Amazon Redshift Serverless

Use cases :

AnyCompany Events manages event ticket sales for various events around the globe. They currently use an in-house solution named TICKIT to track all of the event and sales-related data. TICKIT is a data-intensive digital platform dedicated to event management and ticket sales. It encounters a significant challenge - data silos. With information originating from various sources such as transactional databases, batch processing, and real-time streaming, the data ends up scattered across different systems. This fragmentation impedes TICKIT's ability to have a comprehensive view of its operations, customer preferences, and event-specific data. As a data warehouse specialist at AnyCompany Events, your task is to investigate alternative solutions to enhance your application's performance and data insight capabilities.

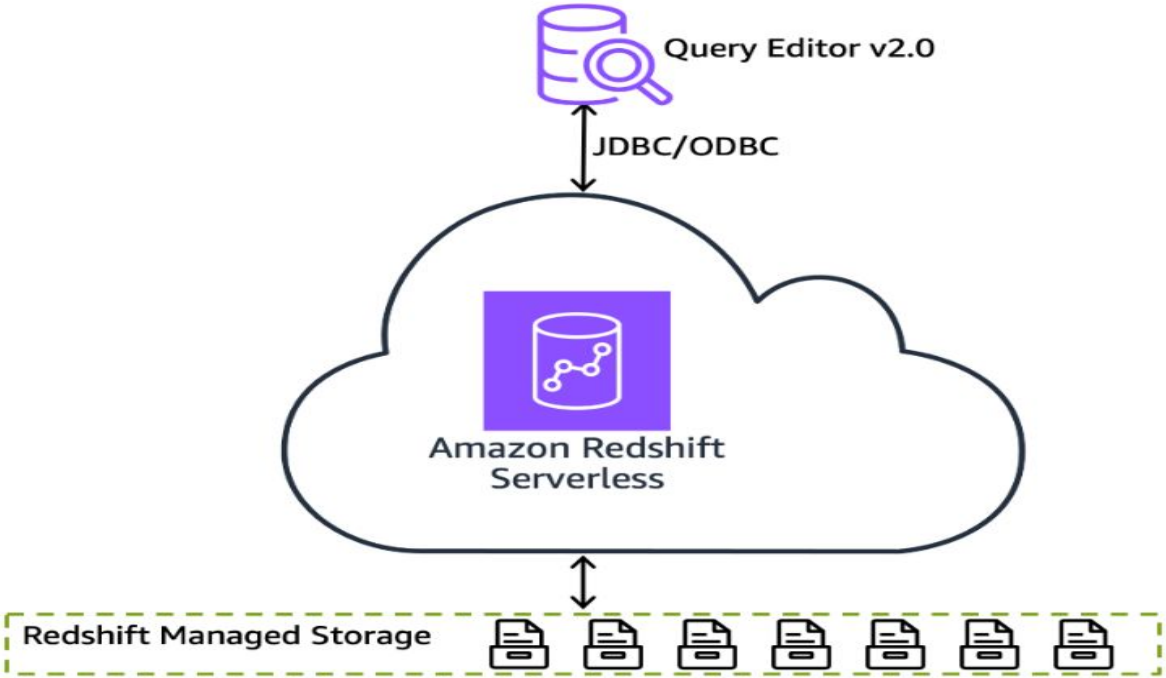
In this lab, you explore an Amazon Redshift Serverless endpoint. You then explore the sample data in your production database. Finally, you run various queries to validate that the data has been imported correctly.

Objectives

By the end of this lab, you should be able to do the following :

- Explore an Amazon Redshift Serverless endpoint.
- Run queries to discover data within the data set.

