# **MySQL databse kommandoer**

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.

CREATE database oblig1;

1. Lager tabellen Film i kapittel 2, oppg. 1, i databasen oblig1

CREATE TABLE Film (

FilmNr INTEGER,

Tittel VARCHAR(100) NOT NULL,

År SMALLINT,

Land VARCHAR(50),

Sjanger VARCHAR(50),

Alder SMALLINT,

Tid SMALLINT,

Pris DECIMAL(8, 2),

CONSTRAINT FilmPN PRIMARY KEY (FilmNr)

) Engine=InnoDB;

1. Setter inn dataene i figur 2.16 i filmtabellen

INSERT INTO Film (FilmNr, Tittel, År, Land, Sjanger, Alder, Tid, Pris) VALUES

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

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

( 3, 'Apocalypse 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, 'Italia', 'Komedie', 11, 123, NULL),

( 7, 'Asterix hos britene', 1988, 'Frankrike', 'Tegnefilm', 7, 78, '149.00'),

( 8, 'Veiviseren', 1987, 'Norge', 'Action', 15, 96, '87.00'),

( 9, 'Salmer fra kjøkkenet', 2002, 'Norge', 'Komedie', 7, 80, '149.00'),

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

(11, 'La Grande bouffe', 1973, 'Frankrike', 'Drama', 15, 129, ' 87.00'),

(12, 'The Blues Brothers', 1980, 'USA', 'Komedie', 11, 124, '135.00'),

(13, 'Beatles: Help', 1965, 'Storbritania', 'Musikk', 11, 144, NULL

);

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

SELECT Tittel, Sjanger, Pris FROM Film WHERE År >= 1988 ORDER BY Pris DESC;

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

SELECT \* FROM Film WHERE Pris IS NULL ORDER BY Alder, Sjanger;

1. Finner antall filmer som er til salgs i hver sjanger, og summen av prisen

SELECT Sjanger, COUNT(\*) AntallTilSalgs, SUM(Pris) AS SumPris FROM Film   
 WHERE Pris IS NOT NULL GROUP BY Sjanger ORDER BY Sjanger;

1. Setter inn en ny rad. Finn på data selv.

INSERT INTO Film (FNr, Tittel, År, Land, Sjanger, Alder, Tid, Pris)

VALUES (14, 'Avatar', 2008, 'USA', 'Drama', 13, 180, NULL);

1. Korrigerer tittelen på filmen «High Noon». Den heter egentlig «High Moon»  
   UPDATE Film SET Tittel = 'High Moon' WHERE Tittel = 'High Noon';
2. Legger på 10% på prisen på alle Action filmer  
   UPDATE Film set Pris = Pris\*1.1 WHERE Sjanger='Action';
3. Sletter filmen Anastasia  
   DELETE FROM Film WHERE Tittel = 'Anastasia';

**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.

CREATE TABLE Kunde (

KundeNr INTEGER,

Fornavn VARCHAR(50) NOT NULL,

Etternavn VARCHAR(50) NOT NULL,

Adresse VARCHAR(100) NOT NULL,

PostNr CHAR(4) NOT NULL,

CONSTRAINT KundePN PRIMARY KEY (KundeNr)

) Engine=InnoDB;

1. Setter inn minst 3 rader med data i tabellen Kunde. En av dem heter Kari Mo.

INSERT INTO Kunde (KundeNr, Fornavn, Etternavn, Adresse, PostNr) VALUES

(5002, 'Paal', 'Aass', 'Lindemans gate79', '1711'),

(5007, 'Joakim', 'Laursen', 'Thomas Heftyes gate 39', '0015'),

(5009, 'Kari', 'Mo', 'Solheimgata 81', '0654');

**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?

CREATE TABLE Faktura (

FakturaNr INT,

KundeNr INT,

FilmNr INT,

Beløp DECIMAL(8,2) NOT NULL,

Dato DATE,

TekstTilKunden VARCHAR(200),

CONSTRAINT FakturaPK PRIMARY KEY (FakturaNr),

CONSTRAINT KundeNrFK FOREIGN KEY (KundeNr) REFERENCES Kunde (KundeNr),

CONSTRAINT FilmNrFK FOREIGN KEY (FilmNr) REFERENCES Film(FilmNr)

