PostgreSQL

Skill Course

Day – 6 : Data types and Constrains in SQL

Numeric Data Types

Data Type	Description	Example Use
INTEGER	Stores whole numbers.	Employee IDs, Age
SERIAL	Auto-incrementing integer.	Primary key auto-increment
BIGINT	Stores large integers.	Population, Large counts
NUMERIC(p, s)	Stores exact numbers with precision (p) and scale (s).	Financial data (salary)
REAL	Stores floating-point numbers (single precision).	Scientific calculations
DOUBLE PRECISION	Stores double-precision floating-point numbers.	High-precision computations

Example

```
CREATE TABLE employees (
    employee_id SERIAL PRIMARY KEY,
    age INTEGER,
    salary NUMERIC(10, 2)
);
```

Character Data Types

Data Type	Description	Example Use
CHAR(n)	Fixed-length string of n characters.	Employee codes
VARCHAR(n)	Variable-length string up to n characters.	Names, Email addresses
TEXT	Unlimited-length string.	Descriptions, Comments

Example

```
CREATE TABLE products (
    product_id SERIAL PRIMARY KEY,
    name VARCHAR(50),
    description TEXT
);
```

Date and Time Data Types

Data Type	Description	Example Use
DATE	Stores date (year, month, day).	Birthdate, Hire date
TIME	Stores time (hour, minute, second).	Appointment times
TIMESTAMP	Stores date and time.	Order timestamps
TIMESTAMPTZ	Stores date and time with timezone info.	Global event tracking
INTERVAL	Stores duration of time.	Duration between events

Example

```
CREATE TABLE events (
    event_id SERIAL PRIMARY KEY,
    event_name VARCHAR(100),
    event_date TIMESTAMP,
    duration INTERVAL
);
```

Boolean Data Type

Data Type	Description	Example Use
BOOLEAN	Stores TRUE, FALSE, or NULL.	Flags, Active status

```
CREATE TABLE users (
    user_id SERIAL PRIMARY KEY,
    username VARCHAR(50),
    is_active BOOLEAN
);
```

PostgreSQL Constraints:

In PostgreSQL, constraints are rules enforced on data in tables to ensure accuracy, consistency, and integrity. They define conditions that the data must meet and are applied to columns or tables during table creation or modification.

- NOT NULL Constraint
- UNIQUE Constraint
- PRIMARY KEY Constraint
- FOREIGN KEY Constraint
- CHECK Constraint
- DEFAULT Constraint
- EXCLUSION Constraint

```
CREATE TABLE employees (
    employee_id SERIAL PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    position VARCHAR(50) NOT NULL
);
```

Assigning Constraints

```sql
CREATE TABLE students (
    student_id SERIAL PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    email VARCHAR(100) UNIQUE,
    age INTEGER CHECK (age >= 18),
    registration_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

PRIMARY KEY: student_id uniquely identifies each student.

NOT NULL: name cannot be NULL

UNIQUE: email must be unique.

CHECK: age must be 18 or older.

DEFAULT: registration_date defaults to the current time.

FOREIGN KEY Constraint

The FOREIGN KEY constraint ensures that values in a column match values in another table's primary key, maintaining referential integrity.

```
CREATE TABLE departments (
    department_id SERIAL PRIMARY KEY,
    department_name VARCHAR(50) NOT NULL
);

CREATE TABLE employees (
    employee_id SERIAL PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    department_id INTEGER,
    FOREIGN KEY (department_id) REFERENCES departments(department_id)
);
```