# **Oblig 1 Databaser og nettverk 23.9.2021 kl. 12:00**

Dette er en SQL øvelse. Lever inn både SQL setningen du lager og resultatet.

MySQL workbench: se kap 30 i refman-8.0

**Oppgave 1**: Lag SQL setninger som:

1. Oppretter en ny database oblig1.
2. Lager tabellen Film i kapittel 2, oppg. 1, i databasen oblig1
3. Setter inn dataene i figur 2.16 i filmtabellen
4. 

CREATE DATABASE Oblig1;

USE Oblig1; OK

CREATE TABLE IF NOT EXISTS Film (

FNr INT,

Title VARCHAR(20) NOT NULL,

Year INT,

Country VARCHAR(50) NOT NULL,

Genre VARCHAR(10) NOT NULL,

Age INT,

RunTime int,

Price float,

CONSTRAINT PK\_Film PRIMARY KEY(FNr)

) Engine=InnoDB; OK

INSERT INTO Film Values

(1, 'Casablanca', 1942, 'USA', 'Drama', 15, 102, 149.00),

(2, 'Fort Apechea', 1948, 'USA', 'Western', 15, 127, NULL),

(3, 'Apocolypse Now', 1979, 'USA', 'Action', 18, 155, 123.00),

(4, 'Streets of Fire', 1984, 'USA', 'Action', 15, 93, NULL),

(5, 'High Noon', 1952, 'USA', 'Western', 15, 85, 123.00),

(6, 'Cinema Paradiso', 1988,'Italy', 'Comedy', 11, 123, NULL),

(7, 'Asterix hos britene', 1988, 'France', 'Animation', 7, 78, 149.00),

(8, 'veiviseren', 1987, 'Norway', 'Action', 15, 96, 87.00),

(9, 'Salmer Fra Kjøkkenet', 2002, 'Norway', 'Comedy', 7, 80, 149.00),

(10, 'Anastasia', 1997, 'USA', 'Animation', 7, 94, 123.00),

(11, 'LA Grande bouffe', 1973, 'France', 'Drama', 15, 129, 87.00),

(12, 'Blues Brothers 2000', 1998, 'USA', 'Comedy', 11, 124, 135.00),

(13, 'Beatles: Help', 1965, 'Great Britain', 'Music', 11, 144, NULL); OK

1. Henter Tittel, Sjanger og Pris for filmer som er produsert i 1988 eller seinere, sortert synkende på pris (=dyrest først)



USE Oblig1;

Select Title,

Year,

Genre,

Price

FROM Film

WHERE Year >= 1988

ORDER BY Price DESC; OK

1. Henter alle kolonner for filmer som ikke er til salgs (mangler pris), sortert på alder og sjanger.

USE Oblig1;

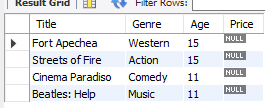
Select Title,

Genre,

Age,

Price

FROM Film

WHERE Price is NULL   
order by Price Desc  
  


Finner antall filmer som er til salgs i hver sjanger, og summen av prisen  
  
USE Oblig1;

Select

Genre,

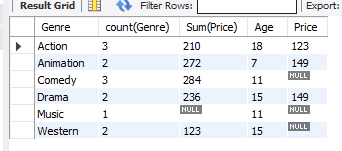
count(Genre),

Sum(Price),

Age,

Price

FROM Film WHERE Price IS NOT NULL

group by Genre;  


Setter inn en ny rad. Finn på data selv.  
USE Oblig1;

ALTER TABLE Film

ADD Score Int AFTER RunTime;   
USE OBlig1;

UPDATE Film SET Score = 10 WHERE FNr = 1;

UPDATE Film SET Score = 7 WHERE FNr = 2;

UPDATE Film SET Score = 10 WHERE FNr = 3;

UPDATE Film SET Score = 7 WHERE FNr = 4;

UPDATE Film SET Score = 10 WHERE FNr = 5;

UPDATE Film SET Score = 7 WHERE FNr = 6;

UPDATE Film SET Score = 10 WHERE FNr = 7;

UPDATE Film SET Score = 7 WHERE FNr = 8;

UPDATE Film SET Score = 10 WHERE FNr = 9;

UPDATE Film SET Score = 7 WHERE FNr = 10;

