Lecture 3. Database Creation & DDL

Creating a Database

Command: `CREATE DATABASE` — creates a new database.

- You must be a **superuser** or have **CREATEDB** privelege.
- Basic syntax:

```
CREATE DATABASE name
[ [ WITH ]
    [ OWNER [=] user_name ]
    [ TEMPLATE [=] template ]
    [ ENCODING [=] encoding ]
    [ LC_COLLATE [=] lc_collate ]
    [ LC_CTYPE [=] lc_ctype ]
    [ TABLESPACE [=] tablespace_name ]
    [ ALLOW_CONNECTIONS [=] allowconn ]
    [ CONNECTION LIMIT [=] connlimit ]
    [ IS_TEMPLATE [=] istemplate ]
];
```

Parameters

- **name** — name of database to create
- **OWNER** = **user_name**
    - Role that will own database
    - `DEFAULT` → default template (usually `template1`)
- **ENCODING** = **encoding**
    - Character set encoding
- **LC_COLLATE = lc_collate**

    - Collation (sorting order) for strings (e.g., `'sv_SE.utf8'`).
- **LC_CTYPE = lc_ctype**
    - Character classification (e.g., upper/lowercase rules).
- **TABLESPACE = tablespace_name**
    - Tablespace that will store the database.
- **ALLOW_CONNECTIONS = allowconn**
    - If `false` → nobody can connect to this DB.
- **CONNECTION LIMIT = connlimit**
    - Max number of concurrent connections. `1` = no limit (default).
- **IS_TEMPLATE = istemplate**
    - If `true` → this database can be cloned by any user with `CREATEDB`.
    - If `false` (default) → only superusers or the owner can clone it.

Examples:

```
CREATE DATABASE music
    LC_COLLATE 'sv_SE.utf8'
    LC_CTYPE   'sv_SE.utf8'
    TEMPLATE template0;

CREATE DATABASE music2
    LC_COLLATE 'sv_SE.iso885915'
    LC_CTYPE   'sv_SE.iso885915'
    ENCODING   LATIN9
    TEMPLATE   template0;
```

Tablespace

> Tablespace = physical location on disk where database objects are stored.

```
CREATE TABLESPACE fastspace
    LOCATION '/ssd1/postgresql/data';
```

```
CREATE TABLESPACE indexspace
    OWNER genevieve
    LOCATION '/data/indexes';
```

Deleting a Database

Command: `DROP DATABASE` — removes a database.

- Removes catalog entries for that database.
- Deletes the directory that contains its data.
- Can be executed **only by database owner**.

```
DROP DATABASE name;
```

DDL – Data Definition Language

> DDL = **commands that define structures** in a database.

DDL statements **create, modify, remove** database objects:

- tables, indexes, users, etc.

Common DDL commands:

- `CREATE`
- `ALTER`
- `DROP`

Table Basics

A table:

- consists of **rows** and **columns**.
- **Number and order of columns is fixed.**
- Each column has:
    - a **name**

    - a **data type**
- Number of **rows is variable** (can grow/shrink).

Creating Tables

Command: `CREATE TABLE` — define a new table.

- Creates a new, initially **empty** table in the current database.
- Table will be owned by the user issuing the command.

Syntax:

```
CREATE TABLE [ IF NOT EXISTS ] table_name (
    { column_name data_type [ COLLATE collation ] [ column_constraint [ ... ] ]
    | table_constraint
    | LIKE source_table [ like_option ... ] }
    [, ... ]
)
[ INHERITS ( parent_table [, ... ] ) ]
[ TABLESPACE tablespace_name ];
```

Parameters

- **IF NOT EXISTS**
    - Do not throw error if relation with same name already exists.
- **table_name**
    - Name of the table (optionally schema-qualified).
- **column_name**
    - Name of a column in the new table.
- **data_type**
    - Data type of column (can include array specifiers).
- **COLLATE collation**
    - Assigns collation to the column (must be collatable type).