) Engine=InnoDB;  
Hva vil du gjøre for å få flere filmer på en faktura?  
Ser at forhold mellom Faktura og Film er M:N, lager ny tabell FakturaFilm:

CREATE TABLE FakturaFilm (

FakturaNr INT,

FilmNr INT,

Beløp DECIMAL(8,2) NOT NULL,

Dato DATE,

CONSTRAINT FakturaFilmPK PRIMARY KEY (FakturaNr, FilmNr),

CONSTRAINT FilmNrFK FOREIGN KEY (FilmNr) REFERENCES Film(FilmNr),

CONSTRAINT FakturaNrFK FOREIGN KEY (FakturaNr) REFERENCES Faktura(FakturaNr)

) Engine=InnoDB;  
Benytter ikke FakturaFilm-tabell i det videre.

1. Legger inn data i faktura tabellen. Legg inn minst 2 fakturaer for Kari Mo

INSERT INTO Faktura (FakturaNr, KundeNr, FilmNr, Beløp, Dato, TekstTilKunden) VALUES

(10,5002, 1, 200, '2021-09-10','Casablanca'),

(11,5009, 7, 180, '2021-09-20','Asterix hos britene'),

(12,5009, 4, 150, '2021-10-03','Streets of Fire')

;

1. Hent alle Kari Mo sine fakturaer.  
   SELECT \* FROM Faktura F

INNER JOIN Kunde K ON F.Kundenr=K.KundeNr

WHERE K.Fornavn='Kari' AND K.Etternavn='Mo';

1. Legger inn en forretningsregel, slik at fakturabeløpet ikke kan være negativt, og ikke større enn kr 10.000,- (CONSTRAINT)  
   ALTER TABLE Faktura ADD  
    CONSTRAINT CHECK(Beløp > 0 AND Beløp < 10000);
2. Prøv å legge inn en faktura rad med «ulovlig» pris. Hva skjer.  
   INSERT INTO Faktura (FakturaNr, KundeNr, FilmNr, Beløp, TekstTilKunden)   
    VALUES (15, 5009, 4, 11111, "Apocalypse Now - 123kr, CasaBlanca - 149kr");  
   Innsettingen godtas. Har for gammel MySQL.
3. Korriger SQLen din så prisen blir «lovlig og kjør SQLen på nytt

INSERT INTO Faktura (FakturaNr, KundeNr, FilmNr, Beløp, TekstTilKunden)   
 VALUES (15, 5009, 4, 123, "Apocalypse Now - 123kr, CasaBlanca - 149kr");

**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

1. Kopier beskrivelse av funksjonen til besvarelsen

<https://mariadb.com/kb/en/library/substring_index/>  


1. Vis at du kan bruke funksjonen

Denne returnerer strengen: Dette er, vet jeg

SELECT SUBSTRING\_INDEX('Dette er, vet jeg, en vanskelig avggjørelse',',',2)   
 AS AltFørAndreKomma;

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**.)

* 1. Hent alla data fra tabellen DB
  2. Finn fram til dokumentasjon for denne systemdatabasen
  3. Hent data fra et par av de andre tabellene

Løsningsforslag:

For å se systemdatabasene i MySQL: Edit | Preferences

Velg SQL-Editor. Under Sidebar, kryss av for «Show Metadata and Internal Schemas»

For å få informasjon om en database (schema): Velg en base i baseoversikten (schema) til venstre i Workbench. Da kommer det opp en skiftnøkkel og en i, Trykk på ien.

Et par eksempelr på hva systemdatabasen mysql inneholder:

* Tabellen mysql.user inneholder om informasjon om hver bruker, dvs. hva slags rettigheter brukeren har etc.
* Tabellen help\_topic inneholder en kort forklaring for hver emne, samt en url til hjelpsystemet til en server et sted (MariaDB sin server).

En annen systemdatabase er information\_schema

* Tabellen columns: Inneholder informasjon om alle kolonnene i databasene som du har opprettet.

