**BÁO CÁO THỰC HÀNH**

**CƠ SỞ DỮ LIỆU**

***Đề Tài: Trợ lý mua sắm cho người tiêu dùng***

**Giáo viên hướng dẫn**

Vũ Tuyết Trinh

**Thành viên nhóm**

1. Lê Ngọc Long
2. Đoàn Anh Tuấn
3. Trần Hữu An

**Phân công công việc**

Công việc chung: Nêu ý tưởng, thiết kế CSDL

Lê Ngọc Long: Thu thập và xử lý dữ liệu, trigger, hàm chức năng, giao diện app

Đoàn Anh Tuấn: Thu thập dữ liệu, hàm chức năng

Trần Hữu An: Thu thập dữ liệu, hàm chức năng

**Mô tả hệ thống**

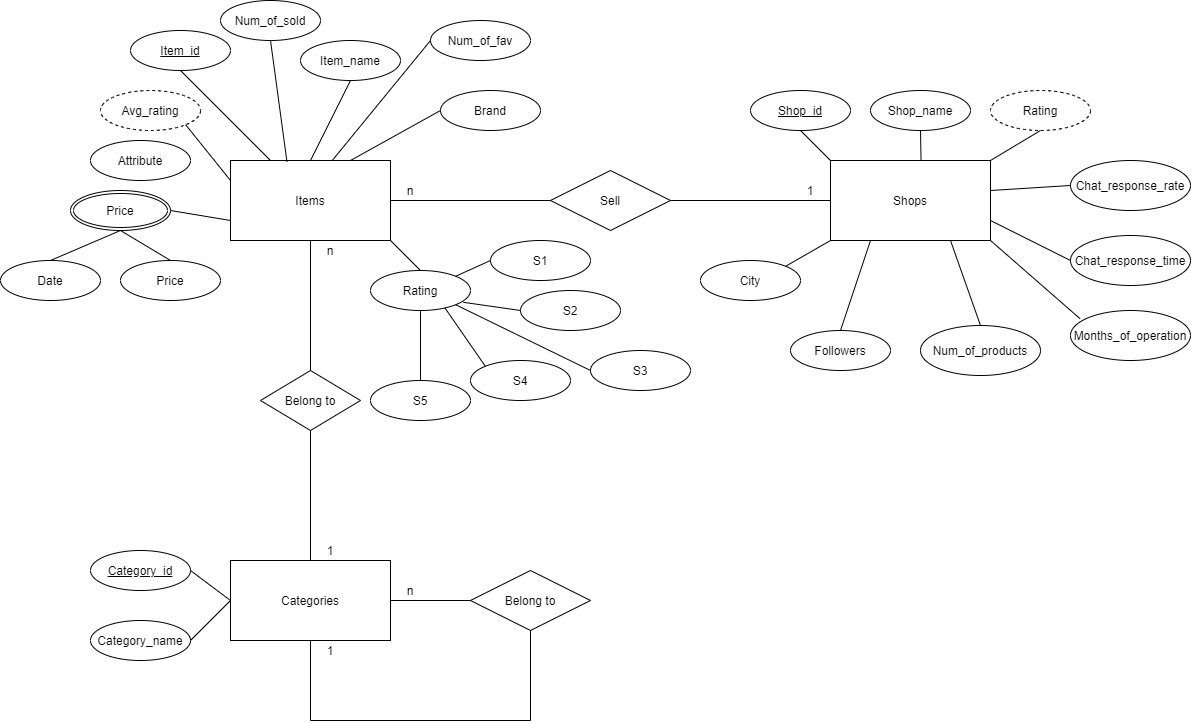
1. **Chức năng chính**

a) Hiển thị thông tin chi tiết của sản phẩm: Thông tin chung, Sơ đồ biến động giá, mô hình hóa số lượng đánh giá, giá thấp nhất, giá cao nhất, giá trung, số lần thay đổi giá, độ dài lịch sử giá; so sánh giá hiện tại so với giá cao nhất và trung bình

b) Hiển thị thông tin chi tiết của shop: Số sản phẩm, thời gian tham gia, tỉ lệ phản hồi chat, số người theo dõi, địa chỉ shop và đánh giá trung bình của shop (thang 5); mô hình hóa số lượng đánh giá, đánh giá xếp hạng chung

c) Tìm kiếm sản phẩm (chưa hoàn thiện): tìm kiếm một sản phẩm được lưu theo tên, khoảng giá, thành phố, đánh giá sản phẩm và sắp xếp theo: giá, lượt thích, lượt mua hay đánh giá.