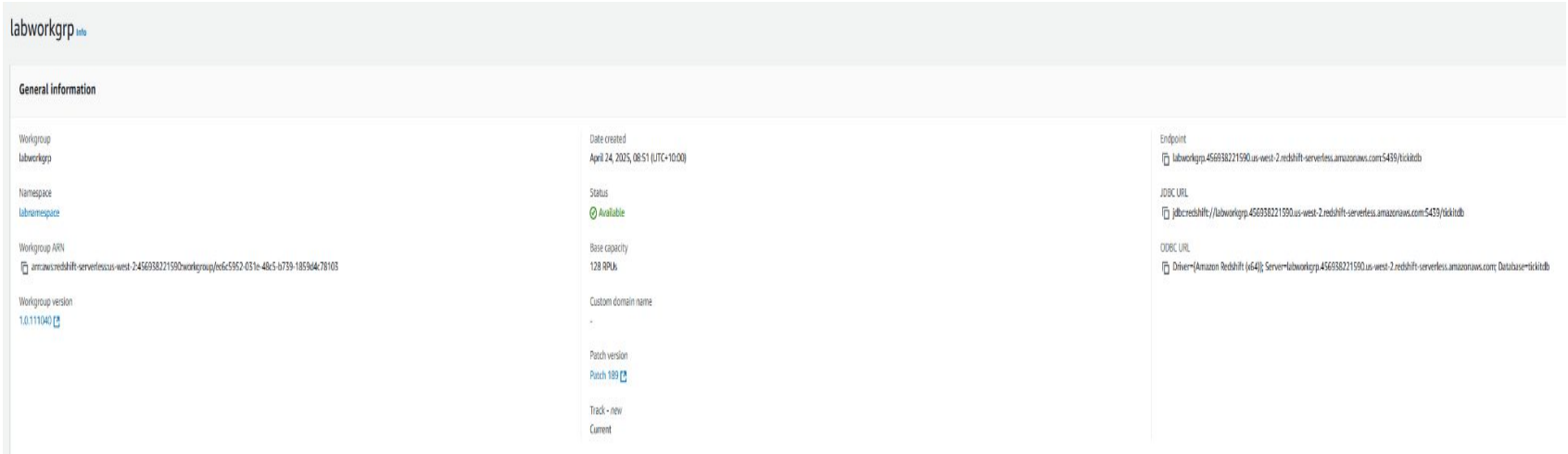
Amazon Redshift Serverless basic Arch



At the top of the AWS Management Console, in the search bar, search for and choose `Amazon Redshift`.

In the left navigation pane, choose Workgroup configuration.
Note: A *workgroup* is a collection of compute resources.

You can select it to view and edit the work group's properties.



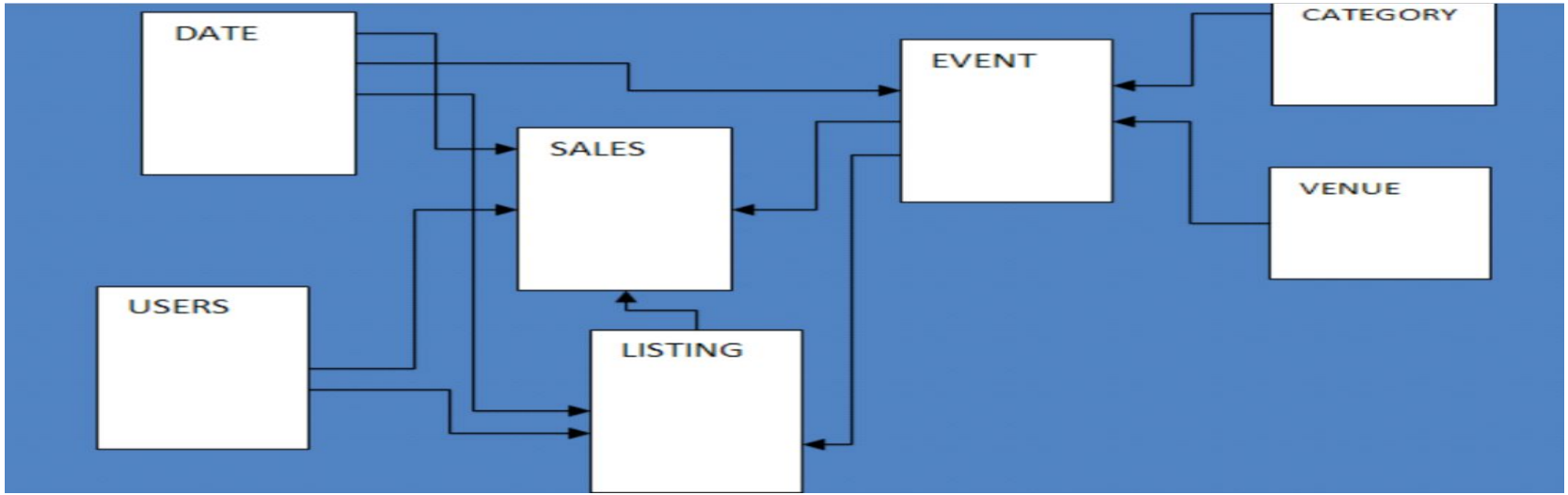
Run queries to discover data within the dataset

In this task, you connect to an Amazon Redshift Serverless endpoint and query the data from the TICKIT database using query editor v2.

