

به نام خدا



دانشگاه صنعتی امیر کبیر
(پلی تکنیک تهران)

دستور کار آزمایشگاه پایگاه داده

جلسه هفتم

امنیت پایگاه داده

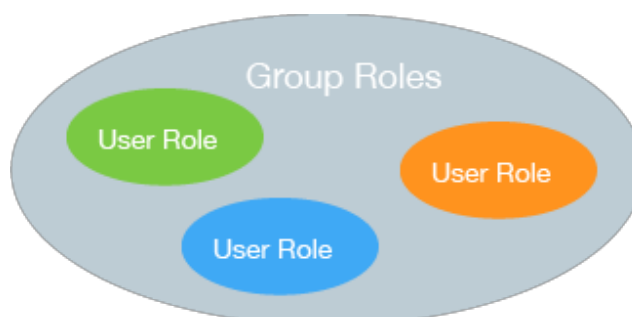
استاد درس

دکتر شهریاری

در این بخش به توضیح برخی مفاهیم مربوط به کاربران و مسائل امنیتی و نحوه‌ی کار با دستورات مربوط به آن پرداخته می‌شود.

ایجاد Role در PostgreSQL

در نسخه‌های جدیدتر PostgreSQL مفهومی به نام Role وجود دارد که عملکرد آن مانند گروه‌های کاربری و کاربر در نسخه‌های قدیمی آن و همچنین برخی از پایگاه‌داده‌های مشابه است. یک role می‌تواند شامل چند role دیگر باشد (مشابه گروه که شامل چندین کاربر است) و همچنین می‌توان به آن امکان login کردن داد (مشابه کاربر). در اصطلاح به role هایی شامل چند role دیگر هستند group role و به role هایی که امکان login را دارند اصطلاحاً login role گفته می‌شود.



برای ایجاد یک role از دستوری به شکل زیر استفاده می‌شود:

1. **CREATE ROLE** role_name;

توجه: با ایجاد یک role در سرور، تمامی دیتابیس‌های موجود در سرور به آن دسترسی دارند.

برای دیدن لیست تمامی role های موجود در سرور فعلی از دستور زیر استفاده می‌کنیم:

1. **SELECT** role_name **FROM** pg_roles;

توجه: role هایی که با pg_ شروع می‌شوند، توسط سیستم ایجاد شده‌اند.

با فرمان `\du` در `psql` نیز لیست تمامی `role` ها در سرور به همراه ویژگی‌های هر کدام را می‌توان نمایش داد. که خروجی آن به شکل زیر است:

List of roles		
Role name	Attributes	Member of
-----+	-----+	-----+
bob	Cannot login	{}
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	{}

هنگامی که یک `role` ساخته می‌شود می‌توان به آن لیستی از ویژگی‌ها (حقوق) را داد. برای این کار از دستوری به شکل زیر استفاده می‌کنیم:

1. **CREATE ROLE** name [**WITH**] option;

در دستور بالا عبارت `option` می‌تواند یک یا چند ویژگی باشد. مثلاً: `CREATEROLE`, `CREATEDB`, `SUPER` و ...

ایجاد role login

برای ساختن یک `role` به نحوی که امکان `login` کردن را داشته باشد از دستور زیر استفاده می‌کنیم:

```
1. CREATE ROLE name
2. LOGIN
3. PASSWORD 'password';
```

ایجاد superuser role

برای ساختن یک `role` به گونه‌ای که بدون محدودیت تمام دسترسی‌های پایگاه داده را داشته باشد از دستوری مشابه با دستور زیر استفاده می‌کنیم:

```
1. CREATE ROLE name
2. SUPERUSER
3. LOGIN
4. PASSWORD 'password';
```

توجه: برای اجرای دستور بالا شما باید درون یک role باشید که یک superuser باشد.

