Creating Tables and Loading Data In Them

You required these tables to conduct data analysis using Amazon Redshift. So, start by creating these tables on the Redshift cluster.

Step1: Create the schema **redshift_demo**.

```
create schema redshift_demo;
```

Step2: Create the table **users**.

```
create table redshift_demo.users(
      userid integer not null distkey sortkey,
      username char(8),
      firstname varchar(30),
      lastname varchar(30),
      city varchar(30),
      state char(2),
      email varchar(100),
      phone char(14),
      likesports boolean,
      liketheatre boolean,
      likeconcerts boolean,
      likejazz boolean,
      likeclassical boolean,
      likeopera boolean,
      likerock boolean,
      likevegas boolean,
      likebroadway boolean,
      likemusicals boolean);
```

Step2: Create the table **venue**.

```
create table redshift_demo.venue(
      venueid smallint not null distkey sortkey,
      venuename varchar(100),
      venuecity varchar(30),
      venuestate char(2),
      venueseats integer);
```

Step 3: Create the table **category**.

```
create table redshift_demo.category(
      catid smallint not null distkey sortkey,
      catgroup varchar(10),
      catname varchar(10),
      catdesc varchar(50));
```

Step 4: Create the table **date**.

```
create table redshift_demo.date(
      dateid smallint not null distkey sortkey,
      caldate date not null,
      day character(3) not null,
      week smallint not null,
      month character(5) not null,
      qtr character(5) not null,
      year smallint not null,
      holiday boolean default('N'));
```

Step 5: Create the table **event**.

```
create table redshift_demo.event(
      eventid integer not null distkey,
      venueid smallint not null,
      catid smallint not null,
      dateid smallint not null sortkey,
      eventname varchar(200),
      starttime timestamp);
```

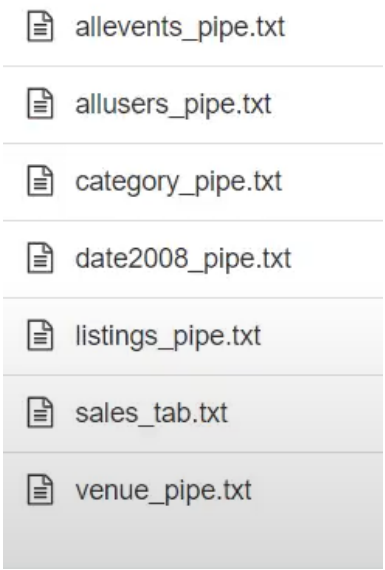
Step 6: Create the table **listing**.

```
create table redshift_demo.listing(
      listid integer not null distkey,
      sellerid integer not null,
      eventid integer not null,
      dateid smallint not null  sortkey,
      numtickets smallint not null,
      priceperticket decimal(8,2),
      totalprice decimal(8,2),
      listtime timestamp);
```

Step 7: Create the table **sales**.

```
create table redshift_demo.sales(
      salesid integer not null,
      listid integer not null distkey,
      sellerid integer not null,
      buyerid integer not null,
      eventid integer not null,
      dateid smallint not null sortkey,
      qtysold smallint not null,
      pricepaid decimal(8,2),
      commission decimal(8,2),
      saletime timestamp);
```

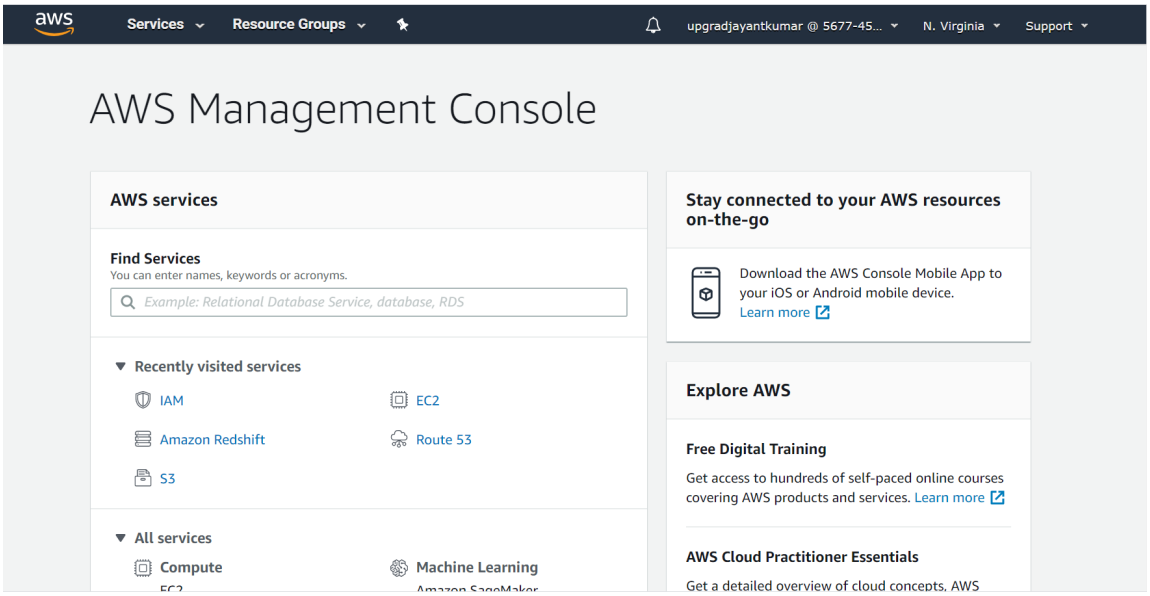
In your S3 bucket, please check whether the data set is present or not. If it is not, then the error 'file not found' will be thrown. So, recheck carefully, and if it is not available, then upload the **tickit** data set to the S3 bucket.



