AWS Redshift

Introduction

Amazon Redshift is a fully managed, petabyte-scale data warehouse service in the cloud. It enables users to analyze large datasets using standard SQL and existing Business Intelligence (BI) tools.

Key Features

1. Scalability

- **Massively Parallel Processing (MPP):** Distributes data and query load across multiple nodes.
- **Elastic Resize:** Adjusts the size of your data warehouse cluster with minimal downtime.

2. Performance

- **Columnar Storage:** Stores data in a columnar format, optimizing I/O and reducing storage costs.
- **Data Compression:** Automatically compresses data to save storage and improve query performance.
- **Query Optimization:** Advanced query optimization techniques enhance performance.

3. Security

- **Data Encryption:** Supports encryption at rest and in transit using AWS Key Management Service (KMS).
- **Network Isolation:** Integration with Amazon VPC provides enhanced security through network isolation.
- **Access Control:** Role-based access control allows fine-grained permissions for users and groups.

4. Cost-Effective

- **Pay-as-you-go Pricing:** Only pay for what you use with no upfront costs.
- **Reserved Instances:** Option to reserve instances for a lower hourly rate.

5. Integration with AWS Ecosystem

- **Seamless Integration:** Works with AWS services like S3, DynamoDB, and AWS Glue for data ingestion and ETL processes.
- **Data Lake Integration:** Easily integrates with data lakes for advanced analytics.

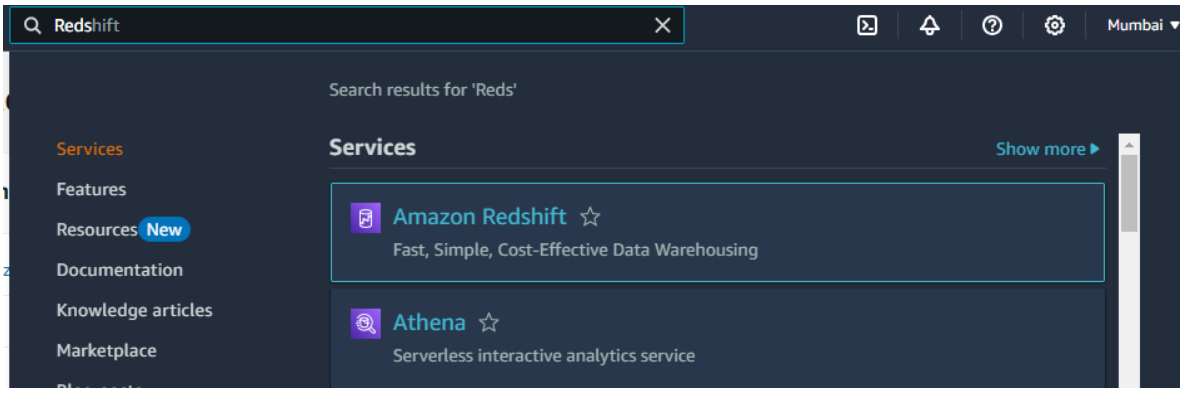
Use Cases

- **Business Intelligence:** Run complex queries and analytics to drive business decisions.
- **Data Warehousing:** Consolidate data from multiple sources for reporting and analysis.
- **ETL Processes:** Efficiently extract, transform, and load large datasets.

