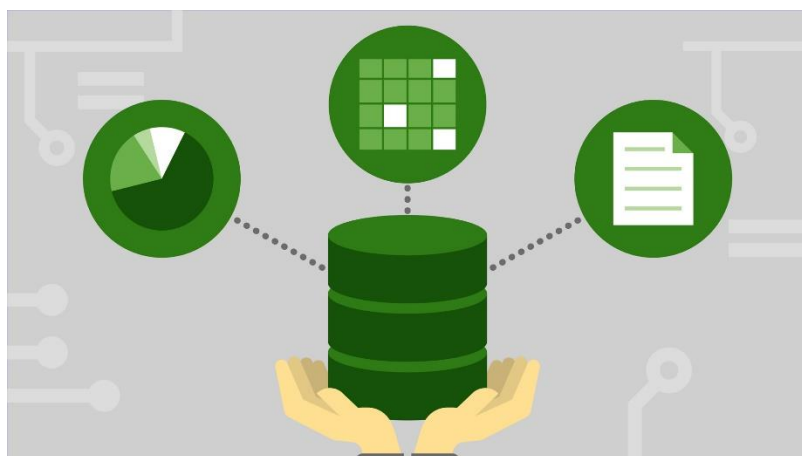


به نام خدا



دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده برق و کامپیوتر



آزمایشگاه پایگاه داده

دستورکار شماره ۵

شماره دانشجویی

۸۱۰۱۹۵۵۲۶

آبان ۹۹

پارسا صدری سینکی

گزارش فعالیت‌های انجام شده

توابع و تریگرها

مثال ۰: در این مثال بجای استفاده از CONSTRAINTS از RAISE EXCEPTION در یک تریگر فانکشن استفاده شده که در هنگام INSERT کردن در TABLE تریگر می شود.

```

CREATE FUNCTION person_bit()
AS $$
BEGIN
IF LENGTH(NEW.login_name) = 0 THEN
RAISE EXCEPTION 'Login name must not be empty.';
END IF;

IF POSITION(' ' IN NEW.login_name) > 0 THEN
RAISE EXCEPTION 'Login name must not include white space.';
END IF;
RETURN NEW;
END;
$$;

INSERT INTO person VALUES ('', 'Felonious Erroneous');
INSERT INTO person VALUES ('space man', 'Major Tom');

```

SQL Error [P0001]: ERROR: Login name must not include white space.
Where: PL/pgSQL function person_bit() line 8 at RAISE

مثال ۱: در این مثال جدولی دیگر ایجاد شد برای ذخیره لاگ ها که این کار توسط اضافه کردن کوئری INSERT به این جدول هنگام صدا زده شدن تریگر فانکشنی که در مثال قبل صدا می زدیم. به این صورت در هنگام INSERT یا UPDATE عملیات لاگ می شود و برای DELETE هم همین تریگر و تریگر فانکشن را دوباره می نویسیم برای DELETE این بار با این تفاوت که در تریگر فانکشن دیگر چک های مربوط INSERT و UPDATE را نداریم.

برای ساخت خود جدول لاگ هم نکته ای که وجود دارد دو ستونی هست که با داده پیش فرض پر می شوند. یکی توسط now() که زمان آن لحظه را ثبت می کند و یکی هم توسط session_user که user_id را ثبت می کند.

```

CREATE TRIGGER person_bdt
BEFORE DELETE ON person
FOR EACH ROW EXECUTE PROCEDURE person_bdt();

INSERT INTO person VALUES ('dfunny', 'Doug Funny');
INSERT INTO person VALUES ('pmayo', 'Patti Mayonnaise');
- SELECT * FROM person;
- SELECT * FROM person_audit;
- UPDATE person SET display_name = 'Doug Yancey Funny' WHERE login_name = 'dfunny';
- SELECT * FROM person;
- SELECT * FROM person_audit ORDER BY effective_at;
- DELETE FROM person WHERE login_name = 'pmayo';
- SELECT * FROM person;
- SELECT * FROM person_audit ORDER BY effective_at;

```

person_audit

login_name	display_name	operation	effective_at	userid	abstract
dfunny	Doug Funny	INSERT	2020-11-19 19:33:22	postgres	
pmayo	Patti Mayonnaise	INSERT	2020-11-19 19:33:30	postgres	
dfunny	Doug Yancey Funny	UPDATE	2020-11-19 19:35:33	postgres	
pmayo	Patti Mayonnaise	DELETE	2020-11-19 19:36:25	postgres	

