Databaser

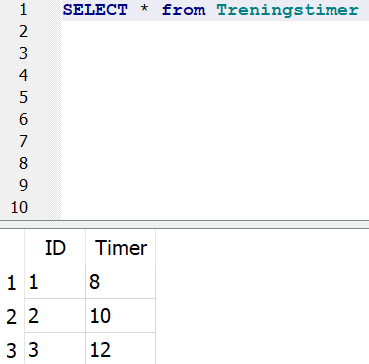
**Create**

Ved brug af Create opretter man en tabel. Den oprettede tabel i eksemplet hedder treningstimer og har kolonerne ID og Timer



Efter at have indsat data i tabellen ser den sådan ud:



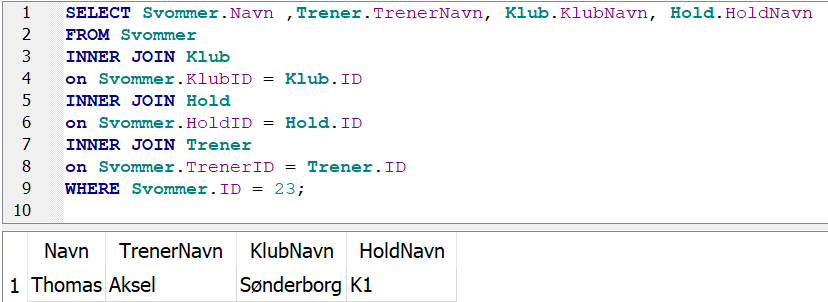


**Insert**

Ved brug af Insert kan der indsættes data i den oprettede tabel. Ved at sige hvilken tabel, heri Svommer, og vælge de kolonner hvor dataene skal indsættes kan der oprettes en ny Svommer til tabellen. I eksemplet er der lavet en Svommer med ID’et 23, som hedder Thomas. De andre ID fås fra de andre tabeller, sådan at han kommer i den rigtige klub, hold og med rigtig træner:

****

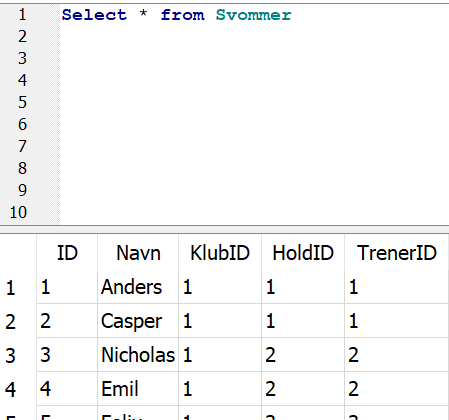
Den nye svømmer kan dermed findes:



Hvor han nu kan se hvilken træner, klub og hold han har.

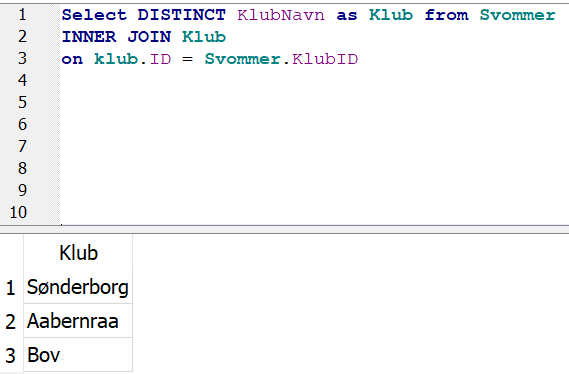
**Select**

Ved brug af Select vælger man hvad man vil se af dataene. \* i programmet siger at jeg gerne vil se alle koloner fra tabellen Svommer. Som det kan ses, kommer der 5 koloner frem:

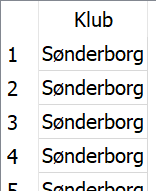
****

**Select Distinct**

Ved brug af Select Distinct får man alle unikke instans. I dette eksempel ser man alle unikke klubber i stedet for at se klubberne flere gange.   
Der er blevet brugt inner join for at få klubnavnet frem i stedet for ID’en for klubben.

