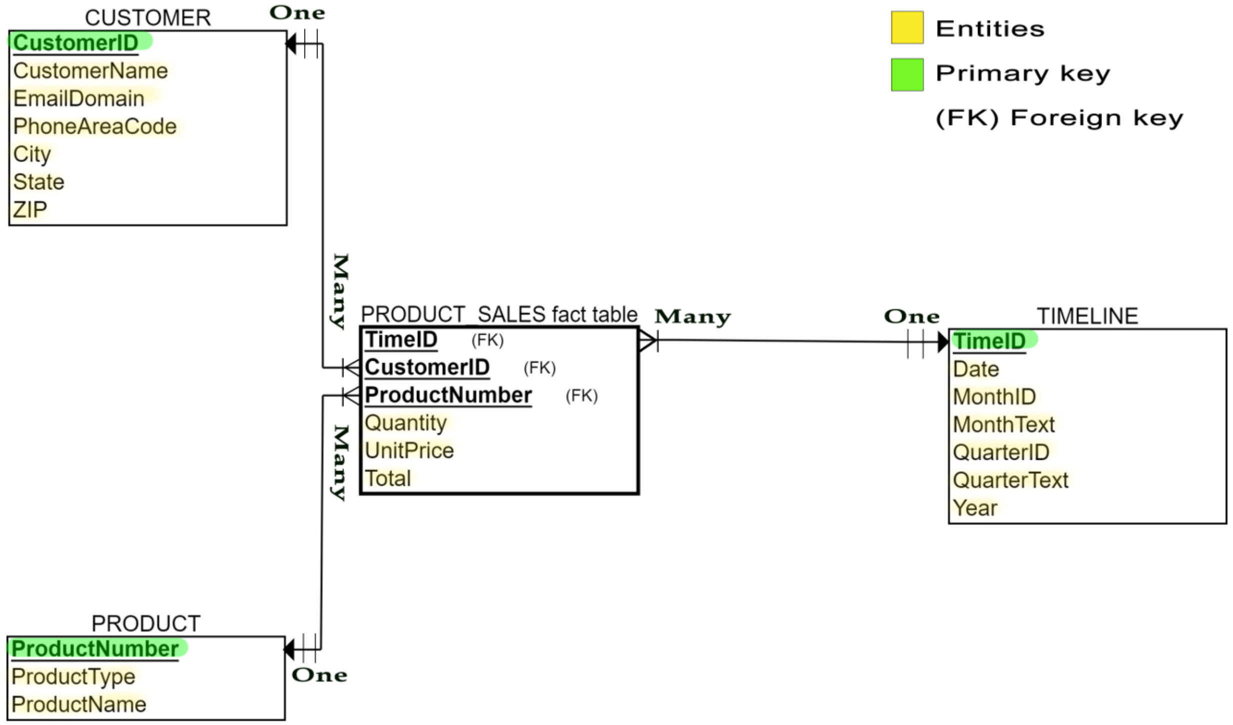
```
SELECT
    a.iata_code AS airport_code,
    an.iata_code AS airline_code,
    an.airline AS airline_name,
    CASE
        WHEN departure_delay >= -18 AND departure_delay <= 100 THEN 'small'
        WHEN departure_delay > 100 AND departure_delay <= 250 THEN 'medium'
        ELSE 'big'
    END AS departure_delay_category,
    COUNT(*) AS delays_ctg
FROM
    flights f
    JOIN airports a ON f.origin_airport = a.iata_code
    JOIN airlines an ON f.airline = an.iata_code
GROUP BY
    departure_delay_category, airport_code, airline_code, airline_name
ORDER BY
    delays_ctg DESC;
```



