PostgreSQL

How to Use PostgreSQL Command line to Manage Databases?

A PostgreSQL **Command line**, in general, is a software program. We execute a command line on a command line interface (CLI) to interact directly with data on a database. Although a **Graphical User Interface (GUI)** provides a much better user experience, PostgreSQL CLI acts vital when managing applications or operating systems with greater control, reliability, and conventionality.

PostgreSQL is an RDBMS (Relational Database Management System) that allows effortless and secured data collection. Today, PostgreSQL is used as the primary data storage repository (as a data warehouse) for many websites, mobile and analytics applications. And, you would have noticed (when you were installing the PostgreSQL database server) some tools that offer data professionals with a high understanding of SQL a seamless way to manage databases in PostgreSQL using command lines.

The tools are known as an interactive terminal program — psql and pgAdmin.

1. **psql: A terminal-based front-end to PostgreSQL database server.**
2. **pgAdmin: A web-based front-end to PostgreSQL database server.**

We can use psql (terminal or command line) and pgAdmin connection tool to connect our PostgreSQL database and manage it, too.

This tutorial article will explain how to use the interactive terminal programs to connect and manage your PostgreSQL database in the Command Line Interface.

How to Connect to a Database in PostgreSQL Command line (CLI)

Now, to get started with psql Connect to database command line you, first have to install PostgreSQL, After installing PostgreSQL, launch psql and enter the following command in the Postgresql CLI.

```
psql -h <hostname> -p <port> -U <username> -d <database>
```

Replace `<h`

Replace `<hostname>`, `<port>`, `<username>`, and `<database>` with your specific values.

- `h`: Specifies the host where the PostgreSQL server is running. Use `localhost` if it's on the same machine.
- `p`: Specifies the port number. The default is `5432`.
- `U`: Specifies the username you want to connect with.
- `d`: Specifies the name of the database to connect to.

Example

```
psql -h localhost -p 5432 -U myusername -d mydatabase
```

You'll be prompted to enter a password if the PostgreSQL server requires it.

How to Create a Database

To create a database using the PostgreSQL CLI, use the following command:

```
CREATE DATABASE database_name;
```

Example

```
CREATE DATABASE books_db;
```

List All Databases

To get a list of all the databases using the PostgreSQL CLI, enter the `\l` prompt.

```
postgres=# \l
```

If you wish to get information for a particular database, enter the `\l` prompt in the PostgreSQL CLI, followed by the database name.

```
postgres=# \l books_db

postgres=# \l
                                    List of databases
   Name    |  Owner   | Encoding |          Collate    |
Ctype       |
-----------+----------+----------+---------------------+-----
-----------------+
 books_db | postgres | UTF8     | English_Ireland.1252 | Engl
ish_Ireland.1252 |
```

Additonally, using `\l` + prompt will display additional information such as the size and the tablespace in the database.

```
\l+
```