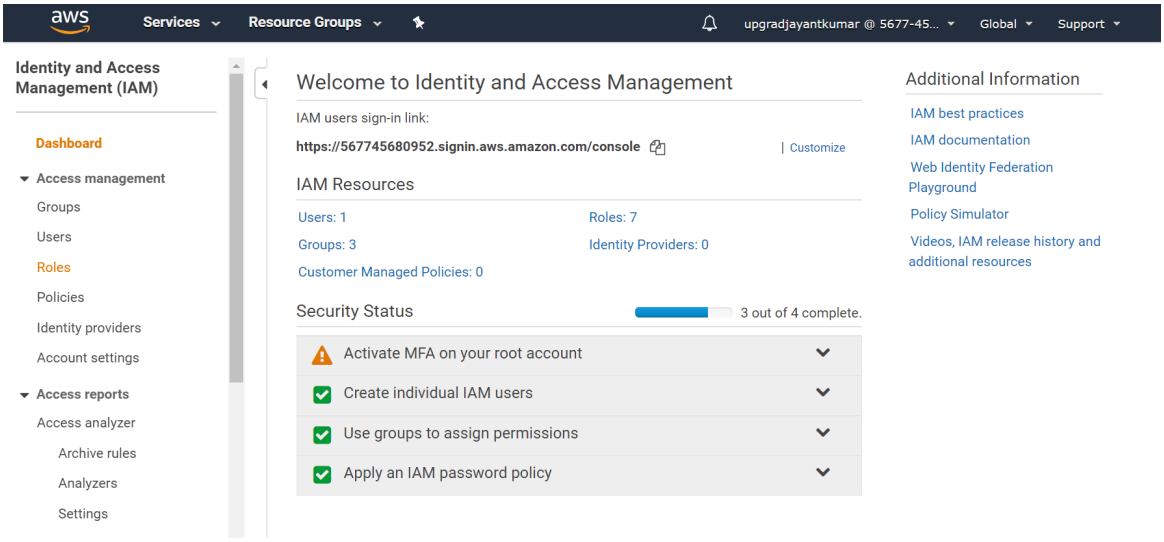
Once the data set is available at your S3 bucket, execute the following commands step by step to load data into tables.

In the copy command, you need the role ARN number.

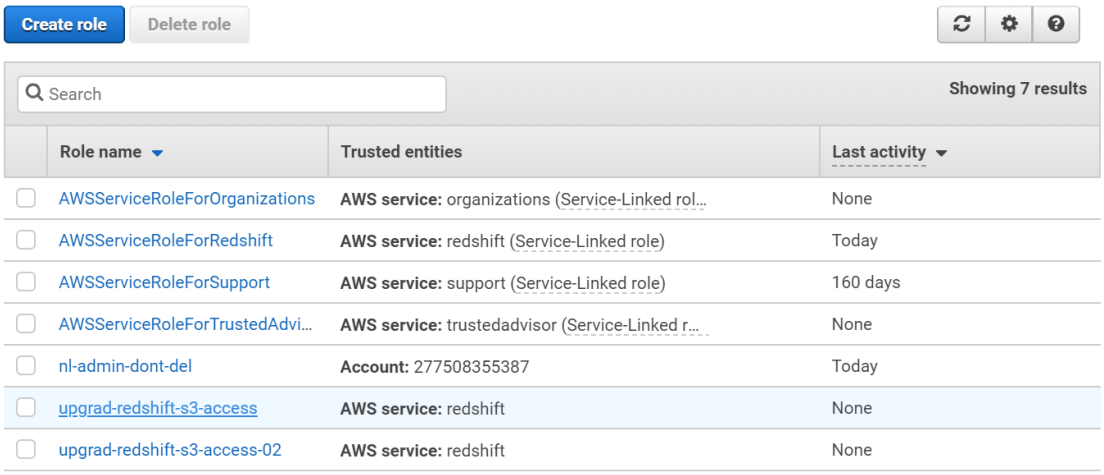
1. Search for the **IAM** in the AWS Dashboard.



