

## 8장. Database Role 관리

PostgreSQL은 Role이라는 개념을 사용하여 데이터베이스 액세스 권한을 관리합니다. Role은 데이터베이스가 설정된 방법에 따라 데이터베이스 사용자 또는 데이터베이스 사용자 그룹으로 생각할 수 있습니다. Role의 개념은 "사용자" 및 "그룹"의 개념을 포함합니다. PostgreSQL 8.1 이전 버전에서 사용자와 그룹은 별개의 엔티티였지만 이제는 Role만 있습니다. 모든 Role은 사용자, 그룹 또는 양쪽 모두로 작용할 수 있습니다. LOGIN 속성이 있는 Role은 "Database User"와 동일합니다.

### 8.1 Role 속성

데이터베이스 role은 권한을 정의하는 속성이 다수 있으며, 클라이언트 인증 시스템과 인터랙션합니다.

- 로그인 권한

데이터베이스 연결을 위한 초기 role 이름으로 LOGIN 속성이 있는 role만 사용할 수 있습니다. LOGIN 속성이 있는 role은 “데이터베이스 사용자”와 동일한 것으로 간주될 수 있다. 로그인 권한이 있는 role을 생성하려면 다음 중 하나를 사용해야 합니다.

```
db=# CREATE ROLE name LOGIN;
```

```
db=# CREATE USER name;
```

(CREATE USER는 디폴트로 LOGIN 권한을 준다는 점에서 CREATE ROLE과 다릅니다.)

- Superuser 상태

데이터베이스 superuser는 로그인 권한을 제외한 모든 권한 검사를 건너 뛴다. 이 권한은 위험하며, 무심코 사용해서는 안 된다. 작업 대부분은 superuser 이외의 다른 role로 수행하는 것이 좋습니다. 새 데이터베이스 superuser를 생성하려면 CREATE ROLE name SUPERUSER를 사용해야 합니다. superuser인 role로 이것을 수행해야 합니다.

- 데이터베이스 생성

데이터베이스를 생성하려면 권한이 명시적으로 role에 주어져야 합니다(모든 권한

검사를 건너 뛰는 **superuser**인 경우 제외). 이러한 **role**을 생성하려면 **CREATE ROLE name CREATEDB**를 사용해야 합니다.

- **role** 생성

**role**을 추가적으로 생성하려면 권한이 명시적으로 **role**에 주어져야 합니다(모든 권한 검사를 건너 뛰는 **superuser**인 경우 제외). 이러한 **role**을 생성하려면 **CREATE ROLE name CREATEROLE**을 사용해야 합니다. **CREATEROLE** 권한이 있는 **role**은 다른 **role**을 변경 및 삭제할 수 있으며, 멤버십을 부여 또는 취소할 수도 있습니다. 단, **superuser** **role**의 멤버십을 생성, 변경(**alter**), 삭제 또는 변경(**change**)하려면 **superuser** 상태가 필요합니다. **CREATEROLE**로는 부족합니다.

- 복제 초기화

**streaming replication**을 초기화하려면 권한이 명시적으로 **role**에 주어져야 합니다(모든 권한 검사를 건너 뛰는 **superuser**인 경우 제외). **streaming replication**에 사용되는 **role**은 항상 **LOGIN** 권한이 있어야 합니다. 이러한 **role**을 생성하려면 **CREATE ROLE name REPLICATION LOGIN**을 사용해야 합니다.

- 비밀번호

패스워드는 데이터베이스에 연결할 때 사용자가 패스워드를 입력해야 하는 클라이언트 인증 방법인 경우에만 중요합니다. **password** 및 **md5** 인증 방법은 패스워드를 이용합니다. 데이터베이스 패스워드는 운영 체제 사용자 패스워드와 구분됩니다. **role** 생성 시 패스워드 지정은 **CREATE ROLE name PASSWORD 'string'**을 사용해야 합니다.

## 8.2 Role 멤버십

권한 관리의 편의상 사용자를 그룹으로 묶는 것이 편리할 수 있습니다. 이렇게 하면 권한을 그룹 단위로 부여하거나 취소할 수 있습니다. 그룹을 나타내는 **role**을 생성한 다음, 그룹 **role**의 멤버십을 개별 사용자 **role**에 부여하면 됩니다. 그룹 **role**을 설정하려면 먼저 **role**을 생성해야 합니다. 일반적으로 그룹으로 사용되는 **role**은 **LOGIN** 속성이 없으며, 원하면 설정은 할 수 있습니다.

그룹 **role**이 존재하는 경우 **GRANT** 및 **REVOKE** 명령을 사용하여 멤버를 추가 및 삭제할 수 있습니다.

```
db=# GRANT group_role TO role1, ... ;
db=# REVOKE group_role FROM role1, ... ;
```

다른 그룹 **role**에도 멤버십을 부여할 수 있습니다.(그룹 **role**과 비 그룹 **role** 사이에 실제로는 구분이 없음). 데이터베이스는 순환식 멤버십 루프의 설정을 허용하지 않습니다. 또한, **role**의 멤버십을 **PUBLIC**에 부여하는 것도 허용하지 않습니다.

그룹 **role**의 멤버는 두 가지 방법으로 **role**의 권한을 사용할 수 있습니다.

첫째, 그룹의 모든 멤버는 명시적으로 **SET ROLE**을 수행하여 일시적으로 그룹 **role**이 “됩니다”. 이 상태에서 데이터베이스 세션은 원래의 로그인 **role**이 아닌 그룹 **role**에 대한 액세스 권한을 가지며, 생성된 데이터베이스 오브젝트는 로그인 **role**이 아닌 그룹 **role**이 소유하는 것으로 간주됩니다.

