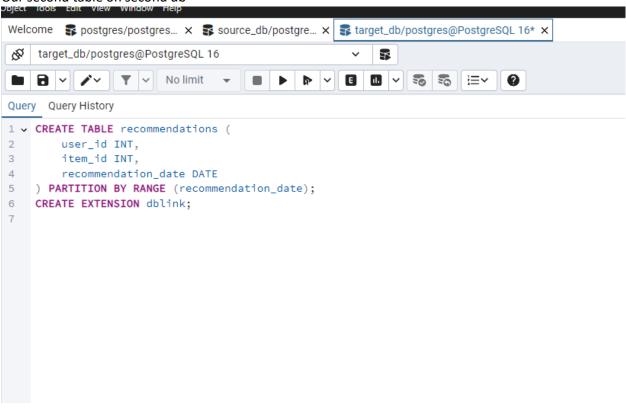
# This is our first table, on a first db

### Our second table on second db



### Create a partition for yesterday as we usually run our code at 0-1 a.m.

```
elcome 🗣 postgres/postgres... × 🗣 source_db/postgre... × 🗣 target_db/postgres@PostgreSQL 16* ×
target_db/postgres@PostgreSQL 16
                                                  Jery Query History

→ CREATE OR REPLACE FUNCTION create_daily_partition()

  RETURNS void AS $$
  DECLARE
      y DATE := CURRENT_DATE - 1;
      t DATE := CURRENT_DATE;
      part_name TEXT := 'recommendations_' || TO_CHAR(y, 'YYYY_MM_DD');

→ BEGIN

          'CREATE TABLE IF NOT EXISTS %I PARTITION OF recommendations FOR VALUES FROM (%L) TO (%L);',
          part_name, y::text, t::text
  END $$ LANGUAGE plpgsql;
  SELECT create_daily_partition();
```

#### Now we move data with dblink

```
★ target_db/postgres@PostgreSQL 16

                                                                          Query History
 1 - DO $$
      DECLARE
           batch_size
                             INT := 10000;
            offset_rows INT := 0;
rows_fetched INT;
           offset_rows
 6 V BEGIN
7 -- Подключаемся к source_db через dblink
8 PERFORM dblink_connect('myconn', 'dbname=source_db user=postgres password=1');
10
               Цикл переноса данных батчами
                EXECUTE format($sql$
12
13
                     INSERT INTO recommendations (user_id, item_id, recommendation_date)

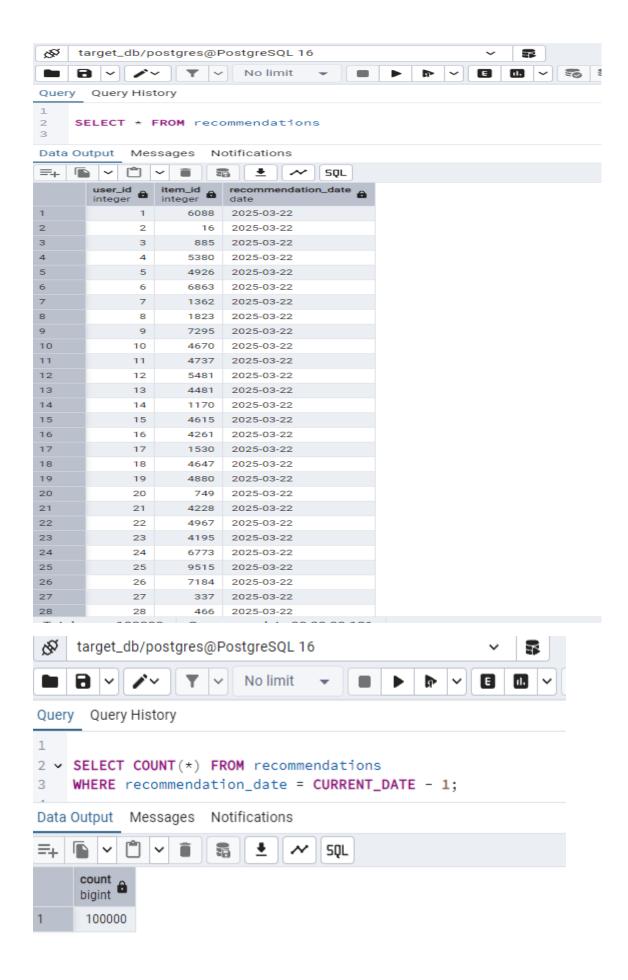
SELECT user_id, item_id, recommendation_date

FROM dblink('myconn', %L)

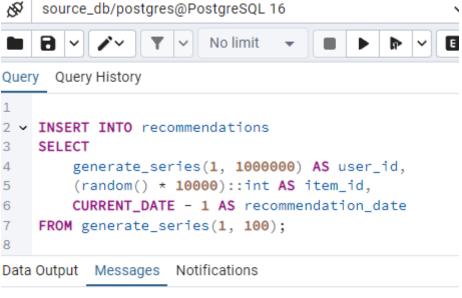
AS t(user_id INT, item_id INT, recommendation_date DATE);
14
15
16
                 format(
19
                      'SELECT user_id, item_id, recommendation_date FROM recommendations
                       WHERE recommendation_date = CURRENT_DATE - INTERVAL ''1 day''
ORDER BY user_id
OFFSET %s LIMIT %s',
21
22
23
                       offset_rows, batch_size
               ));
25
                GET DIAGNOSTICS rows fetched = ROW_COUNT;
28
                EXIT WHEN rows_fetched < batch_size;
offset_rows := offset_rows + batch_size;</pre>
29
30
           END LOOP;
32
             - Отключаемся от dblink
Data Output Messages Notifications
```

Query returned successfully in 1 secs 91 msec.

### Now check it

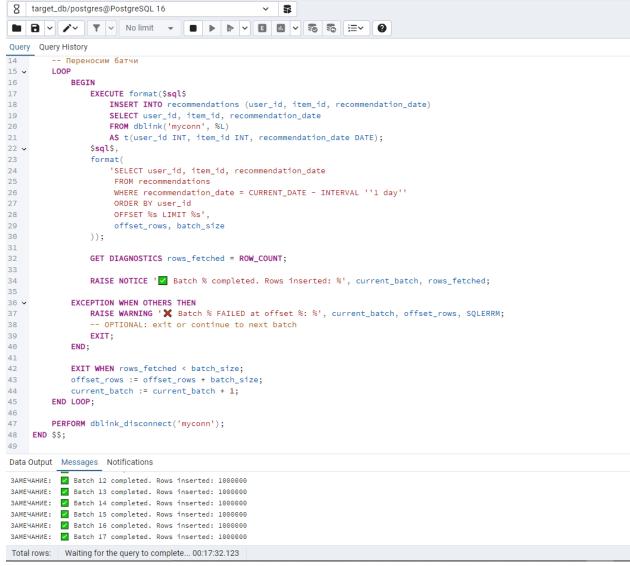


## Now for 1m users and 100 items for each



INSERT 0 100000000

Query returned successfully in 3 min 59 secs.



I decided to batch for 1 million rows, every batch is about a minute so my 100millions rows will be less than 2 hours.

After each batch, I log the batch number and how many rows were inserted