UPDATE Film SET Score = 10 WHERE FNr = 11;

UPDATE Film SET Score = 7 WHERE FNr = 12;

UPDATE Film SET Score = 10 WHERE FNr = 13;

Korrigerer tittelen på filmen «High Noon». Den heter egentlig «High Moon»  
USE Oblig1;

UPDATE Film

SET

Title = 'High Moon'

WHERE

Title = 'High Noon'; OK

Legger på 10% på prisen på alle Action filmer  
USE Oblig1;

UPDATE Film SET Price = Price\*1.1 WHERE Genre = 'Action'; OK

1. Sletter filmen Anastasia

USE Oblig1;

DELETE FROM Film WHERE Title= 'Anastasia'; OK

**Oppgave 2**: Videobutikken har også en tabell Kunde. Se kap. 3, oppgave 4. Lag SQL setninger som:

1. Lager tabellen Kunde.  
   Kundetabellen skal inneholde et kundenummer (som er primærnøkkel), fornavn og etternavn, adresse og postnummer.
2. Setter inn minst 3 rader med data i tabellen Kunde. En av dem heter Kari Mo.

USE Oblig1;

CREATE TABLE IF NOT EXISTS Customers (

KNr INT,

Lastname VARCHAR(20) NOT NULL,

Firstname VARCHAR(20), #hele navn og adresse bør vel være NOT NULL

Adress VARCHAR(20),

Zipcode INT,

CONSTRAINT PK\_Customers PRIMARY KEY(KNr)

) Engine=InnoDB; OK

INSERT INTO Customers Values

(1, 'Mo','Kari', 'GateVeien 5', 1312),

(2, 'Morgan', 'Arthur', 'VeiGata 13', 1153),

(3, 'Castro','Fidel', 'CubaParken', 1953); OK

**Noen stamkunder får lov til å leie uten å betale kontant. De får en faktura isteden.**

1. Lag en fakturatabell. Finn selv ut hvilke felter som bør være med i tabellen. Skriv noen få ord om hva de forskjellige feltene skal inneholde og en begrunnelse for hvorfor feltet skal kunne ha NULL eller ikke. Er det noen felter du vil ha standardverdier (default-verdier) i? Begrunn typen du har valgt for de forskjellige feltene. Hva velger du som PK? Bruker du FK?

Legger inn data i faktura tabellen. Legg inn minst 2 fakturaer for Kari Mo  
USE Oblig1;

USE Oblig1;

CREATE TABLE IF NOT EXISTS Invoice (

INr INT,

LastName VARCHAR(20) NOT NULL,

KID INT,

Cash Float,

CONSTRAINT PK\_Customers PRIMARY KEY(INr),

**Foreign Key (LastName)**

**references Customers(LastName) #dette blir feil**

) Engine=InnoDB;

Hvis du prøver å opprette disse tabellene I Workbench får du feilmelding:   
**Error Code: 1215. Cannot add foreign key constraint**

Prøv å lete på nettet med søkeord "MySQL Error Code 1215"

For oss skal alltid en fremmednøkkelen i en tabell referere til primærnøkkel I en annen tabell. Det gjøres ikke i din fakturatabell.

Dette er sentralt stoff, og du må få det på plass.

Altså: opprett en korrekt fakturatabell, og tilpass resten av oppgave 2 til dette.

INSERT INTO Invoice Values

(5, 'Mo',1234, 'GateVeien 5', 100.00),

(7, 'Mo',1434, 'GateVeien 5', 150.00);  
foreign key satt inn som etternavn for å assigne riktig fakura fra kundelista

Hent alle Kari Mo sine fakturaer.  
 SELECT

INr,

Name,

KID,

Adress,

Cash

from Invoice

Where LastName = 'Mo';

1. Legger inn en forretningsregel, slik at fakturabeløpet ikke kan være negativt, og ikke større enn kr 10.000,- (CONSTRAINT)

delimiter //

CREATE TRIGGER FixPirce BEFORE INSERT ON Invoice FOR EACH ROW

BEGIN

IF NEW.Cash < 0 THEN

SET new.Cash = 0;

END IF;

IF NEW.Cash > 1000 THEN

SET new.Cash = 10000;

End if;

END;//

delimiter ;