	airport_code character varying (15) 🔒	airline_code character varying (2) 🔒	airline_name character varying (28) 🔒	departure_delay_category text 🔒	delays_ctg bigint 🔒
1	ATL	DL	Delta Air Lines Inc.	small	18487
2	DFW	AA	American Airlines Inc.	small	12145
3	CLT	US	US Airways Inc.	small	7648
4	DFW	MQ	American Eagle Airlines Inc.	small	6418
5	LAS	WN	Southwest Airlines Co.	small	5941
6	MDW	WN	Southwest Airlines Co.	small	5929
7	IAH	EV	Atlantic Southeast Airlines	small	5740
8	ATL	EV	Atlantic Southeast Airlines	small	5106
9	BWI	WN	Southwest Airlines Co.	small	5051
10	PHX	WN	Southwest Airlines Co.	small	4745
11	IAH	UA	United Air Lines Inc.	small	4730
12	ORD	MQ	American Eagle Airlines Inc.	small	4693
13	DEN	WN	Southwest Airlines Co.	small	4569
14	DAL	WN	Southwest Airlines Co.	small	4419
15	PHX	US	US Airways Inc.	small	4367
16	HOU	WN	Southwest Airlines Co.	small	4204

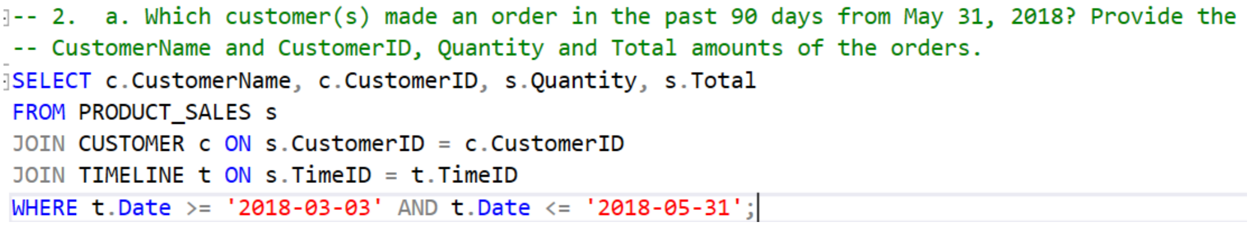
Part B: Data Warehousing & OLAP (50 points)

1. Sketch a representative Star schema for the data warehouse (specifying the relations, the attributes, the primary keys, and the foreign keys).

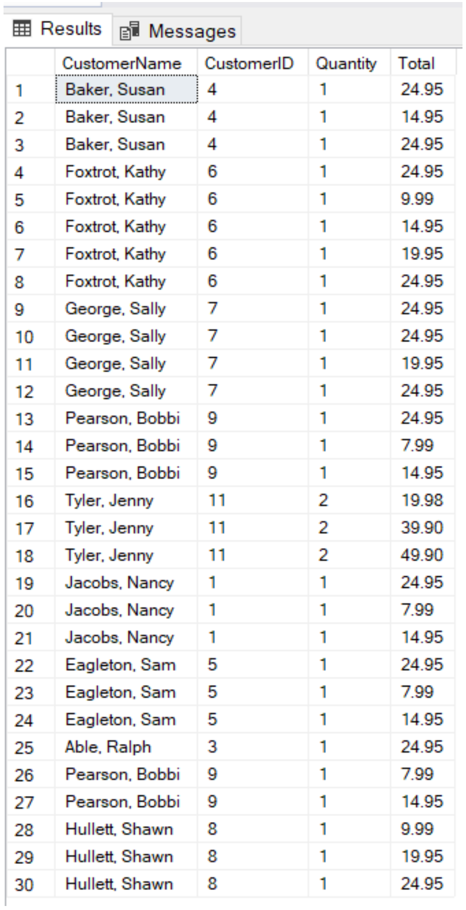


2. Suppose that we want to examine the data of HSD_DW, write SQL queries to answer the following questions:

a.



