POSTGRESQL DEVELOPMENT GUIDELINE

Mục lục

Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!

Prepare environments

Cài đặt PostgreSQL

Bạn có thể bỏ qua phần này nếu đã cài đặt PostgreSQL.

Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!

Tạo gói portable PostgreSQL

Mục đích của phần này sẽ giúp bạn tự tạo một thư mục để chạy PostgreSQL server mà không cần phải cài đặt (setup). Để không bị xung đột với PostgreSQL server đang có trên máy bạn (nếu có cài trước đây) thì tôi sẽ đổi port mặc định 5432 thành 5439.

Download

Tải bản binary tại:

https://www.enterprisedb.com/download-postgresql-binaries

Ví dụ phiên bản postgresql-13.3-2-windows-x64-binaries sau khi tải về, giải nén ra thì chiếm khoản 655M đĩa cứng

Viết script để thiết lập và khởi động PostgreSQL server

Nội dung file start-postgresql.cmd

```
REM Prepare data folder. Then start PostgreSQL with port 5439
@echo off
set PATH="%CD%\bin";%PATH%
set PGDATA=%CD%\pgdata
set PGDATABASE=postgres
set PGUSER=postgres
set PGPORT=5439
set PGLOCALEDIR=%CD%\share\locale
"%CD%\bin\initdb" -E UTF8 -U postgres -A trust
"%CD%\bin\pg_ctl" -D "%PGDATA%" -l logfile start
echo "#### Ready. Press space to shutdown..."
pause > null
"%CD%\bin\pg_ctl" -D "%PGDATA%" stop
```

Nội dung file start_psql.cmd

```
@echo off
set PATH=%CD%\bin;%PATH%
psql -U postgres -p 5439
```

Thay đổi port 5432 thành 5439 trong module pgAdmin

D:\RunNow\postgresql-13.3-2-windows-x64-binaries\pgsql\pgAdmin 4\web\pgadmin\browser\server_groups\servers\static\js

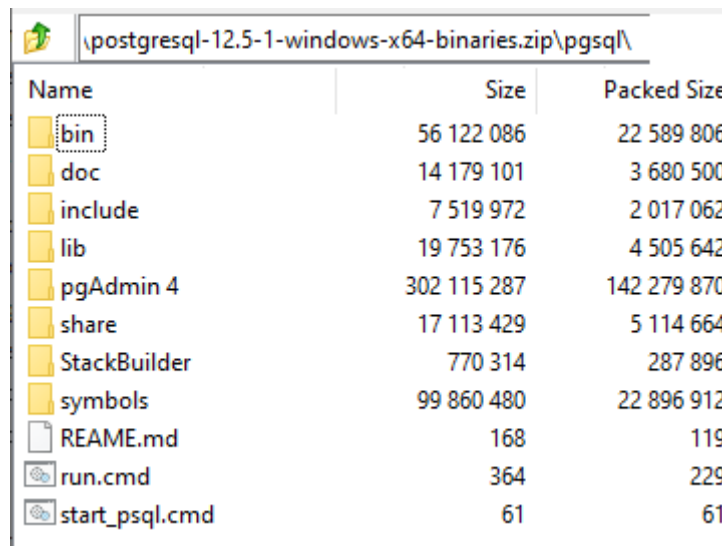Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!

Tải gói binary này:

https://bit.ly/36WioU7

Gói này bao gồm bản postgresql gốc 12.5 được download tại:

https://www.enterprisedb.com/download-postgresql-binaries

và 2 scripts tôi tạo thêm: **run.cmd** và **start_psql.cmd**.
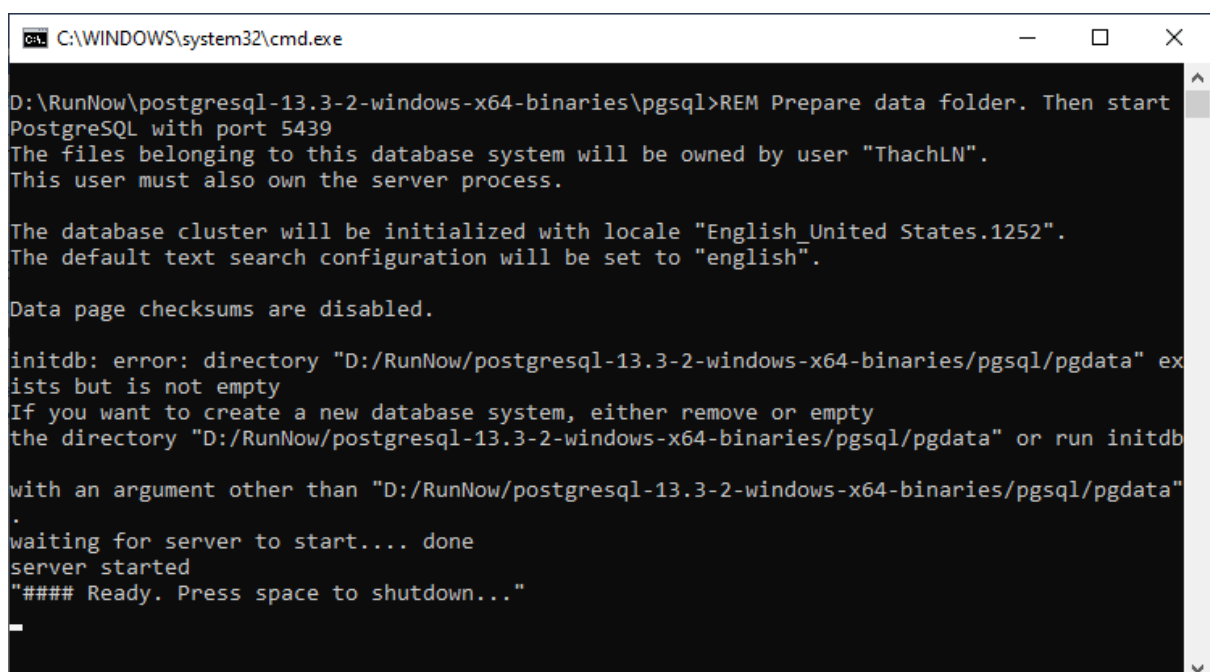
Name	Size	Packed Size
\postgresql-12.5-1-windows-x64-binaries.zip\pgsql\		
bin	56 122 086	22 589 806
doc	14 179 101	3 680 500
include	7 519 972	2 017 062
lib	19 753 176	4 505 642
pgAdmin 4	302 115 287	142 279 870
share	17 113 429	5 114 664
StackBuilder	770 314	287 896
symbols	99 860 480	22 896 912
REAME.md	168	119
run.cmd	364	229
start_psql.cmd	61	61

Sau khi tải về giải nén ra thư mục nào đó. Ví dụ giải nén ra thư mục D:\RunNow thì kết quả là có thư mục D:\RunNow\postgresql-13.3-2-windows-x64-binaries\pgsql.