1. **Sơ đồ thực thể liên kết**



1. **Lược đồ quan hệ**

**Ảnh có chứa bàn

Mô tả được tạo tự động**

**Chi tiết:**

* Mỗi thực thể biểu diễn thành một bảng: items, shops, categories
* Khóa chính được gạch chân và in đậm.
* Khóa ngoài được in nghiêng.
* Quan hệ Shops – Items là quan hệ 1-n => Thêm shop\_id vào bảng Items
* Quan hệ Categories – Items là quan hệ 1-n => Thêm category\_id vào bảng Items
* Thuộc tính Price trong bảng Items là thuộc tính đa trị => Biểu diễn thành một bảng mới Item\_history(**item\_id, date**, price)

a) Bảng shops

* shop\_id: Khóa chính dùng để phân biệt các shop khác nhau
* shop\_name: Tên hiển thị của shop
* chat\_response\_rate: Tỉ lệ phản hồi chat cho biết tần suất Người bán trả lời chat (Phần trăm số tin nhắn trả lời / tổng số tin nhắn nhận)
* chat\_response\_time: Thời gian phản hồi chat kể từ lúc nhận được tin nhắn (theo các mức: trong vài phút, trong vài giờ, trong vài ngày, …)
* months\_of\_operation: Thời gian hoạt động của shop kể từ thời điểm đăng ký bán hàng (tính theo tháng)
* followers: Số người đang theo dõi shop
* City: Thành phố nơi shop hoạt động (là nơi sản phẩm được gửi đi)
* num\_of\_products: Số lượng sản phẩm hiện đang được bán
* Rating: Đánh giá trung bình của người mua (được tính dựa trên đánh giá của tất cả sản phẩm của shop)

b) Bảng items

* item\_id: Khóa chính dùng để phân biệt các sản phẩm khác nhau
* item\_name: Tên hiển thị của sản phẩm
* price: Giá bán hiện tại
* brand: Thương hiệu

shop\_id: Khóa ngoài tham chiếu đến bảng shops cho biết sản phẩm này thuộc shop nào

* category\_id: Khóa ngoài tham chiếu đến bảng categories cho biết sản phẩm thuộc loại mặt hàng nào
* num\_of\_fav: Số người yêu thích sản phẩm
* num\_of\_sold: Số lượng đã bán
* s1, s2, s3, s4, s5: Số lượt đánh giá 1/2/3/4/5 sao

c) Bảng item\_history

* (Item\_id, date): Khóa chính dùng để phân biệt giá của sản phẩm theo ngày
* Item\_id: Khóa ngoài tham chiếu đến bảng items
* Date: Ngày cập nhật giá
* Price: Giá mỗi lần cập nhật

d) Bảng categories

* category\_id: Khóa chính dùng để phân biệt các loại mặt hàng khác nhau
* category\_name: Tên phân loại mặt hàng
* parent\_id: Khóa ngoài tham chiếu đến category\_id trong cùng bảng, biểu thị danh mục cha của danh mục này (danh mục có tối đa 3 mức, parent\_id = 0 ứng với mức cao nhất)

1. **Công việc chi tiết**

**A. Lê Ngọc Long**

**a) Thu thập dữ liệu**

* Viết tool thu thập dữ liệu sử dụng python (selenium), nhập data vào CSDL
* Xử lý dữ liệu: loại bỏ dữ liệu bị thiếu, chuẩn hóa về tên sản phẩm (ký tự đặc biệt), loại bỏ khoảng trắng thừa, …

