Dokumentation for at få adgang til min postgres database:

Du skal bruge psql for at connecte til databasen. Her er et link for at downloade postgres

https://www.postgresql.org/download/

Som nævnt, så har jeg oprettet en postgres database. Hertil vil jeg fremlægge nogle praktiske oplysninger for at få adgang:

Brugernavn: kristian

Password: your_password

Denne kommando vil forbinde til databasen og her bliver der spurgt om dit password.

psql -h aws-0-eu-central-1.pooler.supabase.com -p 5432 -d postgres -U kristian.wczlxxflscuntcyotivh

Brug af Postgres kommandoer:

For at få alle databaser frem:

\list

For at connecte til en database:

\c <databasenavn> (i dette tilfælde hedder databasen postgres)

\c postgres

For at få en oversigt over tabeller:

\dt

Her henter du adgang til de columns du har access til.

SELECT id, title, author FROM books;

Du kan også prøve at skrive:

```
SELECT * FROM books;
```

Dette vil give fejl, da du ikke har adgang til alt dataen.

Denne kommando gør at du kun kan opdatere author inde i databasen:

```
UPDATE books SET author = 'New Author Name test' WHERE id = 2;
```

Denne kommando inserter ny author. Du kan prøve at ændre author til f.eks. ”title”, hvilket vil give en fejl, da du kun har access til at update author.

```
INSERT INTO books (author) VALUES ('New Author Name');
```

Du kan prøve at skrive:

```
INSERT INTO books (title) VALUES ('New Author Name');
```

Dette giver også en fejl.

Undersøgelse i forhold til valg.

Forløbet med at vælge den rigtige database skulle undersøges. Inden undersøgelse havde jeg tre databaser oppe og vende fordi jeg har arbejdet med dem før:

MySQL
PostgreSQL
MongoDB

https://www.playerzero.ai/advanced/product-builder-facts/mysql-vs-postgresql
Denne artikel beskriver forskellen mellem MySQL og PostgreSQL i forhold til de sikkerhedsmæssige features. Det jeg kom frem til at postgres er bedre til granulær adgangskontrol.

Postgres gør det muligt for administratorer at definere sikkerhedspolitikker, der kan styre adgangen til indivduelle coluimns/rækker i en database tabel. Konklusionen var at postgres har en dybere og mere fleksibel tilgang til granulær adgang, hvilket er derfor jeg valgte dette.

MySQL har et system for brugergodkendelse og tilladelser, som giver dig mulighed for at styre, hvilke brugere der har adgang til hvilke databaser og tabeller.

Men når det kommer til de mere finjusterede adgangskontroller, såsom rækkeniveau-sikkerhed og mere komplekse sikkerhedspolitikker, har PostgreSQL traditionelt haft en føring:

Herudover brugte jeg også postgreSQL egen artikel til inspiration:
https://www.postgresql.org/docs/current/ddl-priv.html

https://feedback.mongodb.com/forums/924145-atlas/suggestions/39906208-granular-permissions
Denne artikel handler om at en bruger har lavet noget feedback til MongoDB, omkring at mongoDB ikke understøtter granular permission. Brugeren beskriver problemet med at han ikke kan lave nogen form for granulær kontrol over hvilke tilladelser der tildeles til hver bruger. Dette er en stor ulempe, da han giver sine kollegar mere adgang end nødvendigt. En admin fra MongoDB bekræfter dette d.4 marts 2024:

Fuat (Administrator, MongoDB) kommenterede · mandag, 4. marts 2024, 21:32 · Rapport

Hi Hyung,

Thank you for your feedback. This is a feature currently under active development. I recommend to follow it with existing feedback item: https://feedback.mongodb.com/forums/924145-atlas/suggestions/39906208-granular-permissions

Thank you,

Fuat

-

Derudover er der mange andre kommentarer der understøtter dette. MongoDB tilbyder foruddefineret roller, som f.eks. projekt ejer, projekt medlem og andre projektniveaur. Af den årsag valgte jeg ikke at gå med MongoDB
Kommandoer jeg brugte til granulær adgang:

```
GRANT SELECT(id, title, author) ON books TO kristian;
```

```
GRANT UPDATE(author) ON books TO kristian;
```

```
GRANT INSERT (author) on books TO kristian;
```

Jeg har brugt Supabase til oprettelse af min postgreSQL og hertil bliver databasen hosted på AWS.

https://supabase.com/dashboard/projects

Sådan her ser hele databasen ud.

```
  id bigint primary key generated always as identity,
   title text,
   author text,
   published_date timestamp with time zone
```

id	created_at	title	author	published_year
1	"2024-03-05 11:5	"The Great Gatsb	"F. Scott Fitzge	1925
3	"2024-03-05 11:5	"1984"	"George Orwell"	1949
4	"2024-03-05 11:5	"The Catcher in	"J.D. Salinger"	1951
5	"2024-03-05 11:5	"The Great Gatsb	"F. Scott Fitzge	1925
6	"2024-03-05 11:5	"To Kill a Mockt	"Harper Lee"	1960
7	"2024-03-05 11:5	"1984"	"George Orwell"	1949
8	"2024-03-05 11:5	"The Catcher in	"J.D. Salinger"	1951
9	"2024-03-05 14:2	null	"New Author Name	null
2	"2024-03-05 11:5	"To Kill a Mockt	"New Author"	1960
10	"2024-03-05 14:5	null	"New Author Kris	null