- **INHERITS (parent_table [, ...])**
  - New table automatically **inherits all columns** from listed tables.
  - Parent tables can be normal or foreign tables.
- **LIKE source_table [like_option ...]**
  - Copies all column names, data types, and NOT NULL constraints from `source_table` into new table.
- **CONSTRAINT constraint_name**
  - Optional name for column or table constraint.
- **TABLESPACE tablespace_name**
  - Tablespace where the table is created.

Default Values for Columns

> `DEFAULT` defines a value used when `INSERT` omits the column.

```
CREATE TABLE products (
    product_no integer,
    name       text,
    price      numeric DEFAULT 9.99
);
```

If `INSERT` does not specify `price`, value `9.99` will be used.

Modifying Existing Tables

Command: `ALTER TABLE` — change definition of an existing table.

Operations

- **ADD COLUMN**
  - Adds new column, same syntax as in `CREATE TABLE`.
- **DROP COLUMN**

    - Removes a column.
    - Indexes and constraints involving this column are dropped automatically.
- **SET DATA TYPE / TYPE**
    - Changes column data type.
- **SET DEFAULT / DROP DEFAULT**
    - Set or remove default value for a column.
- **SET NOT NULL / DROP NOT NULL**
    - Mark column to **reject NULLs** or **allow NULLs**.
- **ADD table_constraint**
    - Add new constraint (same syntax as `CREATE TABLE`).

examples:

```
ALTER TABLE distributors
   ADD COLUMN address varchar(30);

ALTER TABLE distributors
   DROP COLUMN address RESTRICT;

ALTER TABLE distributors
   ALTER COLUMN address TYPE varchar(80),
   ALTER COLUMN name    TYPE varchar(100);

ALTER TABLE distributors
   ALTER COLUMN address SET DATA TYPE varchar(80),
   ALTER COLUMN name    SET DATA TYPE varchar(100);
```

Deleting Tables

Command: `DROP TABLE` — removes tables from the database.

Only:

- table owner,

- schema owner,
- or superuser

  can drop a table.

Parameters

- **CASCADE**
  - Automatically drop objects that depend on this table (e.g., views),

    and all objects that depend on those.
- **RESTRICT**
  - Refuse to drop if any objects depend on it.
  - **This is the default.**

```
DROP TABLE table_name [ CASCADE | RESTRICT ];
```

Data Types Overview

PostgreSQL has many native data types. You can also create new types with `CREATE TYPE`

Main Categories

- Numeric types
- Character types
- Binary data types
- Date/Time types
- Boolean types
- Arrays

Integer Types

- **smallint** (2 bytes)
  - Range: **−32768 to +32767** (small-range integer).
- **integer** (4 bytes)

    - Typical choice for int.
    - Range: **−2147483648 to +2147483647**.
- **bigint** (8 bytes)
    - Large-range integer.
    - Range: **−9223372036854775808 to +9223372036854775807**.

Floating-Point Types

- **real** (4 bytes)
    - Variable-precision, inexact.
    - About **6 decimal digits precision**.
- **double** (often `double precision`) (8 bytes)
    - Variable-precision, inexact.
    - About **15 decimal digits precision**.

Serial Types

- **smallserial** (2 bytes)
    - Auto-increment int, range **1 to 32767**.
- **serial** (4 bytes)
    - Auto-increment int, range **1 to 2147483647**.

Character Types

- **varchar(n)**
    - Variable-length string, **with limit** `n`.
- **char(n)**
    - **Fixed-length** string, blank-padded to length `n`.
- **text**
    - Variable-length string, **unlimited length**.

Binary Data Type

bytea

- Variable-length binary string.

Date/Time Types

- **timestamp [(p)] [without time zone]** (8 bytes)
  - Date + time, **no time zone**.
- **timestamp [(p)] with time zone** (8 bytes)
  - Date + time, **with time zone**.
- **date** (4 bytes)
  - Calendar date, **no time of day**.
- **time [(p)] [without time zone]** (8 bytes)
  - Time of day only, **no date**.
- **time [(p)] with time zone** (12 bytes)
  - Time of day only, **with time zone**.
- **interval [fields] [(p)]** (16 bytes)
  - Time interval (e.g., “2 days 3 hours”).

Boolean Type

- **boolean** (1 byte)
  - Logical value: **TRUE** or **FALSE** (and `NULL`).