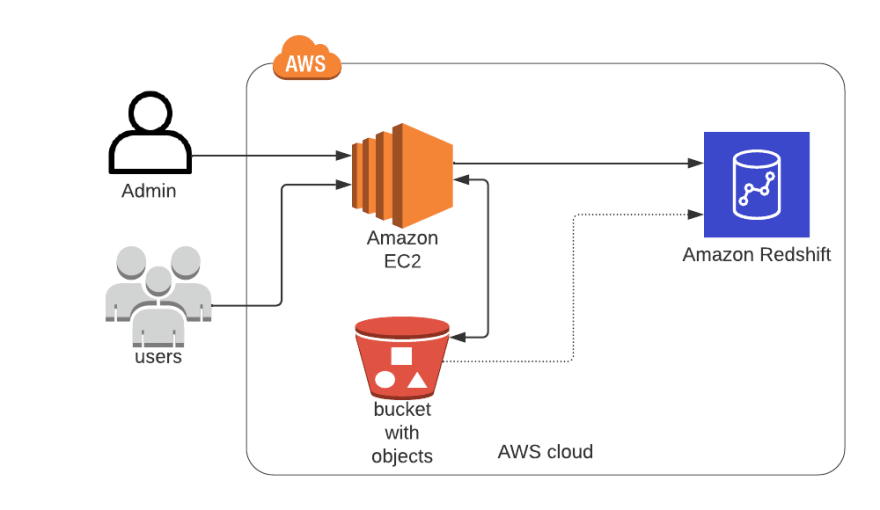
# Lab 3: Working with Amazon Redshift clusters

1. Identify Amazon Redshift as a data warehouse solution.
2. Launch an Amazon Redshift cluster.
3. Create tables with Automatic Table Optimization.
4. Load data from an Amazon S3 bucket to an Amazon Redshift cluster.
5. Identify table columns to designate as sort keys.

**NOT LAB GIVEN DIAGRAM-**



**My thinking - redshift is like EC2 for Data Analysis**

**Meaning it offer ways to handle very large amount of data, and get analysis from it.**

**Reason -**

**Think of a Redshift cluster as a group of super-powerful computers (servers) working together. They are specially designed to handle huge amounts of data and perform complex tasks really quickly.**

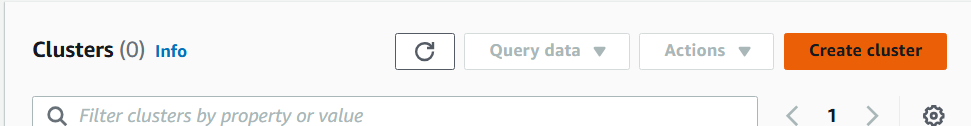
**Amazon Redshift Spectrum - lets you analyze data stored in Amazon S3 without having to load it into Redshift first.**

Pre-step - connect to CLI using Url given

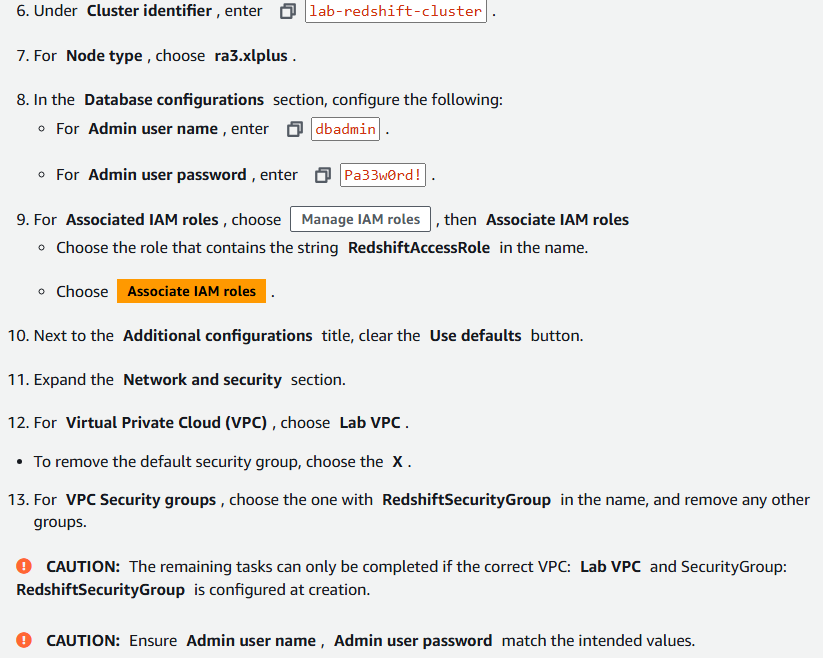
**Task 1: Create a two node Amazon Redshift cluster**