Đứng trong thư mục này bạn khởi động PostgreSQL được cấu hình tại port 5439 bằng cách chạy script:

```
run.cmd
```

Kết quả là cửa sổ hiện ra bên dưới:

```
C:\WINDOWS\system32\cmd.exe                                    —    □    ×

D:\RunNow\postgresql-13.3-2-windows-x64-binaries\pgsql>REM Prepare data folder. Then start
PostgreSQL with port 5439
The files belonging to this database system will be owned by user "ThachLN".
This user must also own the server process.

The database cluster will be initialized with locale "English_United States.1252".
The default text search configuration will be set to "english".

Data page checksums are disabled.

initdb: error: directory "D:/RunNow/postgresql-13.3-2-windows-x64-binaries/pgsql/pgdata" ex
ists but is not empty
If you want to create a new database system, either remove or empty
the directory "D:/RunNow/postgresql-13.3-2-windows-x64-binaries/pgsql/pgdata" or run initdb

with an argument other than "D:/RunNow/postgresql-13.3-2-windows-x64-binaries/pgsql/pgdata"
.
waiting for server to start.... done
server started
"#### Ready. Press space to shutdown..."
```
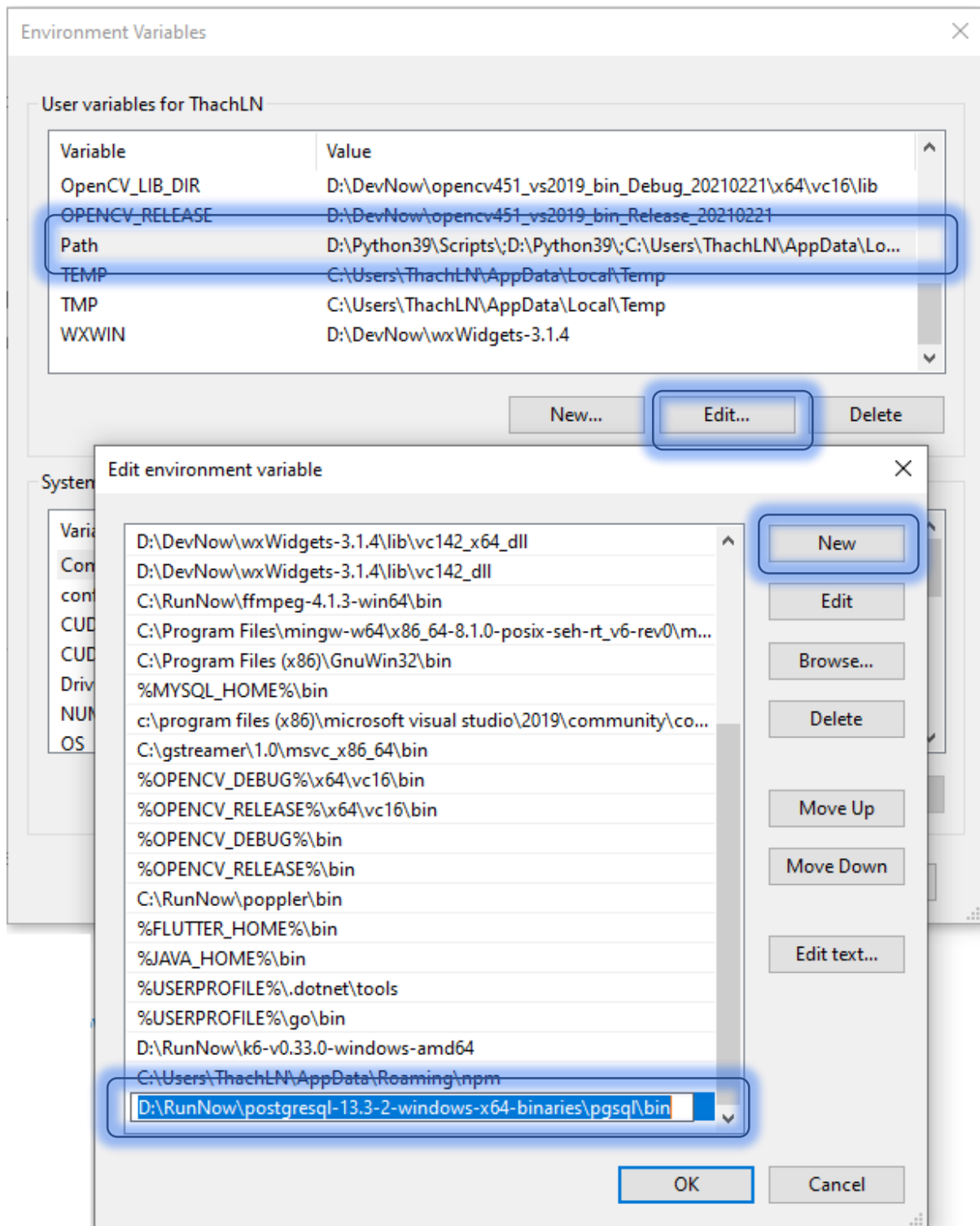
Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!

Sử dụng PostgreSQL

Bổ sung đường dẫn sau vào biến môi trường PATH:

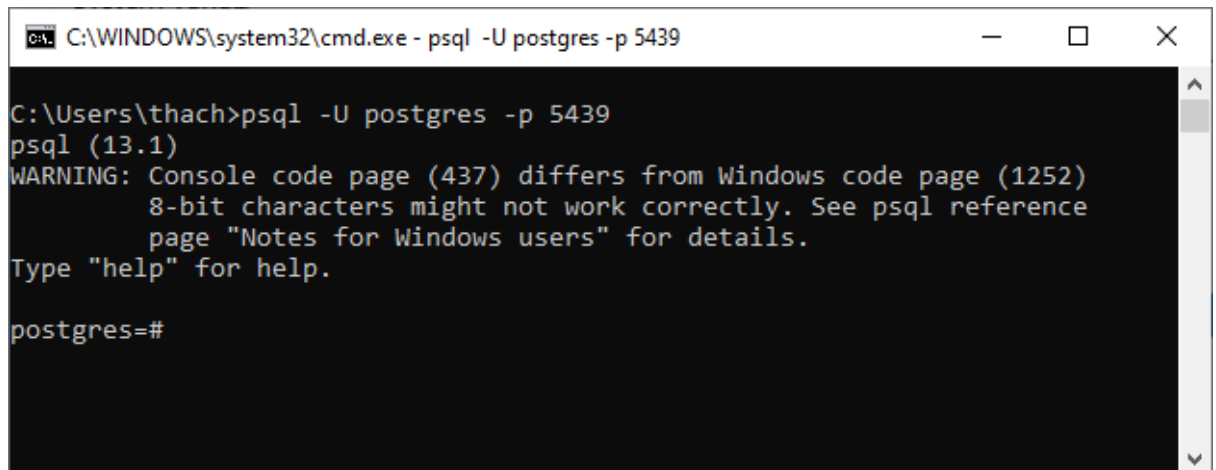D:\RunNow\postgresql-13.3-2-windows-x64-binaries\pgsql\bin

Kết nối server

Thực hiện lệnh sau để kết nối với PostgreSQL, port 5439 với user là "postgres"

```
psql -U postgres -p 5439
```

Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!

