PostgreSQL / SQL Notes

Reference: https://www.pgtutorial.com/

---

1. Introduction to Tables

A **table** stores data in rows and columns. - **Row (Record):** One complete entry - **Column (Field):** One attribute

Example:

```
student_info
----------------------------
roll_number | first_name | last_name
1           | Sanket     | Kamble
```

---

2. Data Types

Data types define what kind of data a column can store.

Data Type	Description	Example
INT	Integer values	10, 25
VARCHAR(n)	Text data	'Sanket'
DATE	Date value	'2025-01-01'
BOOLEAN	True/False	true

---

3. CREATE TABLE

Used to create a table.

```sql
CREATE TABLE student_info (
    roll_number INT PRIMARY KEY,
    first_name VARCHAR NOT NULL,
    last_name VARCHAR NOT NULL
);
```

---

4. INSERT Statement

Used to insert records into a table.

```
INSERT INTO student_info (roll_number, first_name, last_name)
VALUES (1, 'Sanket', 'Kamble');
```

---

5. SELECT Statement

Used to retrieve data.

```
SELECT * FROM student_info;
```

With condition:

```
SELECT * FROM student_info WHERE roll_number = 1;
```

---

6. WHERE Clause

Filters records based on conditions.

Common operators: - `=` Equal - `>` Greater than - `<` Less than - `IN` Multiple values - `BETWEEN` Range

---

7. SQL Constraints (Detailed Explanation)

Constraints are rules applied on table columns to ensure **data accuracy, consistency, and validity**.

They prevent invalid data from entering the database.

2NF (Second Normal Form)

- Must be in 1NF
- No partial dependency

Split data into separate tables.

3NF (Third Normal Form)

- Must be in 2NF
- No transitive dependency

Example:

```
student_id → dept_id → dept_name
```

Split into STUDENT and DEPARTMENT tables.

---

8. PRIMARY KEY

A **Primary Key** uniquely identifies each record in a table.

Rules:

- Must be **unique**
- Cannot be **NULL**
- One primary key per table

Example:

```
roll_number INT PRIMARY KEY
```

This ensures every student has a unique roll number.

---

9. UNIQUE Constraint

Ensures that values in a column or combination of columns are unique.

Single Column UNIQUE

```
first_name VARCHAR UNIQUE
```

Composite UNIQUE (Multiple Columns)

```
UNIQUE(customer_id, order_id)
```

This prevents duplicate customer–order combinations.

---

10. NOT NULL Constraint

Ensures that a column **must have a value**.

```
last_name VARCHAR NOT NULL
```

If you try to insert NULL, the query will fail.

---

11. DEFAULT Constraint

Assigns a default value if no value is provided during insertion.

```
middle_name VARCHAR DEFAULT 'UNKNOWN'
```

Example:

```
INSERT INTO student_info (roll_number, first_name, last_name)
VALUES (1, 'Amit', 'Sharma');
```

Middle name will automatically be set to `UNKNOWN`.

---

12. CHECK Constraint

Restricts values based on a condition.

Example 1: Numeric Check

```
marks INT CHECK (marks > 0)
```

Example 2: Allowed Values

```
subject VARCHAR CHECK (subject IN ('A','B','C'))
```

This ensures only valid values are stored.

---

13. FOREIGN KEY

A **Foreign Key** creates a relationship between two tables.

- It refers to the **Primary Key** of another table

- Maintains **referential integrity**

Example:

```
FOREIGN KEY (course_id)
REFERENCES courses(course_id)
```

---

14. Foreign Key Actions

ON DELETE / ON UPDATE options:

```
ON DELETE CASCADE
ON UPDATE CASCADE
```

Action	Meaning
CASCADE	Automatically update/delete child rows
SET NULL	Set foreign key to NULL
SET DEFAULT	Assign default value
NO ACTION	Reject operation

---

15. ALTER TABLE (Add Constraint After Creation)

Used to add constraints to an existing table.

```
ALTER TABLE sales
ADD CONSTRAINT unique_cus_ord UNIQUE(customer_id, order_id);
```

---

16. DROP TABLE

Deletes a table permanently along with its data.

```
DROP TABLE student_info;
```

⚠️This operation cannot be undone.

UNIQUE

Ensures values are unique.

Single column:

```
first_name VARCHAR UNIQUE
```

Composite unique:

```
UNIQUE(customer_id, order_id)
```

NOT NULL

Column cannot have NULL values.

```
last_name VARCHAR NOT NULL
```

DEFAULT

Provides default value.

```
middle_name VARCHAR DEFAULT 'UNKNOWN'
```

CHECK

Validates column values.

```
CHECK (marks > 0)
```

```
CHECK (subject IN ('A','B','C'))
```

FOREIGN KEY

Creates relationship between tables.

```
FOREIGN KEY (course_id)
REFERENCES courses(course_id)
```

---

9. Foreign Key with Actions

```
ON DELETE CASCADE
ON UPDATE CASCADE
```

Options: - CASCADE - SET NULL - SET DEFAULT - NO ACTION

---

10. Example: Student & Course Relation

```
COURSES                 STUDENTS
--------                ---------
course_id (PK) <----    course_id (FK)
course_name             roll_num
                         fname
```

---

11. ALTER TABLE (Add Constraint)

```
ALTER TABLE sales
ADD CONSTRAINT unique_cus_ord UNIQUE(customer_id, order_id);
```

---

12. DROP TABLE

Deletes table permanently.

```
DROP TABLE student_info;
```

---

13. Quick Summary

- CREATE → create table
- INSERT → add data
- SELECT → fetch data
- WHERE → filter data
- Constraints → validation rules

- Normalization → clean database design

✅These notes are suitable for **exams, interviews, and practice**.