

Data Management Systems

PostgreSQL

Matteo Devigili

May 24th, 2021

Constraints

- ▶ *Not Null*: specifies that a column must not assume the null value
- ▶ *Unique*: ensures that the data contained in a column, or a group of columns, is unique among all the rows in the table
- ▶ *Primary Key*: indicates that a column, or group of columns, can be used as a unique identifier for rows in the table
- ▶ *Check*: allows you to specify that the value in a certain column must satisfy a Boolean (truth-value) expression

Constraints - PostgreSQL

Import Data

psql

- ▶ `\i`: reads input from the file and executes it as though it had been typed on the keyboard.
- ▶ `\copy`: psql reads the file and routes the data between the server and the local file system

PostgreSQL

SQL

- ▶ *COPY FROM*: instructs the PostgreSQL server to directly read from a file

PostgreSQL

SQL - Basics

Part One

SELECT <attributes> FROM <one or more relations>;

1. ***: to select all columns
2. *LIMIT < n >*: *n* rows will be returned
3. *OFFSET < n >*: to skip *n* rows before beginning to return rows
4. *WHERE <condition>*: to filter our data on a condition (such as gender = 'Female')
5. *OR, AND, NOT*: logical operators
6. *IS NULL*: to test for null (i.e. missing) values

SQL - Basics

Part Two

1. *DISTINCT*: eliminates duplicate rows from the result
2. *ORDER BY*: rows are sorted in a specified order (ASC or DESC)
3. *LIKE*: a case-sensitive test for pattern matching (where % stands for any sequence of characters and _ for any single character)
4. *iLIKE*: similar to LIKE but case-insensitive
5. *BETWEEN*: logical operator
6. *EXTRACT*: retrieves subfields such as year or hour from date/time values.
7. *AS*: to assign a temporary name

SQL - Aggregate Functions

ALL | DISTINCT:

- ▶ *COUNT*: returns the count of all or distinct values passed
- ▶ *SUM*: returns the sum of all or distinct values passed
- ▶ *AVG*: returns the average of all or distinct values passed

ALL:

- ▶ *MIN*: returns the minimum value among all values passed
- ▶ *MAX*: returns the maximum value among all values passed

We have seen also

- ▶ *ROUND*: to specify the length or precision of a numeric expression

SQL - GROUP BY

- ▶ *GROUP BY*: to group observations
- ▶ *HAVING*: to filter data on a condition

NB: WHERE applies the condition to rows *before* the GROUP BY, while HAVING *after*.

References

- ▶ Obe, Regina O., and Leo S. Hsu. PostgreSQL: Up and Running: a Practical Guide to the Advanced Open Source Database. "O'Reilly Media, Inc.", 2017.
- ▶ PostgreSQL 12.2 Documentation
<https://www.postgresql.org/docs/12/index.html>