**Redshift Architecture:**

**Amazon Redshift can either be referenced as a redshift instance or redshift cluster where a cluster can go from 1 to 60 according to current documentation.**

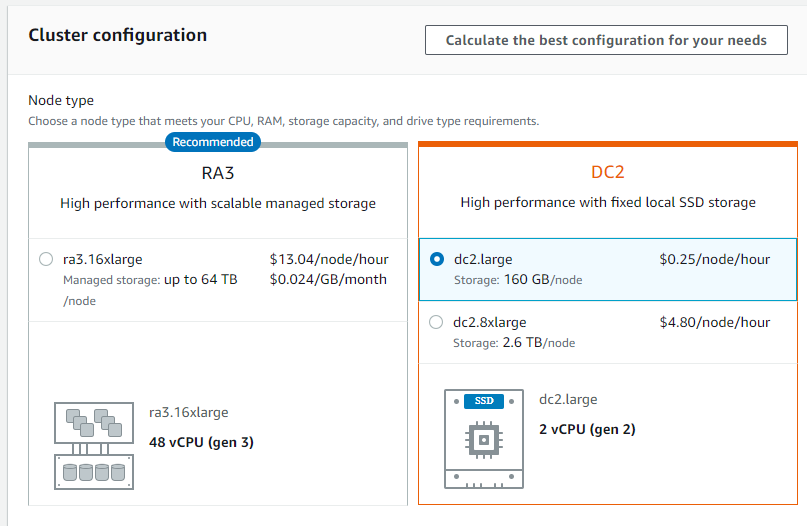
**It consists of one Leader Node and 2 to many compute nodes.**

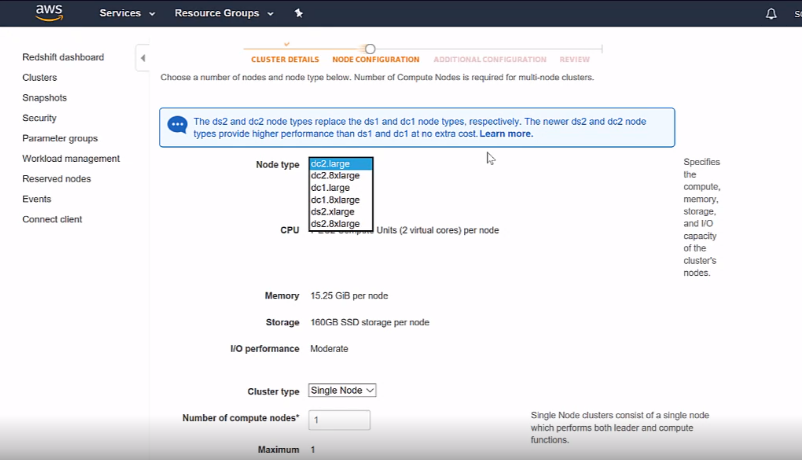
**Client interactions always happens on Leader Node.**



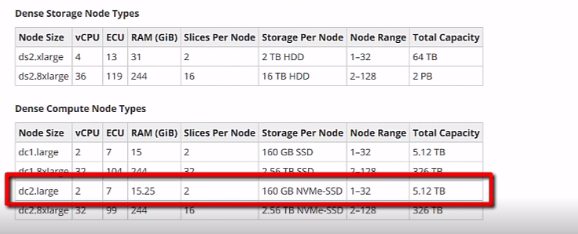
**They can be configured in 2 types of nodes.**

