+ -
Helvetica N...
▼

Step 8 of 8

Hands-on Lab : Views in PostgreSQL

Estimated time needed: 15 minutes

In this lab, you will learn how to create and execute views and materialized views in the PostgreSQL database service using the pgAdmin graphical user interface (GUI) tool. Materialized views behave differently compared to regular views. In materialized views, the result set is materialized, or saved for future use. You can't insert, update, or delete rows like in regular views. Essentially, materialized views store the results of a database query as a separate table-like object so that the query results can be accessed at a later time without having to re-run the query. As a result, materialized views can improve database performance compared to regular views.

Software Used in this Lab

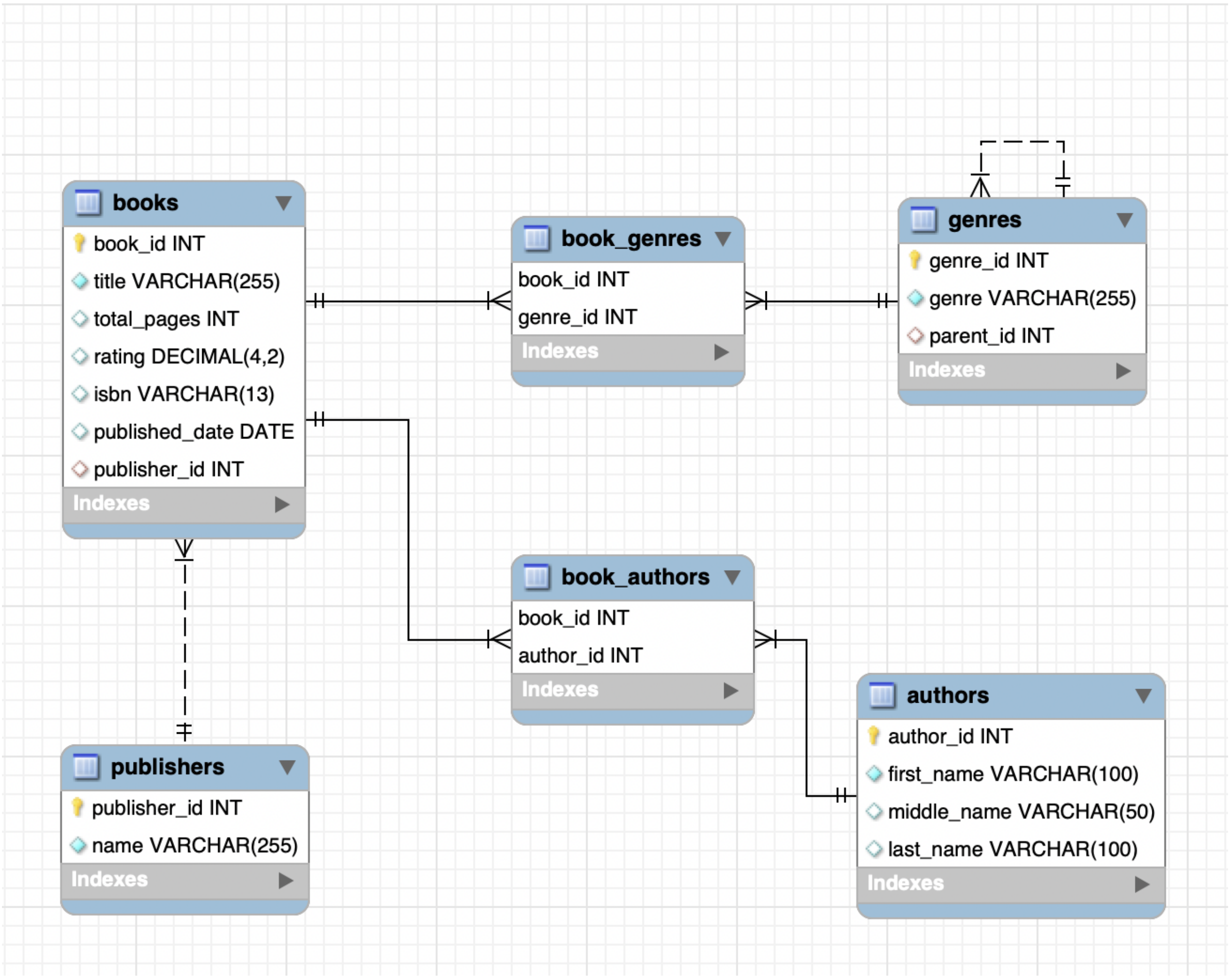
In this lab, you will use the PostgreSQL Database. PostgreSQL is a Relational Database Management System (RDBMS) designed to efficiently store, manipulate, and retrieve the data.

To complete this lab you will utilize the PostgreSQL relational database service available as part of IBM Skills Network Labs (SN Labs) Cloud IDE. SN Labs is a virtual lab environment used in this course.