مثال ۲: در این مثال ستونی برای ذخیره متن اضافه شد و یک ستون دیگر برای جستجو بر روی این متن. ستون مربوط به جستجو نوع داده TSVECTOR را دارد و در تریگر هنگام اضافه شدن متن به جدول توسط تابع `to_tsvector` آماده برای جستجو کردن می شود.

```

END IF;

IF POSITION(' ' IN NEW.login_name) > 0 THEN
    RAISE EXCEPTION 'Login name must not include white space.';
END IF;

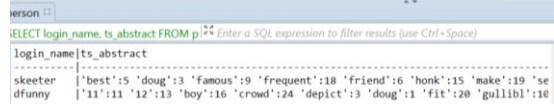
INSERT INTO person_audit (login_name, display_name, operation, abstract)
VALUES (NEW.login_name, NEW.display_name, TG_OP, NEW.abstract);

SELECT to_tsvector(NEW.abstract) INTO NEW.ts_abstract;

RETURN NEW;
END;
$$;

UPDATE person SET abstract = 'Doug is depicted as an introverted, quiet, insecure and gullible';
SELECT login_name, ts_abstract FROM person;

```



login_name	ts_abstract
skeeter	'best':5 'doug':3 'famous':9 'frequent':18 'friend':6 'honk':15 'make':19 'se
dfunny	'11':11 '12':13 'boy':16 'crowd':24 'depict':3 'doug':1 'fit':20 'gullible':10

مثال ۳: در این مثال چون ستون مربوط به جستجو human readable نیست توسط ایجاد یک view آن را از دسترسی عادی حذف کردیم. در این مثال هدف این است که نشان بدهیم هنگام مثلا INSERT کردن بر روی view هنوز تریگر تعریف شده ما به صورت عادی عمل می کند و نتیجه لاگ می شود.

```

UPDATE person SET abstract = 'Doug is depicted as an introverted, quiet, insecure and gullible';
SELECT login_name, ts_abstract FROM person;

/* Example 3 */

CREATE VIEW abridged_person AS SELECT login_name, display_name, abstract FROM person;

INSERT INTO abridged_person VALUES ('skeeter', 'Mosquito Valentine', 'Skeeter is Doug''s best friend');

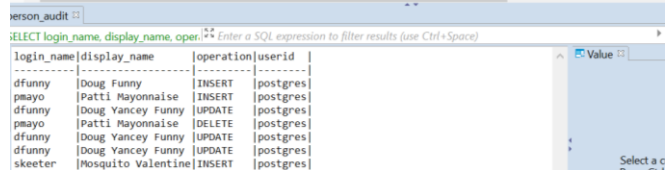
SELECT login_name, ts_abstract FROM person WHERE login_name = 'skeeter';

SELECT login_name, display_name, operation, userid FROM person_audit ORDER BY effective_at;

/* Example 4 */

CREATE TABLE transaction (

```




login_name	display_name	operation	userid
dfunny	Doug Funny	INSERT	postgres
pmayo	Patti Mayonnaise	INSERT	postgres
dfunny	Doug Vancey Funny	UPDATE	postgres
pmayo	Patti Mayonnaise	DELETE	postgres
dfunny	Doug Vancey Funny	UPDATE	postgres
dfunny	Doug Vancey Funny	UPDATE	postgres
skeeter	Mosquito Valentine	INSERT	postgres

مثال ۴: در این مثال جدولی برای transaction ایجاد شد که در آن خرید و فروش های شخص ثبت می شود و به موازات آن یک ستون به جدول قبلی به عنوان balance اضافه شد تا مقدار پولی که کاربر در لحظه دارد را نگه دارد. محاسبه balance به این صورت انجام شد که با ایجاد یک تریگر روی جدول transaction که هنگام INSERT کردن بر روی آن balance محاسبه می شود و در جدول اصلی UPDATE می شود.

نکته این مثال این است که در تریگر فانکشن ابتدا ما balance را UPDATE می کنیم و بعد چک های لازم را برای منفی نشدن و نبودن پول انجام می دهیم. که چون کل تریگر فانکشن به صورت یک transaction دیتابیس انجام می شود مشکلی ایجاد نمی کند.

```
-- CREATE TRIGGER transaction_bit
BEFORE INSERT ON transaction
FOR EACH ROW EXECUTE PROCEDURE transaction_bit();

-- SELECT login_name, balance FROM person WHERE login_name = 'dfunny';
-- INSERT INTO transaction (login_name, post_date, description, credit, debit) VALUES ('dfunny', '2018-01-11', 'ACH CREDI
-- SELECT login_name, balance FROM person WHERE login_name = 'dfunny';
-- INSERT INTO transaction (login_name, post_date, description, credit, debit) VALUES ('dfunny', '2018-01-17', 'FOR:BGE F
-- SELECT login_name, balance FROM person WHERE login_name = 'dfunny';
-- INSERT INTO transaction (login_name, post_date, description, credit, debit) VALUES ('dfunny', '2018-01-17', 'FOR:BGE F
-- SELECT login_name, balance FROM person WHERE login_name = 'dfunny';
-- INSERT INTO transaction (login_name, post_date, description, credit, debit) VALUES ('dfunny', '2018-01-23', 'FOR: ANNE
-- SELECT login_name, balance FROM person WHERE login_name = 'dfunny';
```