1. **Dense Compute nodes(dc)**
2. **Dense Storage nodes(ds)**

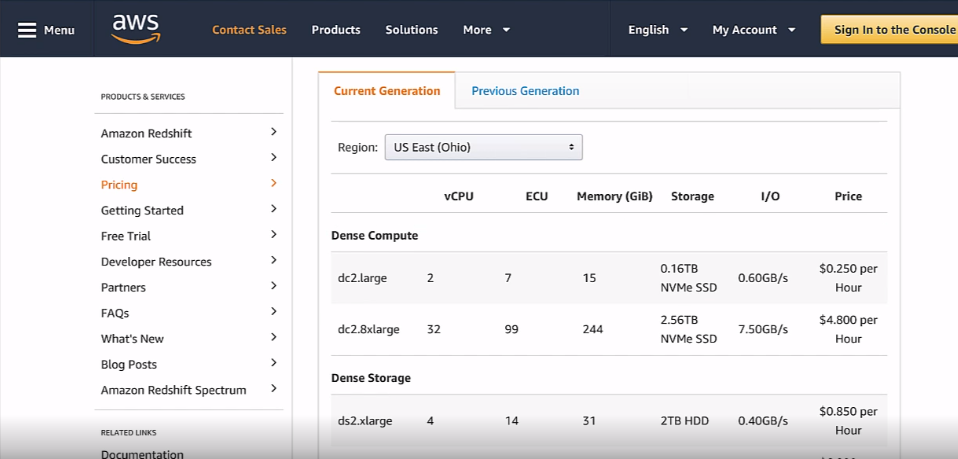




**Node details and you can choose the node based on your requirement.**



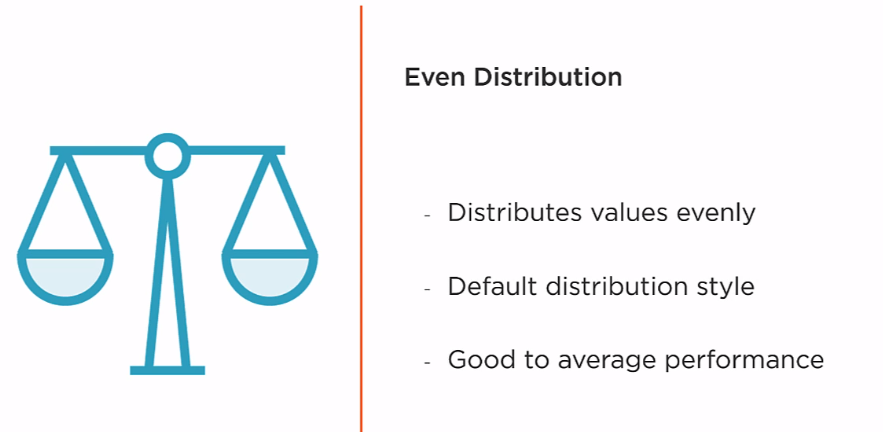
**Example of cluster pricing:**

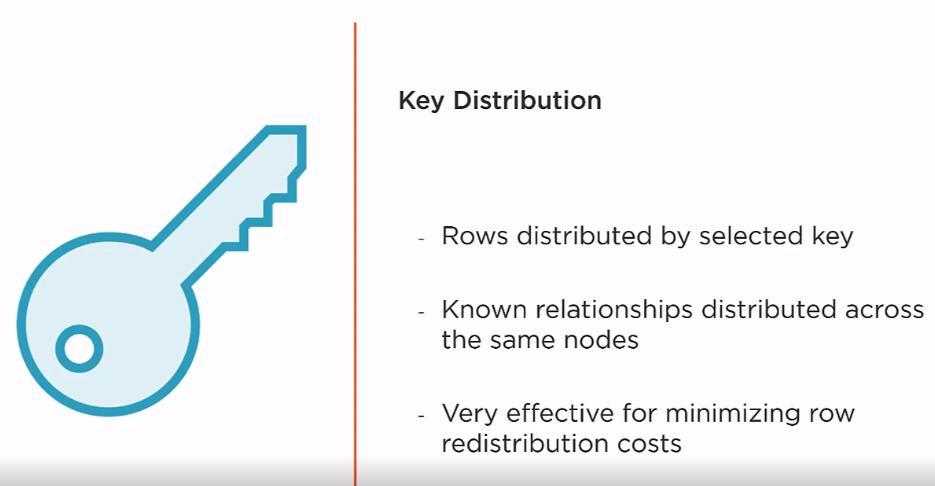


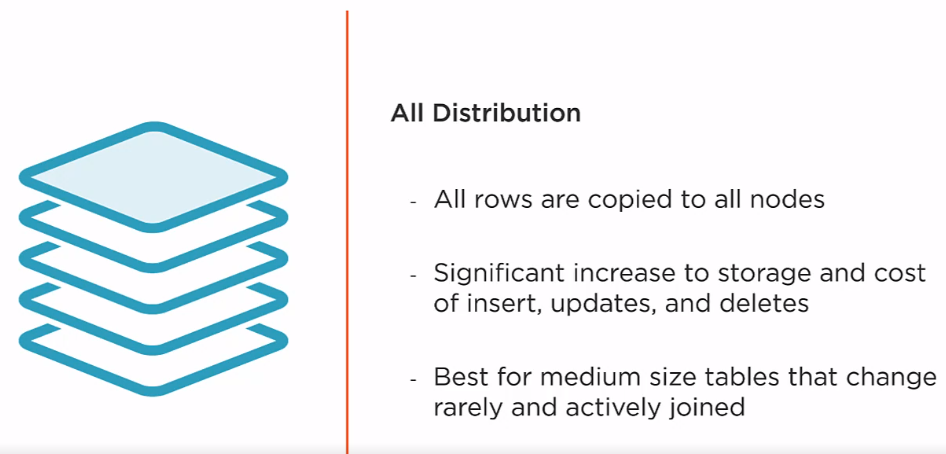
**Why Redshift?**

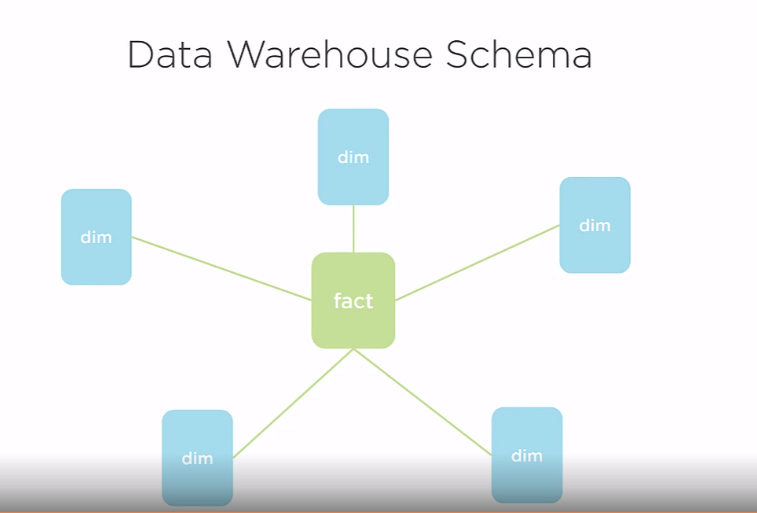
**Below are the Data distribution styles.**







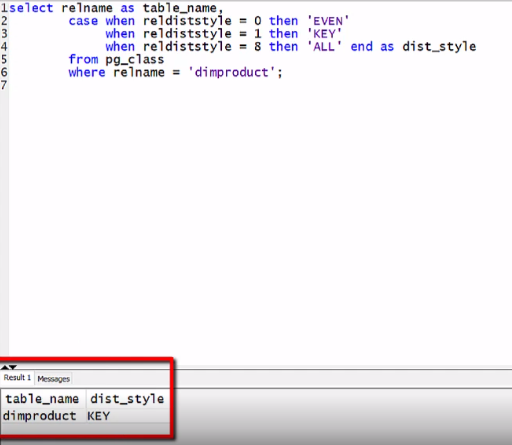




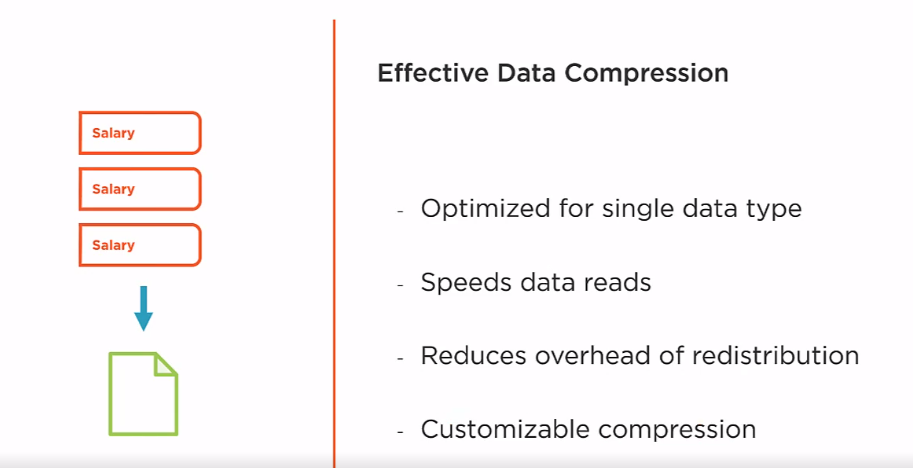
**Below you can see the distribution type of the table designed.**