Database Used in this Lab

The eBooks database has been used in this lab.

The following ERD diagram shows the schema of the complete eBooks database used in this lab:



Objectives

After completing this lab, you will be able to use pgAdmin with PostgreSQL to:

- Restore a database schema and data.
- Create and execute a view.
- Create and execute a materialized view.

Lab Structure

In this exercise, you will go through three tasks where you will learn how to create and execute views and materialized views in the PostgreSQL database service using the pgAdmin graphical user interface (GUI) tool.

Task A: Restore a database schema and data

To get started with this lab, you will first download the relevant **eBooks** database dump file, then launch PostgreSQL and pgAdmin using the Cloud IDE. You can do this by following these steps:

1. Download the **eBooks** PostgreSQL dump file (containing the eBooks database schema and data) below to your local computer storage.
    - eBooks*pgsql*dump.tar
2. Click on the Skills Network extension button on the left side of the window.

3. Open the "DATABASES" drop down menu and click on "PostgreSQL"
4. Click on the "Start" button. PostgreSQL may take a few moments to start.
5. Next, open the pgAdmin Graphical User Interface by clicking the "pgAdmin" button in the Cloud IDE interface.
6. Once the pgAdmin GUI opens, click on the `Servers` tab on the left side of the page. You will be prompted to enter a password.
7. To retrieve your password, click on the "PostgreSQL" tab near the top of the interface.
8. Click on the Copy icon to the left of your password to copy the session password onto your clipboard.
9. Navigate back to the "pgAdmin" tab and paste in your password, then click `OK`
10. You will then be able to access the pgAdmin GUI tool.
11. In the tree-view, expand **Servers** > **postgres** > **Databases**. Enter your PostgreSQL service session password if prompted during the process. Right-click on **Databases** and go to **Create > Database**. Type **eBooks** as name of the database and click **Save**.
12. In the tree-view, expand **eBooks**. Right-click on **eBooks** and select **Restore**.
13. Follow the instructions below to restore and proceed to Task B:
    - On the **General** tab, click on the **Select file** button by the Filename box.
    - Click the **Upload File** button.
    - Double-click on the drop files area and load the **eBooks*pgsql*dump.tar** you downloaded earlier from your local computer storage.
    - When the upload is complete, close the drop files area by clicking the **X** button.
    - Make sure Format is set to **All Files**, select the uploaded **eBooks*pgsql*dump.tar** file from the list, and then click the **Select** button.
    - Now switch to **Restore options** tab.
    - Under Disable, set the Trigger option to **Yes**. Then click **Restore** button.

Task B: Create and execute a view

1. In the tree-view, expand **eBooks > Schemas > public**. Right-click on **Views** and go to **Create > View**.

2. On the **General** tab, type **publisher*and*rating_view** as name of the view. Then switch to **Code** tab.

3. On the **Code** tab, copy and paste the code below. Then click **Save**.

```
SELECT books.title, books.rating, publishers.name
FROM books INNER JOIN publishers ON books.publisher_id = publishers.publisher_id
```

4. In the tree-view, expand **Views**. Right-click on **publisher*and*rating_view** and go to **View/Edit Data > All Rows**.

5. You will access the view you created. This allows you to actually access and view the contents of tables in your database.

Task C: Create and execute a materialized view

1. In the tree-view, expand **eBooks > Schemas > public**. Right-click on **Materialized Views** and go to **Create > Materialized View**.

2. On the **General** tab, type **publisher*and*rating*materialized*view** as name of the view. Then switch to the **Definition** tab.

3. On the **Definition** tab, copy and paste the code below. Then click **Save**.

```
SELECT books.title, books.rating, publishers.name
FROM books INNER JOIN publishers ON books.publisher_id = publishers.publisher_id
```

4. In the tree-view, expand **Materialized Views**. Right-click on **publisher*and*rating*materialized*view** and go to **Refresh View > With data**.

5. Right-click on **publisher*and*rating*materialized*view** again and go to **View/Edit Data > All Rows**.

6. You will access the materialized view you created.

   As you can see, at first glance it doesn't look too different from the regular view you created earlier in this lab - indeed, from the user perspective it's essentially the same: you see the results of a query displayed in a table-like format. The difference is that this materialized view is cached in the database so that the data can be accessed again at a future time without having to re-run the database query, which can be intensive on the server depending on the complexity of the query and the size of the table being queried.

Congratulations! You have completed this lab, and you are ready for the next topic.

Author

- Sandip Saha Joy

Other Contributors

- David Pasternak

Changelog

Date	Version	Changed by	Change Description
2021-03-25	1.0	Sandip Saha Joy	Created initial version
2021-10-18	1.1	David Pasternak	Updated instructions

Previous