Mario Fentler 5CHIT

Benutzerverwaltung in PostgreSQL

In der Übung werden folgende Benutzerrollen angelegt, Berechtigungen erstellt und Beispielaufgaben gelöst. Überprüfen kann man die Berechtigungen in dem man das mit dem User versucht, was man ihm erlaubt/verboten hat.

Rollen:

- Kunde
  darf die Spalte "replacement_cost" **nicht sehen**
- Mitarbeiter
  Zahlungen einsehen und anlegen
- Admin
  Zahlungen einsehen und anlegen
  Zahlungen ändern & löschen
- Redakteur

Rollen erstellen

Eine Rolle in Postgres kann man wie einen User oder eine Gruppe sehen. Dieser Rolle kann man dann bestimmte Berechtigungen geben.
Zuerst muss man sich mit der Datenbank verbinden.

```
psql -h localhost -U Mario -W -d dvdrental
```

Anschließend können die Rollen erstellt werden. Bei Rollen kann man viele verschieden Optionen bei der Erstellung wählen.
Für Gruppenrollen werden meistens die "Login"-Option ausgelassen. Diese findet man bei User-Rollen.

```
CREATE ROLE Kunde;
CREATE ROLE Mitarbeiter;
CREATE ROLE Admin;
CREATE ROLE Redakteur;
```

Diese Rollen kann man Usern geben oder wieder wegnehmen mit folgenden Commands:

```
GRANT rolename TO username;
REVOKE rolename FROM username;
```

Accounts erstellen

In Postgres sind Rollen mit der "Login"-Option dasselbe wie User.
Man kann nun entscheiden wie man sie erstellt. Für das Beispiel wurde CREATE USER verwendet.

```
REATE USER kunde1 WITH PASSWORD 'kunde123';
GRANT Kunde TO kunde1;

CREATE USER mitarbeiter1 WITH PASSWORD 'ma123';
GRANT Mitarbeiter TO mitarbeiter1;

CREATE USER mitarbeiter2 WITH PASSWORD 'ma123';
GRANT Mitarbeiter TO mitarbeiter2;
GRANT Admin To mitarbeiter2;

CREATE USER admin1 WITH PASSWORD 'admin';
GRANT Admin TO admin1;

CREATE USER redakteur1 WITH PASSWORD 'red123';
GRANT Redakteur TO redakteur1;
```

Zugriffsberechtigungen

Daten aus den Tabellen lesen

```
GRANT SELECT ON ALL TABLES IN SCHEMA public TO <Rolle>; //Für alle Rollen
```

Zahlungen einsehen und anlegen

```
GRANT INSERT ON payment TO Admin;
GRANT INSERT ON payment TO Mitarbeiter;
```

Zahlungen ändern und löschen (nur Admin)

```
GRANT UPDATE ON payment TO Admin;
GRANT DELETE ON payment TO Admin;
```

Redakteur und Kunde dürfen die payment Tabelle nicht auslesen.

```
REVOKE SELECT ON payment FROM Kunde;
REVOKE SELECT ON payment FROM Redakteur;
```

Kunde darf eine Spalte nicht sehen -> Grant auf alle Spalten, die er sehen darf.

```
REVOKE SELECT ON film FROM Kunde;
```

```
CREATE SCHEMA film_read_only;
GRANT USAGE ON SCHEMA film_read_only TO Kunde;

CREATE VIEW film_read_only."film" AS SELECT
film_id,title,description,release_year,language_id,rental_duration,rental_rate,leng
th,rating,last_update,special_features,fulltext FROM film;
GRANT SELECT ON film_read_only."film" TO Kunde;
```

Der Kunde hat nun das Recht alles aus der Tabelle "film" zu lesen außer die Spalte "replacement_cost".
Dazu muss er den Select-Befehl aber so ausführen:

```
SELECT * FROM film_read_only."film";
```

Mit dem Befehl **\dp** kann man sich die Permissions anzeigen lassen. Alternativ kann man es auch einfach ausprobieren ob man die Rechte hat.

Berechtigungen über die Datei "pg_hba.conf"

Das File liegt hier, "/etc/postgresql/10/main/pg_hba.conf"
Verbindung zur Datenbank nur über ssl:

```
hostssl all all 0.0.0.0/0 md5
```

Bestimmte IP-Adressen nicht zulassen:

```
host all all shop-ip-addresse/sn-mask reject
```

View und Policy

Ein Marketing Mitarbeiter soll nur auf Kunden und ihre Emails Zugriff haben, die auch aktiv sind. (active=true)

View

Bei der View muss man aufpassen, da hier zwar select * steht, Einträge die allerdings danach erstellt nicht angezeigt werden.

```
CREATE VIEW customers_query AS SELECT * FROM customer WHERE active = 1;
SELECT * FROM customers_query;
```

Policy

Damit Policies auch auf der Tabelle wirken muss man das davor enablen

```
ALTER TABLE customer ENABLE ROW LEVEL SECURITY;
```

Dann wird sie erstellt

```
CREATE POLICY marketing_ma_pol ON customer FOR SELECT TO Kunde USING (active=1);
```

Wenn man nun selected, dann werden nur noch die Kunden angezeigt, die auf active=1 gesetzt sind.

```
SELECT * FROM customer;
SELECT * FROM customer WHERE active=0; //Jeder andere User bekommt hier was
angezeigt
```

Quellen

[1] https://www.postgresql.org/docs/9.1/static/index.html
[2] https://serverfault.com/questions/60508/grant-select-to-all-tables-in-postgresql
[3] https://www.postgresql.org/docs/9.1/static/sql-grant.html
[4] https://www.pg-forum.de/viewtopic.php?t=3785
[5] https://support.chartio.com/knowledgebase/limit-postgresql-user-access-using-schema
[6] https://askubuntu.com/questions/256534/how-do-i-find-the-path-to-pg-hba-conf-from-the-shell
[7] https://www.postgresql.org/docs/9.1/static/auth-pg-hba-conf.html
[8] https://www.postgresql.org/docs/9.5/static/sql-createpolicy.html