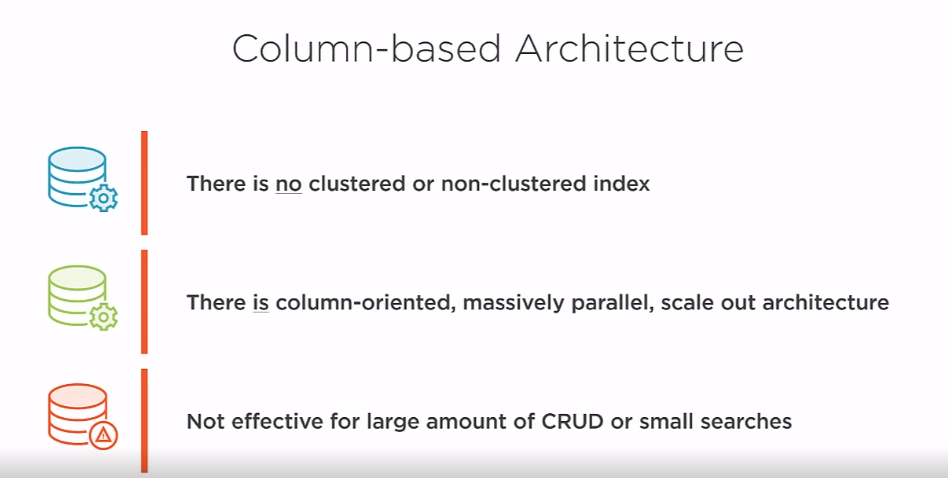


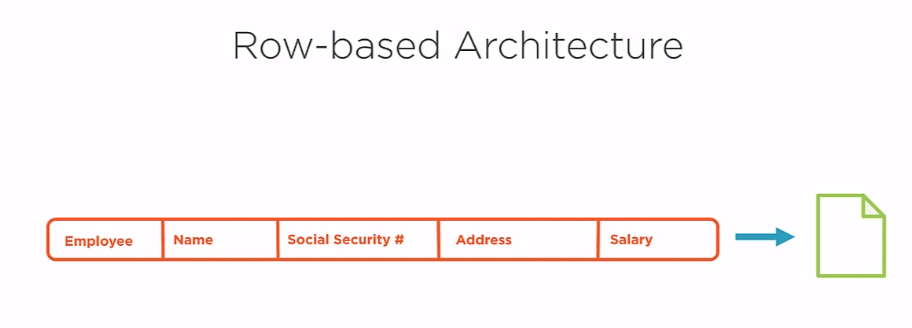
**Below Query will show the type of distribution a table is designed with:**

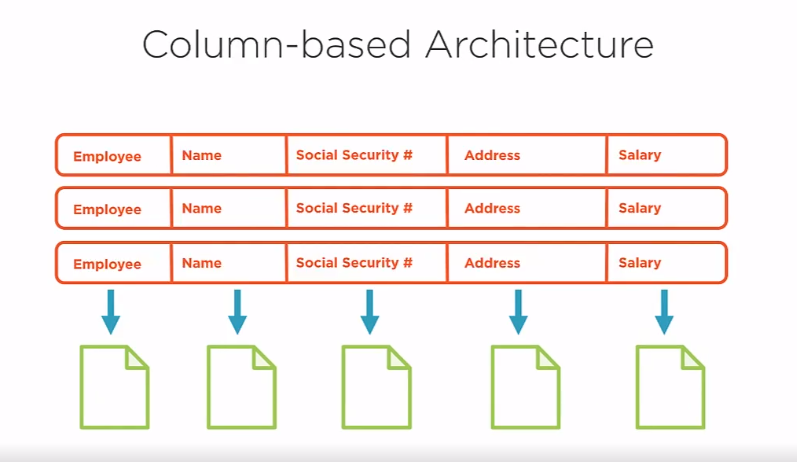


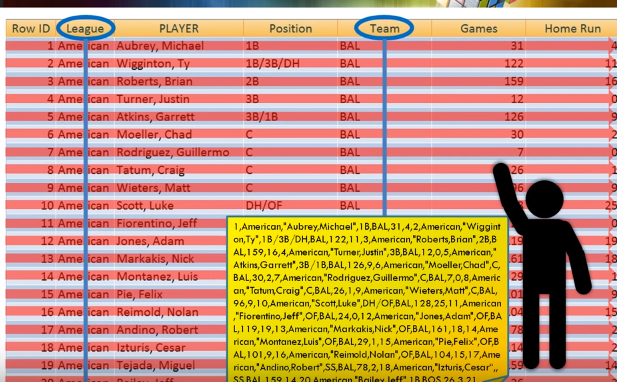
**Improves performance by doing effective data compression as only one data type is stored in a data page so a lot of space is saved.**