login_name	balance
dfunny	\$1,686.19

مثال ۵: در این مثال ما با ایجاد یک ترگر بر روی view ساخته شده مانع تغییر مستقیم ستون balance می شویم و در کد هم مشاهده می کنید که balance تغییر نمی کند بعد از UPDATE انجام شده.

```
-- BEGIN
-- Disallow non-transactional changes to balance
NEW.balance = OLD.balance;
RETURN NEW;
END;
$$;

CREATE TRIGGER abridged_person_iut
INSTEAD OF UPDATE ON abridged_person
FOR EACH ROW EXECUTE PROCEDURE abridged_person_iut();

UPDATE abridged_person SET balance = '100000000.00';

SELECT login_name, balance FROM abridged_person WHERE login_name = 'dfunny';
```



login_name	balance
dfunny	\$1,686.19

توابع پنجره ای

1. در کوئری ۵ مشتری برتر بر هر کشور را از سال ۱۹۹۷ تا ۱۹۹۸ پیدا می کنیم.

```
with q as (
    select c.contact_name || ' - ' || c.contact_title as "customer",
    c.country as "country",
    sum(od.unit_price * od.quantity * (1-od.discount)) as "total_paid"
    from orders o
    inner join customers c on o.customer_id = c.customer_id
    inner join order_details od on o.order_id = od.order_id
    where order_date >= to_date('1997', 'YYYY') and order_date <
    to_date('1998', 'YYYY')
    group by "customer", "country"
)
select t.customer, t.country, t.total_paid
from (
    select *, row_number() over (partition by "country" order by
    total_paid
    desc) as "rank" from q
) as t
where "rank" <= 5;
```

customer	country	total_paid
Sergio Gutiérrez - Sales Representative	Argentina	1149.3999938964844
Yvonne Moncada - Sales Agent	Argentina	429.2000045776367
Patricio Simpson - Sales Agent	Argentina	237.99999618530273
Roland Mendel - Sales Manager	Austria	48096.26317733166
Georg Pipp - Sales Manager	Austria	9305.579979383649
Pascale Cartrain - Accounting Manager	Belgium	6137.480041248798
Catherine Dewey - Sales Agent	Belgium	5297.000019073486
Lúcia Carvalho - Marketing Assistant	Brazil	10132.76750496205
André Fonseca - Sales Associate	Brazil	8008.7849825024605
Mario Pontes - Accounting Manager	Brazil	6022.7699720579385
Paula Parente - Sales Manager	Brazil	4415.149964017421
Janete Limeira - Assistant Sales Agent	Brazil	4283.775011897087

2. در این کوئری ۵ کارمند برتر را در هر ماه پیدا می کنیم (کارمند برتر منظور کارمندی است که در آن ماه بیشترین فروش را داشته است).

```
with q as (
    select e.first_name || ' ' || e.last_name as "employee",
    to_char(order_date, 'month') as "month",
    sum(od.unit_price * od.quantity * (1-od.discount)) as "total_paid"
    from orders o
    inner join employees e on o.employee_id = e.employee_id
    inner join order_details od on o.order_id = od.order_id
    where order_date >= to_date('1997', 'YYYY') and order_date <
    to_date('1998', 'YYYY')
    group by "employee", "month"
)
select t.employee, t.month, t.total_paid
from (
    select *, row_number() over (partition by "month" order by
    total_paid
    desc) as "rank" from q
) as t
where "rank" <= 5;
```

employee	month	total_paid
Margaret Peacock	april	13475.989897621275
Andrew Fuller	april	13118.650159218163
Janet Leverling	april	10252.549891757964
Michael Suyama	april	9593.500004654526
Robert King	april	4986.962445914745
Margaret Peacock	august	16485.53998593986
Robert King	august	6706.5923932668575
Janet Leverling	august	5831.599937453866
Nancy Davolio	august	5104.69999063015
Laura Callahan	august	4493.7249852594
Janet Leverling	december	17636.658917688714

3. در این کوئری ۵ محصول پرفروش هر دسته را پیدا می کنیم.

```
with q as (
    select p.product_name as "product", c.category_name as "category",
           sum(od.unit_price * od.quantity * (1-od.discount)) as "total_cost"
    from products p
         inner join categories c on p.category_id = c.category_id
         inner join order_details od on p.product_id = od.product_id
         group by "product", "category"
)
select t.product, t.category, t.total_cost
from (
    select *, row_number() over (partition by "category" order by
    total_cost
    desc) as "rank" from q
) as t
where "rank" <= 5;
```

product	category	total_cost
Côte de Blaye	Beverages	141396.7356273254
Ipoh Coffee	Beverages	23526.699842727183
Chang	Beverages	16355.959905386866
Lakkalikööri	Beverages	15760.439892498254
Steeleye Stout	Beverages	13643.999849504233
Veggie-spread	Condiments	16701.095047264098
Sirop d'érable	Condiments	14352.599874171614
Louisiana Fiery Hot Pepper Sauce	Condiments	13869.8894459071
Northwoods Cranberry Sauce	Condiments	12771.999989151955
Gula Malacca	Condiments	9915.945213678779
Tarte au sucre	Confections	47234.969978504174