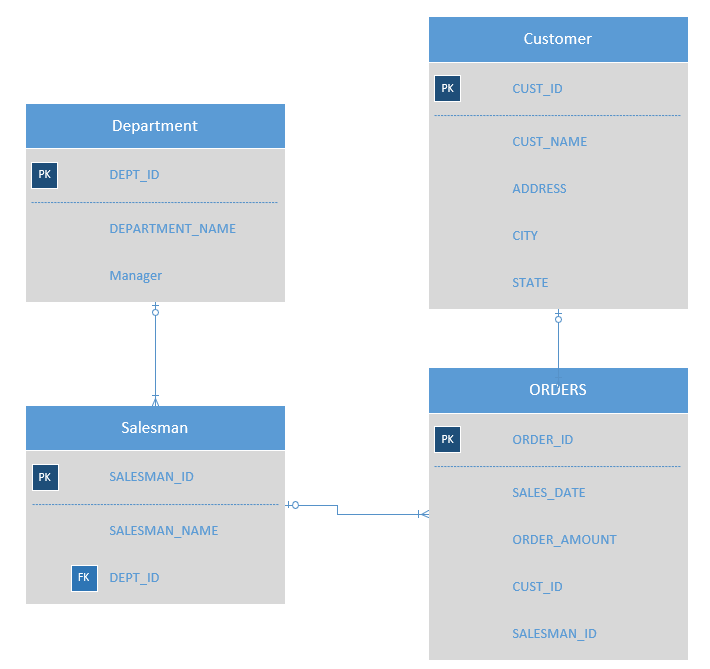
Redshift Challenge Questions

***Refer to the following Data model for Questions 1-5***



1. Write SQL Query to list all the Sales for the state of California in the year 2018

Attached folder has the query[Question1\_Query.sql]

1. Write SQL query to list Sales amount for each department in the year 2018 sorted by the sales amount (Show all the department even if it did not make any sales)

Attached folder has the query[Question2\_Query.sql]

1. Write SQL to list all the salesman who did not make any sales in 2018

Attached folder has the query[Question3\_Query.sql]

1. Write SQL to list Top 10 Salesman in the Year 2018 based on the sales

Attached folder has the query[Question4\_Query.sql]

1. Write SQL to list Top 10 Customers in the Year 2018 based on the sales
2. Attached folder has the query[Question5\_Query.sql]

***Refer to the following Table for Questions 6-7***

**CREATE** **TABLE** TEST\_MSR\_SOURCE (

rpt\_grp\_cd **varchar**(60) ,

lctn\_typ\_cd **varchar**(10) ,

clctn\_prd\_txt **varchar**(8) ,

msr\_cd **varchar**(20),

clcltn\_date **varchar**(10),

grp\_rate\_nmrtr **varchar**(3),

grp\_rate\_dnmntr **varchar**(5) ,

file\_name **varchar**(50),

creat\_ts **varchar**(50),

creat\_user\_id **varchar**(30),

submsn\_cmplt\_cd **varchar**(1))

**CREATE** **TABLE** TEST\_MSR\_TARGET (

TEST\_MSR\_TARGET\_ID **int4** **NOT** **NULL**

rpt\_grp\_cd **varchar**(60),

lctn\_typ\_cd **varchar**(10),

clctn\_prd\_txt **varchar**(8),

msr\_cd **varchar**(20),

clcltn\_date **date**,

grp\_rate\_nmrtr **int4**,

grp\_rate\_dnmntr **numeric**(5),

file\_name **varchar**(50),

finl\_sw **varchar**(1),

creat\_ts **timestamp** **NOT** **NULL**,

creat\_user\_id **varchar**(30) **NOT** **NULL**,

submsn\_cmplt\_cd **varchar**(1))

1. Import .csv file into TEST\_MSR\_SOURCE using Python Script
2. Write a Python script that takes all data from TEST\_MSR\_SOURCE and inserts the data into TEST\_MSR\_TARGET
   1. Database type = Redshift
   2. Number of rows = 10
   3. Sequential key used for TEST\_MSR\_TARGET\_ID
   4. Data transformations from varchar to (date, int, numeric, timestamp)
3. Would your script change if the number of rows = 1,000,000? Why or why not?

Attached python script needs to be edited, to include the correct db/host/username/password and the file location should also be changed to point to the file location on your machine.

Yes, the script and the approach to loading a dataset would change for the datasets that is huge in size.

One approach would be (the approach followed in my script ) to read the CSV file using python’s CSV module and write it to the Redshift table using psycopg2 library. While this approach accomplishes the task of loading in the data into the redshift table. Its not the most efficient way of doing it. As we can see, we had to loop through every single row from the csv file just to insert them into the redshift table(database).

However, this approach would be slow as the number of row increases(and dataset grows in size) or for files that are huge in size. This is makes us think about alternate ways of achieving this task.

There are few possible approaches that would be more efficient and wouldn’t lead to any errors

1. One approach would be to parse the .csv file using big data tools such as Apache Spark(pyspark). Preprocess the data using pyspark and store it in Apache Parquet file format(using compression techniques such as Snappy) and stage it in AWS S3 Bucket.

Once the data lands on S3 we can use redshifts “COPY” command to load data available in S3 to a table in redshift DB

Example Command:

COPY table\_name

FROM 's3://<bucket-name>/load/fileName.csv'

credentials 'aws\_access\_key\_id=<Your-Access-Key- ID>;aws\_secret\_access\_key=<Your-Secret-Access-Key>'

CSV;

The last line in command is the file type of the file available on S3 that needs to be moved to a table in Redshift

In addition to thinking about the optimal way of copying or moving datasets of huge size from on premise servers to Cloud services such as AWS, Azure and Google cloud, before migrating each of the tables from source system to the target system, we need to analyze the following to optimize the data migration process and save costs

1. What are we going to do with the data?

2. What’s the query pattern?

3. Do we use all the columns from the source system(table) and how often is the data scanned

4. Is the query speed important and how important is the cluster cost charged by the service provider too expensive?

The other approach to writing large files into a redshift table split up the files into smaller parts that corresponds to the Redshift cluster's size, so that we can take advantage of Redshift's parallel processing