Kết quả sẽ ra dấu nhắc của postgres như bên dưới:

Bạn có thể thử vài lệnh sau cho quen:

\l (xuyệt trái và chữ L thường) để liệt kê (list) các database

\q để thoát dấu nhắc lệnh của PostgreSQL

Kết nối PostgreSQL ngay trong Ubuntu

```
sudo -u postgres psql
```

Tạo role và gán quyền

Mở kết nối tới server bằng lệnh sau:

```
psql -U postgres -p 5439
```

Thực hiện các lệnh bên dưới để trải nghiệm kỹ năng tạo database trong dấu nhắc lệnh của PostgreSQL:

```
CREATE DATABASE mydata;
CREATE USER mydata_user WITH PASSWORD 'p@ssw0rd';
GRANT CONNECT ON DATABASE mydata TO mydata_user;
GRANT ALL PRIVILEGES ON DATABASE "mydata" to mydata_user;
```

Mở cửa sổ cmd hỗ trợ UTF8

Tham khảo:

https://stackoverflow.com/questions/57131654/using-utf-8-encoding-chcp-65001-in-command-prompt-windows-powershell-window

Kết nối database

Thử kết nối vào database **mydata** với username và password đã tạo bằng lệnh sau:

```
psql -U mydata_user -d mydata -h 127.0.0.1 -p 5439 -W
```

Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!

Gõ mật khẩu: p@ssw0rd

Nhấn Enter thì kết quả sẽ ra dấu nhắc lệnh như bên dưới:

Tạo dữ liệu dạng cây

Thử 2 câu lệnh sau để tạo ra table family để mô tả mối quan hệ như sau:


```
CREATE TABLE family (person text PRIMARY KEY, parent text
REFERENCES family);

INSERT INTO family VALUES ('Alan', NULL), ('Bert', 'Alan'),
('Bob', 'Alan'), ('Carl', 'Bert'), ('Carmen', 'Bert'),
('Cecil', 'Bob'), ('Dave', 'Cecil'), ('Den', 'Cecil');
```

Thực hiện 2 câu lệnh sau:

```
WITH RECURSIVE genealogy (bloodline, person, level) AS

(

SELECT person, person, 0 FROM family WHERE parent IS NULL
UNION ALL

SELECT g.bloodline || ' -> ' || f.person, f.person, g.level +
1

FROM family f, genealogy g WHERE f.parent = g.person

)
```

Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!

```
SELECT bloodline, level FROM genealogy;

          bloodline           | level
------------------------------+-------
 Alan                         |   0
 Alan -> Bert                  |   1
 Alan -> Bob                   |   1
 Alan -> Bert -> Carl          |   2
 Alan -> Bert -> Carmen        |   2
 Alan -> Bob -> Cecil          |   2
 Alan -> Bob -> Cecil -> Dave  |   3
 Alan -> Bob -> Cecil -> Den   |   3
(8 rows)
```

Tham khảo cách tạo database và insert dữ liệu tại:

- https://github.com/PacktPublishing/Learning-PostgreSQL-10-Second-Edition/blob/master/Chapter03/building_blocks/schema.sql

- https://github.com/PacktPublishing/Learning-PostgreSQL-10-Second-Edition/blob/master/Chapter03/building_blocks/data.sql

Tạo database với user riêng

```
CREATE DATABASE db_name;

CREATE USER db_user WITH PASSWORD 'Abc!123';

GRANT CONNECT ON DATABASE db_name TO db_user;

GRANT ALL PRIVILEGES ON DATABASE db_name to db_user;
```

Xóa database

```
REVOKE CONNECT ON DATABASE db_name FROM public;


SELECT pg_terminate_backend(pg_stat_activity.pid) FROM
pg_stat_activity WHERE pg_stat_activity.datname = 'db_name';

drop database db_name;
```

Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!
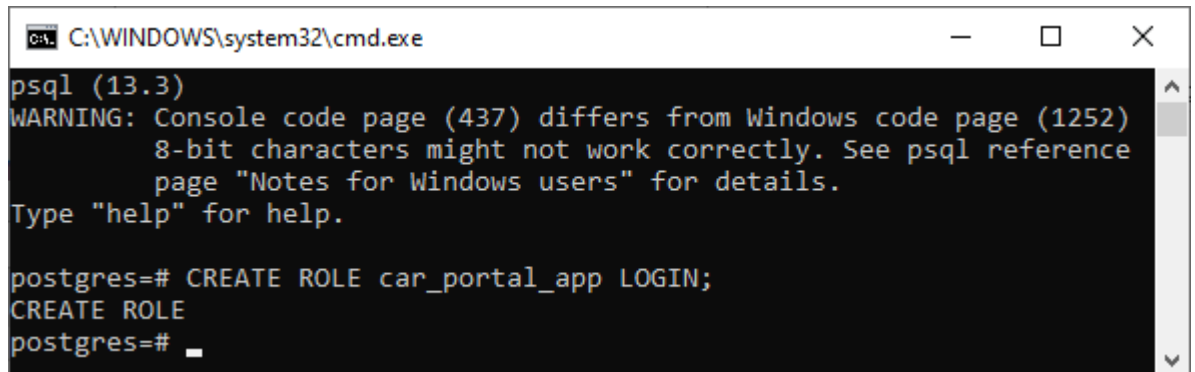
Trải nghiệm Store Procedure

Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!