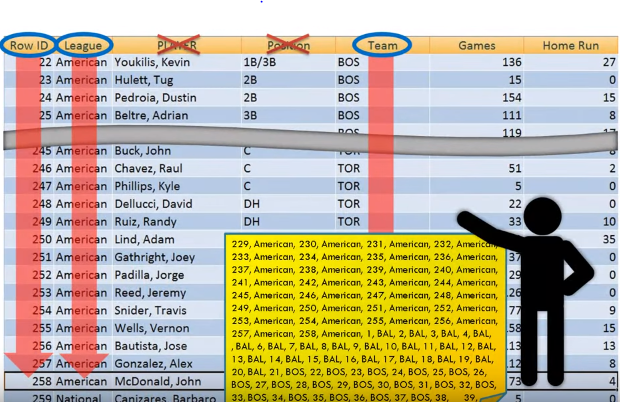




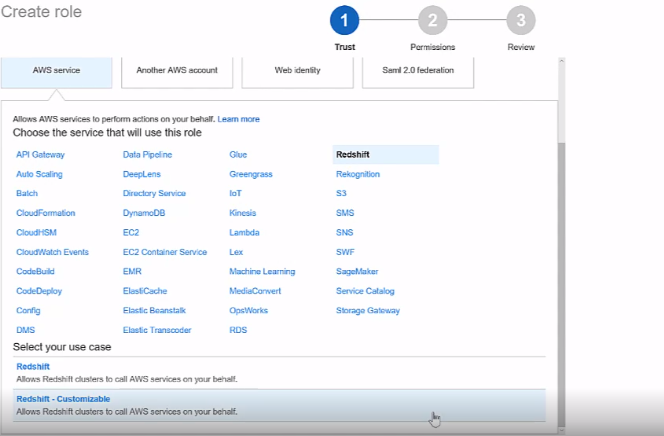


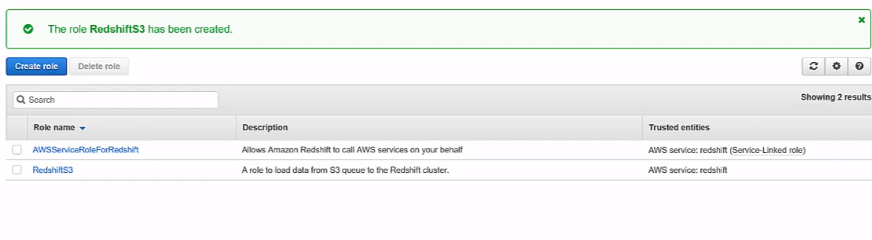




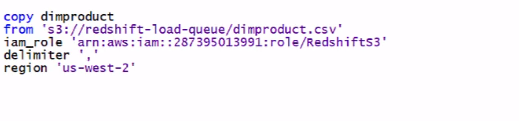


**In order to read a file form S3, Create a S3 Read only Access Role from IAM under the Redshift customizable menu.**

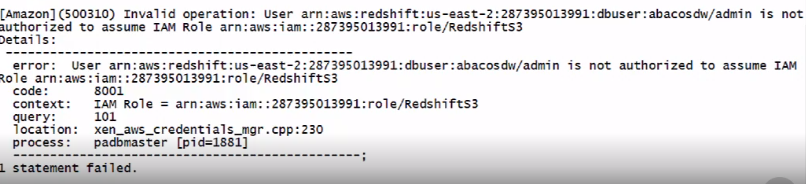




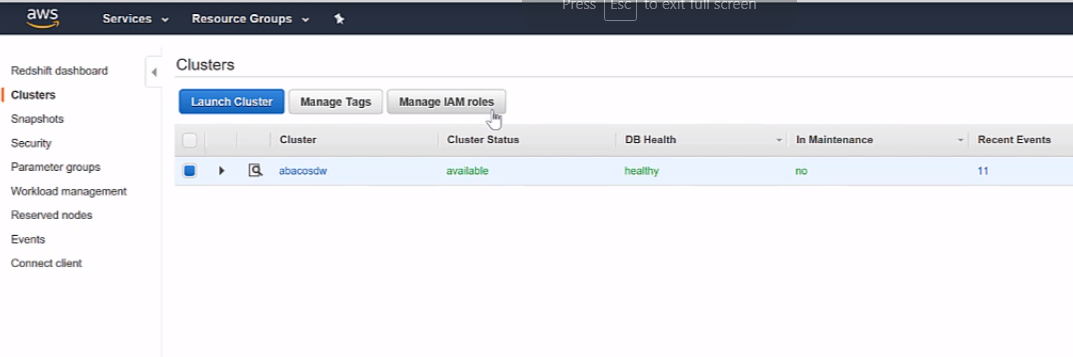
Below is the command to copy the data ( A csv file) from the S3 bucket to The Redshift Cluster.



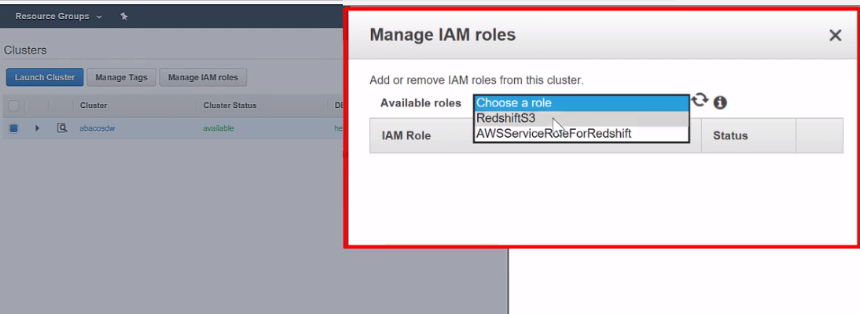
**It will fail if you do not have proper role assigned to the redshift cluster.**



**Now Give/Assign the same Role that we created to the Redshift Cluster.**



**So Assign the Role that we just created to the Redshift Cluster as well.**



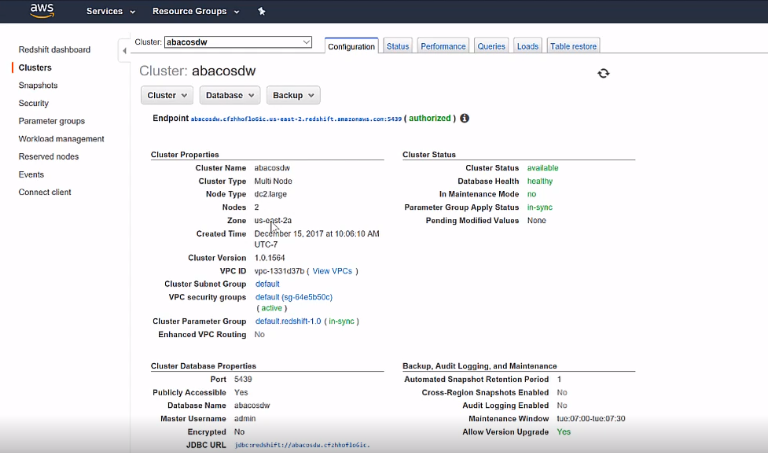
Now try loading the data.



**The Data is loaded successfully.**

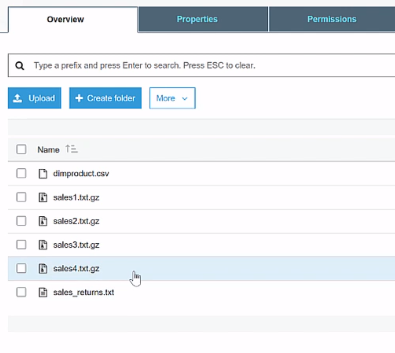
**If you notice it took 55 seconds to load 606 records which is pretty time consuming so how do we improve the performance?**

**If you see below we have 2 Compute Nodes on our Redshift Cluster with each node having 2 node slices which accounts to 4 storage areas and we need to utilize that in order to improve performance.**





**So we need to split our files in to four small files.**



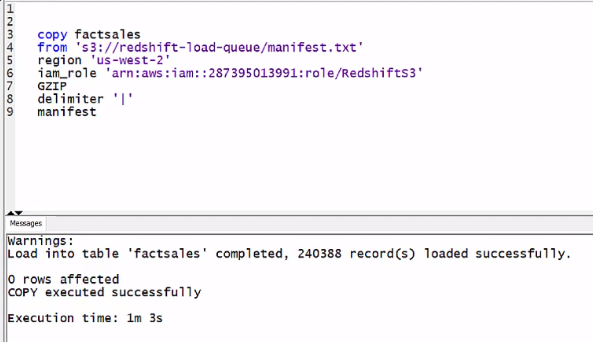
**Also, If you see above there is an unwanted ‘sales’ prefix file in the same path, In order to avoid loading that file we need to use ‘manifest’ option.**

**Load the files using a manifest file which will have the names of all the files that needs to be loaded.**

**The manifest file is like a JSON file and store it in the same S3 Bucket and reference this in your COPY command using ‘manifest’ option.**