**b) Giao diện ứng dụng**

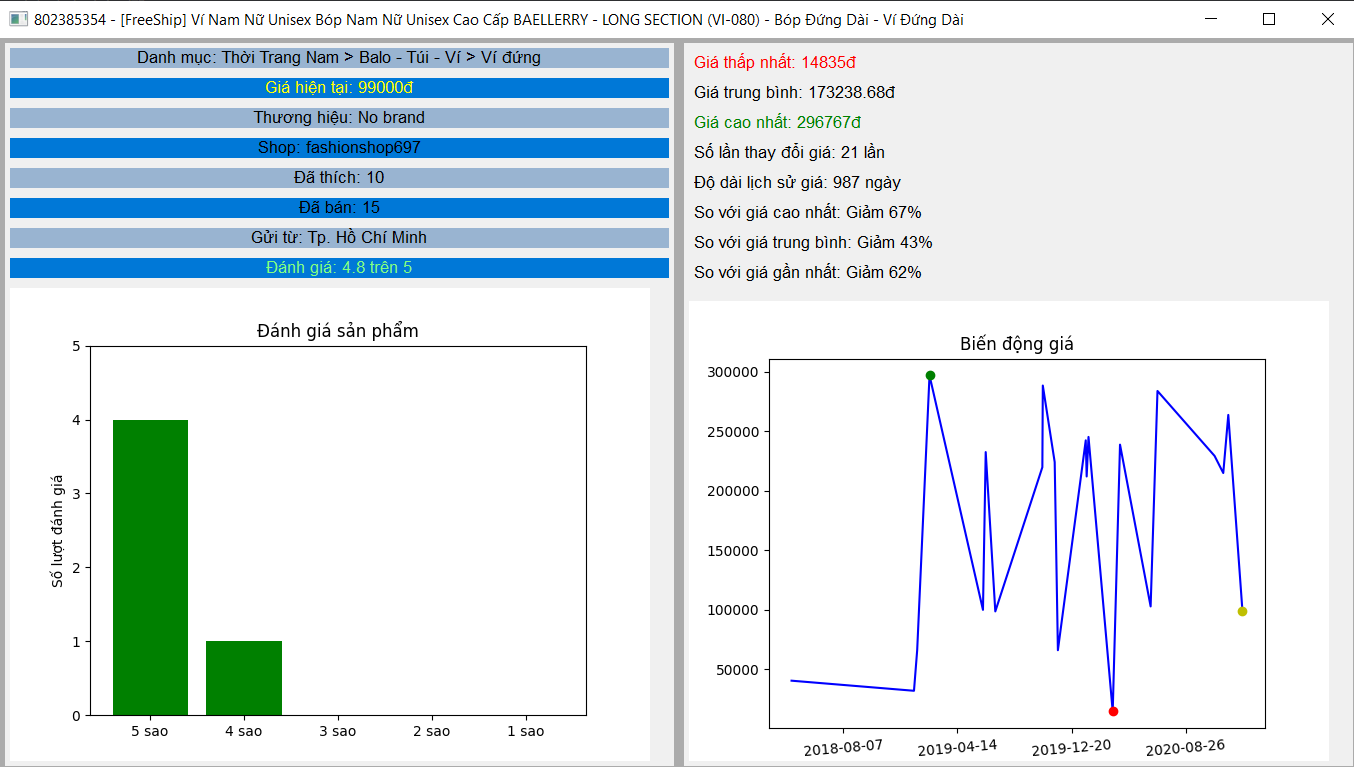
* **Sử dụng WxPython để tạo giao diện đơn giản**

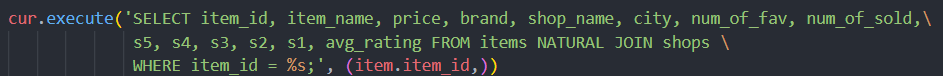
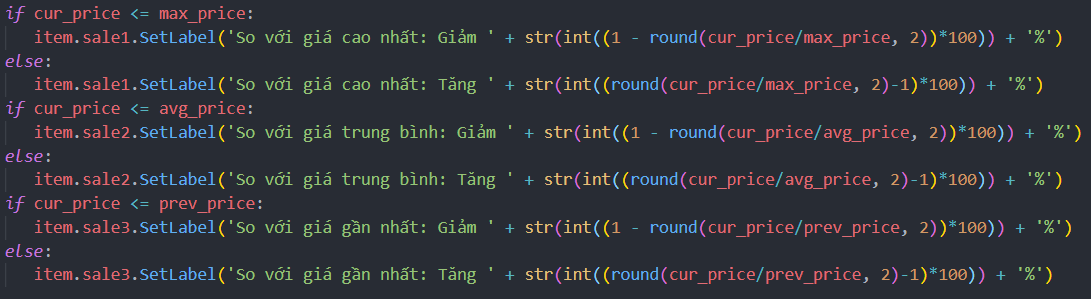
**c) Triggers**

Trigger được dùng để tự động cập nhật dữ liệu về: item\_history, đánh giá của sản phẩm (avg\_rating), số sản phẩm của shop (num\_of\_products) và đánh giá của shop (rating)