Amazon Redshift Serverless connects to the serverless environment in the provisioned AWS Region. Amazon Redshift Serverless runs in a VPC within the port ranges port ranges 5431-5455 and 8191-8215. The default is 5439.

This sample database application helps analysts track sales activity for the fictional TICKIT web site, where users buy and sell tickets online for sporting events, shows, and concerts.

In particular, analysts can identify ticket movement over time, success rates for sellers, and the best-selling events, venues, and seasons. Analysts can use this information to provide incentives to buyers and sellers who frequent the site, to attract new users, and to drive advertising and promotions.



On the lab namespace page, choose **Query data**.In the new browser tab, you might get the following caution in a red color banner: User information couldn't be retrieved. You must have an account to use Redshift Query Editor V2. Choose x to close the prompt.

To create an account to use Redshift Query Editor V2, on the AWS KMS encryption page, leave all the values at the default settings and choose **Configure account**.
The query editor now opens in a new browser tab.

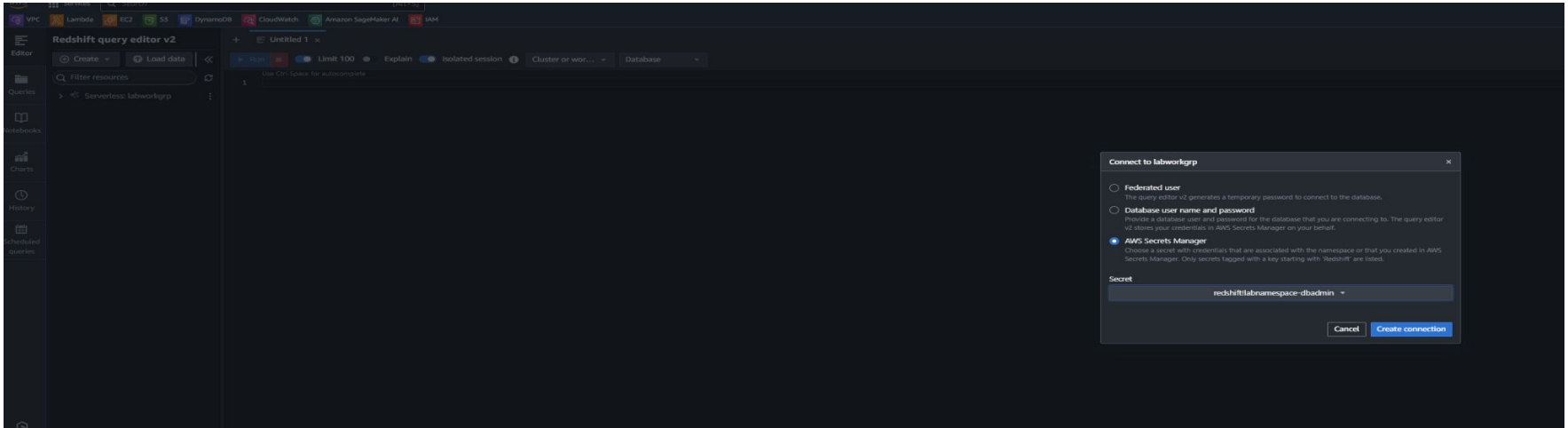
In the Redshift query editor v2 pane, choose the vertical ellipses (three dots) at Serverless:labworksgrp.

Choose Create connection.