ایجاد یک role با دسترسی ساخت پایگاه داده

```
1. CREATE ROLE name
2. CREATEDB
3. LOGIN
4. PASSWORD 'password';
```

واسپاری حقوق به role

هنگامی که یک login role ساخته می‌شود، تنها امکان ورود به پایگاه داده را خواهد داشت ولی امکان ایجاد هیچ تغییری در این پایگاه داده را نخواهد داشت. برای مثال یک login user به صورت پیش‌فرض امکان انجام Select یا اجرای توابع را نخواهد داشت. برای فراهم سازی این قابلیت‌ها برای این role از دستور GRANT استفاده می‌نماییم. به طور مثال برای دادن دسترسی یک جدول به یک role به صورت زیر عمل می‌کنیم:

```
1. GRANT privilege_list
2. ON table_name
3. TO role_name;
```

در دستور بالا برای privilege_list می‌توان یک یا چند مورد از SELECT, INSERT, UPDATE, DELETE و ... استفاده کرد. همچنین برای دادن تمامی دسترسی‌های مربوط به این جدول عبارت ALL را وارد می‌کنیم.

اگر بخواهیم برخی از دسترسی‌های مربوط به تمامی جدول‌ها را بدهیم، در قسمت table_name عبارت ALL TABLES را وارد می‌کنیم. همچنین به جای اسم جدول از اسم هر شیء دیگری می‌توان استفاده کرد.

برای مثال برای دادن همه‌ی دسترسی‌های همه‌ی جدول‌های درون یک schema از دستوری به این شکل استفاده می‌شود:

```
1. GRANT ALL
2. ON ALL TABLES
3. IN SCHEMA "schema_name"
4. TO role_name;
```

سلب حقوق از role

برای لغو حقوقی که به یک role داده شده است، از دستور REVOKE استفاده می‌شود. به طور مثال برای لغو حقوق یک role در مورد یک یا چند جدول به صورت زیر عمل می‌شود:

1. **REVOKE** privilege_list
2. **ON TABLE** table_name
3. **FROM** role_name;

تغییر در role

برای ایجاد تغییرات مربوط به یک role از دستور ALTER ROLE به صورت زیر استفاده می‌شود:

1. **ALTER ROLE** role_name [**WITH**] option;

برای option می‌توان مقادیر جدول زیر را وارد کرد:

توضیح	option
مشخص می‌کند که آیا superuser هست یا خیر	SUPERUSER NOSUPERUSER
امکان ایجاد پایگاه داده	CREATEDB NOCREATEDB
امکان ایجاد role	CREATEROLE NOCREATEROLE
ارث بری از role های والد	INHERIT NOINHERIT
امکان login کردن	LOGIN NOLOGIN
آیا row level security را نادیده بگیرد یا خیر	BYPASSRLS NOBYPASSRLS
حداکثر تعداد اتصال همزمان	CONNECTION LIMIT limit
تغییر کلمه‌ی عبور	PASSWORD 'password' PASSWORD NULL
ایجاد محدودیت زمانی برای کلمه‌ی عبور	VALID UNTIL 'timestamp'

توجه: برای انجام تغییرات بالا یا باید از طریق `superuser` اقدام شود و یا توسط یک `role` با دسترسی `createrole` برای `role`هایی غیر از `superuser`.

سوال: Replicated Role چیست؟

برای تغییر نام یک `role` می‌توان از دستوری به شکل زیر استفاده کرد:

```
1. ALTER ROLE role_name
2. TO new_name;
```

توجه: برای تغییر نام باید از طریق `superuser` و یا یک `role` با دسترسی `createrole` اقدام کرد. همچنین امکان تغییر نام `role` فعلی وجود ندارد.

حذف role

به منظور حذف یک `role` به صورت زیر عمل می‌کنیم:

```
1. DROP ROLE [IF EXISTS] target_role;
```

توجه: برای حذف یک `superuser` باید از `superuser` دیگر و برای حذف یک `role` غیر از `superuser` باید از طریق یک `role` با دسترسی `createrole` اقدام کرد.