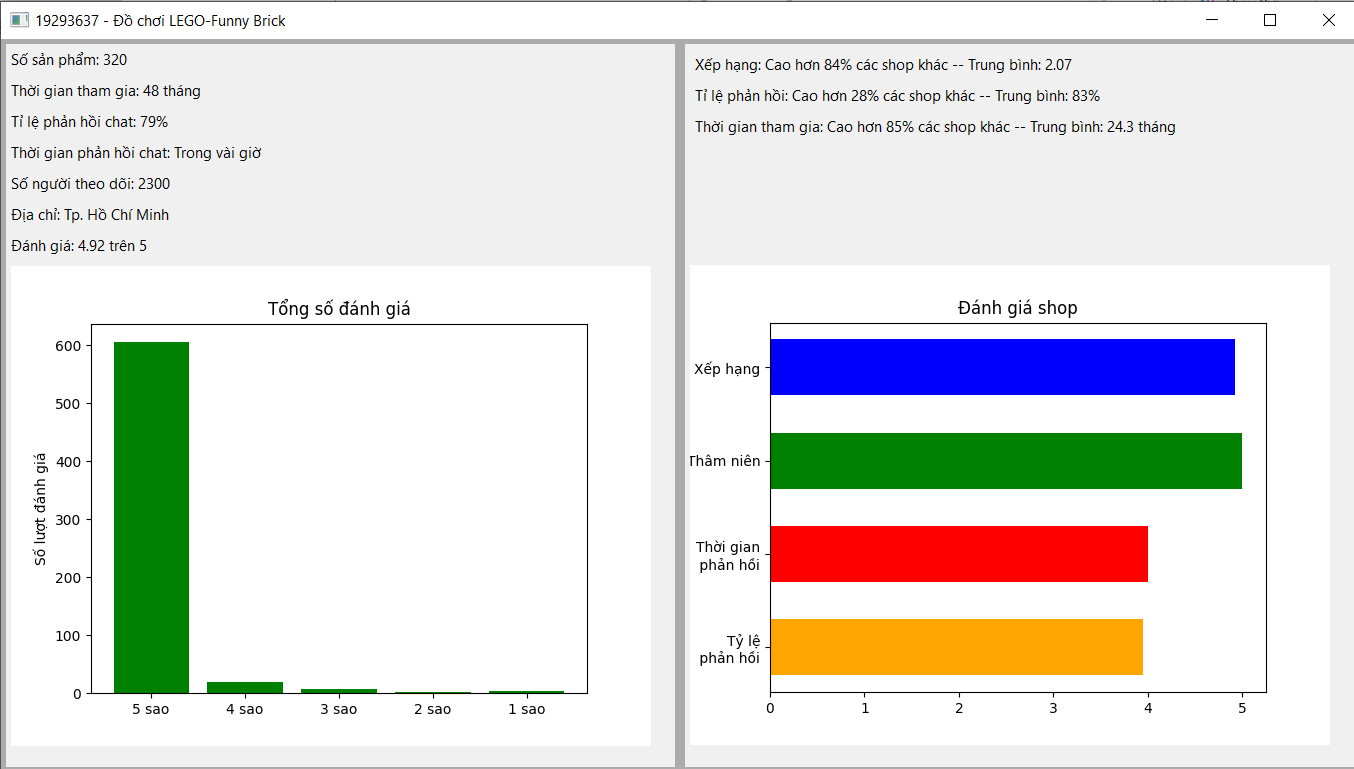
* insert\_into\_item\_history\_when\_insert\_item: Khi thêm một sản phẩm mới.
* insert\_into\_item\_history\_when\_update\_item\_price: Khi cập nhật lại giá của một sản phẩm.
* update\_item\_avg\_rating\_when\_insert\_update: Khi thêm mới hoặc khi cập nhật số lượng đánh giá (s1, s2, s3, s4, s5) của một sản phẩm.
* update\_shop\_num\_of\_products\_after\_delete\_item: Khi xóa một sản phẩm.
* update\_shop\_num\_of\_products\_after\_insert\_item: Khi thêm một sản phẩm.
* update\_shop\_rating\_after\_delete\_item: Khi xóa một sản phẩm.
* update\_shop\_rating\_after\_insert\_update\_item: Khi thêm một sản phẩm hoặc cập nhật số lượng đánh giá (s1, s2, s3, s4, s5) của một sản phẩm.