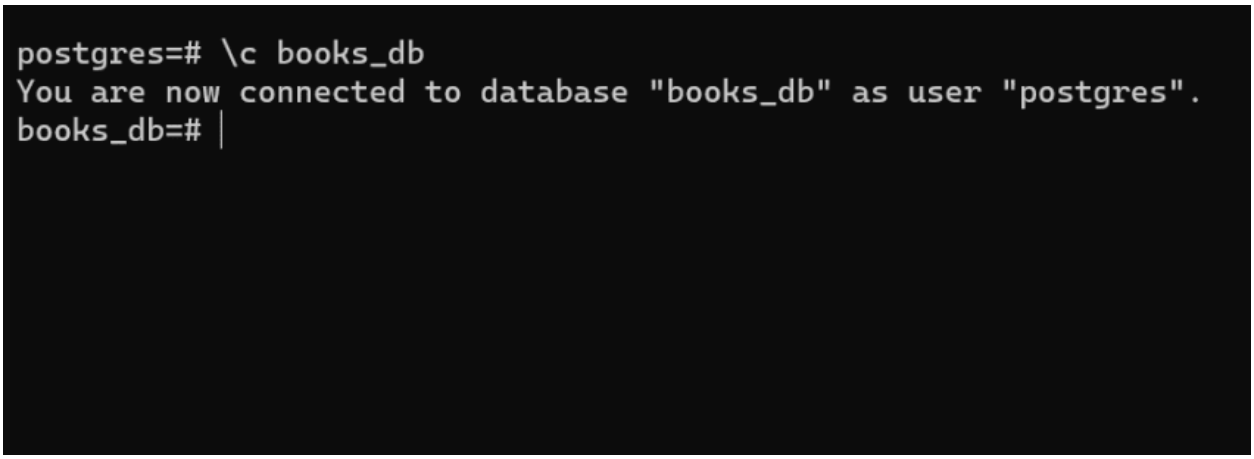
Command to Switch Databases

You remain connected to the default postgres database, to switch the database, enter the following command in the PostgreSQL CLI:

```
\c database_name
```

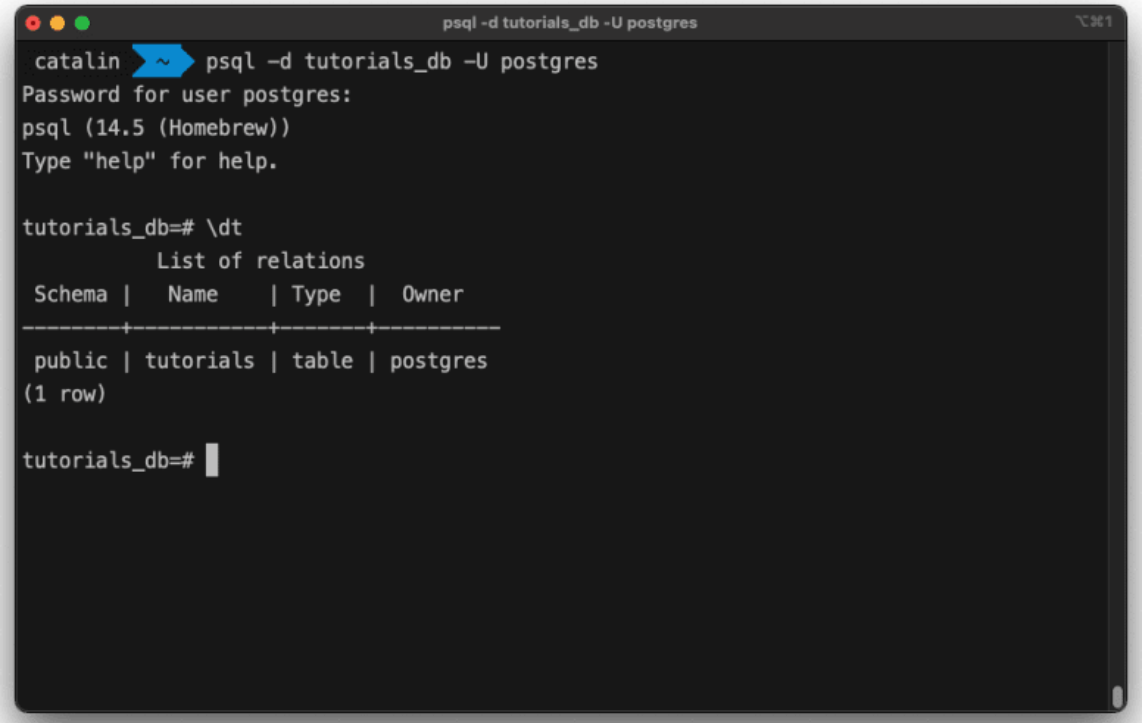
Example

```
\c books_db
```



How to List Database Tables

If you want to list all the tables of a database, use the \dt command. For example, in the database tutorials_db, you can see the following tables:

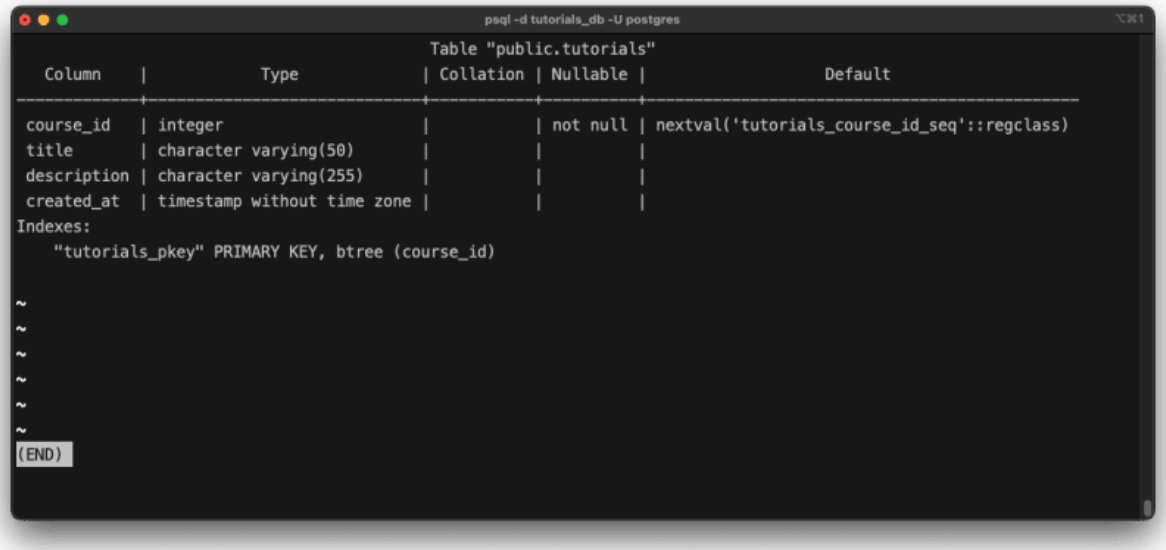


How to describe a Table?

psql also has a command that helps you to describe the table.

```
\d <table-name>

// example
\d tutorials
```



How to Rename a Table?

You can change the name of a table by using the following PostgreSQL Command Line:

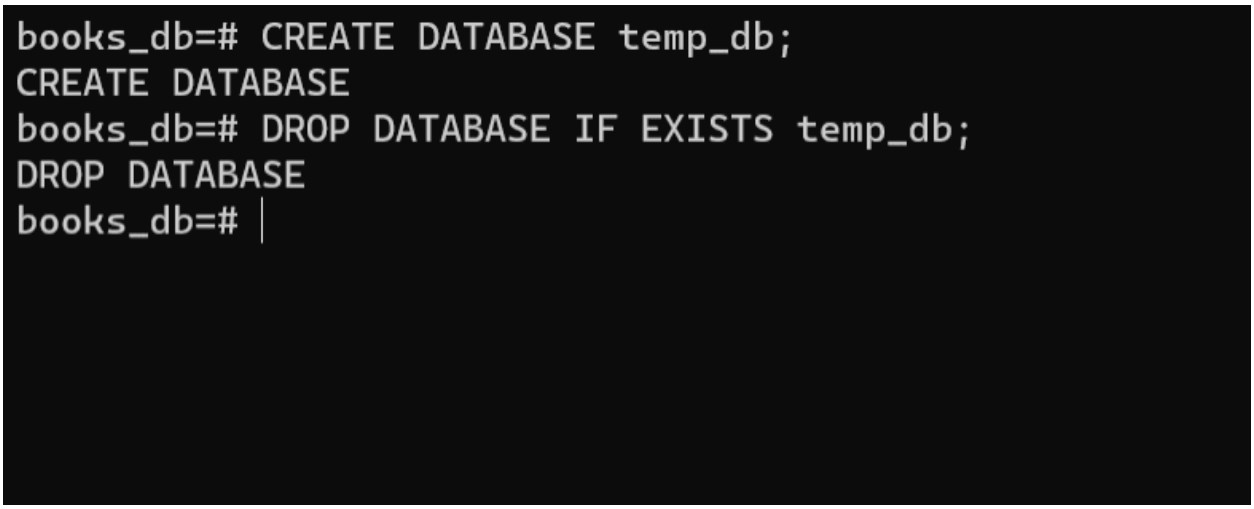
```
ALTER TABLE table_name RENAME TO new_table_name;
```

How to Delete a Database

To delete a database, use the DROP DATABASE command.

```
DROP DATABASE database_name;
```

You will get an error if you try to delete a database that does not exist; use IF EXISTS to get a notice.



How to Quit psql?

You can quit the PostgreSQL CLI using the \q command.

Connect Using pgAdmin in PostgreSQL Command line

Used as a graphical tool in PostgreSQL to manage databases, pgAdmin is a web-based front-end to PostgreSQL database server. To access the database using pgAdmin, first, you will need to install and configure the latest version of your browser and create a new pgAdmin user account. An email and a password are required to authenticate access.

```
python /usr/lib/python2.7/site-packages/pgadmin4-web/setup.py
```

After the authentication process is complete, you can access the phAdmin 4 interface using the command line given below:

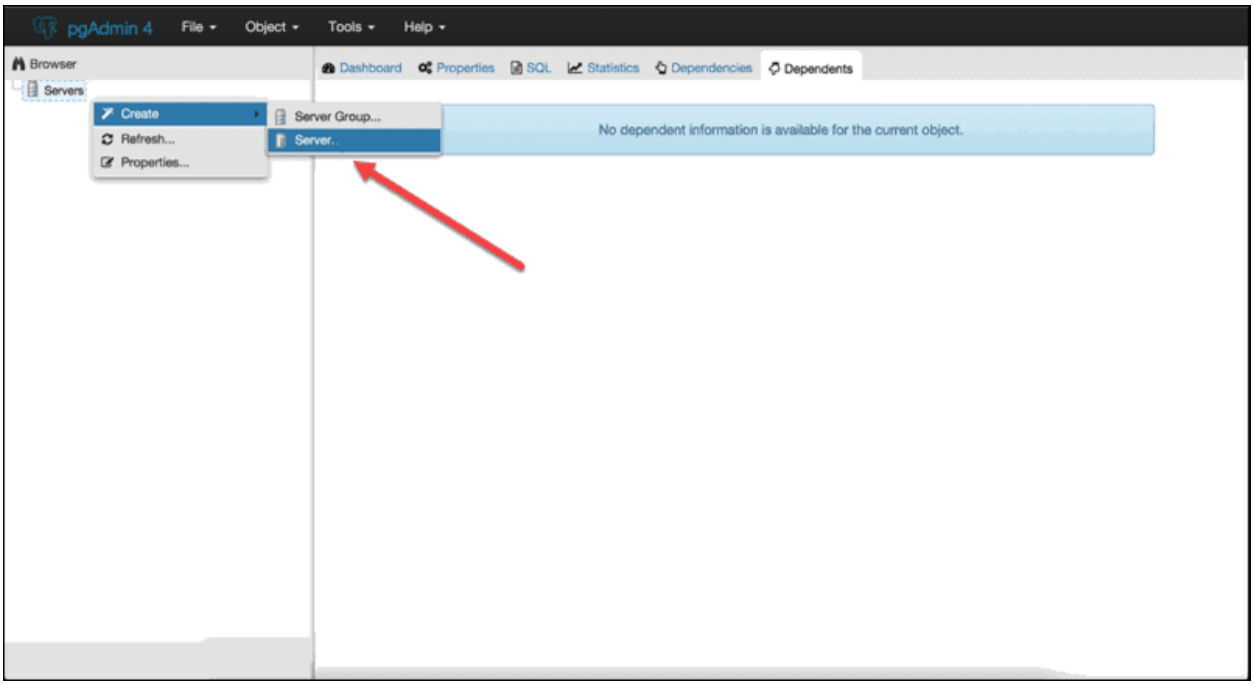
```
http://localhost/pgadmin4
```

or

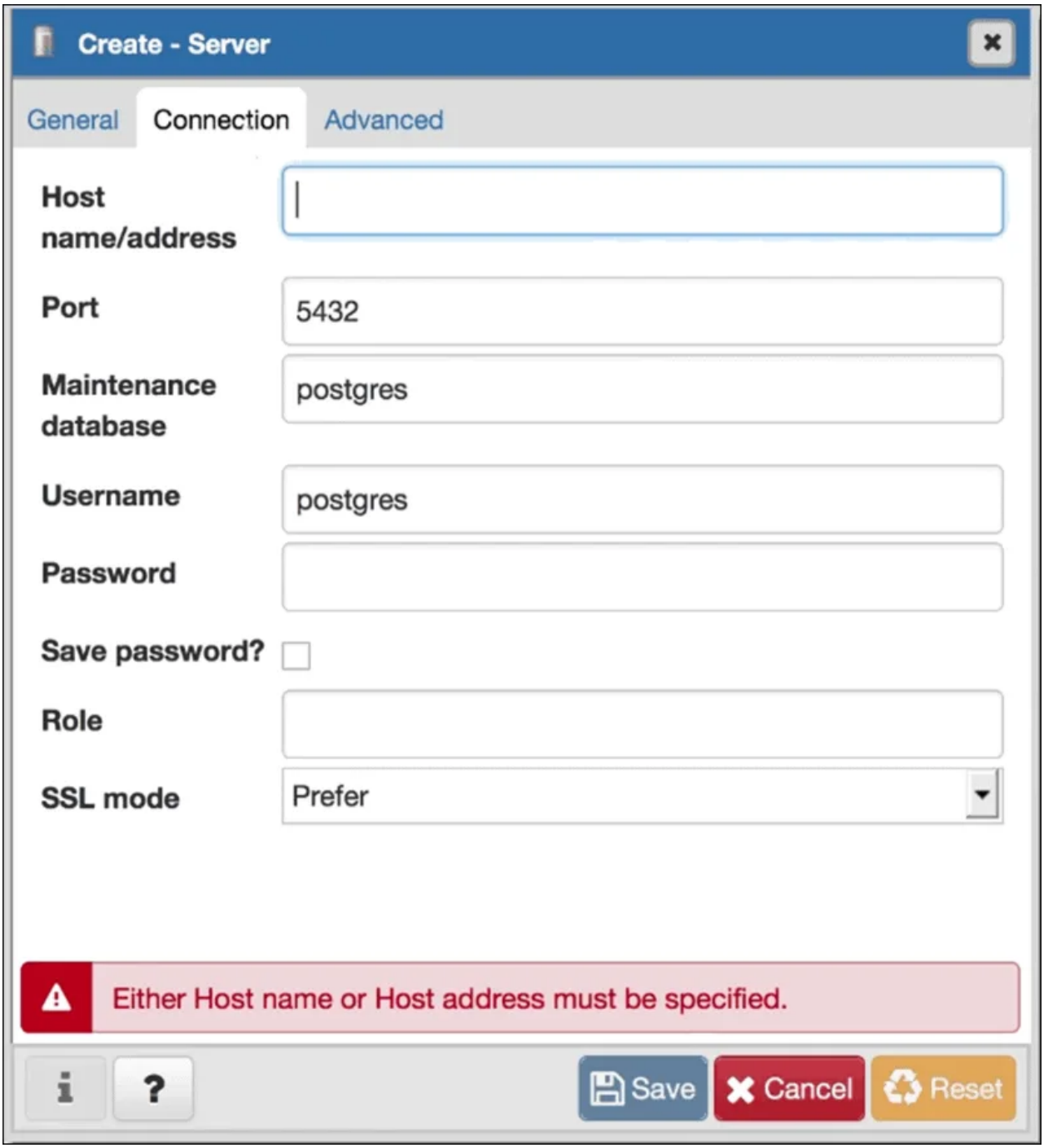
```
http://ip-adress/pgadmin4
```

Now, use the email and password created previously to authenticate the user. Now, the user interface will load and now you would need to navigate as shown in the image

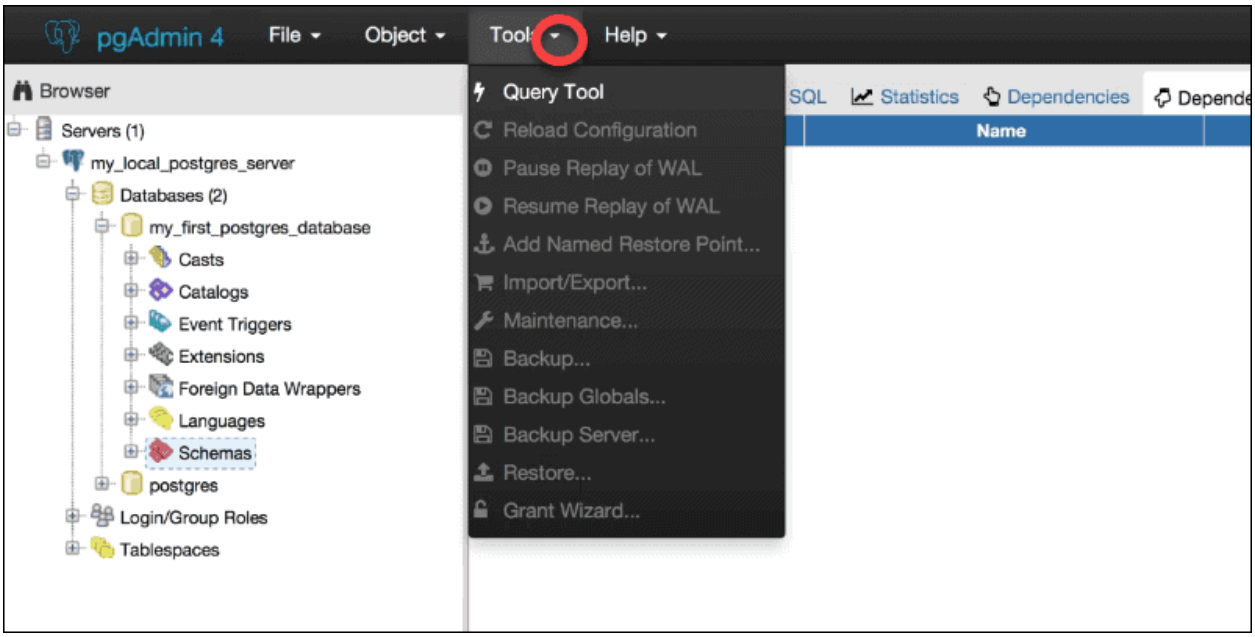
below. (Servers>Create>Server)



Under the “Create – Server” section, the General and Connection tabs will now be used to enter values such as server name and user credentials for the database.



Now that the connection to your database has been established, the following interface will provide an overview of the user's databases. You can execute queries by clicking on Tools > Query Tool or by pressing ALT + Shift + Q.



Conclusion

Let's conclude. Through this PostgreSQL command line tutorial article, we provided two ways to connect to the database command line. First, by using the psql method, and second, by using the pgAdmin method. And if you need to learn more on the subject, either of these two articles can help.

1. Documentation for pgAdmin and psql connect to database command line in PostgreSQL 8.4
2. Documentation for pgAdmin and psql connect to database command line in PostgreSQL 9.1

Here are some essential reads for a deeper dive into PostgreSQL functionalities:

1. PostgreSQL date_part: Syntax & 7 Use Cases Simplified
2. PostgreSQL SELECT Statement: Syntax & Example Queries | A 101 Guide
3. PostgreSQL TRIM() Function: Syntax & Practical Examples | A 101 Guide

On that note, extracting and managing complex data, from a diverse set of data sources can be a challenging task and this is where Hevo can help!

Hevo Data offers a faster way to move data from Databases or SaaS applications into your Data Warehouse to be visualized in a BI tool. Hevo is fully automated and hence does not require you to code.