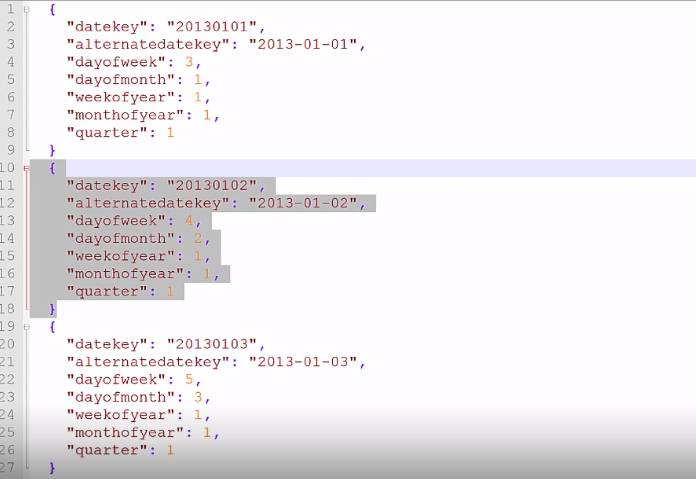


**Data is loaded fast for a quarter million records in a minute using parallelisim.**



**Example of Loading Data directly from JSON file:**

**Example of JSON file**



**To see errors from loading a JSON file go to ‘stl\_load\_errors’ table to see the error message.**



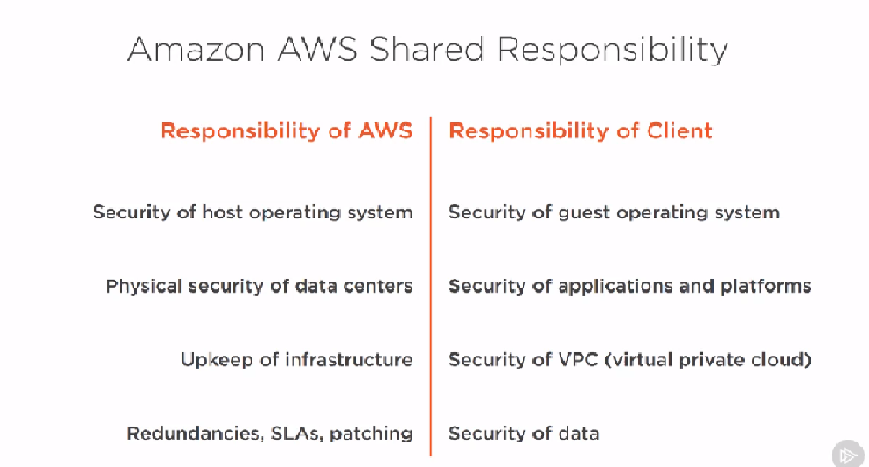
**Below is the command to load JSON file.**



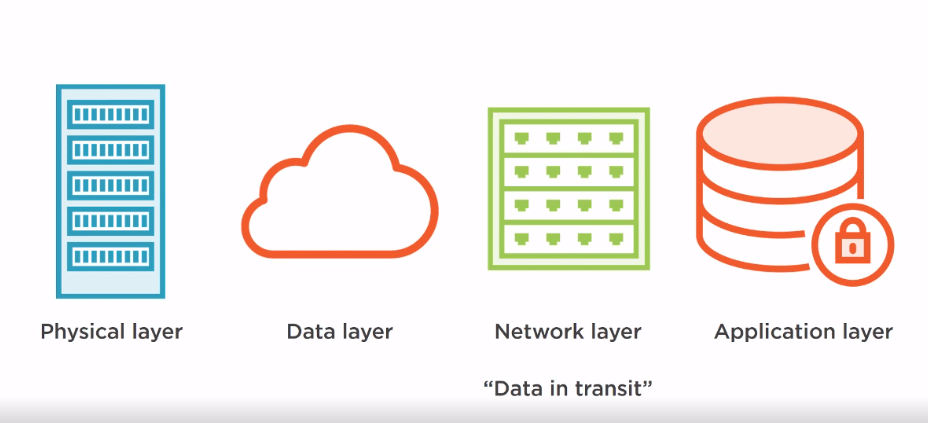
**Example of Loading Data directly from DynamoDB:**



**AWS RedShift Security**

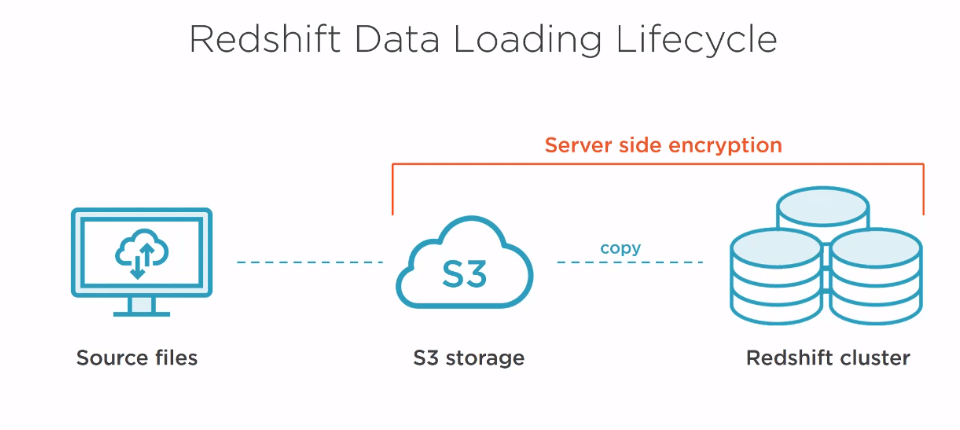


**Below shows the layers that we need to consider about security**



**Below is Server Side Encryption:**

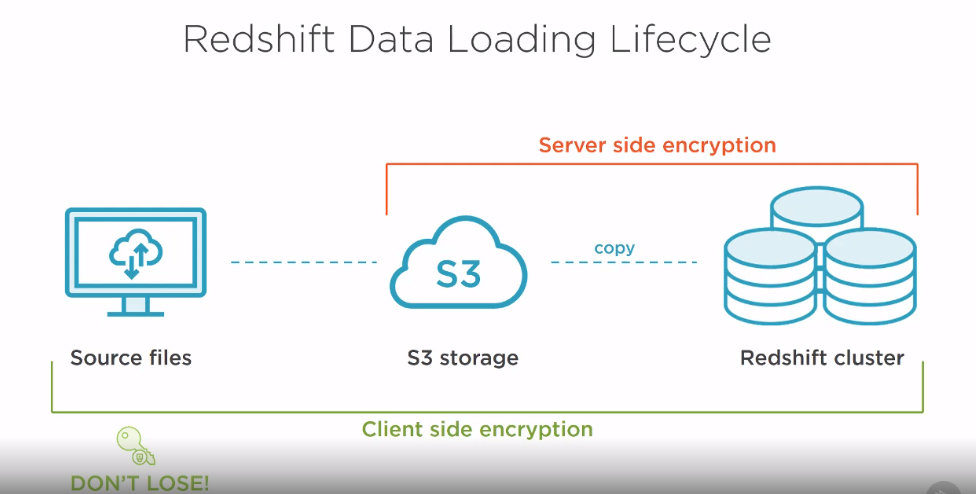
Where Data is encrypted between S3 and Resdshift Cluster and AWS is responsible for that when you choose that option.