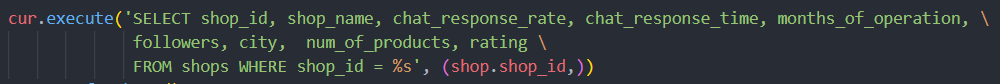
**d) Hàm chức năng**

**i. Hiển thị sản phẩm** ****

* Lấy thông tin sản phẩm: 
* Lấy tên danh mục sản phẩm (danh mục sản phẩm có tối đa 3 mức, dữ liệu sẽ được tiếp tục xử lý trong ứng dụng)
* Lấy thông tin về giá thấp nhất, giá trung bình, giá cao nhất, số lần thay đổi giá và độ dài lịch sử giá của sản phẩm:
  + Số lần thay đổi giá (Trong Item\_history bao gồm giá mỗi lần thay đổi của sản phẩm) = Tổng số bản ghi – 1
  + Độ dài lịch sử giá (tính từ khi thêm mới sản phẩm đến lần cập nhật giá cuối cùng) = Ngày cập nhật giá gần nhất – Ngày thêm mới sản phẩm
* So sánh giá sản phẩm so với giá cao nhất, trung bình và giá gần nhất của nó:

**ii. Hiển thị shop**



* Lấy thông tin của shop: 
* Lấy số lượng đánh giá để xây dựng biểu đồ:

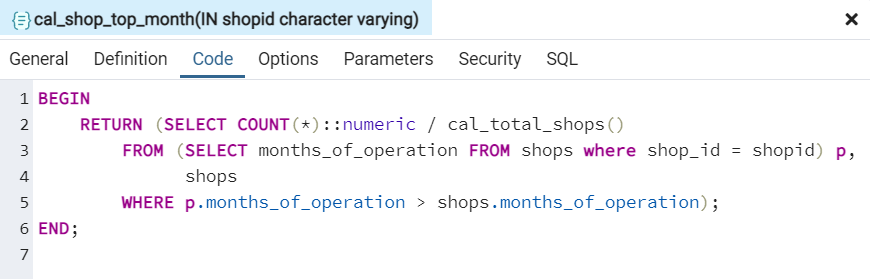
SELECT SUM(s5), SUM(s4), SUM(s3), SUM(s2), SUM(s1) FROM items WHERE shop\_id = {shop\_id};

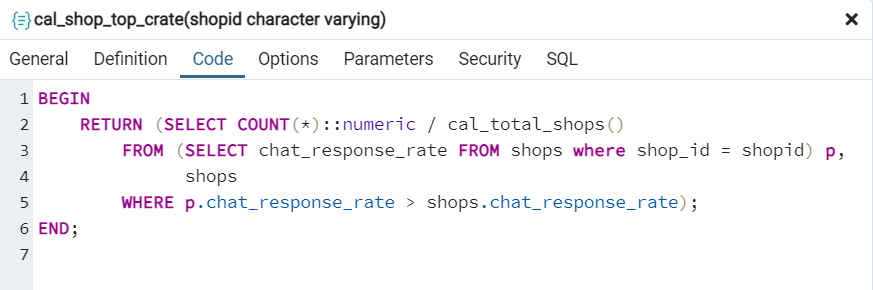
* Tính trung bình của Đánh giá (xếp hạng), Tỉ lệ phản hồi, Thời gian tham gia:

SELECT AVG(rating), AVG(chat\_response\_rate), AVG(months\_of\_operation) FROM shops WHERE shop\_id = {shop\_id};

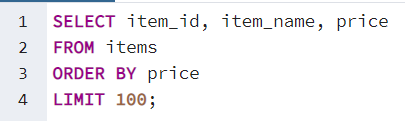
* cal\_shop\_top\_rating: Tính phần trăm số shop có đánh giá thấp hơn (Đếm số shop có đánh giá thấp hơn chia cho tổng số shop hiện lưu)



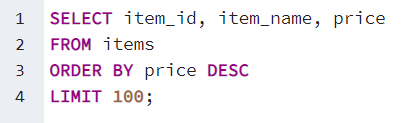
* cal\_shop\_top\_month: Tính phần trăm số shop có thời gian hoạt động thấp hơn (Đếm số shop có thời gian hoạt động thấp hơn chia cho tổng số shop hiện lưu) 
* cal\_shop\_top\_crate: Tính phần trăm số shop có Tỷ lệ phản hồi thấp hơn (Đếm số shop có Tỷ lệ phản hồi thấp hơn chia cho tổng số shop hiện lưu)