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)

Det er ikke lov å bruke internett, unntatt til å gå inn på Inspera. Praktisk utførelse av dette:

* Du kobler deg opp mot Inspera for å hente ut oppgavetekst og noen oppgitte data.
* Så slår du av internett på PCen (SLÅ PÅ FLYMODUS)
* Når oppgaven skal leveres, slår du på internett for å bruke Inspera

Et par ting dere må passe på:

* Når dere lager en tabell, sett alltid Engine til InnoDB (de fleste har det som default). Hvis det er problemer, sjekk at dere har engine InnoDB på alle tabeller som er involvert.
* Når Workbench lager en index, er det i scriptet med ordet VISIBLE etter hver index. MariaDB og MySQL liker ikke den, så den må slettes (kommer an på versjon av databasene)
* Dersom dere lager en FK med flere felter, må den tabellen som refereres ha en index på disse feltene med samme rekkefølge på feltene som det er i din FK.

OPPGAVE 1 35%

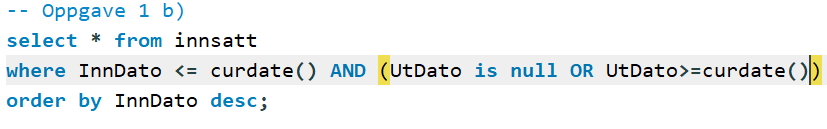
Her er en beskrivelse av et databasesystem for å holde orden på innsatte i fengsler. Det følger med ei script fil som lager denne databasen. NB! Sone i denne sammenhengen betyr å sitte i fengsel (sone straffen sin), og har ingenting med at fengselet er delt opp i soner.

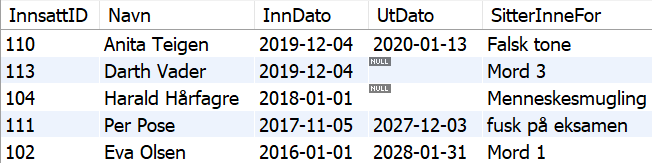
Databasen inneholder følgende tabeller:

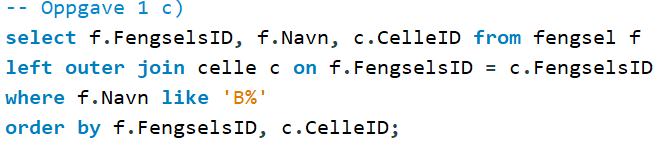
* **Fengsel** – FengselsID og navn på fengslene
* **Celle** – CelleID samt hvilket fengsel cella er i (celler er nummerert lokalt for hvert fengsel)
* **Innsatt** – ID og navn på innsatt, samt dato for soning. InnsattID er nummerert slik at de er unike uavhengig av fengsel (dvs. ikke lokale for hvert fengsel)
* **SonerICelle** – kobling mellom innsatt og celle, slik at en ser i hvilken celle og fengsel en innsatt sitter i.

Opprett databasen prison, med tabeller og data fra medfølgende script

Lag SQL-setninger som gjør følgende:

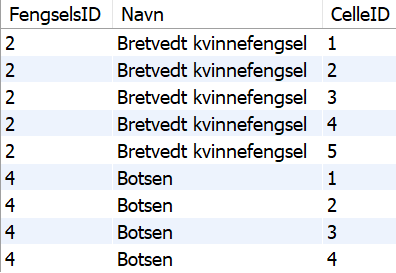
1. Lister opp alle innsatte. 1a) select \* from innsatt;
2. Lister opp alle innsatte (alle i tabellen innsatt) som sitter inn nå, sortert synkende på datoen de ble satt inn (innsatt.InnDato)



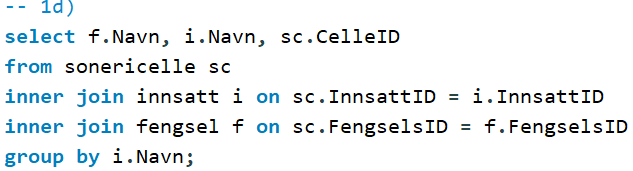


1. Lister opp alle fengsler som begynner på boksatven B, med tilhørende celler, sortert på fengselsid og celleid

LØSN: Det er greit med INNER JOIN, men da får en bare med de fengslene som har registrerte celler (men hva skal en med et fengsel uten celler?)



