Temporary Structures in PostgreSQL

CTEs vs Temp Tables vs PL/pgSQL Variables

@AshishZope

Temporary Table (TEMP TABLE)

- Definition: A real table that exists temporarily in a session.
- Use Case: When you need to store and reuse intermediate results across multiple queries in a session or transaction.
- Key Points:
  - Created with: CREATE TEMP TABLE temp_name (...)
    Data persists for the session or transaction (depending on how it's declared).
  - Can be indexed and analyzed.
    Slower than CTEs for one-time use but great for reuse and complex logic.

```sql
CREATE TEMP TABLE temp_users AS
SELECT * FROM users WHERE active = true;

SELECT * FROM temp_users WHERE age > 30;
```

CTE (Common Table Expression)

- CTE (Common Table Expression)
- Definition: A temporary result set that is defined within the execution scope of a single SQL statement.
- Use Case: When you want to break a complex query into readable parts or reuse subqueries.
- Key Points:
  - Defined using WITH clause.
  - Only exists for the duration of the query.
  - Not materialized by default in PostgreSQL 12+ (query planner may inline it).

```sql
WITH active_users AS (
    SELECT * FROM users WHERE active = true
)
SELECT * FROM active_users WHERE age > 30;
```

Temp Variable (PL/pgSQL Variable)

- Definition: Variables defined in PL/pgSQL procedures or functions to store temporary values.
- Use Case: When writing procedural logic and need temporary storage (e.g., loop counters, intermediate values).
- Key Points:
  - Declared inside functions or DO blocks.
  - Exist only during the function execution.
  - Cannot be queried like tables.

```
DO $$
DECLARE
    user_count INTEGER;
BEGIN

    SELECT COUNT(*) INTO user_count FROM users
    WHERE active = true;

    RAISE NOTICE 'Active users: %', user_count;

END $$;
```

Found Helpful..?

Repost

Follow for more..!

@AshishZope