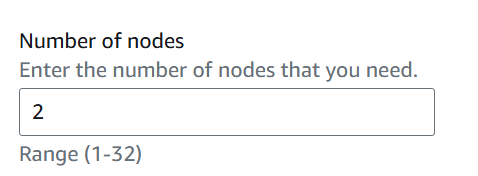
**Two nodes mean two computer/instances will work from RedShift side**

1.1 open redshift in console, click on create cluster

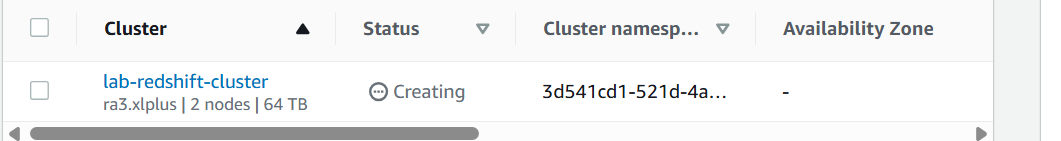


1.2 enter these values

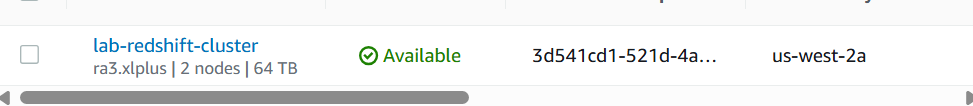


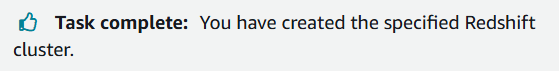


1.3 click create, wait



1.4 done





**Task 2: Creating tables in Amazon Redshift**

use the psql interface to create tables on the Amazon Redshift cluster that holds data.

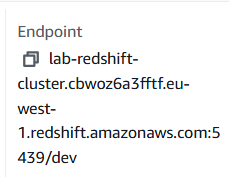
Tables are where your data is stored within the cluster.

We use psql to create tables in redshift

We will send command to redshift cluster DB, and we will do this via psql (via CLI).

2.1 Open the created cluster by clicking on it

2.2 Copy the endpoint value somewhere



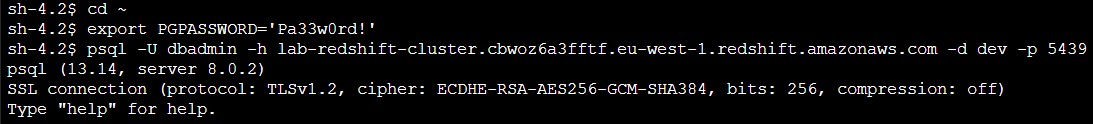
2.3 enter this in CLI

cd ~

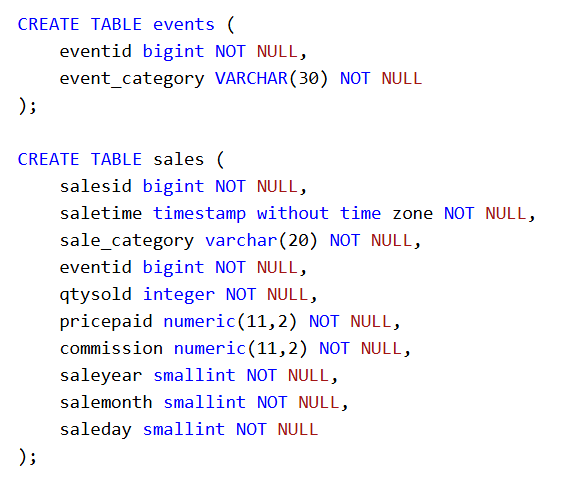
export PGPASSWORD='Pa33w0rd!'