**Client Side encryption:**

**Where client is responsible for the encryption all the way from client until redshift cluster.**

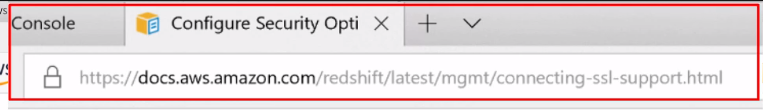
**NOTE: If you lose the key file then there is no way for AWS to retrieve that data.**



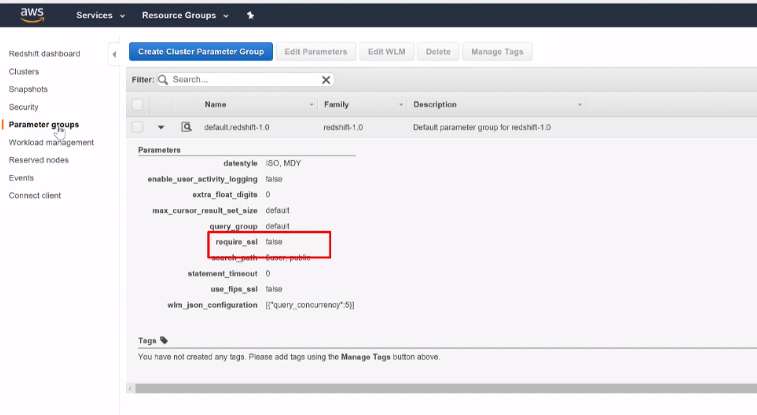
**Now the data at transit between the client to AWS and from AWS to client needs to be protected by SSL/TLS.**

**SSL certificates are installed by default on the cluster and fully supported at the client as long as you have the latest ODBC and JDBC Drivers.**

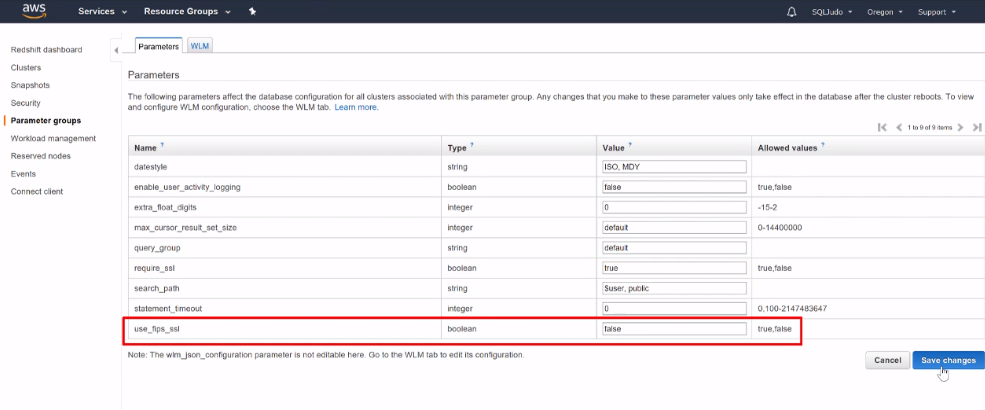
**Below URL will give how to setup SSL in Redshift**



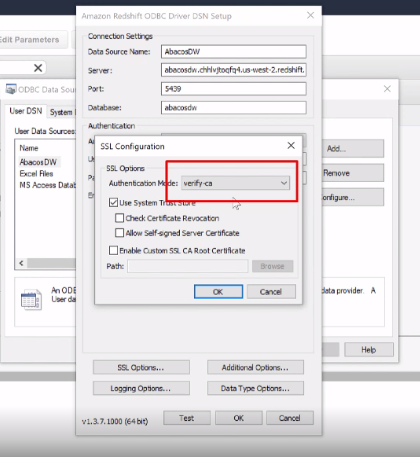
**SSL on the Cluster by default is always enabled but by default it is not required. In other words, SSL will be used only when the client requests for it or uses it.**



**You can enable SSL by creating your own parameter group and assign it to this redshift cluster. In addition you can enable FIPS mode if you need it.**

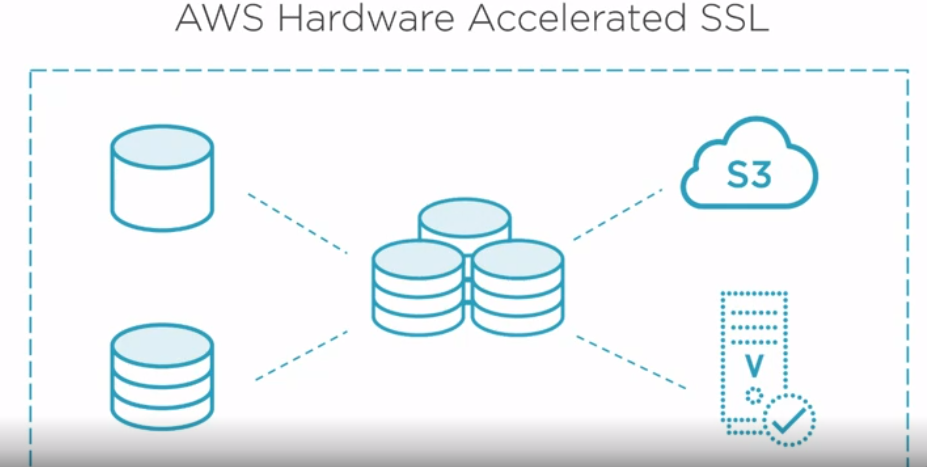


**Now, in the client ODBC, enable the SSL by setting the SSL options below:**

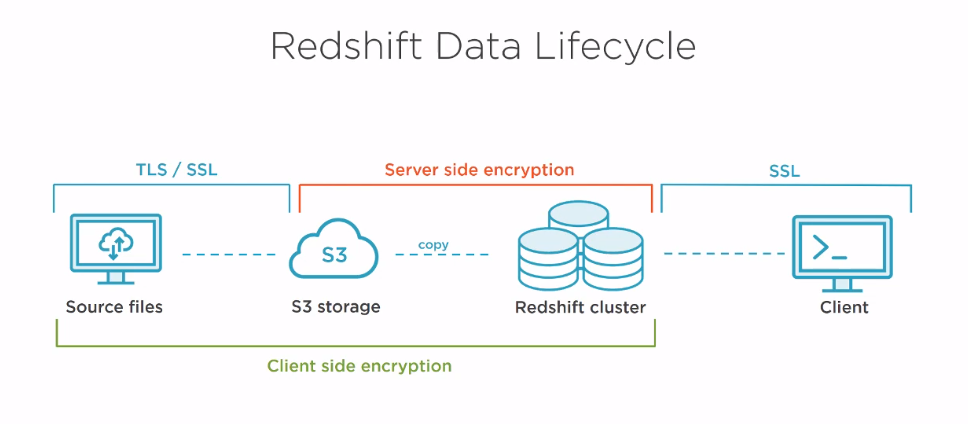


**Now the same SSL should be enabled between the source files and the AWS by establishing SSL within your code for the upload of files.**