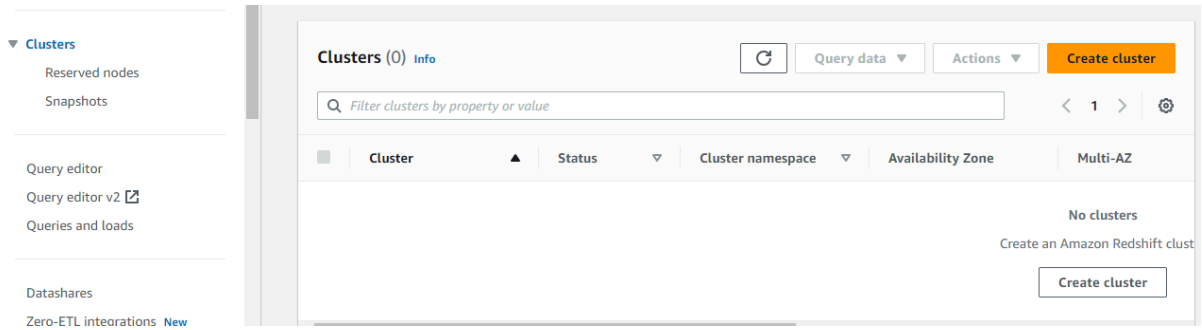
AWS Redshift Cluster

1. Create an Amazon Redshift Cluster

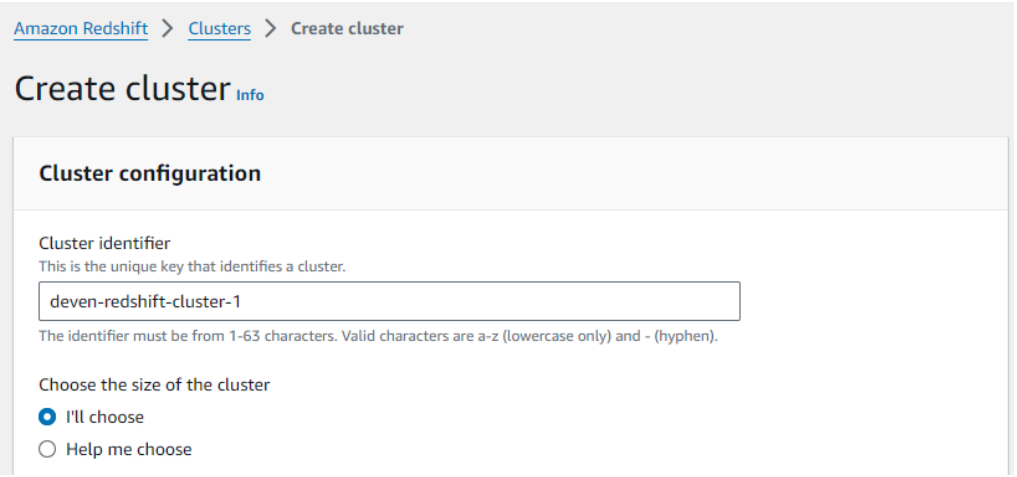
1. **Log in to AWS Management Console**:
    - Go to the AWS Management Console.
2. **Navigate to Amazon Redshift**:
    - Search for and select **Amazon Redshift** from the services menu.



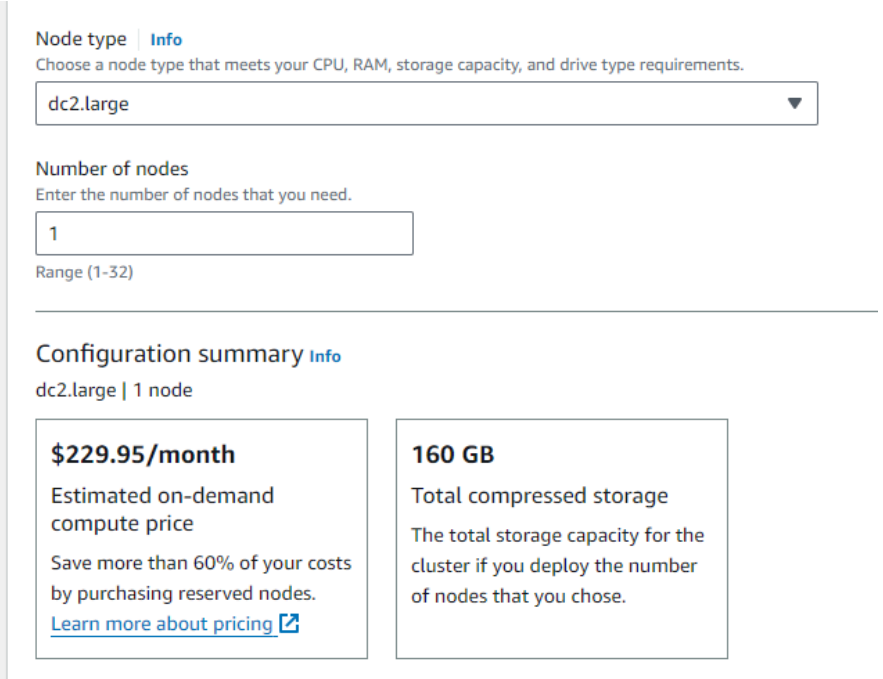
3. **Create a Cluster**:
    - Click **Create cluster**.

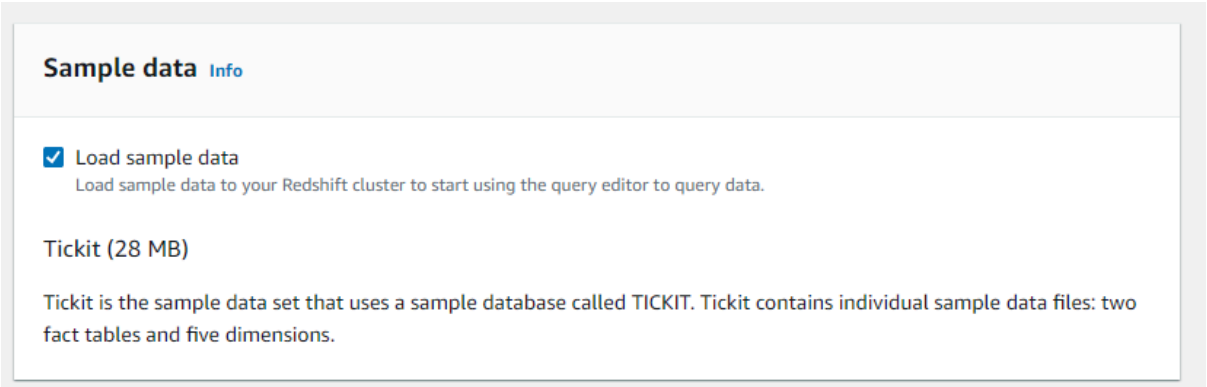


- **Cluster Identifier**: Choose a unique name (e.g., `deven-redshift-cluster-1`).