Chuẩn bị CSDL

Tạo ROLE

Mở cửa sổ lệnh pgsql, thực hiện lệnh:

```
CREATE ROLE car_portal_app LOGIN;
```



```
psql (13.3)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

postgres=# CREATE ROLE car_portal_app LOGIN;
CREATE ROLE
postgres=#
```
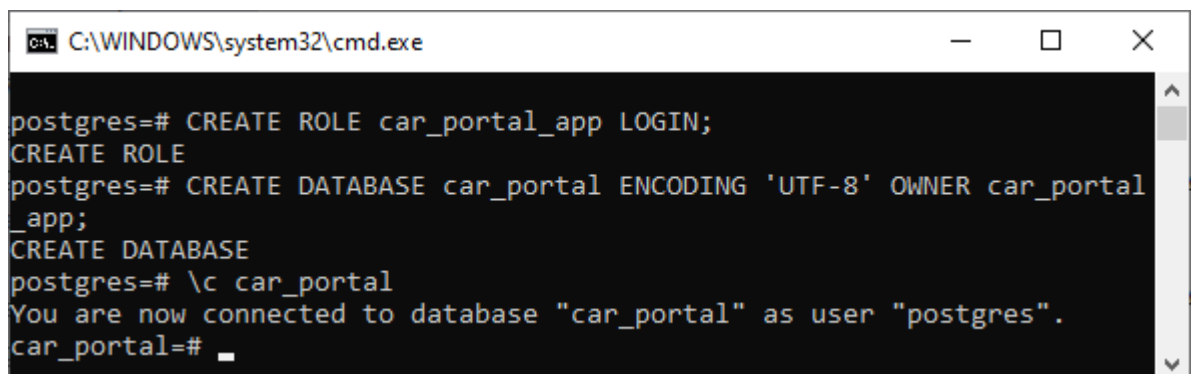
Tạo database

```
CREATE DATABASE car_portal ENCODING 'UTF-8' OWNER
car_portal_app;
```

Kết nối vào database đã tạo

```
\c car_portal
```



```
postgres=# CREATE ROLE car_portal_app LOGIN;
CREATE ROLE
postgres=# CREATE DATABASE car_portal ENCODING 'UTF-8' OWNER car_portal
_app;
CREATE DATABASE
postgres=# \c car_portal
You are now connected to database "car_portal" as user "postgres".
car_portal=#
```

Khởi tạo schema

```
CREATE SCHEMA car_portal_app AUTHORIZATION car_portal_app;


SET search_path to car_portal_app;
SET ROLE car_portal_app;
```

Khởi tạo các table

```
CREATE TABLE account (
```

```sql
    account_id SERIAL PRIMARY KEY,
    first_name TEXT NOT NULL,
    last_name TEXT NOT NULL,
    email TEXT NOT NULL UNIQUE,
    password TEXT NOT NULL,
    CHECK(first_name !~ '\s' AND last_name !~ '\s'),
    CHECK (email ~* '^\w+@\w+[.]\w+$'),
    CHECK (char_length(password)>=8)
);

CREATE TABLE account_history (
    account_history_id BIGSERIAL PRIMARY KEY,
    account_id INT NOT NULL REFERENCES account(account_id),
    search_key TEXT NOT NULL,
    search_date DATE NOT NULL,
    UNIQUE (account_id, search_key, search_date)
);

CREATE TABLE seller_account (
    seller_account_id SERIAL PRIMARY KEY,
    account_id INT NOT NULL REFERENCES account(account_id),
    total_rank FLOAT,
    number_of_advertisement INT,
    street_name TEXT NOT NULL,
    street_number TEXT NOT NULL,
    zip_code TEXT NOT NULL,
    city TEXT NOT NULL
);

CREATE TABLE car_model
(
    car_model_id SERIAL PRIMARY KEY,
    make text,
    model text,
    UNIQUE (make, model)
);

CREATE TABLE car (
    car_id SERIAL PRIMARY KEY,
```

```sql
    number_of_owners INT NOT NULL,
    registration_number TEXT UNIQUE NOT NULL,
    manufacture_year INT NOT NULL,
    number_of_doors INT DEFAULT 5 NOT NULL,
    car_model_id INT NOT NULL REFERENCES car_model (car_model_id),
    mileage INT
);


CREATE TABLE advertisement(
    advertisement_id SERIAL PRIMARY KEY,
    advertisement_date TIMESTAMP WITH TIME ZONE NOT  NULL,
    car_id INT NOT NULL REFERENCES car(car_id),
    seller_account_id INT NOT NULL REFERENCES seller_account (seller_account_id)
);


CREATE TABLE advertisement_picture(
    advertisement_picture_id SERIAL PRIMARY KEY,
    advertisement_id INT REFERENCES advertisement(advertisement_id),
    picture_location TEXT UNIQUE
);


CREATE TABLE advertisement_rating (
    advertisement_rating_id SERIAL PRIMARY KEY,
    advertisement_id INT NOT NULL REFERENCES advertisement(advertisement_id),
    account_id INT NOT NULL REFERENCES account(account_id),
    advertisement_rating_date DATE NOT NULL,
    rank INT NOT NULL,
    review TEXT NOT NULL,
    CHECK (char_length(review)<= 200),
    CHECK (rank IN (1,2,3,4,5))
);


CREATE TABLE favorite_ads(
    account_id INT NOT NULL REFERENCES account(account_id),
    advertisement_id INT NOT NULL REFERENCES advertisement(advertisement_id),
```

```
        primary key(account_id,advertisement_id)
);
```

Kiểm tra lại kết quả trên pgAdmin
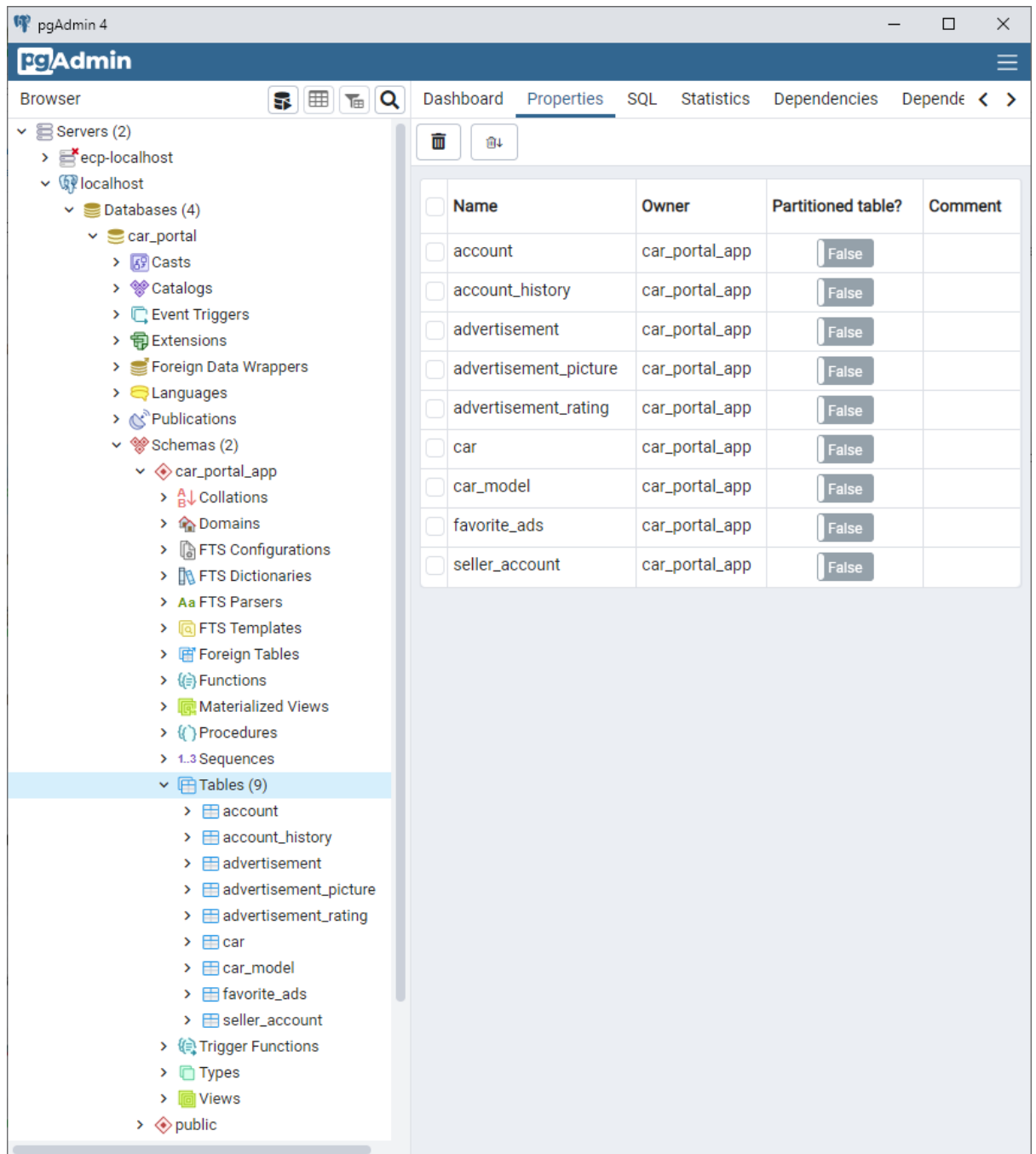
Chuẩn bị dữ liệu mẫu

