DBMS Lab

PostgresSQL

Introduction

PostgreSQL is a powerful, open source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads. The origins of PostgreSQL date back to 1986 as part of the POSTGRES project at the University of California at Berkeley and has more than 30 years of active development on the core platform.

Database A database is a collection of data stored in some organized fashion.

DBMS

Database Management System (DBMS) is a software for storing and retrieving users' data while considering appropriate security measures. It consists of a group of programs which manipulate the database. DBMS does all the work of storing, retrieving, managing, and manipulating data.PostgreSQL is also a DBMS.

Table A structured list of data of a specific type.

Column A single field in a table. All tables are made up of one or more columns.

Datatype A type of allowed data. Every table column has an associated data type that restricts (or allows) specific data in that column. Datatypes restrict the type of data that can be stored in a column.

Row A record in a table.

Primary Key A column (or set of columns) whose values uniquely identify every row in a table.

SQL (Structured Query Language) SQL is a language designed specifically for communicating with databases.

Working with PostgreSQL

Issuing PostgreSQL statements, and obtaining information about databases and Tables.

Selecting a Database
Syntax : \c database_name;

Learning About Databases and Tables
Information about databases, tables, columns.

\l; Returns a list of available databases

\dt; Returns a list of available tables in the currently selected database.

\d table_name; It returns a row for each field containing the field name, its datatype, whether NULL is allowed, key information, default value, and extra information (such as auto_increment for field cust_id).

Creating Data

A relational database consists of multiple related tables. A table consists of rows and columns. Tables allow you to store structured data.

Creating Database
Syntax: CREATE DATABASE db_name;

Creating Table
Syntax: CREATE TABLE table_name (column1 datatype column_constraint,
column2 datatype column_constraint,table_constraints);

Constraints

PostgreSQL includes the following column constraints:

1. **NOT NULL** - ensures that values in a column cannot be **NULL**
2. **UNIQUE** – ensures the values in a column unique across the rows within the same table.
3. **PRIMARY KEY** – a primary key column uniquely identify rows in a table. A table can have one and only one primary key. The primary key constraint allows you to define the primary key of a table.
4. **CHECK** – a **CHECK** constraint ensures the data must satisfy a boolean expression.
5. **FOREIGN KEY** – ensures values in a column or a group of columns from a table exists in a column or group of columns in another table. Unlike the primary key, a table can have many foreign keys.

Table constraints are similar to column constraints except that they are applied to more than one column.

Data Types

1. **Boolean**
2. **Character** types such as char, varchar, and text.
3. **Numeric** types such as integer,floating-point,serial and arbitrary precision numbers.
4. **Date/Time** types such as date, time, timestamp, and interval
5. **UUID** for storing Universally Unique Identifiers
6. **Array** for storing array strings, numbers, etc.
7. **JSON**

Insert Data

Inserting Single Row
Syntax: INSERT INTO table_name(column1, column2, . . .) VALUES (value1, value2, . . .);

Inserting Multiple Row
Syntax: INSERT INTO table_name(column1, column2, . . .) VALUES (value1, value2, . . .),(value1, value2, . . .),(value1, value2, . . .);

Retrieving Data

Using the SELECT statement to retrieve one or more columns of data from a table.

The SELECT Statement :
Retrieving Individual Columns:

Syntax: SELECT column_name FROM table_name;

Note : It is important to note that SQL statements are not case sensitive, so SELECT is the same as select, which is the same as Select.

Retrieving Multiple Columns:
To retrieve multiple columns from a table, the same SELECT statement is used. The only difference is that multiple column names must be specified after the SELECT keyword, and each column must be separated by a comma.

Syntax: SELECT column1,column2 FROM table_name;

Retrieving All Columns :

*Syntax: SELECT * FROM table_name;*

When a wildcard (*) is specified, all the columns in the table are returned. The columns are in the order in which the columns appear in the table definition.

Retrieving Distinct Rows :
DISTINCT keyword:
SELECT DISTINCT column1 FROM table_name;

It return only distinct (unique) column1 rows,

Note : If u apply distinct on multiple columns all rows would be retrieved unless all of the specified columns were distinct.

Limiting Results :
SELECT statements return all matched rows, possibly every row in the specified table. To return just the first row or rows, use the LIMIT clause.
*Syntax: SELECT * FROM table_name LIMIT 5;*

LIMIT 5 instructs PostgreSQL to return no more than five rows.
LIMIT 3,4 means 3 rows starting from row 4.

Using Fully Qualified Table Names:
It is also possible to refer to columns using fully qualified names (using both the table and column names)
Syntax: SELECT table_name.column1 FROM table_name;

Filtering Data

Retrieving just the data you want involves specifying search criteria, also known as a filter Condition.

Within a SELECT statement, data is filtered by specifying search criteria in the WHERE clause. The WHERE clause is specified right after the table name.

Syntax: SELECT column1,column2 FROM table_name where column1 = 2.50;

This statement retrieves two columns from the table, but instead of returning all rows, only rows with a column1 value of 2.50 are returned.

The WHERE Clause Operators :

Operator	Description
=	Equality
<>	Nonequality
!=	Nonequality
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
BETWEEN	Between two specified values

Checking for a Range of Values

To check for a range of values, you can use the BETWEEN operator.

Syntax: SELECT column1,column2 FROM table_name where column1 BETWEEN 5 and 10;

Combining WHERE Clauses Using the AND Operator :
To filter by more than one column, you use the AND operator to append conditions to your WHERE clause.

Syntax: SELECT column1,column2 FROM table_name where column1 = 1000 AND column2 <= 10;

Using the IN Operator
The IN operator is used to specify a range of conditions, any of which can be matched. IN takes a comma-delimited list of valid values, all enclosed within parentheses.

Syntax: SELECT column1,column2 FROM table_name where column1 in (1002,1003);

Updating Data

Using the UPDATE statement to modify the data in the table.

Updating all columns

Syntax: UPDATE table_name SET column1 = value1,column2 = value2;

Updating columns based on condition

Syntax: UPDATE table_name SET column1 = value1,column2 = value2 where condition;

Updating Table Structure

Using the ALTER statement to modify the structure of an existing table.

PostgreSQL provides many actions to update structure:

1. Add a column
2. Drop a column
3. Change the data type of a column
4. Rename a column
5. Set a default value for the column.
6. Add a constraint to a column.
7. Rename a table

Add a new column to a table

Syntax: ALTER TABLE table_name ADD COLUMN column_name datatype column_constraint;

Drop a column from a table

Syntax: ALTER TABLE table_name DROP COLUMN column_name;

Rename a column

Syntax: ALTER TABLE table_name RENAME COLUMN column_name TO new_column_name;

Deleting Data

Delete both record and structure of the table

Syntax: DROP TABLE table_name;

Delete one or more rows

Syntax: DELETE FROM table_name WHERE condition;

Remove all data from table

Syntax: TRUNCATE TABLE table_name;

Delete database

Syntax: DROP DATABASE db_name;

Import/Export

Import to PostgreSQL Database:

To import an existing dump file into PostgreSQL, you will have to create a new database. This database will hold the imported data.

psql -h hostname -d database_name -U user_name -f data-dump.sql

Export PostgreSQL Database :

pg_dump -U username -h hostname -d database_name >> data-dump.sql

Reading Resources

1. `https://www.postgresqltutorial.com/`
2. `https://www.postgresql.org/docs/9.5`
3. `https://www.javatpoint.com/`