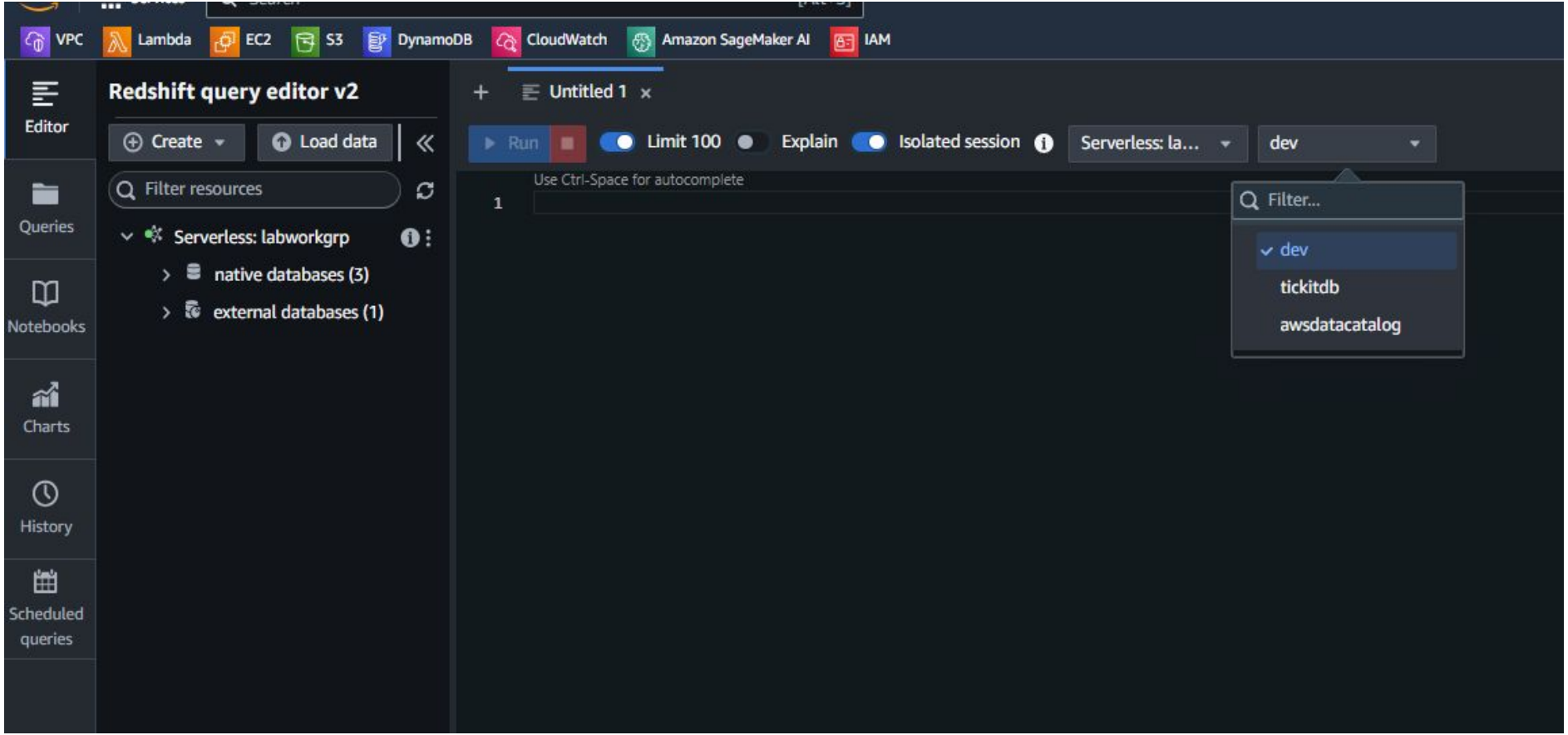
On the Connect to labworkgroup window prompt, choose AWS Secrets Manager.

Select Choose a secret drop-down menu and choose redshift!labnamespace-dbadmin.