Thực thi câu lệnh SQL ở đây:

```
https://raw.githubusercontent.com/PacktPublishing/Learning-
PostgreSQL-10-Second-Edition/master/Chapter07/ch-07/data.sql
```

Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!
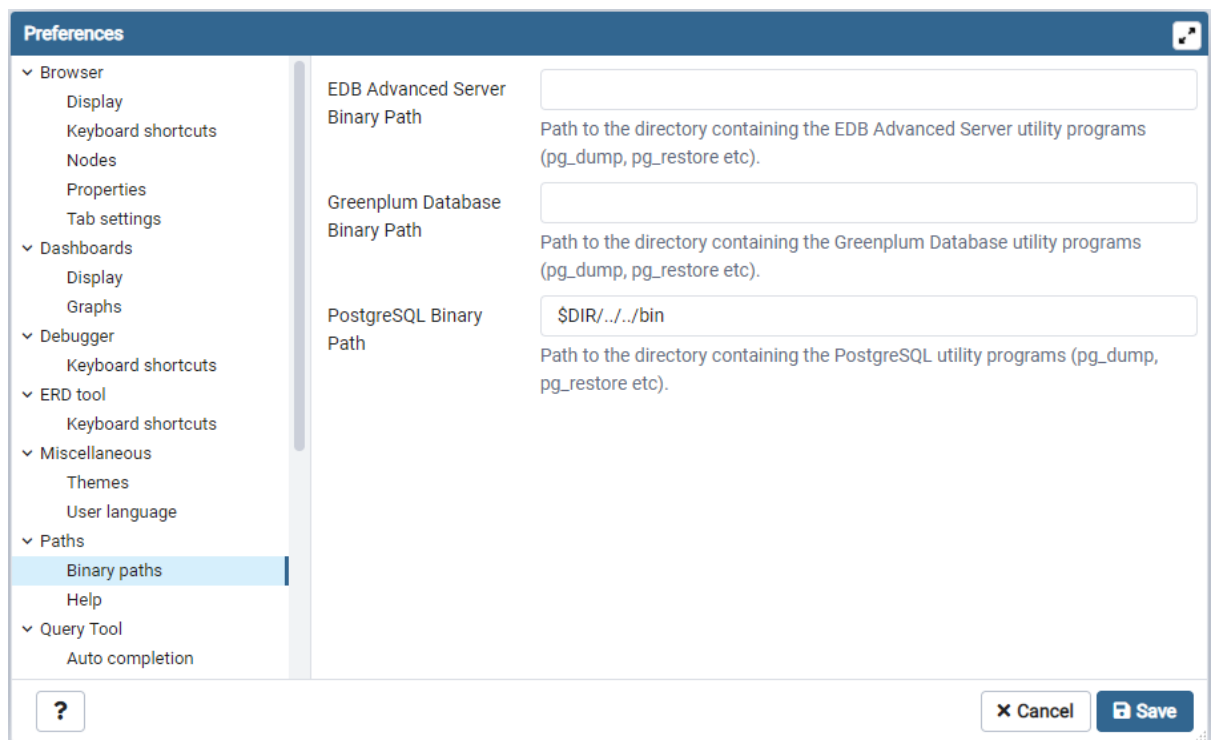
Trải nghiệm Store Procedure

Kết nối CSDL

Mở cửa sổ lệnh pgsql

Mở cửa sổ Windows Command Prompt tại thư mục:

```
D:\RunNow\postgresql-13.3-2-windows-x64-binaries\pgsql\bin
```

Thực hiện lệnh sau:

```
psql -U car_portal_app -p 5439  car_portal
```
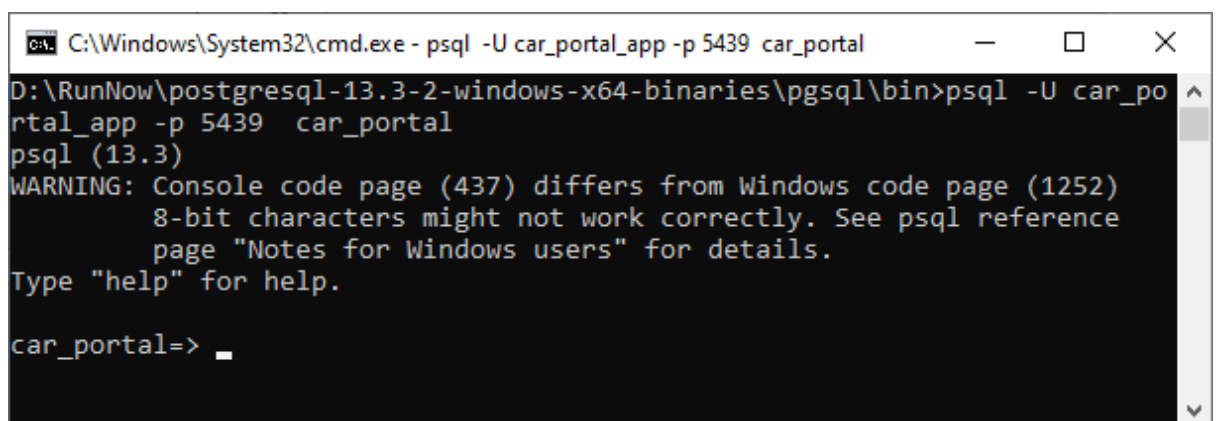