1. Prøv å legge inn en faktura rad med «ulovlig» pris. Hva skjer.  
   - i databasen vil tallet bli satt til 0 om tallet er lavere enn 0 eller bli satt til 1000 om det er høyere enn det  
   INSERT INTO Invoice Values

(611, 'Mo',1234, 'GateVeien 5', -100000.00);

1. Korriger SQLen din så prisen blir «lovlig og kjør SQLen på nytt  
   INSERT INTO Invoice Values
2. (611, 'Mo',1234, 'GateVeien 5', -100000.00);

**NB! 2f, 2g og 2h fungerer kanskje ikke, men prøv. De krever en forholdvis ny utgave av MySQL.**

**Oppgave 3**

Finn dokumentasjonen på den innebygde funksjonen SUBSTRING\_INDEX på mariadb.com

Kopier beskrivelse av funksjonen til besvarelsen

Returns the substring from string *str* before count occurrences of the delimiter *delim*. If *count* is positive, everything to the left of the final delimiter (counting from the left) is returned. If *count* is negative, everything to the right of the final delimiter (counting from the right) is returned. SUBSTRING\_INDEX() performs a case-sensitive match when searching for *delim*.

1. Vis at du kan bruke funksjonen

SELECT SUBSTRING\_INDEX('1-2-3-4','-', 1); - gir 1

SELECT SUBSTRING\_INDEX('1-2-3-4','-', 2); - gir 1-2

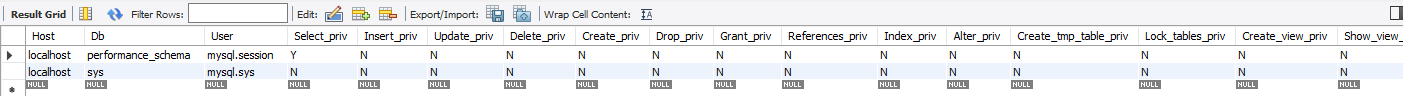
SELECT SUBSTRING\_INDEX('1-2-3-4','-', -3); gir 2-3-4  
  
dette kan brukes der man har nettsider hvor man ikke vil ha med .com eller annet info som ikke er relevant som like vel er med i dataen man behandler.

**Oppgave 4**

MySQL har en system database (schema) som heter mysql. Utfør «USE mysql;» så kan du jobbe mot denne basen. (Eller: finn denne databasen i **phpMyAdmin**.)

Hent alla data fra tabellen DB  
Use myql;

Select \* from DB;

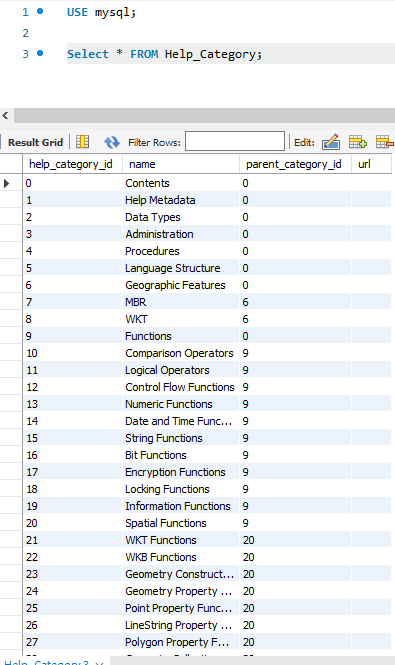


* 1. Finn fram til dokumentasjon for denne systemdatabasen

<https://dev.mysql.com/doc/refman/8.0/en/system-schema.html>

* 1. Hent data fra et par av de andre tabellene  
     Use mysql;

Select \* from Help\_category



Info:

For å se systemdatabasene i object-browseren i Workbench:

Edit | Preferences | SQL Editor | Show Metadata Schemata

Every MySQL is shipped with default system schemas/databases. Those are:

* **mysql** - is the system database that contains tables that store information required by the MySQL server
* **information\_schema** - provides access to database metadata (kap 24 i refman-5.7)
* **performance\_schema** - is a feature for monitoring MySQL Server execution at a low level (kap 26 i refman-8.0)
* **sys** - a set of objects that helps **DBA**s and developers interpret data collected by the Performance Schema (kap 27 i refman-8.0)