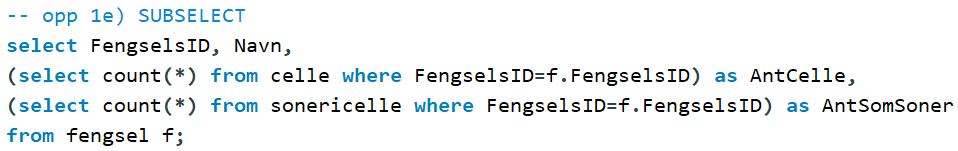
1. Lister opp alle innsatte, med navn på fengsel og person, og id på celle

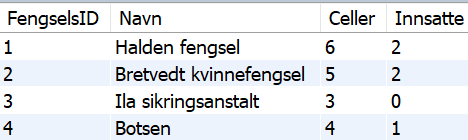




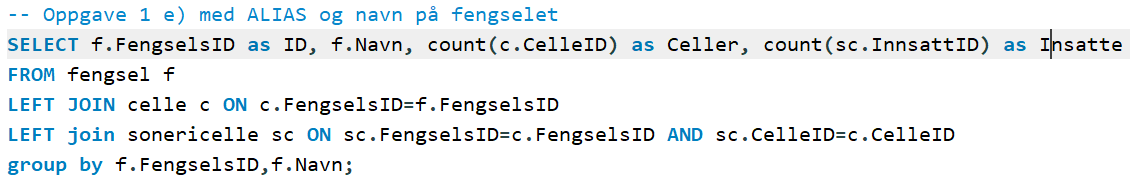
1. Lister opp alle fengsler (en rad pr. fengsel), med antall celler pr fengsel og antall innsatte pr fengsel.

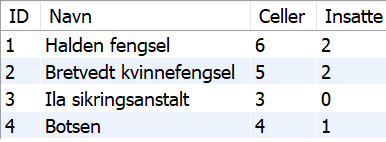
LØSNINGSFORSLAG:





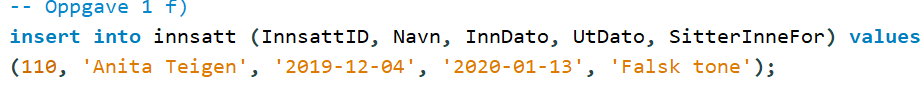
Denne SELECT vil også fungere, med outer join



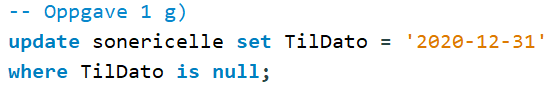


1. Setter inn en rad i tabellen innsatt

Løsningsforslag: Dersom en ikke har InsattID som autoincrement, men en angi en unik verdi for InsattID.



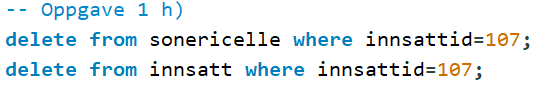
1. Oppdaterer tabellen sonericelle med TilDato '2020-12-31' på alle rader som mangler TilDato



1. Sletter en innsatt som er ferdig med soningen

LØSN: Fordi sonericelle har en FK som kobler InnsattID mot tabellen Innsatt, må en først slette raden i conericell før en kal slette den innsatte i tabellen innsatt.

Når en innsatt slipper ut (eller flyttes), er det naturlig å sette en tildato istedenfor å slette raden. Men etter ei stund kan det hende dataene skal slettes. Og dersom det er lagt inn noe feil, bør en kunne slette.



OPPGAVE 2 35%

* Oppdater databasen fra oppgave 1 slik at databasen kan brukes til å holde rede på ansatte også. Fengslene er delt opp i avdelinger. Cellene skal kobles til avdelinger
* Lag en vaktliste tabell, som en slags logg over hvem (ID for ansatt) som har vært på vakt på hvilken avdeling, og når han/hun hadde vakt (fra og til dato og klokkeslett). Denne tabellen kan også fylles ut på forhånd, og dermed brukes til å se hvem som skal på vakt.

Oppdater databasen fra oppgave 1 slik at databasen kan brukes til å holde rede på ansatte også. Fengslene er delt opp i avdelinger. Basen skal kunne inneholde ei vaktliste om når den enkelte skal jobbe, og på hvilken avdeling han/hun skal jobbe. Cellene skal også kobles til avdelinger.

1. Lag ei spørring som viser hvem som jobber på hvilke avdelinger for et angitt tidspunkt.
2. Lag ei spørring som returnerer navn på både innsatte og ansatte, med en kode e.l. som forteller om det er en ansatt eller en innsatt.