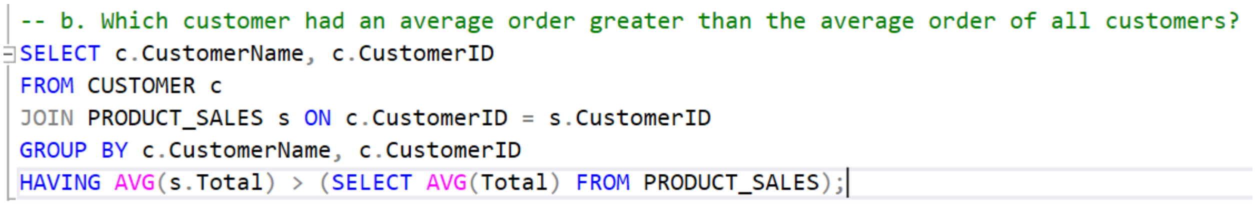
```
-- 2.  a. Which customer(s) made an order in the past 90 days from May 31, 2018? Provide the
-- CustomerName and CustomerID, Quantity and Total amounts of the orders.
SELECT c.CustomerName, c.CustomerID, s.Quantity, s.Total
FROM PRODUCT_SALES s
JOIN CUSTOMER c ON s.CustomerID = c.CustomerID
JOIN TIMELINE t ON s.TimeID = t.TimeID
WHERE t.Date >= '2018-03-03' AND t.Date <= '2018-05-31';
```



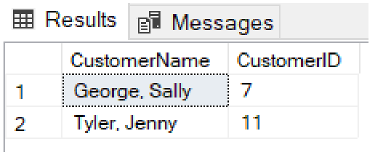
Results Messages

	CustomerName	CustomerID	Quantity	Total
1	Baker, Susan	4	1	24.95
2	Baker, Susan	4	1	14.95
3	Baker, Susan	4	1	24.95
4	Foxtrot, Kathy	6	1	24.95
5	Foxtrot, Kathy	6	1	9.99
6	Foxtrot, Kathy	6	1	14.95
7	Foxtrot, Kathy	6	1	19.95
8	Foxtrot, Kathy	6	1	24.95
9	George, Sally	7	1	24.95
10	George, Sally	7	1	24.95
11	George, Sally	7	1	19.95
12	George, Sally	7	1	24.95
13	Pearson, Bobbi	9	1	24.95
14	Pearson, Bobbi	9	1	7.99
15	Pearson, Bobbi	9	1	14.95
16	Tyler, Jenny	11	2	19.98
17	Tyler, Jenny	11	2	39.90
18	Tyler, Jenny	11	2	49.90
19	Jacobs, Nancy	1	1	24.95
20	Jacobs, Nancy	1	1	7.99
21	Jacobs, Nancy	1	1	14.95
22	Eagleton, Sam	5	1	24.95
23	Eagleton, Sam	5	1	7.99
24	Eagleton, Sam	5	1	14.95
25	Able, Ralph	3	1	24.95
26	Pearson, Bobbi	9	1	7.99
27	Pearson, Bobbi	9	1	14.95
28	Hullett, Shawn	8	1	9.99
29	Hullett, Shawn	8	1	19.95
30	Hullett, Shawn	8	1	24.95

b.