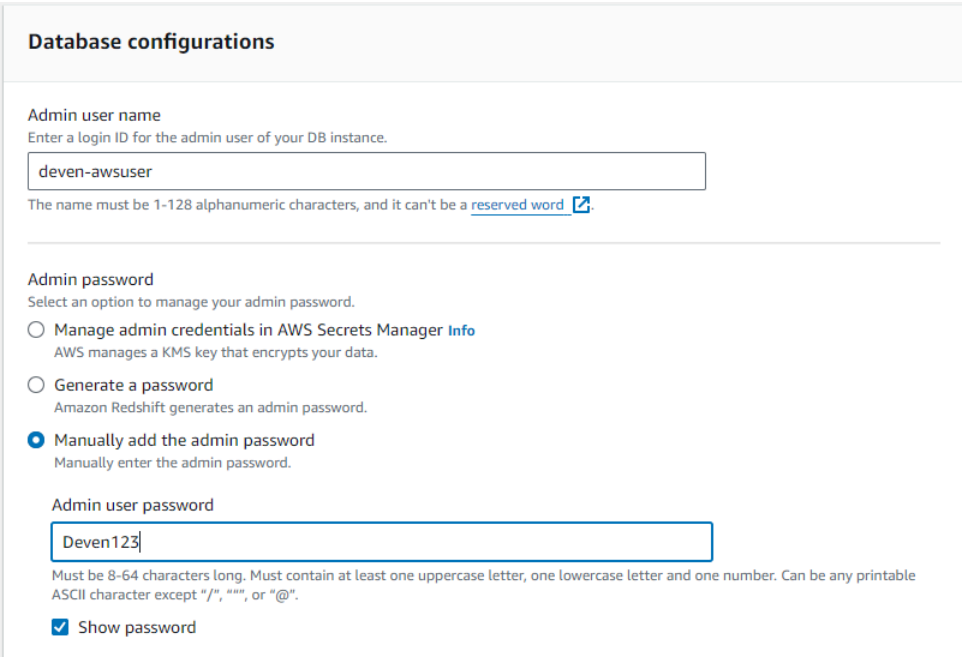


- **Node Type**: Select the appropriate node type for your workload (e.g., `dc2.large`). and Start with one node for practice.

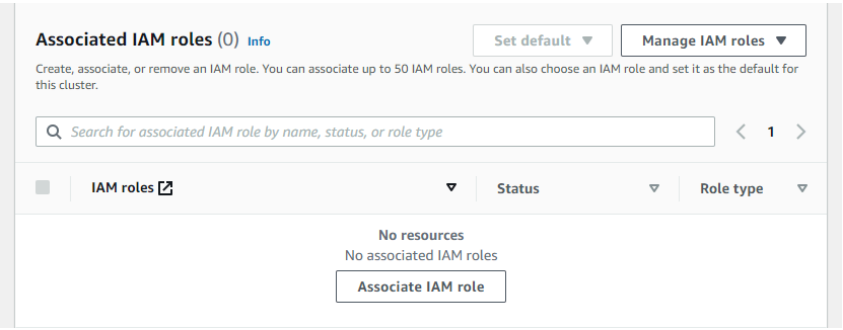


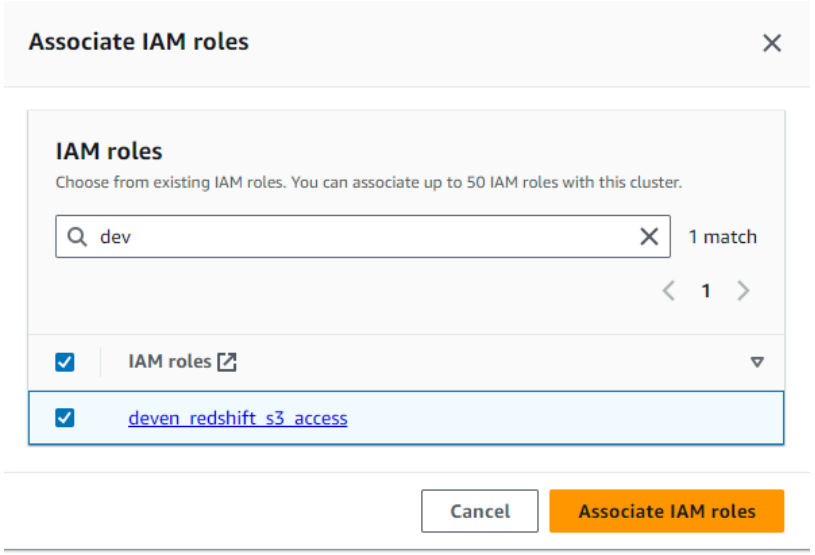


- **Master Username**: Choose an admin username and **Master Password**: Set a password and confirm it.