psql -U dbadmin -h RedshiftClusterEndpoint -d dev -p 5439

Replace RedshiftClusterEndpoint with 2.2 value

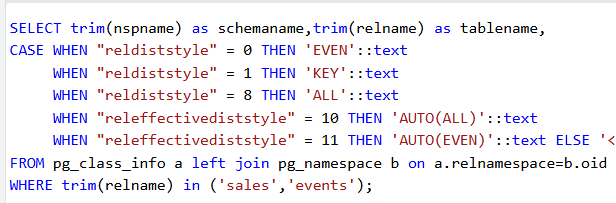


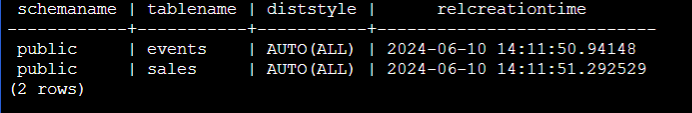


2.4 run these sql commands:  


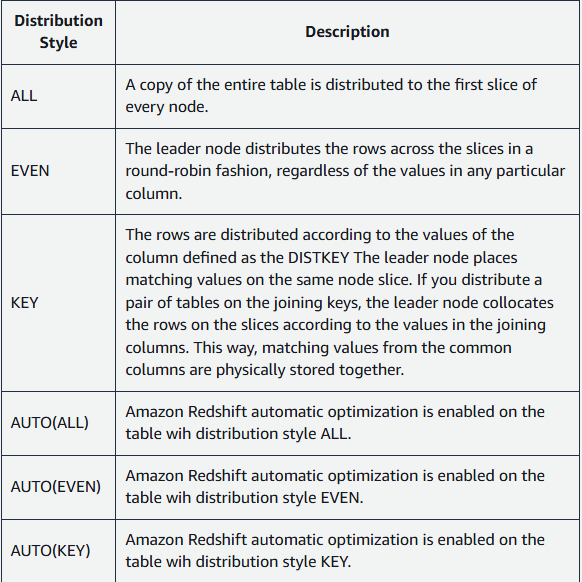
Tables created

2.5 now run these



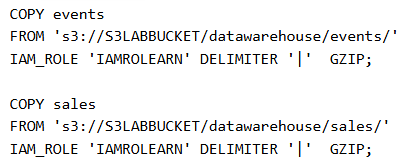


What this command does:  
This SQL query retrieves information about tables named "sales" and "events" from the Amazon Redshift system catalog tables.



**Task 3: Load tables with data from S3 objects**

3.1 use this to copy data



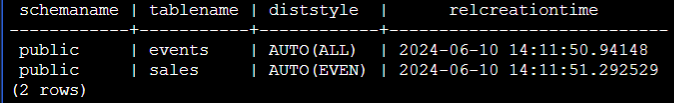
Replace S3 bucket path, and IAM ARN

Output-





3.2 view the distribution style once again, it has changed



**Task 4: Working with sort keys**

We have to assign columns as sort keys

4.1 Run this to get sort keys for both tables

SELECT sti.table, encoded, diststyle, sortkey1

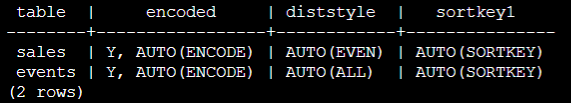
FROM SVV\_TABLE\_INFO sti

WHERE sti.database ='dev' AND sti.table in ('events','sales') order by size desc;

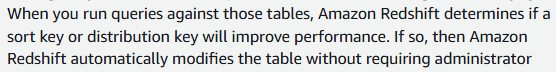
sti.table retrieves the name of the table.