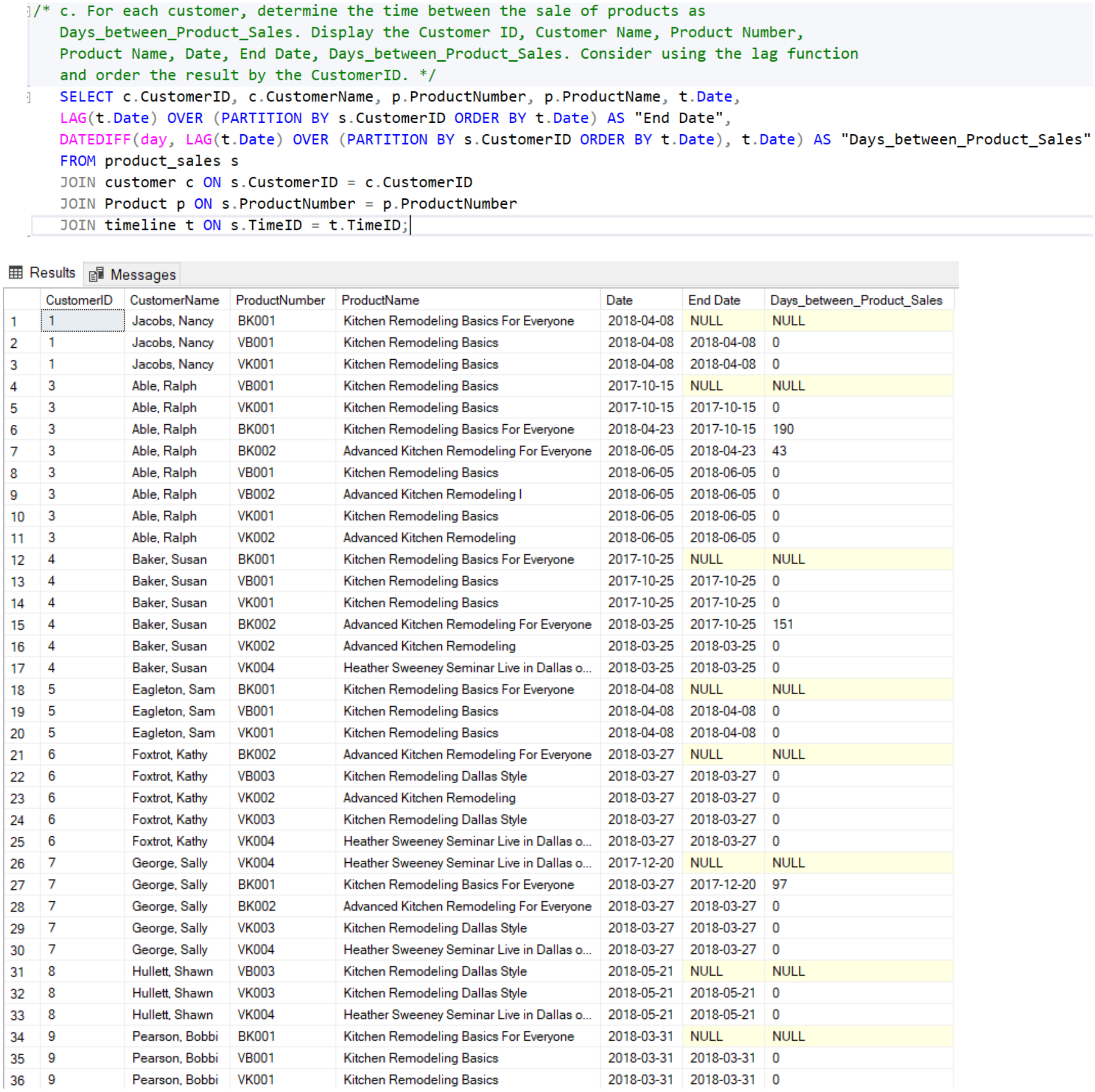
```
-- b. Which customer had an average order greater than the average order of all customers?
SELECT c.CustomerName, c.CustomerID
FROM CUSTOMER c
JOIN PRODUCT_SALES s ON c.CustomerID = s.CustomerID
GROUP BY c.CustomerName, c.CustomerID
HAVING AVG(s.Total) > (SELECT AVG(Total) FROM PRODUCT_SALES);
```