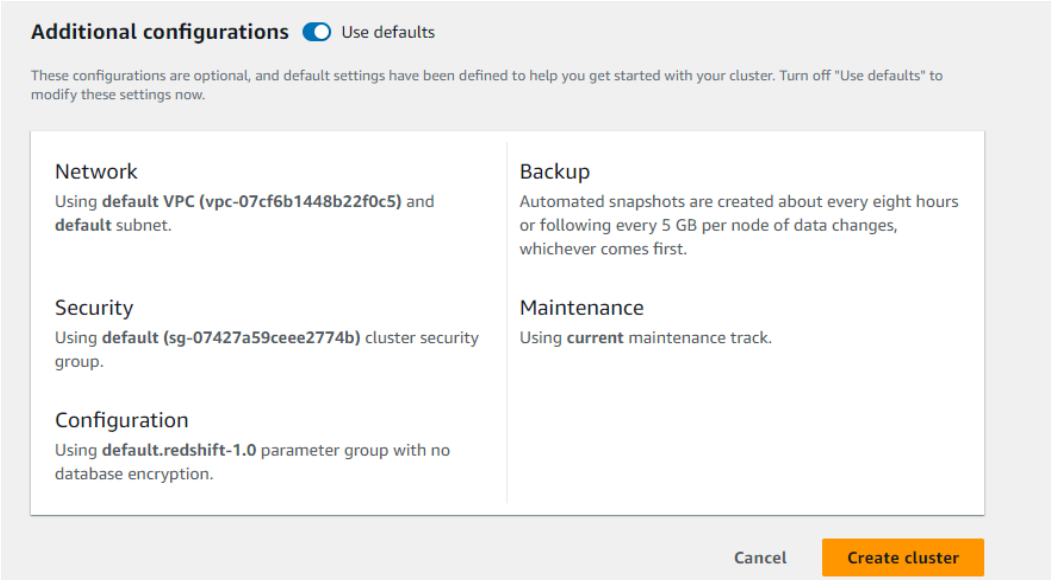


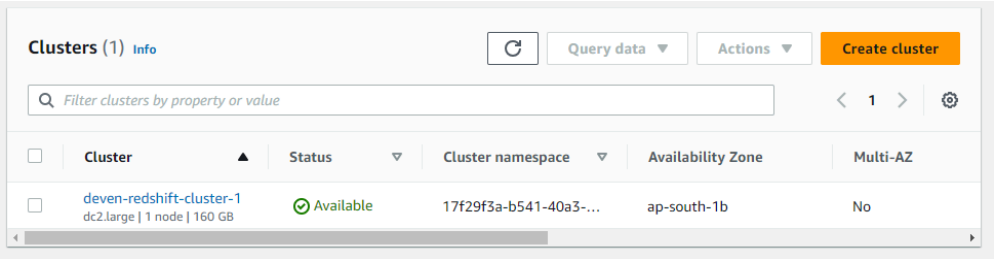
4. **Configure Security Settings**:
   - Choose or create a new VPC and configure security groups to allow access to the cluster (e.g., open port 5439 for PostgreSQL).



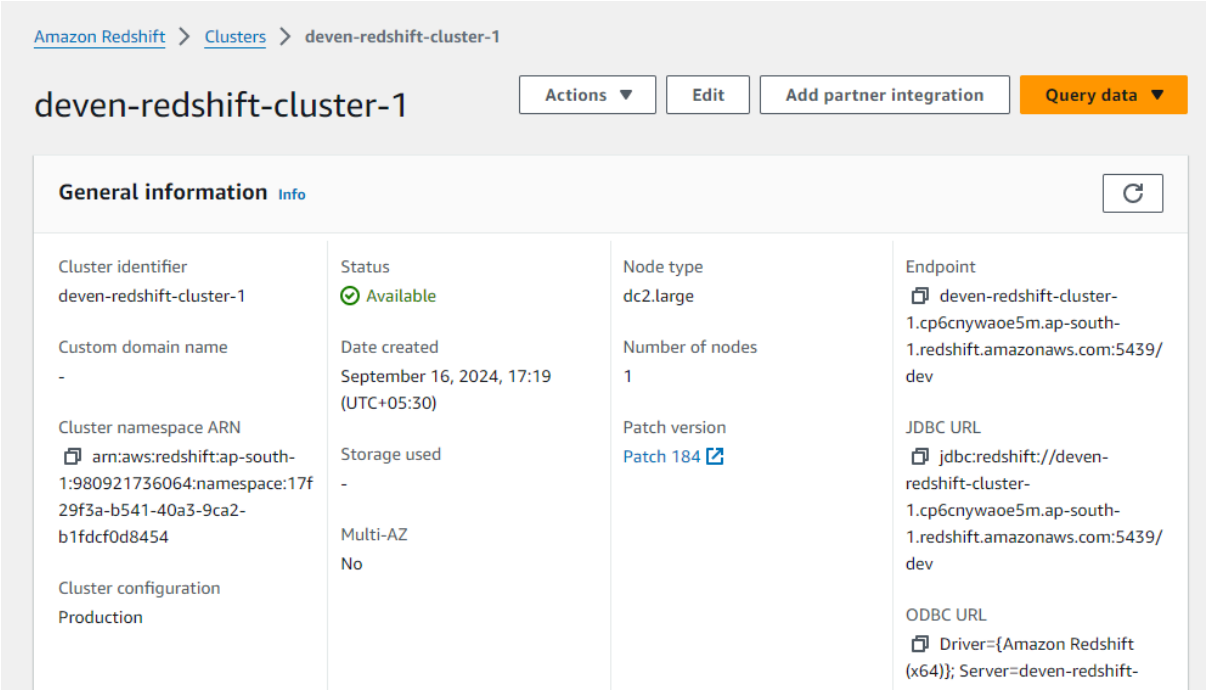


5. **Launch the Cluster**:
   - Review your settings and click **Create cluster**. Wait for the cluster status to change to **Available**.