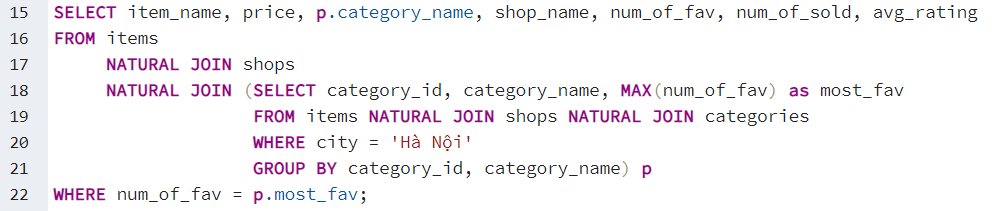


**iii. Hiển thị 100 sản phẩm giá thấp nhất**

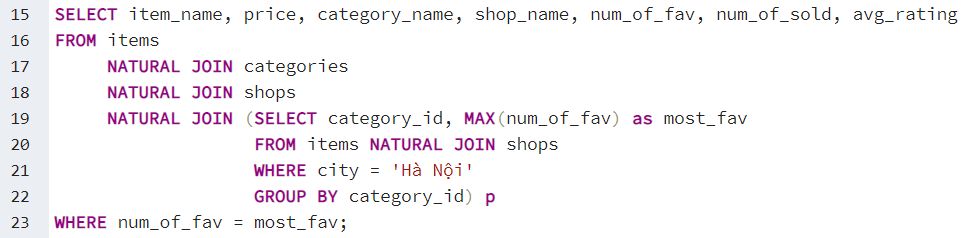
****

**Giá cao nhất**

****

**iv. Hiển thị các sản phẩm được yêu thích nhất của mỗi loại danh mục mặt hàng ở Hà Nội** 

190 ~ 220 ms – 972 rows affected

****

79 ~ 108 msec – 972 rows affected

Trần Hữu An

----

--Search with name

DROP FUNCTION search\_name(text);

CREATE OR REPLACE FUNCTION search\_name (x text)

RETURNS TABLE (item\_id varchar(20), item\_name varchar(200), price int, shop\_name varchar(50), brand varchar(20), favorite int, sold int, rate numeric)

AS $$

SELECT items.item\_id, items.item\_name, items.price, shops.shop\_name, items.brand, items.num\_of\_fav, items.num\_of\_sold, ROUND ((items.s5\*5+items.s4\*4+items.s3\*3+items.s2\*2+items.s1\*1)/(items.s5+items.s4+items.s3+items.s2+items.s1+0.1), 2) AS "rate"

FROM items, shops, item2

WHERE (items.shop\_id = shops.shop\_id) AND (items.item\_id = item2.item\_id) AND (item2.converttvkdau LIKE '%'||converttvkdau(LOWER(x))||'%')

GROUP BY items.item\_id, shops.shop\_name;

$$ LANGUAGE sql;

-----

SELECT \* FROM public.search\_name('cdjevbekr');

-----

----chuyen doi tieng Viet khong dau

CREATE OR REPLACE FUNCTION convertTVkdau (x text) RETURNS text AS

$$

DECLARE

codau text; kdau text; r text;

BEGIN

codau = 'áàảãạâấầẩẫậăắằẳẵặđéèẻẽẹêếềểễệíìỉĩịóòỏõọôốồổỗộơớờởỡợúùủũụưứừửữựýỳỷỹỵ';

kdau = 'aaaaaaaaaaaaaaaaadeeeeeeeeeeeiiiiiooooooooooooooooouuuuuuuuuuuyyyyy';

r = x;

FOR i IN 0..length(codau)

LOOP

r = replace(r, substr(codau,i,1), substr(kdau,i,1));

END LOOP;

RETURN r;

END;

$$ LANGUAGE plpgsql;

----

----

create table item2 as (select item\_id, converttvkdau(LOWER(items.item\_name)) from items);

CREATE OR REPLACE FUNCTION public.update\_into\_item2\_when\_insert\_item()

RETURNS trigger

AS $$

BEGIN

INSERT INTO item2

VALUES (NEW.item\_id, converttvkdau(LOWER(NEW.item\_name)));

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

----

---Search item with Shop

DROP FUNCTION search\_shop(text, text);

CREATE OR REPLACE FUNCTION search\_shop (x text, y text)

RETURNS TABLE (item\_id varchar(20), item\_name varchar(200), price int, shop\_name varchar(50), brand varchar(20), favorite int, sold int, rate numeric)

AS $$

SELECT \*

FROM public.search\_name(x) a

WHERE converttvkdau(LOWER(a.shop\_name)) LIKE '%'||converttvkdau(LOWER(y))||'%'

$$ LANGUAGE sql;

----

SELECT \* FROM search\_shop('ao khoac ni', 'thoi trang');