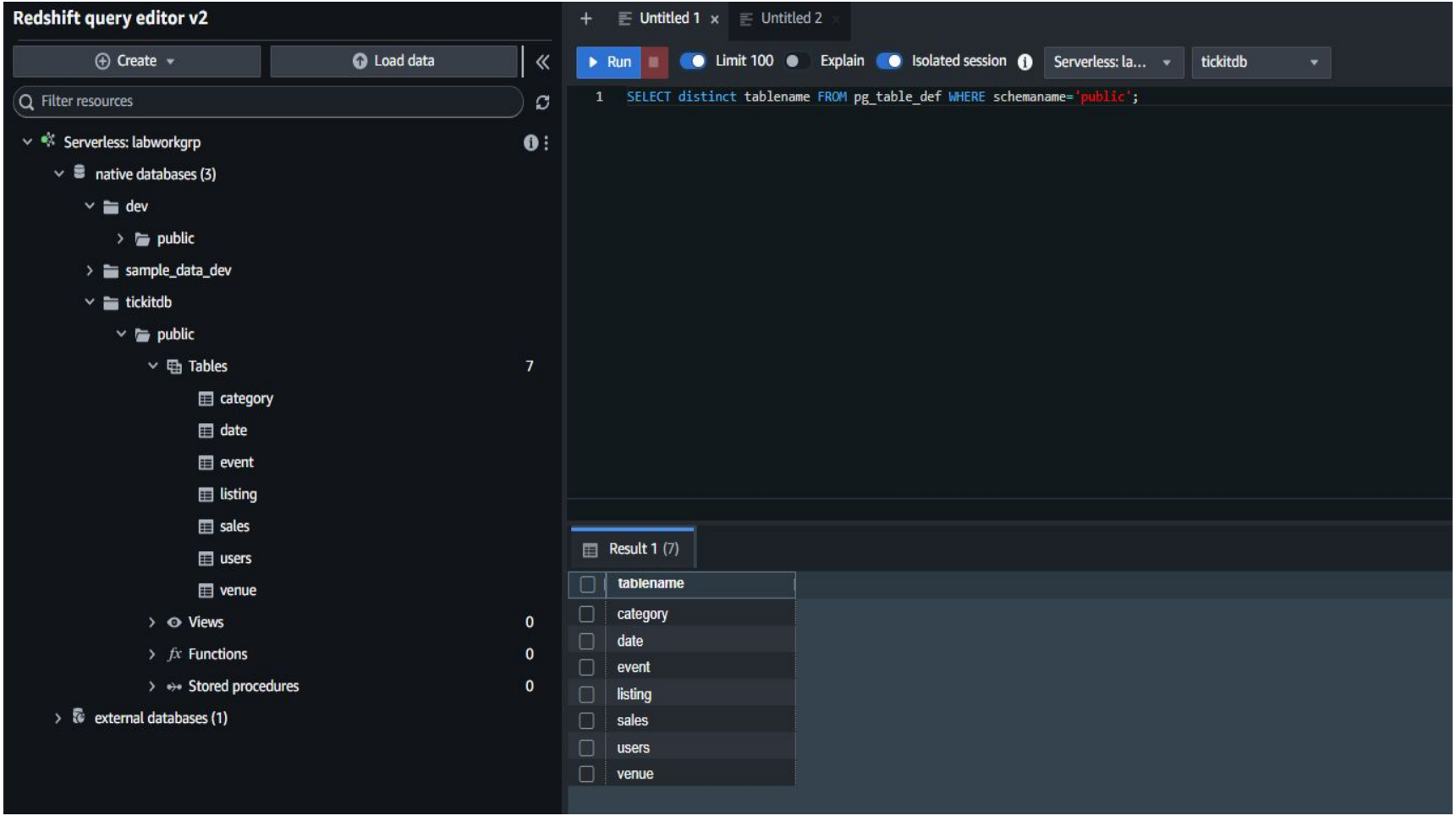
Choose **Create connection**. On the top right, choose the newly created tickitdb database instead of the *dev* database.



Choose the newly created tickitdb database instead of the *dev* database.



Choose the newly created tickitdb database instead of the *dev* database.



Saple Quries

Command: To find total sales on a given calendar date, run the following query:

```
SELECT sum(qtysold)
FROM   sales, date
WHERE  sales.dateid = date.dateid
  AND    caldate = '2008-01-05';
```

Command: To find the top five sellers in San Diego based on the number of tickets sold in 2008, run the following query:

```
select sellerid, username, (firstname ||' '|| lastname) as name,
city, sum(qtysold)
from sales, date, users
where sales.sellerid = users.userid
and sales.dateid = date.dateid
and year = 2008
and city = 'San Diego'
group by sellerid, username, name, city
order by 5 desc
  limit 5;
```