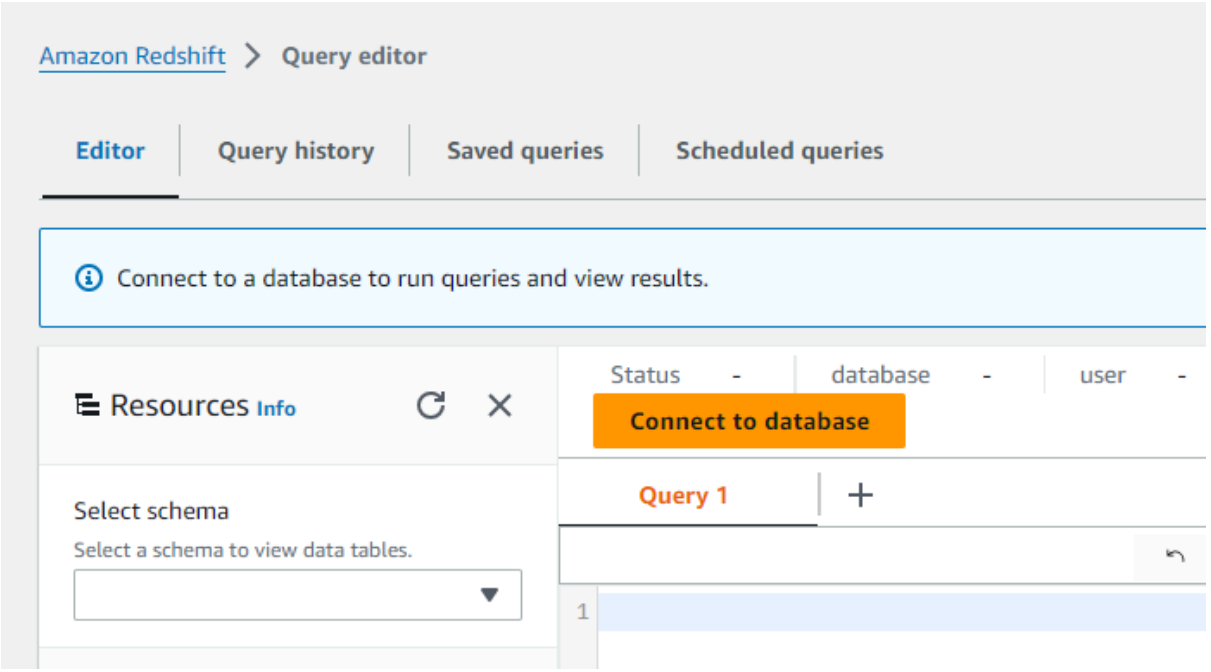


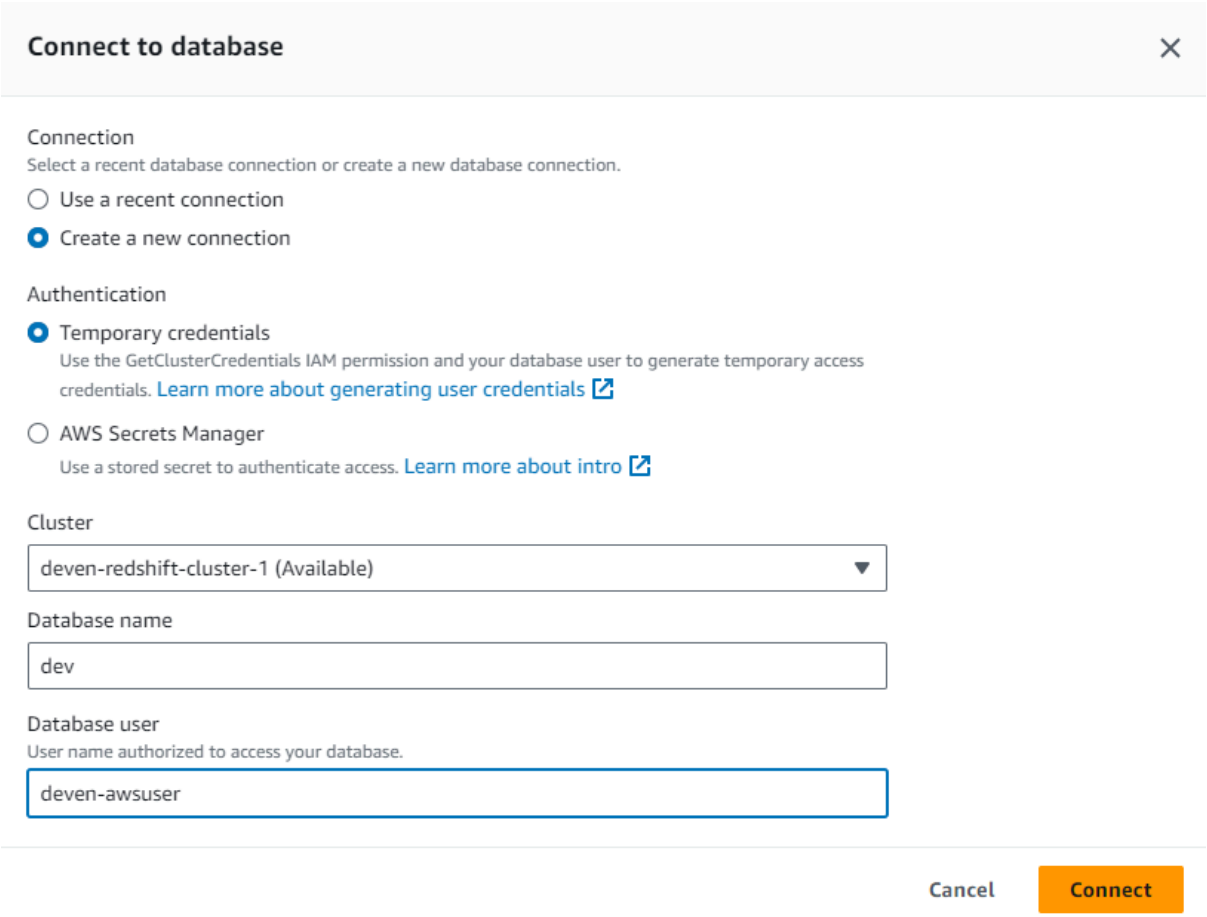


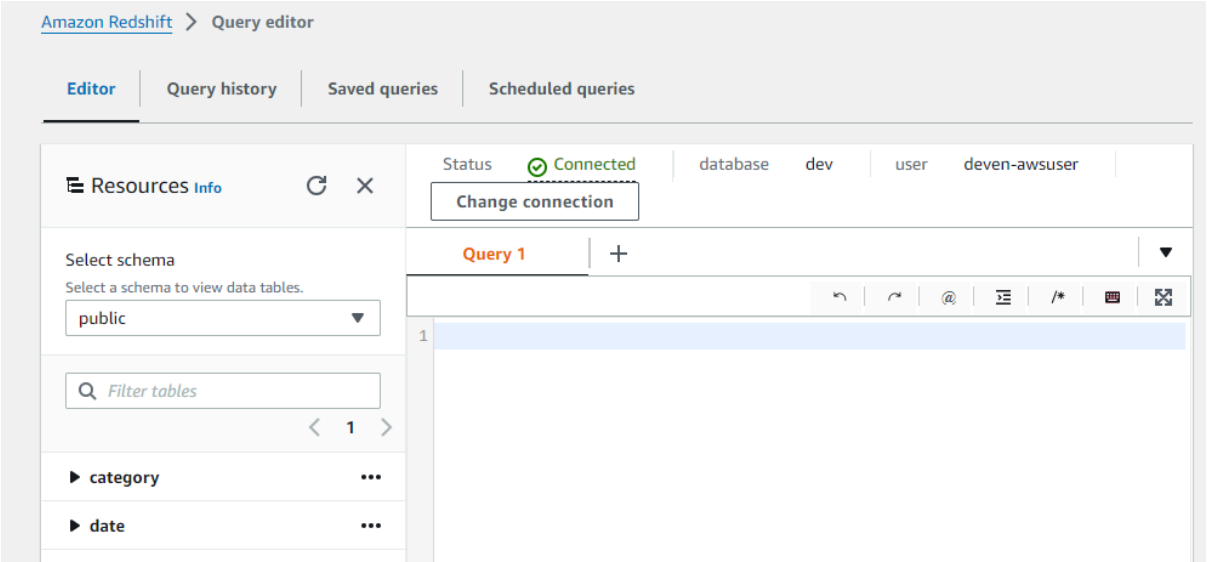
   - Click on **Query data**



- In Editor click connect to database and make connection with through your created master username and password