----

---Search item with price

DROP FUNCTION search\_price(text, int, int);

CREATE OR REPLACE FUNCTION search\_price (x text, min int, max int)

RETURNS TABLE (item\_id varchar(20), item\_name varchar(200), price int, shop\_name varchar(50), brand varchar(20), favorite int, sold int, rate numeric)

AS $$

SELECT \*

FROM public.search\_name(x) a

WHERE (a.price > min) AND (a.price < max)

ORDER BY a.price ASC;

$$ LANGUAGE sql;

----

SELECT \* FROM search\_price('ao khoac ni', 100000, 300000);

----

---Search with location

DROP FUNCTION search\_location(text, text);

CREATE OR REPLACE FUNCTION search\_location (x text, y text)

RETURNS TABLE (item\_id varchar(20), item\_name varchar(200), price int, shop\_name varchar(50), city varchar(20), brand varchar(20), favorite int, sold int, rate numeric)

AS $$

SELECT a.item\_id, a.item\_name, a.price, a.shop\_name, cities.city\_name, a.brand, a.favorite, a.sold, a.rate

FROM public.search\_name(x) a, shops, cities, items

WHERE (a.item\_id = items.item\_id) AND (shops.shop\_id = items.shop\_id) AND (shops.city\_id = cities.city\_id) AND (converttvkdau(LOWER(cities.city\_name)) LIKE '%'||converttvkdau(LOWER(y))||'%')

$$ LANGUAGE sql;

----

SELECT \* FROM search\_location ('truyen','Ha Noi');

----

---Search with brand

CREATE OR REPLACE FUNCTION search\_brand (x text, y text)

RETURNS TABLE (item\_id varchar(20), item\_name varchar(200), price int, shop\_name varchar(50), brand varchar(20), favorite int, sold int, rate numeric)

AS $$

SELECT \*

FROM public.search\_name(x) a

WHERE converttvkdau(LOWER(a.brand)) LIKE '%'||converttvkdau(LOWER(y))||'%'

$$ LANGUAGE sql;

----

SELECT \* FROM search\_brand('sac', 'pisen');

----

---Search item with sale (san pham dang giam gia)

CREATE OR REPLACE FUNCTION search\_sale (x text)

RETURNS TABLE (item\_id varchar(20), item\_name varchar(200), price int, shop\_name varchar(50), brand varchar(20), favorite int, sold int, rate numeric)

AS $$

SELECT \*

FROM public.search\_name(x) a

WHERE (get\_item\_prev\_price(a.item\_id)-a.price > 0)

ORDER BY a.price ASC

$$ LANGUAGE sql;

----

SELECT \* FROM search\_sale('quat');

----

CREATE OR REPLACE FUNCTION get\_item\_prev\_price(itemid character varying)

RETURNS bigint

AS $$

BEGIN

RETURN (SELECT price

FROM (SELECT ROW\_NUMBER() OVER (ORDER BY date DESC), price

FROM item\_history

WHERE item\_id = itemid) p

WHERE row\_number = 2);

END;

$$ LANGUAGE plpgsql;

----

Đoàn Anh Tuấn

--1. Danh sach nhung san pham thuoc loai x

create or replace function get\_item\_by\_categories(x text)

returns TABLE (item\_id varchar(20), item\_name varchar(200),price int, shop\_id varchar(20),brand varchar(20), num\_of\_fav int, num\_of\_sold int)

as $$

select i.item\_id, i.item\_name, i.price, i.shop\_id, i.brand, i.num\_of\_fav, i.num\_of\_sold FROM items i NATURAL JOIN categories WHERE category\_name = x;

$$ language sql;

select \* from get\_item\_by\_categories('Đồng hồ điện tử');

--2. Danh sach cac shop sap xep theo rating

create or replace function sort\_shops\_by\_rating()

returns table(shop\_id varchar(20), shop\_name varchar(50),num\_of\_product int, rating double precision)

as $$

SELECT shop\_id, shop\_name, num\_of\_products, rating

FROM items NATURAL JOIN shops

GROUP BY shop\_id, shop\_name, num\_of\_products, rating

ORDER BY rating DESC;

$$ language sql;

select \* from sort\_shops\_by\_rating();

--3. Chi ra shop co nhieu mat hang nhat tai moi tinh thanh

SELECT city\_name, shop\_id, shop\_name, sl\_items

FROM (

SELECT DISTINCT ON (city\_id) \*

FROM (

SELECT shop\_id, shop\_name, city\_id, COUNT(\*) AS sl\_items

FROM items NATURAL JOIN shops

GROUP BY shop\_id, shop\_name, city\_id

) a

ORDER BY city\_id, sl\_items DESC

) r NATURAL JOIN cities;