SVV\_TABLE\_INFO is a system view in Amazon Redshift that contains metadata information about tables in the database.

WHERE - filters the results to include only tables in the 'dev' database with names 'events' or 'sales'.







4.2 creating a new table

DROP TABLE IF EXISTS sales\_nosort;

CREATE TABLE sales\_nosort (

salesid bigint NOT NULL,

saletime timestamp without time zone NOT NULL,

sale\_category varchar(20) NOT NULL,

eventid bigint NOT NULL,

qtysold integer NOT NULL,

pricepaid numeric(11,2) NOT NULL,

commission numeric(11,2) NOT NULL,

saleyear smallint NOT NULL,

salemonth smallint NOT NULL,

saleday smallint NOT NULL

);

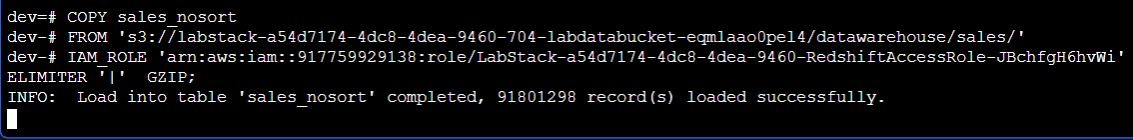
4.3 load data into this table

COPY sales\_nosort

FROM 's3://S3LABBUCKET/datawarehouse/sales/'

IAM\_ROLE 'IAMROLEARN' DELIMITER '|' GZIP;

Replace the values of path and ARN

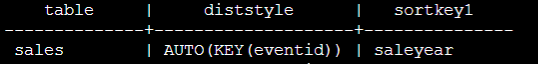


4.4 Run this to set the sort keys on the original sales table:

(here, we are setting saleyear, month as sort keys)

This will disable auto table optimizn

ALTER TABLE sales ALTER SORTKEY (saleyear, salemonth);



4.5 This query is performed on both the sales and sales\_nosort tables to compare how much of a difference the sort key makes.

SELECT s.saleyear,s.salemonth,e.event\_category, sum(s.commission), sum(s.pricepaid)

FROM sales s, events e

WHERE s.saleyear BETWEEN 2021 AND 2022 AND s.salemonth=1

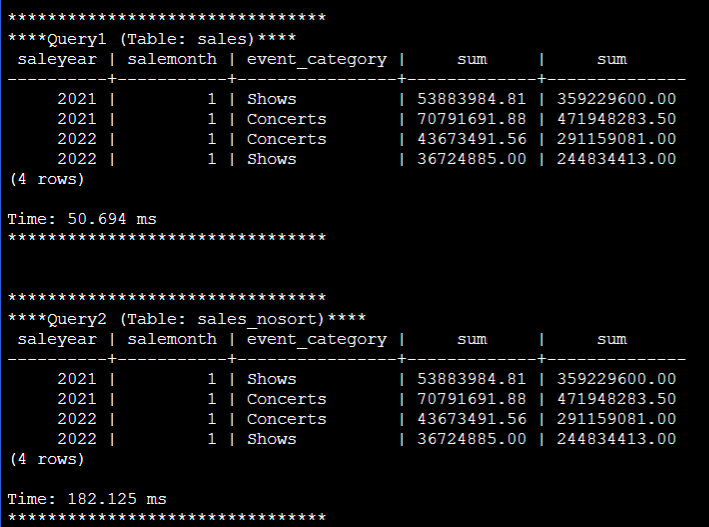
AND s.eventid = e.eventid

GROUP BY s.saleyear,s.salemonth,e.event\_category

ORDER BY s.saleyear,s.salemonth;

And

for i in {00..05}; do psql -U dbadmin -h lab-redshift-cluster.cbwoz6a3fftf.eu-west-1.redshift.amazonaws.com -d dev -p 5439 -f /temp/compare-sort.sql; sleep 2; done



Notice the time difference