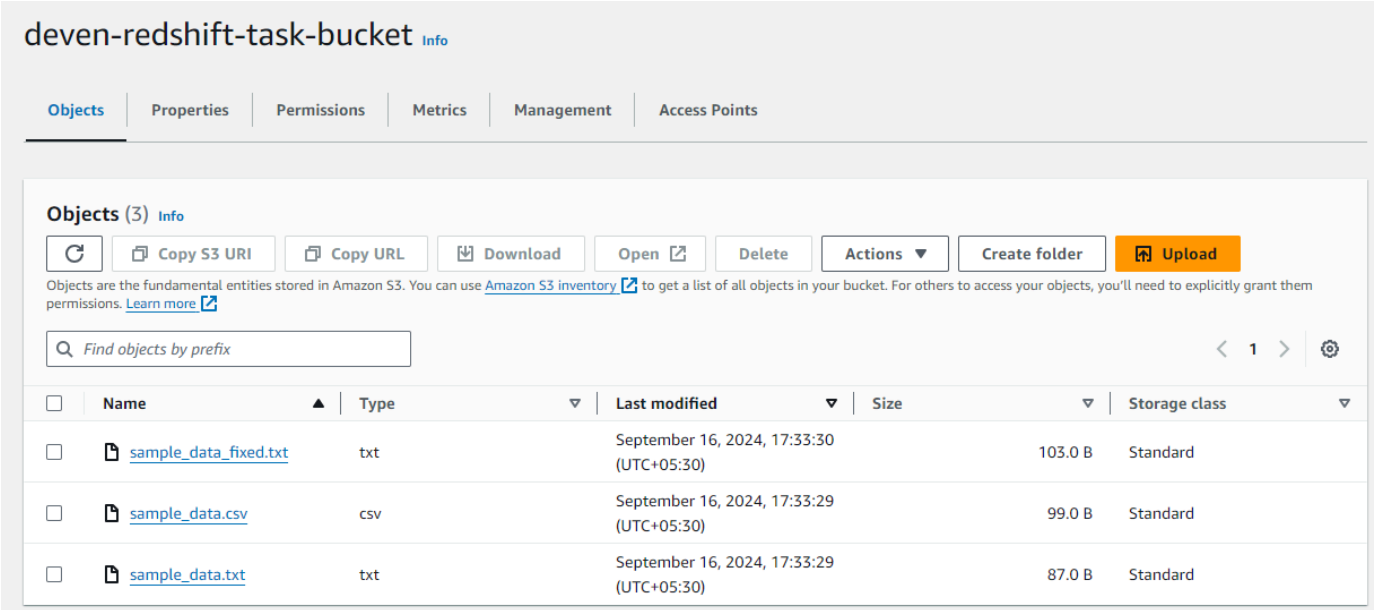






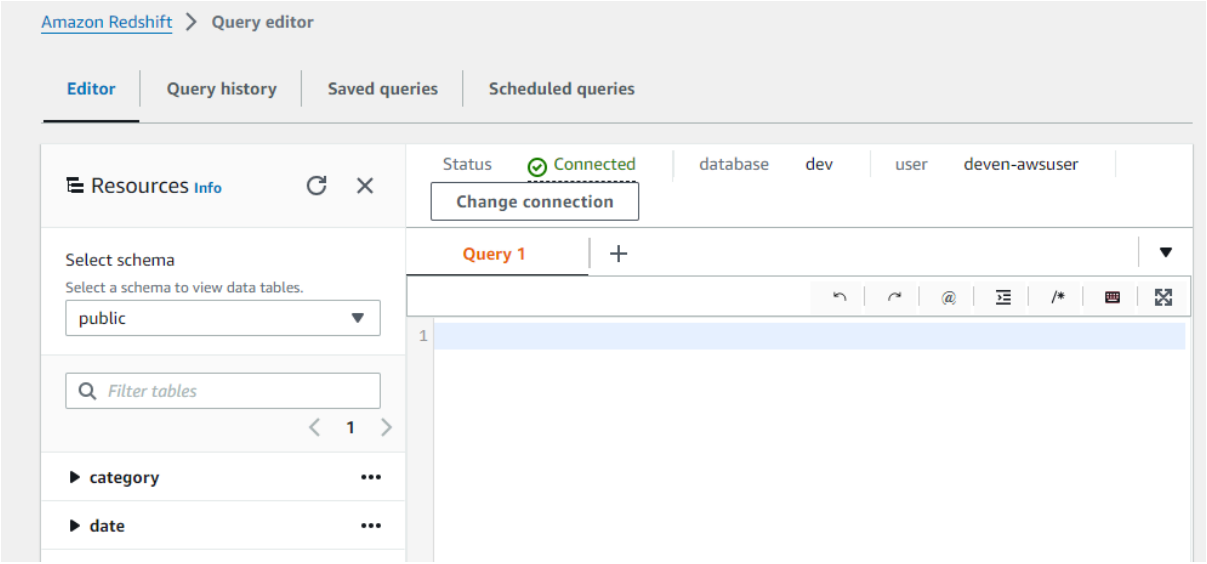
2. Upload Data Files to S3

- **CSV File**: `sample_data.csv`
- **Character-Delimited File**: `sample_data.txt` (using a character like `|` for delimiting)
- **Fixed Width File**: `sample_data_fixed.txt` (columns have fixed widths)



4. Create Tables in the Redshift Database

1. **Run Query in Redshift Editor**:



Create Tables: Write SQL commands to create tables in your Redshift database. For example:

code

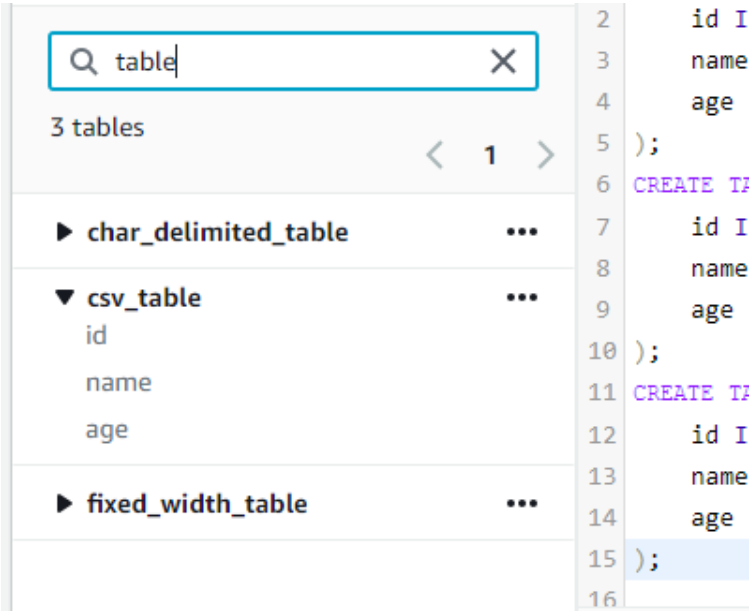
```
-- For CSV file
CREATE TABLE csv_table (
    id INT,
    name VARCHAR(100),
    age INT
);

-- For character-delimited file (using | as delimiter)
CREATE TABLE char_delimited_table (
    id INT,
    name VARCHAR(100),
    age INT
);

-- For fixed width file (assuming fixed width for each column)
CREATE TABLE fixed_width_table (
    id INT,
    name VARCHAR(100),
    age INT
);
```

```sql
CREATE TABLE csv_table (
    id INT,
    name VARCHAR(100),
    age INT
);
CREATE TABLE char_delimited_table (
    id INT,
    name VARCHAR(100),
    age INT
);
CREATE TABLE fixed_width_table (
    id INT,
    name VARCHAR(100),
    age INT
);
```

Result:



5. Transfer Data with COPY Commands

Write COPY Commands:
For **CSV**:
code

```sql
COPY csv_table
FROM 's3://deven-redshift-task-bucket/sample_data.csv'
IAM_ROLE
'arn:aws:iam::your-account-id:role/deven_redshift_s3_access'
CSV;
```

For **Character-Delimited** (e.g., using `|`):
code
```
COPY char_delimited_table
FROM 's3://deven-redshift-task-bucket/sample_data.txt'
IAM_ROLE
'arn:aws:iam::your-account-id:role/deven_redshift_s3_access'
DELIMITER '|';
```

For **Fixed Width** (Redshift doesn't support fixed-width directly; you might need to preprocess the data into a delimited format before loading):

code
```
COPY fixed_width_table
FROM 's3://deven-redshift-task-bucket/sample_data_fixed.txt'
IAM_ROLE
'arn:aws:iam::your-account-id:role/deven_redshift_s3_access'
FIXEDWIDTH '1,16,2';
```

Note- Above is old method for fixed-width but still it is running……..

Query 1

```
COPY csv_table
FROM 's3://deven-redshift-task-bucket/sample_data.csv'
IAM_ROLE 'arn:aws:iam::980921736064:role/deven_redshift_s3_access'
CSV
IGNOREHEADER 1;
COPY char_delimited_table
FROM 's3://deven-redshift-task-bucket/sample_data.txt'
IAM_ROLE 'arn:aws:iam::980921736064:role/deven_redshift_s3_access'
DELIMITER '|'
IGNOREHEADER 1;

COPY fixed_width_table
FROM 's3://deven-redshift-task-bucket/sample_data_fixed.txt'
IAM_ROLE 'arn:aws:iam::980921736064:role/deven_redshift_s3_access'
DELIMITER '|'
IGNOREHEADER 1;  -- Assuming preprocessing to delimit
```

6. Analyze the Database

1. **Run Queries**:
   - Use SQL queries to analyze the data. For example:

code

```
-- Sample query to check data
SELECT * FROM csv_table;
```

id	name	age
1	John Doe	28
2	Jane Smith	34
3	Bob Johnson	45
4	Emily Davis	29
5	Michael Brown	38

```
-- Aggregate data
SELECT AVG(age) AS average_age FROM csv_table;
```

Rows returned (1)

Export

Search rows

average_age
34

2. **Use Redshift Performance Tools**:

Review the performance using system tables and views:
code

```
SELECT * FROM stv_blocklist;  -- Check block-level data distribution
```

Rows returned (1035)

Export

Search rows

1 2 3 4 5 6 7 ... 104

slice	col	tbl	blocknum	num_values
0	1	-2	0	10
0	1	-2	1	8
0	1	-2	4	18

```
SELECT * FROM stv_recents;    -- Recent queries and their performance
```

Rows returned (100)

Export

Search rows

1 2 3 4 5 6 7 ... 10

userid	status	starttime	duration	user_name
1	Done	2024-09-16 12:34:42.572665	357	rdsdb
1	Done	2024-09-16 12:34:42.573168	11	rdsdb
1	Done	2024-09-16 12:34:44.096828	505	rdsdb
1	Done	2024-09-16 12:34:44.097541	17	rdsdb
1	Done	2024-09-16 12:34:44.098138	311	rdsdb
1	Done	2024-09-16 12:34:44.098498	31	rdsdb
1	Done	2024-09-16	349	rdsdb

3. **Visualize Data**:
   - Optionally, you can use Amazon QuickSight or other BI tools to create dashboards and visualizations from your Redshift data.