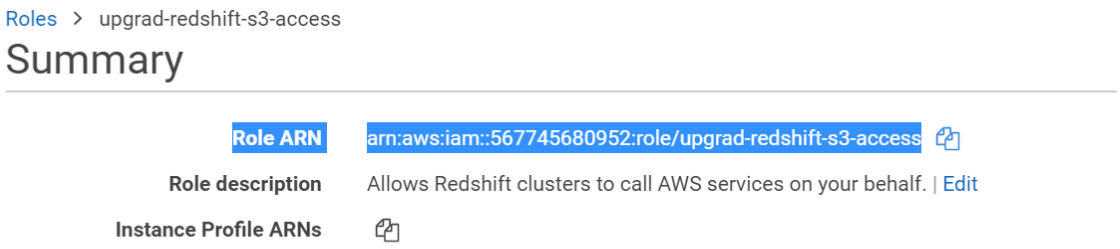
2. Click on **Roles** that are present on the left-hand side.



3. Click on the role that you created. Here, the role name is '**upgrad-redshift-s3-access**'.



4. On the right-hand side of the image given below at the **Role ARN** option, you will notice a two-file symbol that you can use to copy the **Role ARN.**



Step 1: Load the data into the table **users**, and paste the **Role ARN** to **iam_role** in the query given below.

```
copy redshift_demo.users from
's3://redshift-demo-upgrad/tickit/allusers_pipe.txt'
iam_role 'arn:aws:iam::567745680952:role/upgrad-redshift-s3-access'
delimiter '|' region 'us-east-1';
```

Step 2: Load the data into the table **venue,** and paste the **Role ARN** to **iam_role** in the query given below.

```
copy redshift_demo.venue from
's3://redshift-demo-upgrad/tickit/venue_pipe.txt'
iam_role 'arn:aws:iam::567745680952:role/upgrad-redshift-s3-access'
delimiter '|' region 'us-east-1';
```

Step 3: Load the data into the table **category,** and paste the **Role ARN** to **iam_role** in the query given below.

```
copy redshift_demo.category from
's3://redshift-demo-upgrad/tickit/category_pipe.txt'
iam_role 'arn:aws:iam::567745680952:role/upgrad-redshift-s3-access'
delimiter '|' region 'us-east-1';
```

Step 4: Load the data into the table **date,** and paste the **Role ARN** to **iam_role** in the query given below.

```
copy redshift_demo.date from
's3://redshift-demo-upgrad/tickit/date2008_pipe.txt'
iam_role 'arn:aws:iam::567745680952:role/upgrad-redshift-s3-access'
delimiter '|' region 'us-east-1';
```

Step 5: Load data into the table **event,** and paste the **Role ARN** to **iam_role** in the query given below.

```
copy redshift_demo.event from
's3://redshift-demo-upgrad/tickit/allevents_pipe.txt'
iam_role 'arn:aws:iam::567745680952:role/upgrad-redshift-s3-access'
delimiter '|' region 'us-east-1';
```

Step 6: Load the data into the table **listing,** and paste the **Role ARN** to **iam_role** in the query

```
copy redshift_demo.listing from
's3://redshift-demo-upgrad/tickit/listings_pipe.txt'
iam_role 'arn:aws:iam::567745680952:role/upgrad-redshift-s3-access'
delimiter '|' region 'us-east-1';
```

Step 7: Load the data into the table **sales,** and paste the **Role ARN** to **iam_role** in the following query.

```
copy redshift_demo.sales from 's3://redshift-demo-upgrad/tickit/sales_tab.txt'
iam_role 'arn:aws:iam::567745680952:role/upgrad-redshift-s3-access'
delimiter '\t' region 'us-east-1' timeformat 'MM/DD/YYYY HH:MI:SS';
```