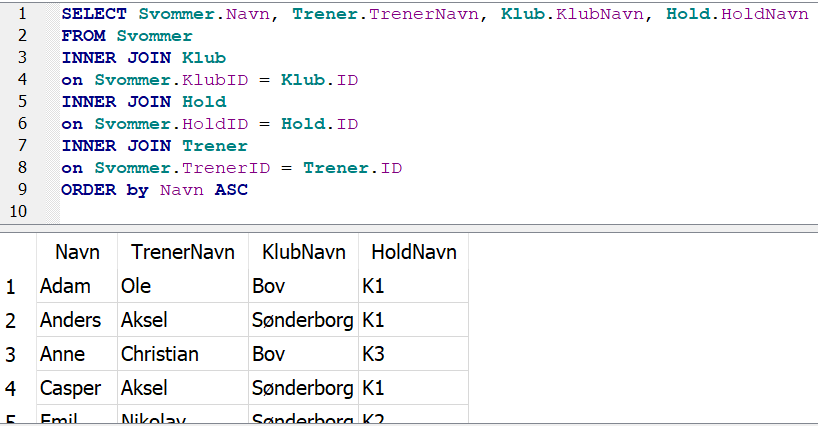
****

Uden distinct ville det se sådan ud:



**Select from Order (Order by)**

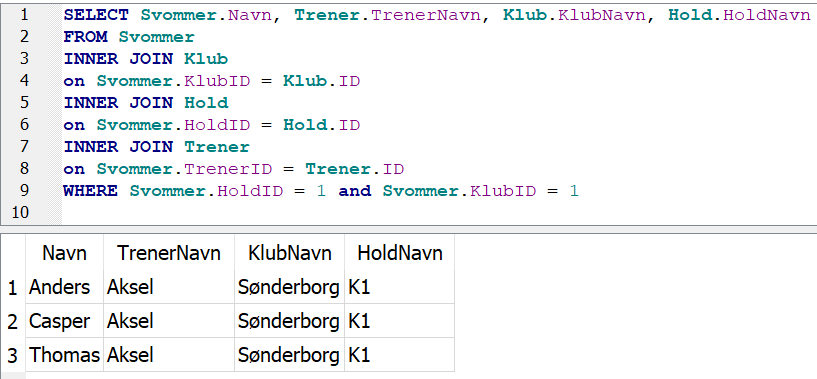
Ved brug af Order by kan man vælge at få dataene vist i alfabetisk eller numerisk rækkefølge. I tabellen er navnen på en masse svømmere og tabellen er blevet sat i orden efter navnene i stigende rækkefølge får man dem der starter med A først:

****

Dette sker ved at skrive Order by *kolonnenavn* ASC, hvor ASC står for ascending hvilket gør at, når det er navne, A står øverst.

**Select from Where**

Ved brug af et Where-statement kan man søge på noget specifikt. Når det handler om en klub, kan jeg finde dem på et specifikt hold, eller dem med en specifik træner:



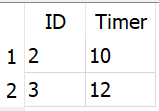
Hermed får jeg altså dem fra Sønderborg, der går på K1

**Delete**

Ved brug af Delete kan man slette en tupel i en tabel, hvis den ikke længere er nødvendigt:



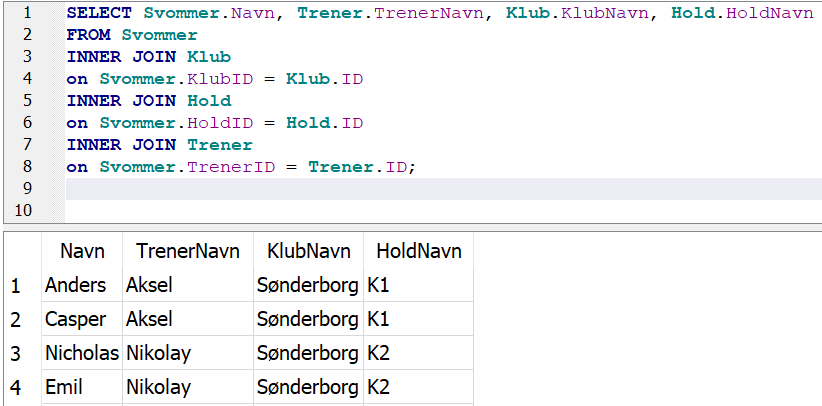
Dermed vil tabellen have fjernet denne tupel:



**Joins**

I min tabel er der ikke eksempler på Left og RightJoin, da alle dataene passer sammen, derfor er det kun InnerJoin der giver mening at bruge her.

**InnerJoin** viser de steder hvor datene er fælles. Hvis datene ikke er fælles i nogen af de to taballer, der bliver benyttet viser dette ikke i den tabel man får ud:



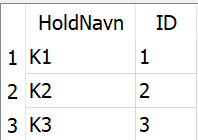
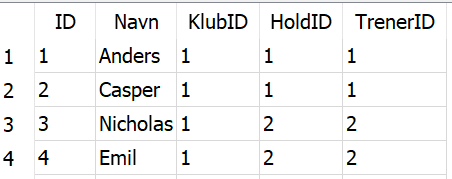
Her er alle de forskellige tabeller blevet sat sammen med tabellen Svommer, der indeholder en masse ID’er

**LeftJoin** er som sagt ikke relevant i min database. Hvis jeg tog hensyn til at der er en masse skabe i en svømmehal, og nogle af svømmerne havde et bestem skab ville jeg ved at få vist Navnene på svømmerne og SkabsID, få alle navnene og de skabe der har en tilhørende svømmer.

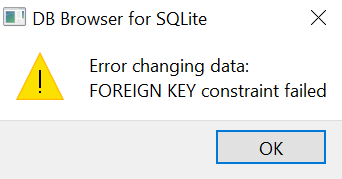
**RightJoin**, ligesom LeftJoin, ville kun vise de relevante data fra den ene tabel og alle data fra den anden tabel. Ligesom med skabene ville man her bare få vist de svømmer, der har et skab, og tilsvarende skab. Udover dette vil man også få vist alle andre skabe, der ikke er blevet tildelt en svømmer, og alle de svømmere der ikke har et skab ville ikke blive vist

**Primary og Foreign Key**

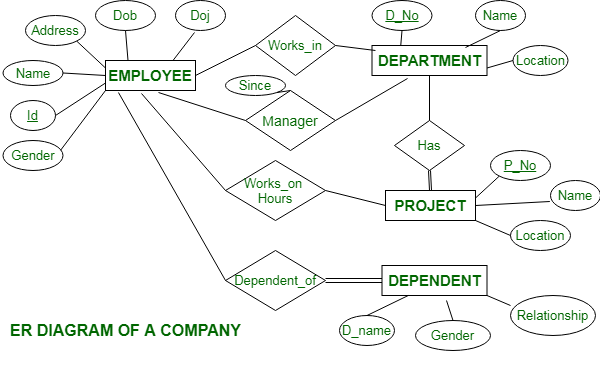
En Primary key er en unik hovednøgle, der sættes på en kolonne i en af tabellerne i databasen. Ved at sætte en anden kolonne fra en anden tabel som foreign key som afhængig af en primary key, kan værdierne fra kolonnen med en foreign key ikke være unikke fra de værdier der optræder i kolonnen med en primary key:



De to tabeller Hold og Svommer, hvor ID i Hold er primary key til HoldID i Svommer. Dette betyder at der ikke kan være et hold med en ID på 4 i HoldID, da denne ikke findes i ID:



**E/R Diagram**



**Normalformer**

1NF: Ingen kolonner må antage en anden kolonnes værdi, i samme tabel.

2NF: Hvis man har en sammensat nøgle, så skal alle andre kolonner afhænge af den samlede nøgle.

3NF: Ingen felter udenfor primærnøglen må afhænge af hinanden.

### Relationel algebra

I SQL:

Sigma er det samme som select.

I SQL: