# Introduction to SQL

## Part-3

**Hitesh Kumar Sharma**
Instructor, Pluralsight

# Table Commands in PostgreSQL

# DDL Commands

In PostgreSQL, Data Definition Language (DDL) commands are used to define, manage, and modify the structure of database objects. They allow you to create, modify, and delete database objects such as tables, views, indexes, and sequences. Here are some common DDL commands in PostgreSQL:

- **CREATE**
- **ALTER**
- **DROP**

These DDL commands are essential for managing the structure of the database in PostgreSQL. By using these commands, you can create and manipulate tables, indexes, views, sequences, and schemas, allowing for effective database administration and maintenance.

# Create Table

To create a table in PostgreSQL, you can use the CREATE TABLE SQL command. Here's a basic example:

**Syntax:**

```
CREATE TABLE table_name (
    column1 datatype1,
    column2 datatype2,
    column3 datatype3,
    ...
);
```

# Create Table (Example)

To create a table in PostgreSQL, you can use the CREATE TABLE SQL command. Here's a basic example:

**Syntax:**

```
CREATE TABLE employees (
    employee_id serial PRIMARY KEY,
    first_name VARCHAR (50) NOT NULL,
    last_name VARCHAR (50) NOT NULL,
    email VARCHAR (255) UNIQUE NOT NULL,
    date_of_birth DATE
);
```

```
postgres=# CREATE TABLE employees (
postgres(#     employee_id serial PRIMARY KEY,
postgres(#     first_name VARCHAR (50) NOT NULL,
postgres(#     last_name VARCHAR (50) NOT NULL,
postgres(#     email VARCHAR (255) UNIQUE NOT NULL,
postgres(#     date_of_birth DATE
postgres(# );
CREATE TABLE
postgres=# \dt
          List of relations
 Schema |   Name    | Type  |  Owner
--------+-----------+-------+----------
 public | employees | table | postgres
(1 row)
```

# Hands-On Lab Exercise-3

## (Topic: DDL Commands)

# DML Commands

In PostgreSQL, Data Manipulation Language (DML) commands are used to retrieve, insert, update, and delete data in the database. They allow you to manage the data stored in the tables. Here are some common DML commands in PostgreSQL:

- **INSERT**
- **UPDATE**
- **DELETE**
- **TRUNCATE**
- **MERGE**

DML commands are crucial for managing and manipulating data within the database. They allow you to insert, retrieve, update, and delete data, ensuring effective data management and maintenance.

# Hands-On Lab Exercise-4

## (Topic: DML Commands)

# Integrity Constraints in PostgreSQL

# Integrity Constraints

In PostgreSQL, integrity constraints ensure the accuracy and reliability of the data stored in a database. They help enforce data integrity rules, maintaining the consistency and quality of the data. There are several types of integrity constraints commonly used in PostgreSQL:

1. **Primary Key**
2. **Unique Key**
3. **Not Null**
4. **Foreign Key**
5. **Check**
6. **Default**
7. **Exclusion**

# Integrity Constraints

1. **Primary Key Constraint:** Ensures that each record in a table is uniquely identifiable. It is a combination of a unique and a not-null constraint.
2. **Foreign Key Constraint:** Establishes a link between data in two tables, ensuring referential integrity. It typically references the primary key of another table.
3. **Unique Constraint:** Ensures that the values in a column or a group of columns are unique among all the rows in the table.

# Integrity Constraints

4. **Check Constraint:** Allows you to define a condition that each row must satisfy. It is typically used to limit the values that can be inserted into a column.
5. **Not-Null Constraint:** Ensures that a column does not accept null values. It guarantees that every row will have a value in the specified column.
6. **Exclusion Constraint:** Allows you to specify that if any two rows are compared on the specified columns using the specified operators, at least one of these operator comparisons must return false or null.

# Primary Key Constraint

In PostgreSQL, the primary key constraint is used to uniquely identify each record (or row) in a table. It ensures that the designated column or combination of columns has unique values, and it does not contain null values. Here's how you can create a primary key constraint in PostgreSQL:

```
CREATE TABLE table_name (
    id SERIAL PRIMARY KEY,
    column1 datatype1,
    column2 datatype2,
    -- other columns
);
```

Adding a primary key constraint to an existing table:

```
ALTER TABLE table_name
ADD CONSTRAINT table_name_pkey PRIMARY KEY (column_name);
```

# Hands-On Lab Exercise-5

## (Topic: Primary Key Constraint)

# Foreign Key Constraint

In PostgreSQL, a foreign key constraint is used to enforce referential integrity between tables. It ensures that values in a column (or a group of columns) in one table exist as values in another table's primary key. Here's how you can create a foreign key constraint in PostgreSQL:

```
CREATE TABLE table1 (
    id SERIAL PRIMARY KEY,
    column1 datatype1,
    -- other columns
);

CREATE TABLE table2 (
    id SERIAL PRIMARY KEY,
    table1_id INT REFERENCES table1(id),
    column2 datatype2,
    -- other columns
);
```

Adding a constraint to an existing table:

```
ALTER TABLE table_name
ADD CONSTRAINT constraint_name
FOREIGN KEY (column_name)
REFERENCES parent_table (parent_column);
```

# Hands-On Lab Exercise-6

## (Topic: Foreign Key Constraint)

# Check Constraint

In PostgreSQL, the CHECK constraint is used to ensure that the values in a column or a group of columns meet specific conditions. It allows you to define custom rules or conditions that must be satisfied for the data in a column. Here's how you can create a CHECK constraint in PostgreSQL:

```
CREATE TABLE table_name (
    column1 datatype1,
    column2 datatype2,
    CHECK (condition)
);
```

Adding constraint to an existing table:

```
ALTER TABLE table_name
ADD CONSTRAINT constraint_name
CHECK (condition);
```

# Hands-On Lab Exercise-7

## (Topic: Check Constraint)

# Default Constraint

In PostgreSQL, a DEFAULT constraint is used to assign a default value to a column when no explicit value is specified during an INSERT operation. This ensures that a default value is automatically assigned to the column if no other value is provided. Here's how you can create a DEFAULT constraint in PostgreSQL:

```
CREATE TABLE table_name (
    column1 datatype1 DEFAULT default_value,
    column2 datatype2 DEFAULT default_value,
    -- other columns
);
```

Adding a constraint to an existing table:

```
ALTER TABLE table_name
ALTER COLUMN column_name SET DEFAULT default_value;
```

# Hands-On Lab Exercise-8

## (Topic: Default Constraint)

# Exclusion Constraint

In PostgreSQL, an exclusion constraint is used to ensure that if any two rows are compared on a specified set of columns using a specified operator, at least one of the specified conditions must hold true. This is useful for preventing conflicts or overlaps between rows based on certain criteria. Here's how you can create an exclusion constraint in PostgreSQL:

```
CREATE TABLE table_name (
    column1 datatype1,
    column2 datatype2,
    EXCLUDE USING gist (column1 WITH =, column2 WITH <>)
);
```

Adding a constraint to an existing table:

```
ALTER TABLE table_name
ADD CONSTRAINT constraint_name
EXCLUDE USING gist (column1 WITH =, column2 WITH <>);
```

# Hands-On Lab Exercise-9

## (Topic: Exclusion Constraint)