**The data is encrypted with in AWS hardwire so we do not need to boher about that.**

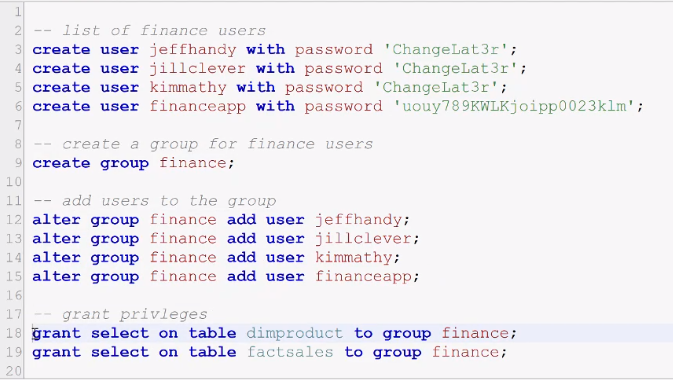


**Below is the complete security setup picture.**



**SECURITY inside Redshift:**

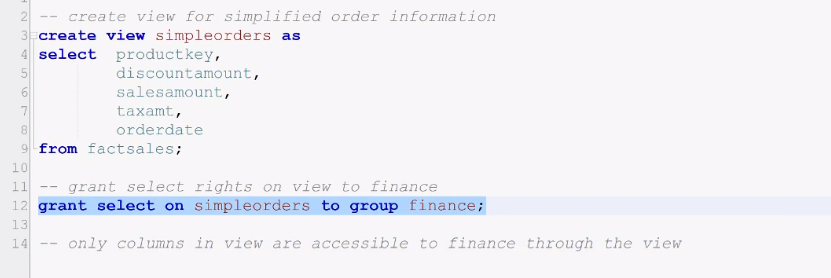
**You can establish security within Redshift by creating Users,Groups,privleges**



**You can crate schema and create tables inside that secured schema and give access to only certain users or groups .**

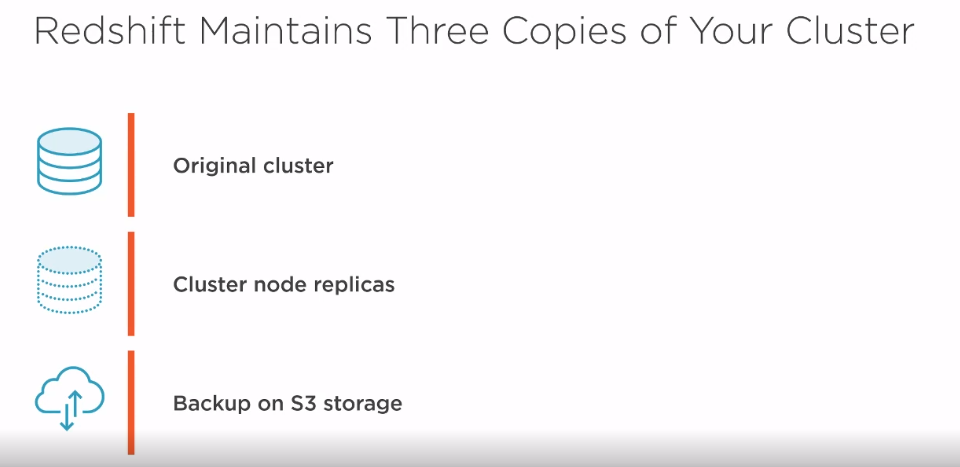


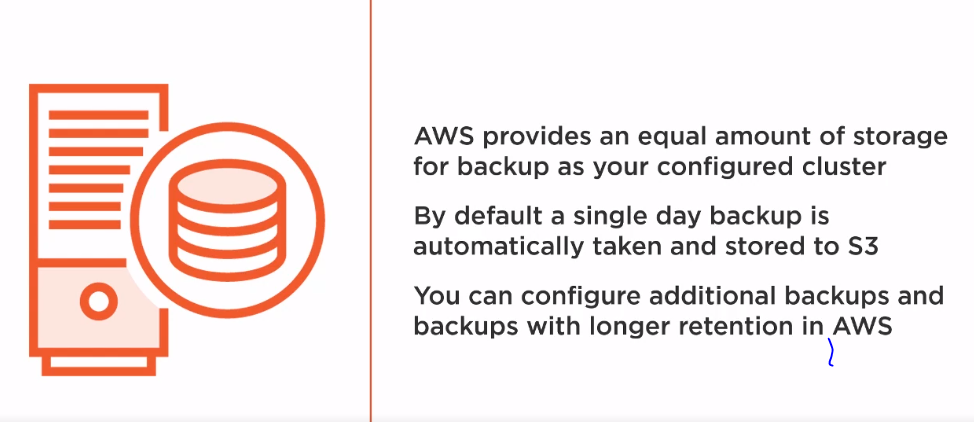
**You can also create View and restrict access to the actual table.**



**RedShift Backups:**

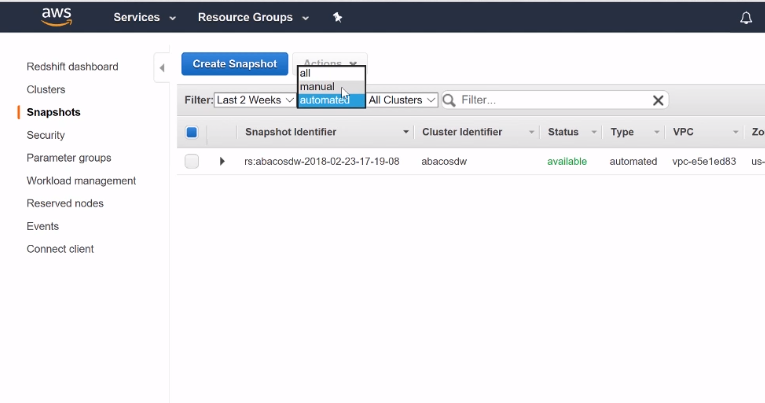
**Amazon Redshift cluster maintains 3 copies of your cluster at all times.**