Results Messages

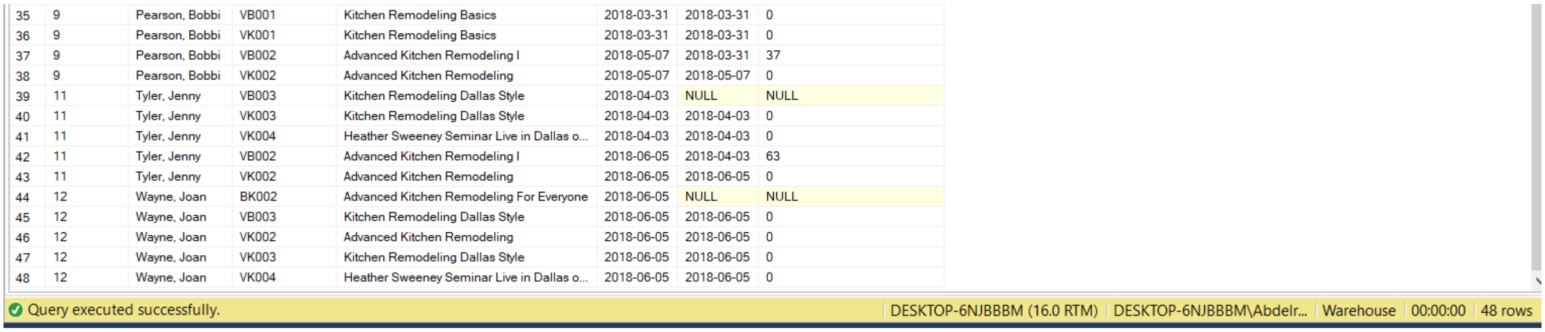
	CustomerName	CustomerID
1	George, Sally	7
2	Tyler, Jenny	11

c.



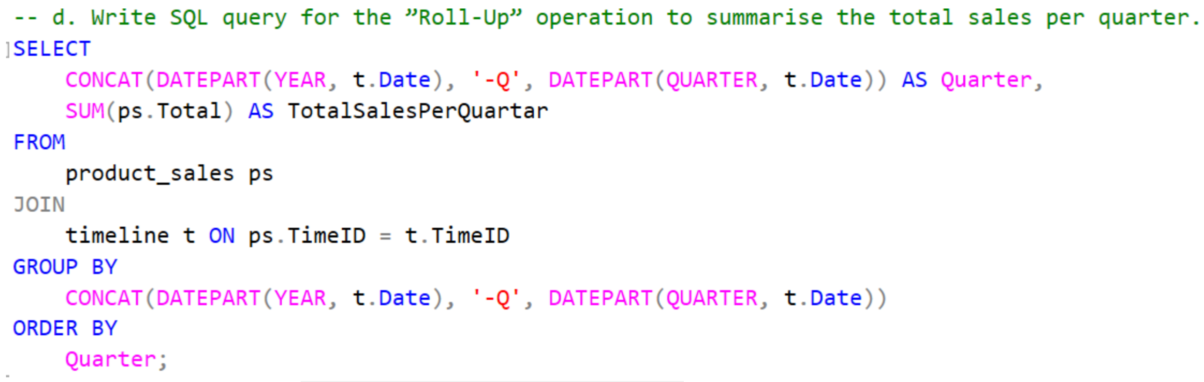
```
/* c. For each customer, determine the time between the sale of products as
   Days_between_Product_Sales. Display the Customer ID, Customer Name, Product Number,
   Product Name, Date, End Date, Days_between_Product_Sales. Consider using the lag function
   and order the result by the CustomerID. */
   SELECT c.CustomerID, c.CustomerName, p.ProductNumber, p.ProductName, t.Date,
   LAG(t.Date) OVER (PARTITION BY s.CustomerID ORDER BY t.Date) AS "End Date",
   DATEDIFF(day, LAG(t.Date) OVER (PARTITION BY s.CustomerID ORDER BY t.Date), t.Date) AS "Days_between_Product_Sales"
   FROM product_sales s
   JOIN customer c ON s.CustomerID = c.CustomerID
   JOIN Product p ON s.ProductNumber = p.ProductNumber
   JOIN timeline t ON s.TimeID = t.TimeID;
```

	CustomerID	CustomerName	ProductNumber	ProductName	Date	End Date	Days_between_Product_Sales
1	1	Jacobs, Nancy	BK001	Kitchen Remodeling Basics For Everyone	2018-04-08	NULL	NULL
2	1	Jacobs, Nancy	VB001	Kitchen Remodeling Basics	2018-04-08	2018-04-08	0
3	1	Jacobs, Nancy	VK001	Kitchen Remodeling Basics	2018-04-08	2018-04-08	0
4	3	Able, Ralph	VB001	Kitchen Remodeling Basics	2017-10-15	NULL	NULL
5	3	Able, Ralph	VK001	Kitchen Remodeling Basics	2017-10-15	2017-10-15	0
6	3	Able, Ralph	BK001	Kitchen Remodeling Basics For Everyone	2018-04-23	2017-10-15	190
7	3	Able, Ralph	BK002	Advanced Kitchen Remodeling For Everyone	2018-06-05	2018-04-23	43
8	3	Able, Ralph	VB001	Kitchen Remodeling Basics	2018-06-05	2018-06-05	0
9	3	Able, Ralph	VB002	Advanced Kitchen Remodeling I	2018-06-05	2018-06-05	0
10	3	Able, Ralph	VK001	Kitchen Remodeling Basics	2018-06-05	2018-06-05	0
11	3	Able, Ralph	VK002	Advanced Kitchen Remodeling	2018-06-05	2018-06-05	0
12	4	Baker, Susan	BK001	Kitchen Remodeling Basics For Everyone	2017-10-25	NULL	NULL
13	4	Baker, Susan	VB001	Kitchen Remodeling Basics	2017-10-25	2017-10-25	0
14	4	Baker, Susan	VK001	Kitchen Remodeling Basics	2017-10-25	2017-10-25	0
15	4	Baker, Susan	BK002	Advanced Kitchen Remodeling For Everyone	2018-03-25	2017-10-25	151
16	4	Baker, Susan	VK002	Advanced Kitchen Remodeling	2018-03-25	2018-03-25	0
17	4	Baker, Susan	VK004	Heather Sweeney Seminar Live in Dallas o...	2018-03-25	2018-03-25	0
18	5	Eagleton, Sam	BK001	Kitchen Remodeling Basics For Everyone	2018-04-08	NULL	NULL
19	5	Eagleton, Sam	VB001	Kitchen Remodeling Basics	2018-04-08	2018-04-08	0
20	5	Eagleton, Sam	VK001	Kitchen Remodeling Basics	2018-04-08	2018-04-08	0
21	6	Foxtrot, Kathy	BK002	Advanced Kitchen Remodeling For Everyone	2018-03-27	NULL	NULL
22	6	Foxtrot, Kathy	VB003	Kitchen Remodeling Dallas Style	2018-03-27	2018-03-27	0
23	6	Foxtrot, Kathy	VK002	Advanced Kitchen Remodeling	2018-03-27	2018-03-27	0
24	6	Foxtrot, Kathy	VK003	Kitchen Remodeling Dallas Style	2018-03-27	2018-03-27	0
25	6	Foxtrot, Kathy	VK004	Heather Sweeney Seminar Live in Dallas o...	2018-03-27	2018-03-27	0
26	7	George, Sally	VK004	Heather Sweeney Seminar Live in Dallas o...	2017-12-20	NULL	NULL
27	7	George, Sally	BK001	Kitchen Remodeling Basics For Everyone	2018-03-27	2017-12-20	97
28	7	George, Sally	BK002	Advanced Kitchen Remodeling For Everyone	2018-03-27	2018-03-27	0
29	7	George, Sally	VK003	Kitchen Remodeling Dallas Style	2018-03-27	2018-03-27	0
30	7	George, Sally	VK004	Heather Sweeney Seminar Live in Dallas o...	2018-03-27	2018-03-27	0
31	8	Hullett, Shawn	VB003	Kitchen Remodeling Dallas Style	2018-05-21	NULL	NULL
32	8	Hullett, Shawn	VK003	Kitchen Remodeling Dallas Style	2018-05-21	2018-05-21	0
33	8	Hullett, Shawn	VK004	Heather Sweeney Seminar Live in Dallas o...	2018-05-21	2018-05-21	0
34	9	Pearson, Bobbi	BK001	Kitchen Remodeling Basics For Everyone	2018-03-31	NULL	NULL
35	9	Pearson, Bobbi	VB001	Kitchen Remodeling Basics	2018-03-31	2018-03-31	0
36	9	Pearson, Bobbi	VK001	Kitchen Remodeling Basics	2018-03-31	2018-03-31	0

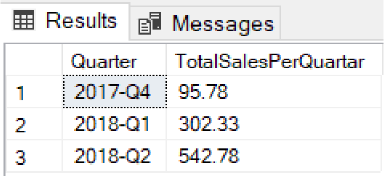


35	9	Pearson, Bobbi	VB001	Kitchen Remodeling Basics	2018-03-31	2018-03-31	0
36	9	Pearson, Bobbi	VK001	Kitchen Remodeling Basics	2018-03-31	2018-03-31	0
37	9	Pearson, Bobbi	VB002	Advanced Kitchen Remodeling I	2018-05-07	2018-03-31	37
38	9	Pearson, Bobbi	VK002	Advanced Kitchen Remodeling	2018-05-07	2018-05-07	0
39	11	Tyler, Jenny	VB003	Kitchen Remodeling Dallas Style	2018-04-03	NULL	NULL
40	11	Tyler, Jenny	VK003	Kitchen Remodeling Dallas Style	2018-04-03	2018-04-03	0
41	11	Tyler, Jenny	VK004	Heather Sweeney Seminar Live in Dallas o...	2018-04-03	2018-04-03	0
42	11	Tyler, Jenny	VB002	Advanced Kitchen Remodeling I	2018-06-05	2018-04-03	63
43	11	Tyler, Jenny	VK002	Advanced Kitchen Remodeling	2018-06-05	2018-06-05	0
44	12	Wayne, Joan	BK002	Advanced Kitchen Remodeling For Everyone	2018-06-05	NULL	NULL
45	12	Wayne, Joan	VB003	Kitchen Remodeling Dallas Style	2018-06-05	2018-06-05	0
46	12	Wayne, Joan	VK002	Advanced Kitchen Remodeling	2018-06-05	2018-06-05	0
47	12	Wayne, Joan	VK003	Kitchen Remodeling Dallas Style	2018-06-05	2018-06-05	0
48	12	Wayne, Joan	VK004	Heather Sweeney Seminar Live in Dallas o...	2018-06-05	2018-06-05	0

✓ Query executed successfully. | DESKTOP-6NJBBBM (16.0 RTM) | DESKTOP-6NJBBBM\Abdelr... | Warehouse | 00:00:00 | 48 rows d.



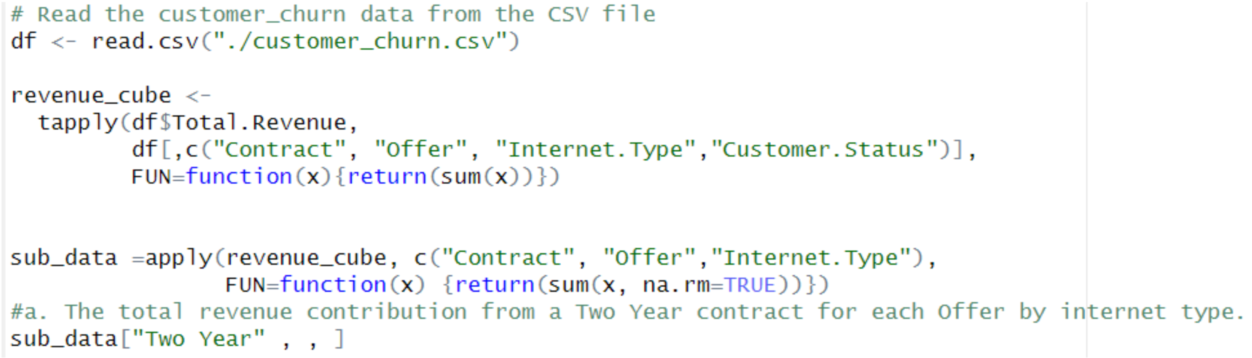
```
-- d. Write SQL query for the ”Roll-Up” operation to summarise the total sales per quarter.
SELECT
    CONCAT(DATEPART(YEAR, t.Date), '-Q', DATEPART(QUARTER, t.Date)) AS Quarter,
    SUM(ps.Total) AS TotalSalesPerQuartar
FROM
    product_sales ps
JOIN
    timeline t ON ps.TimeID = t.TimeID
GROUP BY
    CONCAT(DATEPART(YEAR, t.Date), '-Q', DATEPART(QUARTER, t.Date))
ORDER BY
    Quarter;
```



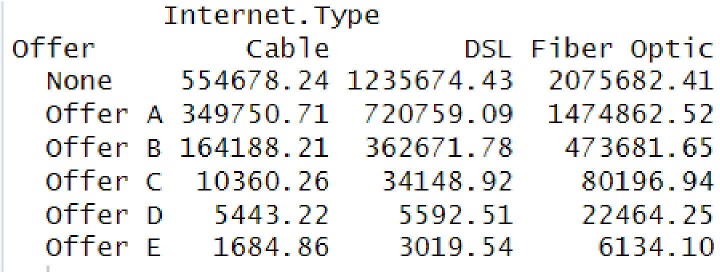
	Quarter	TotalSalesPerQuartar
1	2017-Q4	95.78
2	2018-Q1	302.33
3	2018-Q2	542.78

3. Customer churn is a huge problem for telecoms providers. Analyze the customer_churn dataset provided to determine ways to improve customer retention. Using Excel or R, build an OLAP cube to determine the following:

a.

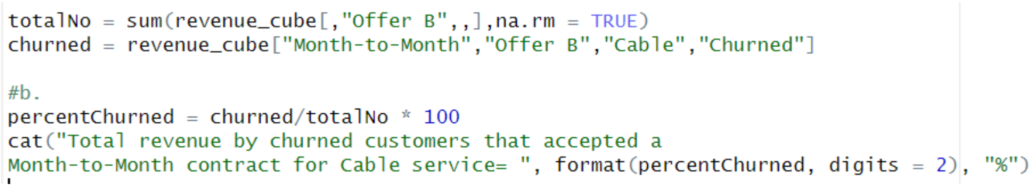


```
# Read the customer_churn data from the CSV file
df <- read.csv("./customer_churn.csv")

revenue_cube <-
  tapply(df$Total.Revenue,
         df[,c("Contract", "Offer", "Internet.Type","Customer.Status")],
         FUN=function(x){return(sum(x))})


sub_data =apply(revenue_cube, c("Contract", "Offer","Internet.Type"),
                FUN=function(x) {return(sum(x, na.rm=TRUE))})
#a. The total revenue contribution from a Two Year contract for each Offer by internet type.
sub_data["Two Year" , , ]
```

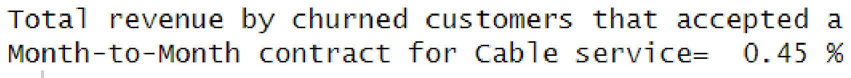


```
           Internet.Type
Offer            Cable        DSL  Fiber Optic
  None       554678.24 1235674.43  2075682.41
  Offer A    349750.71  720759.09  1474862.52
  Offer B    164188.21  362671.78   473681.65
  Offer C     10360.26   34148.92    80196.94
  Offer D      5443.22    5592.51    22464.25
  Offer E      1684.86    3019.54     6134.10
```

b.



```
totalNo = sum(revenue_cube[,"Offer B",,],na.rm = TRUE)
churned = revenue_cube["Month-to-Month","Offer B","Cable","Churned"]

#b.
percentChurned = churned/totalNo * 100
cat("Total revenue by churned customers that accepted a
Month-to-Month contract for Cable service= ", format(percentChurned, digits = 2), "%")
```



```
Total revenue by churned customers that accepted a
Month-to-Month contract for Cable service=  0.45 %
```