هنگام حذف یک `role` که از هرکدام از پایگاه داده‌های یک سرور به آن اشاره شده است با پیغام خطا مواجه می‌شویم. برای رفع این مورد ابتدا باید مالکیت اشیاء مربوط به این `role` را به یک `role` دیگر واگذار کرد و سپس مجوزهای مربوط به آن را لغو کرد. برای این منظور از دستوری به این شکل استفاده می‌کنیم:

```
1. REASSIGN OWNED BY target_role TO another_role;
2. DROP OWNED BY target_role;
3. DROP ROLE target_role;
```

عضویت در role

به منظور مدیریت بهتر و راحت تر role ها می توان از مفهوم group role استفاده کرد و با اعطا و لغو مجوز به یک گروه از انجام تک به تک تغییرات مشابه برای چندین role جلوگیری کرد. برای این کار یک role به عنوان گروه ایجاد کرده و تعدادی role را به عضویت آن درمی آوریم.

برای ایجاد یک group role دستور زیر را وارد می کنیم:

```
1. CREATE ROLE group_role_name;
```

برای عضویت یک role درون group role از دستور زیر استفاده می کنیم:

```
1. GRANT group_role TO user_role;
```

پس از اجرای دستورات بالا با اجرای دستور \du برای مثال با عضویت alice درون sales خروجی به شکل زیر نمایش داده می شود:

List of roles		
Role name	Attributes	Member of
alice		{sales}
api	1000 connections	{}
bob	Cannot login	{}
dba	Create DB	{}
john	Superuser	{}
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	{}
sales	Cannot login	{}

برای لغو عضویت یک role از یک group role از دستور زیر استفاده می کنیم:

```
1. REVOKE group_role FROM user_role;
```

امنیت در سطح سطر

در جدول‌ها مفهومی تحت عنوان row level security یا به اختصار RLS داریم که مجوز دسترسی را در سطح سطرهای یک جدول را فراهم می‌کند. برای اجرای این کار باید از دستور CREATE POLICY استفاده کرد. به عنوان مثال برای ایجاد سیاست روی جدول accounts به گونه‌ای که تنها اعضای گروه managers به سطرهای مربوط به خودشان در جدول دسترسی داشته باشند، دستورات زیر را وارد می‌کنیم:

```
1. ALTER TABLE accounts ENABLE ROW LEVEL SECURITY;  
2.  
3. CREATE POLICY account_managers  
4. ON accounts  
5. TO managers  
6. USING (manager = current_user);
```

توجه: خط اول دستورات بالا row level security را در جدول accounts فعال می‌کند. این مقدار به صورت پیش‌فرض غیرفعال است.

تزریق SQL

دسترسی‌هایی که برای جستجو و ایجاد تغییرات در پایگاه داده از طریق برنامه‌ی کاربردی به کاربران نهایی داده می‌شود می‌تواند باعث ایجاد خطر لو رفتن اطلاعات پایگاه داده و ایجاد اختلال در آن شود. در واقع برنامه‌ی کاربردی درخواست‌های کاربر را از طریق query به پایگاه داده ارسال می‌کند. هکر می‌تواند در فرم‌های سایت query مخربی وارد کند و سیستم در حین اجرا این کد SQL را اجرا می‌کند. به این عمل SQL Injection یا تزریق SQL گفته می‌شود. برای مثال هکر می‌تواند با تزریق دستور DROP TABLE باعث از بین رفتن اطلاعات شود و یا با SELECT اطلاعات کل پایگاه داده را دانلود کند.

به عنوان مثال در قسمتی که مربوط به پیگیری مرسولات کاربر است، چنین query وارد شده است:

```
1. SELECT * FROM orders WHERE order_id= ' user + input '
```

با وارد کردن یک کد SQL در قسمت input به جای شناسه‌ی کالا می‌توان تمامی نام کاربری و رمز عبور مشتریان را استخراج کرد:

```
1. SELECT * FROM orders WHERE order_id= '123 UNION SELECT username, password FROM customers --'
```


برای پیشگیری از این اقدامات چندین راه وجود دارد، از جمله:

- **اعتبارسنجی ورودی‌ها:** در هنگام ورود اطلاعات توسط کاربر، قبل از ارسال ورودی‌های کاربر به پایگاه داده، می‌توان ورودی را از یک فیلترینگ عبور داد تا در صورت وجود مقادیری مانند DROP TABLE یا SELECT * یا ۱=۱ درون ورودی از ارسال آن به پایگاه داده جلوگیری کرد.
- **استفاده از queryهای پارامتری شده:** در این روش با استفاده از دستور PREPARE می‌توان queryها را به دو بخش دستورات و پارامترهای ورودی تقسیم کرد و صرفاً از طریق ارسال پارامترها به عنوان مقدار ورودی درخواست را اجرا کرد. به عنوان مثال در دستوری که بالاتر به آن اشاره شد برای اجرای پارامتری شده‌ی آن می‌توان به صورت زیر عمل کرد:

```
1. PREPARE ordertrack (int) AS
2. SELECT * FROM orders WHERE order_id=$1
3. EXECUTE ordertrack(123);
```

- **حسابرسی در پایگاه داده:** در این روش با استفاده از logهای ذخیره شده در سیستم می‌توان فعالیت‌های مشکوک را مورد بررسی قرار داد و از آن‌ها جلوگیری کرد. برای مثال درخواست کاربر به یک جدول که اجازه‌ی دسترسی به آن را ندارد، می‌تواند به عنوان فعالیت مشکوک قلمداد شود. برای ایجاد چنین گزارش‌هایی می‌توان از افزونه‌ی pgaudit استفاده کرد.

همچنین برای جلوگیری از دسترسی کاربران به اطلاعات ساختاری پایگاه داده (مانند نام جدول‌ها) باید خطاهای پایگاه داده را پیش‌بینی کرد و از نمایش جزئیات خطا به کاربر جلوگیری کرد.

تمرین

۱- تفاوت دستور CREATE USER با دستور CREATE ROLE چیست؟

۲- اقدامات خواسته شده را به ترتیب انجام دهید و در هر مرحله کوئری‌ها و خروجی را در گزارش خود بیاورید:

- یک پایگاه داده ایجاد کنید.
- یک جدول با ستون‌های شماره پرسنلی، نام کامل و نام کاربری بسازید و ۵ سطر دلخواه در آن وارد کنید.
- یک superuser با نام دلخواه ایجاد کنید.
- از طریق superuser لاگین کنید.
- یک group role با مجوز ایجاد نقش و محدودیت زمانی گذرواژه تا "دوم مارس ۲۰۳۰" با نام دلخواه ایجاد کنید.
- دو نقش با نام‌های testrole1 و testrole2 و مجوز login ایجاد کنید.
- دو نقش ایجاد شده را عضو group role ایجاد شده در مراحل قبل کنید.
- به testrole1 مجوز نادیده گرفتن RLS و مجوز ارت بری را بدهید.
- نام testrole2 را به newtstrole2 تغییر دهید.
- وارد testrole1 شوید و تمامی سطرهای جدول ایجاد شده را select کنید. (تصویر خروجی را در گزارش خود بیاورید.) اگر با پیغام خطا مواجه شدید راه حل آن را ذکر کنید و دستورات مربوطه را وارد کنید.
- لیست نقش‌های ساخته شده به همراه ویژگی‌هایشان را نمایش دهید.
- مجدداً وارد superuser شوید و برای سطرهای جدول به صورت دلخواه یک RLS ایجاد کرده و آن را روی جدول فعال کنید.
- سیاست ساخته شده روی کدامیک از role‌های موجود محدودیت ایجاد می‌کند؟ مختصراً توضیح دهید.
- در انتها newtstrole2 را حذف کنید.

۳- در صورتی که در پایگاه داده‌ی خود تنها یک superuser ایجاد کنیم و دسترسی‌ها را بدون ایجاد تغییرات در ساختار role‌ها در back-end کنترل کنیم، منجر به چه مخاطرات امنیتی در پایگاه داده می‌شود؟ مثال بزنید.