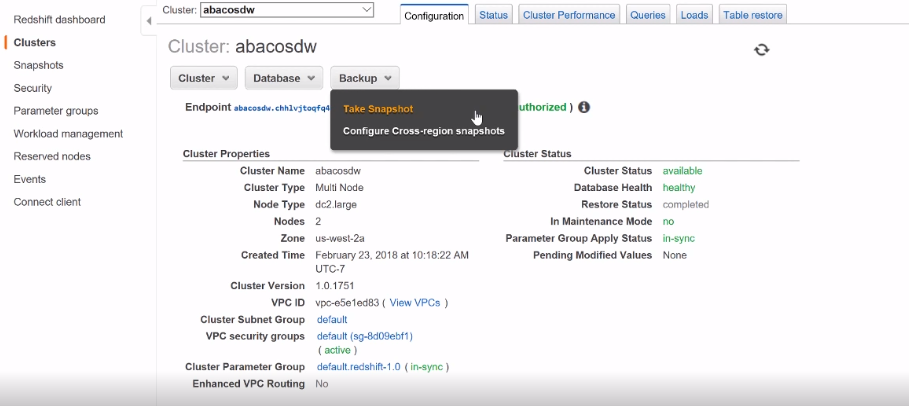




**Take Backups by using below screen.**

**Under Snapshots, you can view all automatic and manual snapshots that have been taken from the beginning.**



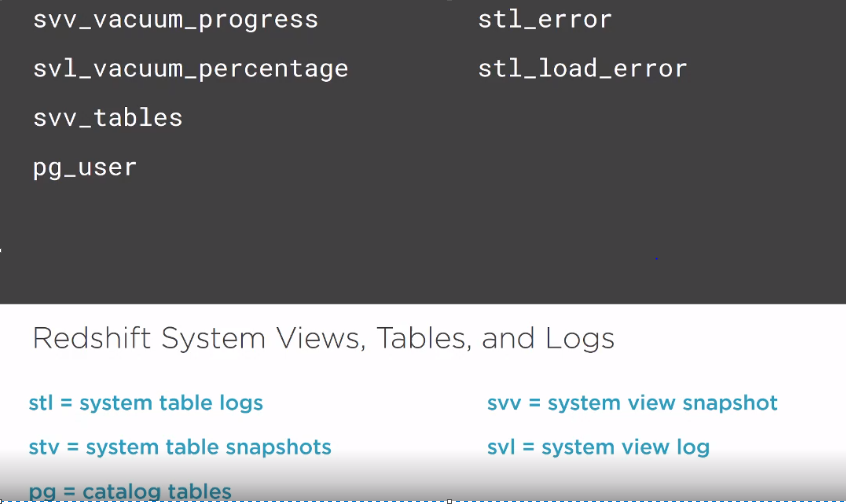


**Enable Cross Region Replication for Disaster Recovery:**

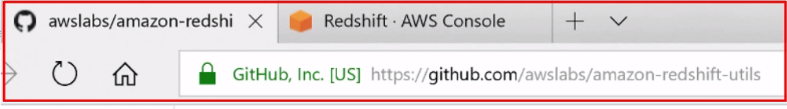


**System Log views:**

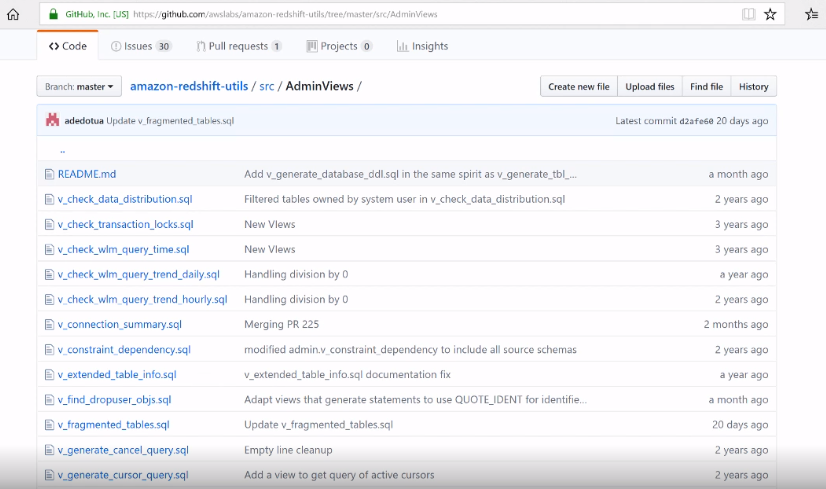
Below tables shows what happens in redshift by viewing this views to see the logs, system Info and other related things.

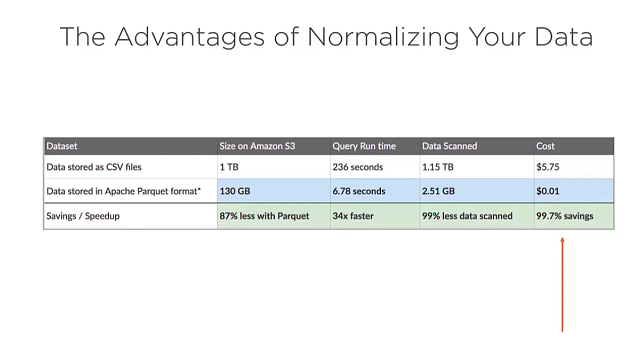


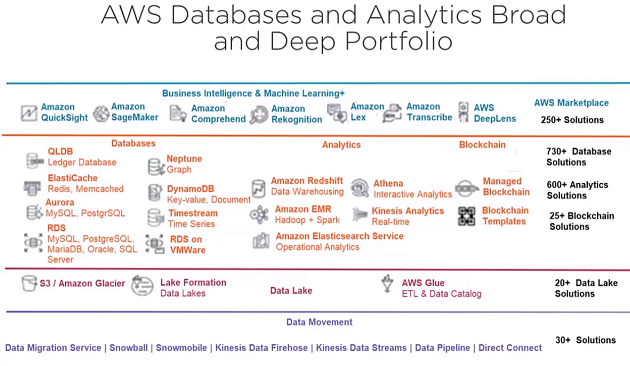
**Go to below GITHUB link to see/get more system utilities and their uses.**



**Example where you can a lot of system view related queries:**







You can load the file form s3 to Redshift by creating a lambda function.



Lambda function:

Below lambda function will read the file form s3 and will load it in database directly.