Tạo FUNCTION

Tạo một FUNCTION tên là a bằng lệnh sau:

```
CREATE OR REPLACE FUNCTION a() RETURNS SETOF INTEGER AS $$
```

Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!

```
    SELECT 1;
$$ LANGUAGE SQL;
```

Xem kết quả trên pgAdmin
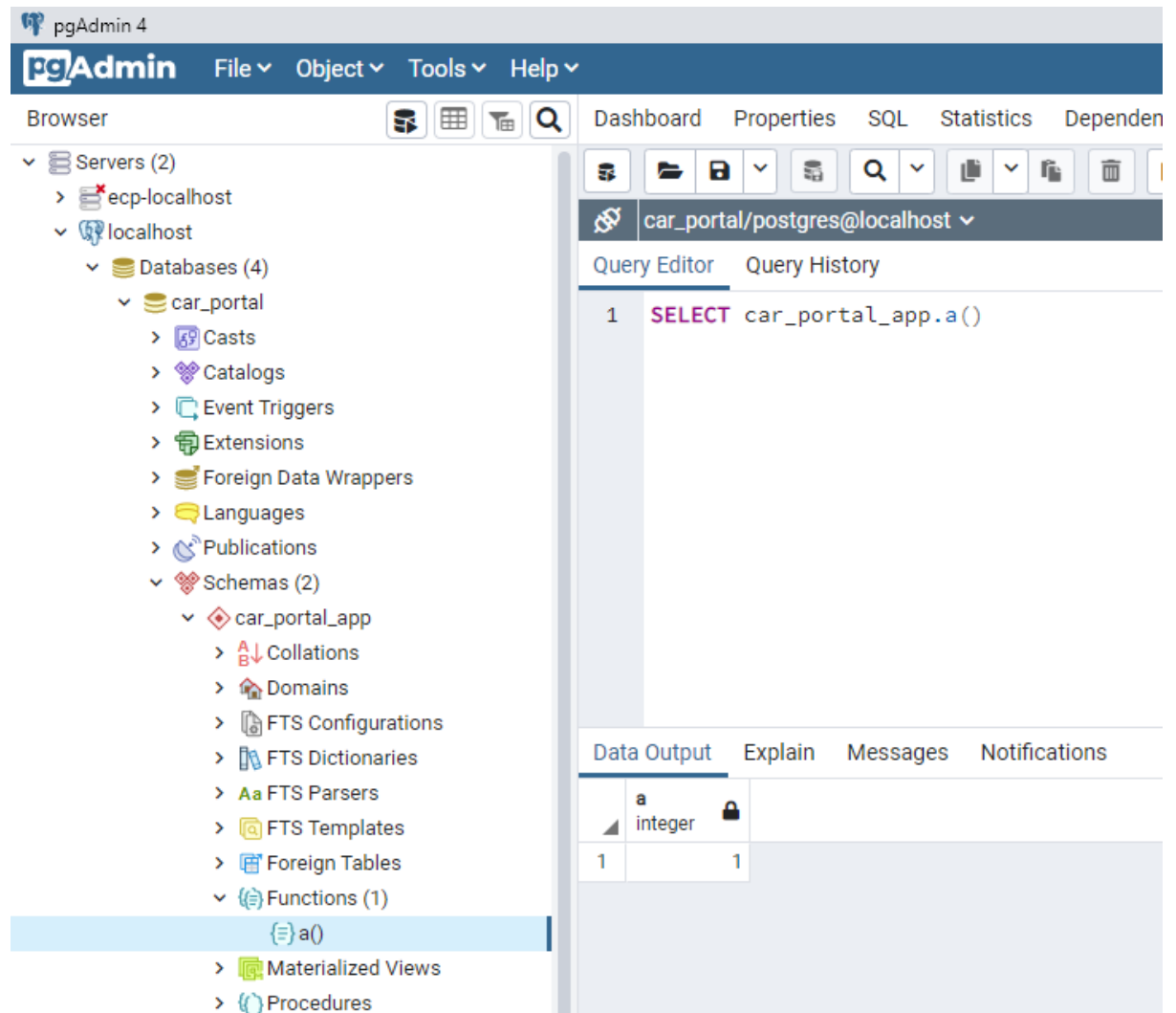
Sử dụng PostgreSQL cho Data Warehouse

