11장. Schema 관리

데이터베이스 클러스터에는 하나 이상의 데이터베이스가 포함됩니다. Role 및 일부 다른 유형의 오브젝트는 클러스터 전체에서 공유됩니다. 서버에 연결된 클라이언트 연결은 단일 데이터베이스, 즉 연결 요청에 지정된 데이터베이스의 데이터에만 액세스 할 수 있습니다. 데이터베이스에는 하나 이상의 Schema가 포함되고, Schema에는 테이블이 포함됩니다. Schema에는 데이터 형식, 함수 및 연산자와 같은 다른 오브젝트들도 포함됩니다. 동일한 오브젝트명을 다른 Schema에서 사용해도 충돌이 발생하지 않습니다. 예를 들어, schema1과 myschema Schema 모두에 mytable이라는 테이블이 포함될 수 있습니다. Schema는 데이터베이스와 달리 엄격 하게 분리되어 있지 않으므로 사용자는 권한이 있으면 연결된 데이터베이스의 모든 Schema 오브젝트에 액세스 할 수 있습니다.

Schema 사용을 선호하는 몇 가지 이유가 있습니다.

- 하나의 데이터베이스를 여러 사용자가 서로 간섭하지 않고 사용할 수 있도록 합니다.
- 관리하기 쉽도록 데이터베이스 오브젝트를 논리 그룹으로 구성합니다.
- 타사 응용 프로그램을 별도의 Schema에 넣어 다른 오브젝트명과의 충돌을 방지합니다.

Schema는 중첩할 수 없다는 점을 제외하고 운영 체제 디렉토리와 유사합니다.

11.1 Schema 생성

Schema를 작성하려면 CREATE SCHEMA 명령을 사용하십시오. Schema에 자유롭게 이름을 지정합니다. 예시는 다음과 같습니다.

```
CREATE SCHEMA myschema;
```

Schema에 오브젝트를 작성하거나 액세스하려면 Schema 이름과 테이블 이름을 점으로 구분된 이름을 작성하십시오.

```
schema.table
```

Schema에 테이블을 만드는 구문은 다음과 같습니다.

```
CREATE TABLE myshema.mytable(
…);
```

다른 사용자가 소유하는 Schema를 만들고 싶을 수 있습니다. 이를 위한 구문은 다음과 같습니다.

```
CREATE SCHEMA schema_name AUTHORIZATION user_name;
```

Schema 이름은 생략할 수도 있으며, 이 경우 Schema 이름은 사용자 이름과 동일합니다. pg_ 로 시작하는 Schema 이름은 시스템에서 사용하기 위해 예약되어 있으며 사용자가 만들 수 없습니다.

11.2 Schema 삭제

빈 Schema(모든 객체가 삭제된 Schema)를 삭제하려면 다음을 수행합니다.

```
DROP SCHEMA myschema;
```

Schema의 모든 오브젝트를 포함하여 Schema를 삭제하는 구문은 다음과 같습니다.

```
DROP SCHEMA myschema CASCADE;
```

11.3 public SCHEMA

앞에서 Schema 이름을 지정하지 않고 테이블을 만들었습니다. 기본적으로 이런 테이블(및 다른 오브젝트)은 자동으로 “public”이라는 Schema에 배치됩니다. 모든 새 데이터베이스에는 이러한 Schema가 포함되어 있습니다. 따라서 다음 두 구문은 동일합니다.

```
CREATE TABLE products(...) ;

또는

CREATE TABLE public.products(...)
```

11.4 Schema 검색 경로

검색 경로의 첫 번째 열거된 Schema를 현재 Schema라고 합니다. 현재 Schema는 검색되는 첫 번째 Schema이며 Schema 이름을 지정하지 않고 CREATE TABLE 명령으로 테이블을 작성한 경우 새 테이블이 작성되는 Schema입니다.

현재 검색 경로를 표시하는 구문은 다음과 같습니다.

```
SHOW search_path

search path
------------

“$user”, public
```

첫 번째 요소는 현재 사용자와 이름이 같은 Schema를 검색하도록 지정합니다. 이러한 Schema가 없으면 이 항목은 무시됩니다. 두 번째 요소는 앞에서 설명한 공용 Schema를 참조합니다. 존재하는 Schema 중 검색 경로에서 처음 나타나는 Schema는 새 오브젝트가 생성되는 기본 위치입니다. 이것이 디폴트로 오브젝트가 public schema에 작성되는 이유입니다. 오브젝트가 Schema 규정없이 다른 컨텍스트에서 참조되는 경우(테이블 변경, 데이터 변경 또는 조회 명령 등), 일치하는 오브젝트가 발견될 때까지 검색경로에서 탐색됩니다. 따라서 기본 구성에서는 정규화 되지 않은 액세스는 공용 Schema만 참조할 수 있습니다.

새 Schema를 경로에 추가하려면 다음을 수행합니다.

```
SET search_path TO myschema.public
```

그런 다음 Schema 규정 없이 테이블에 액세스합니다.

```
DROP TABLE mytable;
```

또, myschema 는 패스 내의 최초의 요소이므로, 새로운 오브젝트는 디폴트로 여기에 작성됩니다.

다음과 같이 쓸 수도 있습니다.

```
SET search_path TO myschema;
```

이렇게 하면 앞으로는 규정된 이름없이 public Schema에 액세스 할 수 없게 됩니다. public Schema는 기본적으로 존재한다는 것 외에 특별한 의미는 없습니다. 다른 Schema와 마찬가지로 삭제할 수도 있습니다.

11.5 Schema 권한

사용자는 기본적으로 소유하지 않은 Schema의 객체에 접근할 수 없습니다. 접근하려면 Schema 소유자에게 Schema USAGE권한을 부여해야 합니다. Schema의 오브젝트에 대해 작업을 수행하려면 해당 오브젝트에 따라 추가 권한이 필요할 수 있습니다.

다른 사용자의 Schema에서 오브젝트를 만들 수 있습니다. 이를 위해서는 Schema에서 CREATE권한이 부여되어야 합니다. 기본적으로 공용 Schema의 경우 모든 사용자에게 CREATE 및 USAGE 권한이 있습니다. 즉, 모든 사용자는 사용자가 연결할 수 있는 모든 데이터베이스의 공용 Schema에 오브젝트를 만들 수 있습니다.

```
REVOKE CREATE ON SCHEMA public FROM PUBLIC;
```

첫 번째 'public'은 Schema입니다. 두 번째 'public'은 모든 사용자를 의미합니다.

11.6 System Catalog SCHEMA

각 데이터베이스에는 public 및 사용자가 정의한 Schema외에 pg_catalog Schema가 포함되어 있습니다. 이 Schema에는 시스템 테이블과 모든 내장 데이터 유형, 함수 및 연산자가 포함됩니다. pg_catalog는 항상 검색 경로에 포함되어 있습니다. 경로에 명시적으로 나열되지 않은 경우 경로Schema를 검색하기 전에 암시적으로 검색됩니다. 이렇게 하면 기본 제공 이름을 항상 검색할 수 있습니다. 그러나 사용자 정의 이름으로 내장 이름을 덮어 쓰는 경우 pg_catalog를 경로의 끝에 명시 적으로 넣을 수 있습니다. 시스템 카탈로그의 이름은 pg_로 시작하므로 이러한 이름은 사용하지 않는 것이 좋습니다.