--4. chi ra nhung shop nao vua kinh doanh dong ho dien tu, vua kinh doanh may tinh bo tui

SELECT k1.shop\_id, k1.shop\_name

FROM

(

SELECT item\_id, shop\_id, category\_name, shop\_name

FROM

(

SELECT item\_id, shop\_id, category\_id, shop\_name

FROM items NATURAL JOIN shops

) e JOIN categories ON e.category\_id = categories.category\_id

WHERE category\_name = 'Đồng hồ điện tử'

) k1

JOIN

(

SELECT item\_id, shop\_id, category\_name, shop\_name

FROM

(

SELECT item\_id, shop\_id, category\_id, shop\_name

FROM items NATURAL JOIN shops

) e JOIN categories ON e.category\_id = categories.category\_id

WHERE category\_name = 'Máy tính bỏ túi'

) k2

ON k1.shop\_id = k2.shop\_id

GROUP BY k1.shop\_id, k1.shop\_name;

--5 dua ra cac shop dang hoat dong tai x

create or replace function get\_shop\_by\_cities(x text)

returns table (shop\_id varchar(20),shop\_name varchar(50),city\_name varchar(20))

as $$

select shop\_id, shop\_name,city\_name

from shops natural join cities

where city\_name = x;

$$ language sql;

select \* from get\_shop\_by\_cities('Hà Nội');

--6 dua ra tat ca san pham cua 1 shop co ten la x

create or replace function cal\_item\_by\_name\_shop(x text)

returns table (item\_id varchar(20), item\_name varchar(200), price int, brand varchar(20), category\_id varchar(10), num\_of\_fav int, num\_of\_sold int)

as $$

select i.item\_id, i.item\_name, i.price, i.brand, i.category\_id, i.num\_of\_fav, i.num\_of\_sold

from items i natural join shops

where shop\_name = x;;

$$ language sql;

select \* from cal\_item\_by\_name\_shop('HADA LABO');

--7.dua ra sp ban nhieu nhat cua tung shop

create or replace function get\_item\_max\_of\_sold\_each\_shop()

returns table (shop\_id varchar(20), item\_id varchar(20), item\_name varchar(200), price int, brand varchar(20), category\_id varchar(10),num\_of\_fav int,num\_of\_sold int)

as $$

select i.shop\_id, item\_id, item\_name, price, brand, category\_id ,num\_of\_fav,num\_of\_sold

from

items i join (select max(num\_of\_sold),shop\_id from items natural join shops group by shop\_id) as a on i.shop\_id = a.shop\_id

where max = num\_of\_sold;

$$ language sql;

select \* from get\_item\_max\_of\_sold\_each\_shop();

--8.tim kiem san pham thuoc 1 mat hang tai 1 thanh pho va co muc gia ...

create or replace function search\_item\_by\_etc(x text,y text,a int,b int)

returns table (item\_id varchar(20), item\_name varchar(200), shop\_id varchar(20), shop\_name varchar(50), category\_name varchar(50), city\_name varchar(20),price int, brand varchar(20), rating double precision)

as $$

select item\_id, item\_name, shop\_id, shop\_name, category\_name, city\_name,price, brand, rating

from items natural join shops natural join cities natural join categories

where category\_name = x and city\_name = y and price > a and price < b;

$$ language sql;

select \* from search\_item\_by\_etc('Kem và sữa dưỡng da','Tp. Hồ Chí Minh',300000,500000);

--9. thay doi ten cua 1 shop

create or replace procedure alter\_shop\_name(IN shopid varchar, x text)

language sql

as $body$

update shops

set shop\_name = x

where shop\_id = shopid;

$body$

select \* from shops where shop\_id = '22130260';

call alter\_shop\_name('22130260', 'Wear VN NEW');

--10 . dua ra 5 sp ban nhieu nhat cua 1 shop dc nhap

create or replace function get\_5\_item\_max\_sold\_a\_shop(x text)

returns table (item\_id varchar(20), item\_name varchar(200), shop\_name varchar (50), num\_of\_sold int)

as $$

select item\_id, item\_name, shop\_name, num\_of\_sold

from items natural join shops

where shop\_name = x

group by 1,2,3

order by 4 desc

limit 5;

$$ language sql;

select \* from get\_5\_item\_max\_sold\_a\_shop('hinminh');