둘째, **INHERIT** 속성이 있는 멤버 **role**은 해당 **role**에서 상속된 모든 권한을 비롯하여 멤버로서 **role**의 권한을 자동으로 갖습니다. 예를 들면, 다음을 실행했다고 가정하면,

```
db=# CREATE ROLE joe LOGIN INHERIT;
db=# CREATE ROLE admin NOINHERIT;
db=# CREATE ROLE wheel NOINHERIT;
db=# GRANT admin TO joe;
db=# GRANT wheel TO admin;
```

**joe role**로 연결한 직후에 데이터베이스 세션은 **joe**에 직접 부여된 권한 외에도, **joe**는 **admin**의 권한을 “상속 받기” 때문에 **admin**에 부여된 권한도 사용합니다. 그러나, **joe**가 간접적으로 **wheel**의 멤버지만 이 멤버십은 **NOINHERIT** 속성을 갖는 **admin**을 통한 것이므로 **wheel**에 부여된 권한은 사용할 수 없습니다.

```
db=# SET ROLE admin;
```

위 명령을 실행하면, 세션은 **admin**에 부여된 이러한 권한만 사용하고 **joe**에 부여된 권한은 사용하지 않습니다.

```
db=# SET ROLE wheel;
```

위 명령을 실행하면, 세션은 **wheel**에 부여된 이러한 권한만 사용하고 **joe** 또는 **admin**에 부여된 권한은 사용하지 않습니다. 원래의 권한 상태는 다음 중 하나를 사용하면 복원됩니다.

```
db=# SET ROLE joe;
db=# SET ROLE NONE;
db=# RESET ROLE;
```

role 속성 LOGIN 및 SUPERUSER, CREATEDB, CREATEROLE은 특수한 권한으로 생각될 수 있지만 데이터베이스 오브젝트의 일상적인 권한으로 상속되지 않습니다. 속성을 사용하려면 이러한 속성 중 하나를 보유한 특정 role에 실제로 SET ROLE을 해야 합니다. 위의 예시에 이어서, CREATEDB 및 CREATEROLE을 admin role에 부여할 수도 있습니다. 그러면, joe role로 연결하는 세션은 이러한 권한을 즉각 갖지는 못하며, SET ROLE admin을 수행한 이후에만 권한이 부여됩니다.

그룹 role을 소멸하려면 DROP ROLE을 사용해야 합니다.

```
db=# DROP ROLE name;
```

그룹 role의 멤버십이 자동 취소됩니다(단, 멤버 role은 영향을 받지 않음). 그룹 role이 소유한 오브젝트를 먼저 삭제하거나 다른 소유자에게 재할당해야 하며, 그룹 role에 부여된 권한은 취소해야 한다는 점을 유의해야 합니다.

## 8.3 Role 조회

PSQL에서 아래의 명령어로 Role 조회가 가능합니다.

```
postgres=# \du
```

List of roles

Role name	Attributes	Member of
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	{}

아래의 Catalog View를 통해 Role 조회가 가능합니다.

```
select * from pg_user ;
```

```
select * from pg_shadow ;
```

```
Select * from pg_roles;
```

## 8.4 Role 생성

### 8.4.1 Syntax

```
CREATE USER username [[ WITH ] option [ ... ]]
```

SUPERUSER | NOSUPERUSER : SUPERUSER 권한 (Default : NOSUPERUSER)

CREATEDB | NOCREATEDB : DATABASE 생성 권한 (Default: NOCREATEDB)

CREATEUSER | NOCREATEUSER : 새로운 User를 생성하는 권한 (Default: NOCREATEUSER)

INHERIT | NOINHERIT : Database의 권한을 다른 구성원들에게 상속하는 역할 (Default: INHERIT)

LOGIN | NOLOGIN : LOGIN 역할 부여 (Default: NOLOGIN)

[ENCRYPTED | UNCRYPTED] PASSWORD 'password' : 'password'를 입력하고 인증이 필요 없는 경우 옵션 생략 가능

### 8.4.2 Example

```
CREATE USER test1; (동일) CREATE ROLE test1 LOGIN;
```

```
CREATE USER test2 with PASSWORD 'test2';
```

```
CREATE USER test3 with PASSWORD 'test3' SUPERUSER;
```

## 8.5 Role 삭제

### 8.5.1 Syntax

```
CREATE USER username [[ WITH ] option [ ... ]]
```

### 8.5.2 Example

```
Drop user test1; (동일) Drop role test1;
```

```
Drop role test2, test3;
```

```
Drop role if exists test3;
```

## 8.6 Role 변경

### 8.6.1 Syntax

```
ALTER ROLE role_specification [ WITH ] option [ ... ]
```

### 8.6.2 Example

```
# Change a role's password
ALTER ROLE davide WITH PASSWORD 'hu8jmn3';

# Remove a role's password
ALTER ROLE davide WITH PASSWORD NULL;

# Change a password expiration date
ALTER ROLE chris VALID UNTIL 'May 4 12:00:00 2015 +1';

# Make a password valid forever
ALTER ROLE fred VALID UNTIL 'infinity';

# Give a role a non-default setting
ALTER ROLE worker_bee SET maintenance_work_mem = 100000 ;

# Give a role a default setting
ALTER ROLE worker_bee RESET maintenance_work_mem ;

# Give a role a non-default, database-specific setting
ALTER ROLE fred IN DATABASE devel SET client_min_messages = DEBUG;
ALTER ROLE fred IN DATABASE devel SET client_min_messages FROM CURRENT ;
ALTER ROLE fred IN DATABASE devel RESET ALL ;
ALTER ROLE name RENAME TO new_name ;
```