Mở cửa số psql, connect vào database có sẵn, thực hiện câu lệnh sau bằng cách copy và paste vào cửa sổ lệnh:

```
DROP TABLE if exists d_date;
CREATE TABLE d_date
(
  date_dim_id          INT NOT NULL,
  date_actual          DATE NOT NULL,
  epoch                BIGINT NOT NULL,
  day_suffix           VARCHAR(4) NOT NULL,
  day_name             VARCHAR(9) NOT NULL,
  day_of_week          INT NOT NULL,
  day_of_month         INT NOT NULL,
```

```sql
    day_of_quarter          INT NOT NULL,
    day_of_year             INT NOT NULL,
    week_of_month           INT NOT NULL,
    week_of_year            INT NOT NULL,
    week_of_year_iso        CHAR(10) NOT NULL,
    month_actual            INT NOT NULL,
    month_name              VARCHAR(9) NOT NULL,
    month_name_abbreviated  CHAR(3) NOT NULL,
    quarter_actual          INT NOT NULL,
    quarter_name            VARCHAR(9) NOT NULL,
    year_actual             INT NOT NULL,
    first_day_of_week       DATE NOT NULL,
    last_day_of_week        DATE NOT NULL,
    first_day_of_month      DATE NOT NULL,
    last_day_of_month       DATE NOT NULL,
    first_day_of_quarter    DATE NOT NULL,
    last_day_of_quarter     DATE NOT NULL,
    first_day_of_year       DATE NOT NULL,
    last_day_of_year        DATE NOT NULL,
    mmyyyy                  CHAR(6) NOT NULL,
    mmddyyyy                CHAR(10) NOT NULL,
    weekend_indr            BOOLEAN NOT NULL
);

ALTER TABLE d_date ADD CONSTRAINT d_date_date_dim_id_pk PRIMARY KEY (date_dim_id);

CREATE INDEX d_date_date_actual_idx
  ON d_date(date_actual);

COMMIT;

INSERT INTO d_date
SELECT TO_CHAR(datum, 'yyyymmdd')::INT AS date_dim_id,
       datum AS date_actual,
       EXTRACT(EPOCH FROM datum) AS epoch,
       TO_CHAR(datum, 'fmDDth') AS day_suffix,
       TO_CHAR(datum, 'TMDay') AS day_name,
       EXTRACT(ISODOW FROM datum) AS day_of_week,
       EXTRACT(DAY FROM datum) AS day_of_month,
       datum - DATE_TRUNC('quarter', datum)::DATE + 1 AS day_of_quarter,
       EXTRACT(DOY FROM datum) AS day_of_year,
       TO_CHAR(datum, 'W')::INT AS week_of_month,
       EXTRACT(WEEK FROM datum) AS week_of_year,
       EXTRACT(ISOYEAR FROM datum) || TO_CHAR(datum, '"-W"IW-') || EXTRACT(ISODOW FROM
datum) AS week_of_year_iso,
       EXTRACT(MONTH FROM datum) AS month_actual,
       TO_CHAR(datum, 'TMMonth') AS month_name,
```
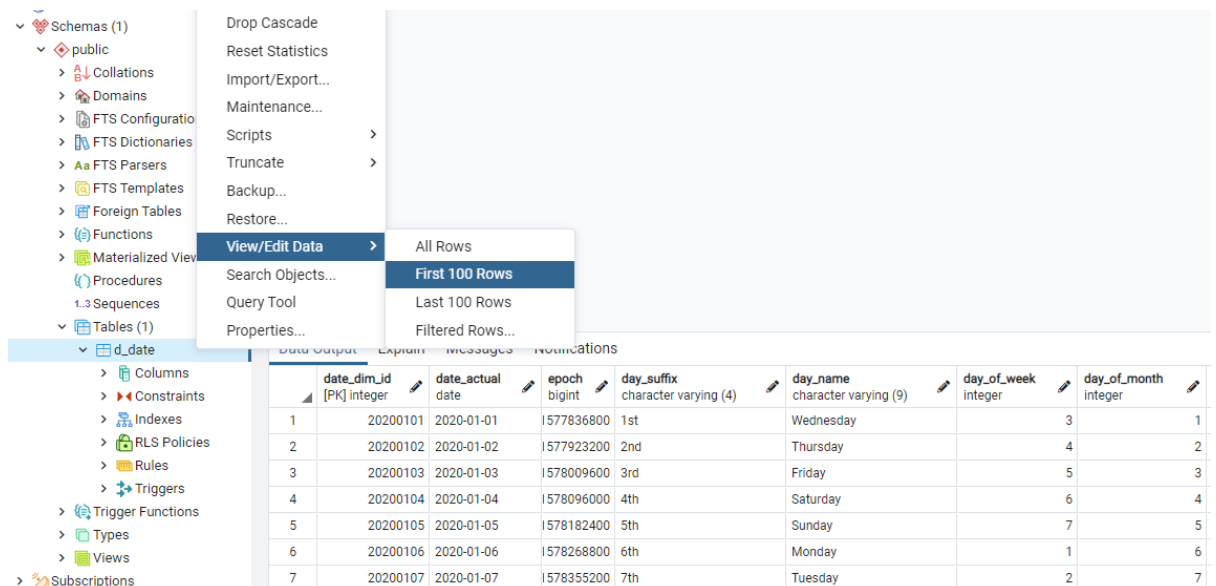
```sql
        TO_CHAR(datum, 'Mon') AS month_name_abbreviated,
        EXTRACT(QUARTER FROM datum) AS quarter_actual,
        CASE
            WHEN EXTRACT(QUARTER FROM datum) = 1 THEN 'First'
            WHEN EXTRACT(QUARTER FROM datum) = 2 THEN 'Second'
            WHEN EXTRACT(QUARTER FROM datum) = 3 THEN 'Third'
            WHEN EXTRACT(QUARTER FROM datum) = 4 THEN 'Fourth'
            END AS quarter_name,
        EXTRACT(YEAR FROM datum) AS year_actual,
        datum + (1 - EXTRACT(ISODOW FROM datum))::INT AS first_day_of_week,
        datum + (7 - EXTRACT(ISODOW FROM datum))::INT AS last_day_of_week,
        datum + (1 - EXTRACT(DAY FROM datum))::INT AS first_day_of_month,
        (DATE_TRUNC('MONTH', datum) + INTERVAL '1 MONTH - 1 day')::DATE AS
last_day_of_month,
        DATE_TRUNC('quarter', datum)::DATE AS first_day_of_quarter,
        (DATE_TRUNC('quarter', datum) + INTERVAL '3 MONTH - 1 day')::DATE AS
last_day_of_quarter,
        TO_DATE(EXTRACT(YEAR FROM datum) || '-01-01', 'YYYY-MM-DD') AS first_day_of_year,
        TO_DATE(EXTRACT(YEAR FROM datum) || '-12-31', 'YYYY-MM-DD') AS last_day_of_year,
        TO_CHAR(datum, 'mmyyyy') AS mmyyyy,
        TO_CHAR(datum, 'mmddyyyy') AS mmddyyyy,
        CASE
            WHEN EXTRACT(ISODOW FROM datum) IN (6, 7) THEN TRUE
            ELSE FALSE
            END AS weekend_indr
FROM (SELECT '2020-01-01'::DATE + SEQUENCE.DAY AS datum
    FROM GENERATE_SERIES(0, 1826) AS SEQUENCE (DAY)
    GROUP BY SEQUENCE.DAY) DQ
ORDER BY 1;


COMMIT;
```

Xem thành quả với pgAdmin:

Đây là eBook của riêng bạn – đề nghị không chia sẻ cho ai khác nhé!

Store Procedure:

FACT f_sales

```sql
DROP TABLE if exists f_sales;
CREATE TABLE f_sales
(
  id            INT NOT NULL,
  dim_date_id        INT NOT NULL,
  dim_product_id      INT NOT NULL,
  dim_seller_id      INT NOT NULL,
  dim_customer_id      INT NOT NULL,
  dim_store_id      INT NOT NULL,
  dim_channel_id      INT NOT NULL,

  m1        FLOAT,
  m2        FLOAT,
  m3        FLOAT,
  m4        FLOAT,
  m5        FLOAT,
  m21        FLOAT8,
  m22        FLOAT8,
  m23        FLOAT8,
  m24        FLOAT8,
  m25        FLOAT8,
  m31        DECIMAL,
  m32        DECIMAL,
  m33        DECIMAL,
  m34        DECIMAL,
  m35        DECIMAL
);

INSERT INTO f_sales
SELECT 111 as ID,
```

```
        123 AS dim_date_id,
        123 AS   dim_product_id,
        123 AS   dim_seller_id,
        123 AS   dim_customer_id,
        123 AS   dim_store_id,
        123 AS   dim_channel_id,
      10 AS m1,
        10 AS m2,
        10 AS m3,
        10 AS m4,
        10 AS m5,
        1000000.5 AS m21,
        1000000.5 AS m22,
        1000000.5 AS m23,
        1000000.5 AS m24,
        1000000.5 AS m25,
        1000000000.5 AS m31,
        1000000000.5 AS m32,
        1000000000.5 AS m33,
        1000000000.5 AS m34,
        1000000000.5 AS m35
    FROM (SELECT '2020-01-01'::DATE + SEQUENCE.DAY AS datum
        FROM GENERATE_SERIES(0, 1826) AS SEQUENCE (DAY)
        GROUP BY SEQUENCE.DAY) DQ
    ORDER BY 1;


    COMMIT;
```

```sql
CREATE OR REPLACE PROCEDURE create_dim_date()
      LANGUAGE sql
AS $procedure$
      BEGIN
            DROP d_date if exists d_date;
            CREATE TABLE d_date
            (
              date_dim_id              INT NOT NULL,
              date_actual              DATE NOT NULL,
              epoch                    BIGINT NOT NULL,
              day_suffix               VARCHAR(4) NOT NULL,
              day_name                 VARCHAR(9) NOT NULL,
              day_of_week              INT NOT NULL,
              day_of_month             INT NOT NULL,
              day_of_quarter           INT NOT NULL,
              day_of_year              INT NOT NULL,
              week_of_month            INT NOT NULL,
              week_of_year             INT NOT NULL,
              week_of_year_iso         CHAR(10) NOT NULL,
              month_actual             INT NOT NULL,
              month_name               VARCHAR(9) NOT NULL,
              month_name_abbreviated   CHAR(3) NOT NULL,
              quarter_actual           INT NOT NULL,
              quarter_name             VARCHAR(9) NOT NULL,
              year_actual              INT NOT NULL,
              first_day_of_week        DATE NOT NULL,
              last_day_of_week         DATE NOT NULL,
              first_day_of_month       DATE NOT NULL,
              last_day_of_month        DATE NOT NULL,
              first_day_of_quarter     DATE NOT NULL,
              last_day_of_quarter      DATE NOT NULL,
              first_day_of_year        DATE NOT NULL,
              last_day_of_year         DATE NOT NULL,
              mmyyyy                   CHAR(6) NOT NULL,
              mmddyyyy                 CHAR(10) NOT NULL,
              weekend_indr             BOOLEAN NOT NULL
            );


            ALTER TABLE d_date ADD CONSTRAINT d_date_date_dim_id_pk PRIMARY KEY
(date_dim_id);


            CREATE INDEX d_date_date_actual_idx
              ON d_date(date_actual);


            COMMIT;


            INSERT INTO d_date
            SELECT TO_CHAR(datum, 'yyyymmdd')::INT AS date_dim_id,
                   datum AS date_actual,
                   EXTRACT(EPOCH FROM datum) AS epoch,
                   TO_CHAR(datum, 'fmDDth') AS day_suffix,
                   TO_CHAR(datum, 'TMDay') AS day_name,
                   EXTRACT(ISODOW FROM datum) AS day_of_week,
                   EXTRACT(DAY FROM datum) AS day_of_month,
```

```sql
                              datum - DATE_TRUNC('quarter', datum)::DATE + 1 AS
day_of_quarter,

                        EXTRACT(DOY FROM datum) AS day_of_year,
                        TO_CHAR(datum, 'W')::INT AS week_of_month,
                        EXTRACT(WEEK FROM datum) AS week_of_year,
                        EXTRACT(ISOYEAR FROM datum) || TO_CHAR(datum, '"-W"IW-') ||
EXTRACT(ISODOW FROM datum) AS week_of_year_iso,
                        EXTRACT(MONTH FROM datum) AS month_actual,
                        TO_CHAR(datum, 'TMMonth') AS month_name,
                        TO_CHAR(datum, 'Mon') AS month_name_abbreviated,
                        EXTRACT(QUARTER FROM datum) AS quarter_actual,
                        CASE
                            WHEN EXTRACT(QUARTER FROM datum) = 1 THEN 'First'
                            WHEN EXTRACT(QUARTER FROM datum) = 2 THEN 'Second'
                            WHEN EXTRACT(QUARTER FROM datum) = 3 THEN 'Third'
                            WHEN EXTRACT(QUARTER FROM datum) = 4 THEN 'Fourth'
                            END AS quarter_name,
                        EXTRACT(YEAR FROM datum) AS year_actual,
                        datum + (1 - EXTRACT(ISODOW FROM datum))::INT AS
first_day_of_week,

                        datum + (7 - EXTRACT(ISODOW FROM datum))::INT AS
last_day_of_week,

                        datum + (1 - EXTRACT(DAY FROM datum))::INT AS
first_day_of_month,

                        (DATE_TRUNC('MONTH', datum) + INTERVAL '1 MONTH - 1
day')::DATE AS last_day_of_month,
                        DATE_TRUNC('quarter', datum)::DATE AS first_day_of_quarter,
                        (DATE_TRUNC('quarter', datum) + INTERVAL '3 MONTH - 1
day')::DATE AS last_day_of_quarter,
                        TO_DATE(EXTRACT(YEAR FROM datum) || '-01-01', 'YYYY-MM-DD')
AS first_day_of_year,
                        TO_DATE(EXTRACT(YEAR FROM datum) || '-12-31', 'YYYY-MM-DD')
AS last_day_of_year,
                        TO_CHAR(datum, 'mmyyyy') AS mmyyyy,
                        TO_CHAR(datum, 'mmddyyyy') AS mmddyyyy,
                        CASE
                            WHEN EXTRACT(ISODOW FROM datum) IN (6, 7) THEN TRUE
                            ELSE FALSE
                            END AS weekend_indr
                FROM (SELECT '2020-01-01'::DATE + SEQUENCE.DAY AS datum
                        FROM GENERATE_SERIES(0, 29219) AS SEQUENCE (DAY)
                        GROUP BY SEQUENCE.DAY) DQ
            ORDER BY 1;


            COMMIT;


    END;
$procedure$
```

Simple Function:

```sql
CREATE OR REPLACE FUNCTION public.my_sum(integer, integer)
 RETURNS integer
 LANGUAGE sql
AS $function$
SELECT $1 + $2;
$function$
;
```

Tham khảo:

https://duffn.medium.com/creating-a-date-dimension-table-in-postgresql-af3f8e2941ac

https://www.nuwavesolutions.com/simple-hierarchical-dimensions-html/

https://www.nuwavesolutions.com/ragged_hierarchical_dimensions/

https://wiki.postgresql.org/images/3/38/PGDay2009-EN-Datawarehousing_with_PostgreSQL.pdf