LØSNINGSFORSLAG: 3 tabeller:

* Ansatt – med data om de ansatte. Et alternativ er å utvide tabellen Innsatt til å inneholde ansatte også, og kalle den person e.l. Men det er gode grunner til å holde innsatte og ansatte adskilt, også i basen.
* Avdeling – informasjon om hver enkelt avdeling. Det er 2 måter å navngi/nummerere avdelingene på, enten pr fengsel eller fortløpende uavhengig av fengsel.
* Vaktliste – holder rede på hvem som har vakt (er på jobb) på hvilken avdeling for hvilke tidsrom.
* I tillegg må tabellen celle utvides med AvdelingsID.

Punkt en er å plassere data i riktig tabell. Det naturlige er at ansatte er ansatt i et bestemt fengsel. For store fengsler med store avdelinger kan en kanskje tenke seg at en ansatt er knyttet til en bestemt avdeling. Men de er ikke ansatt i en bestemt celle, så CelleID i ansatt tabellen blir veldig galt.

Typer er også viktig. IDer skal være heltall. Dato og klokkeslett skal være Date, DateTime eller Time.

Alle tabeller skal ha en fornuftig PK

Faste koblinger i basen bør ha FK, for å sikre datakvalitet:

* I Avdeling bør FengselsID vær FK mot tabellen Fengsel
* I Celle bør AvdelingsId vær FK mot tabellen Avdeling
* I Vaktliste bør FengselsID, AvdelingsID vær FK mot Avdeling og AnsattID vær FK mot tabellen Ansatt.

1. Skriv **SQL**-kode som oppretter en database (schema) **Eksamen2020.** Gjer til gjeldende database**.**  
   Skriv **SQL**-kode som oppretter tabellene du lagde foran.   
   Gjengi koden i besvarelsen med alle skranker (Constraints) du finner naturlig/nødvendig. Du skal la de ansattes **Fornavn**+**Etternavn** være entydig.  
   **CREATE DATABASE Eksamen2020;  
   USE Eksamen2020;  
   CREATE TABLE Poststed (  
    PostNr INT,  
    Poststed VARCHAR(45) NOT NULL,  
    CONSTRAINT PRIMARY KEY(PostNr)**

**) Engine=InnoDB;**

**CREATE TABLE Ansatt (  
 AnsNr INT,  
 Fornavn VARCHAR(45) NOT NULL,  
 Etternavn VARCHAR(45) NOT NULL,**

**Gate VARCHAR(45) NOT NULL,  
 PostNr INT NOT NULL,  
 UNIQUE (Fornavn, Etternavn),   
 CONSTRAINT PRIMARY KEY(AnsNr),  
 CONSTRAINT FOREIGN KEY(PostNr) REFERENCES Poststed(PostNr)**

**) Engine=InnoDB;  
CREATE TABLE Telefon (  
 AnsNr INT NOT NULL,   
 TelefonNr INT,  
 CONSTRAINT PRIMARY KEY(AnsNr, TelefonNr),  
 CONSTRAINT FOREIGN KEY(AnsNr) REFERENCES Ansatt(AnsNr)**

**) Engine=InnoDB;**

Under finner du data tatt fra **MongoDB**-databasen:   
**Per, Hansen, 35950001, 35951151, Byvegen 12, 2300, Hamar,  
Lise, Jensen, 35950002, 91523344, Liavegen 125, 2312, Tangen  
Anders, Lie, 35950003, 35952249, Stangevegen 17, 2310, Stange  
Johanne, Amundsen, 35950001, 35951317, 44827364, Fossbakka 38, 2330, Vallset  
Arne, Lie, 35950004, Grønnegata 100, 2300, Hamar**

Skriv **SQL**-setninger som legger disse dataene inn i tabellene du har laget.  
**INSERT INTO Poststed VALUES   
 (2300,'Hamar'), (2312,'Tangen'), (2310,'Stange'), (2330,'Vallset');  
INSERT INTO Ansatt (AnsNr,Fornavn,Etternavn,Gate,PostNr)   
 VALUES (1, 'Per', 'Hansen', 'Byvegen 12', 2300),   
 (2, 'Lise', 'Jensen', 'Liavegen 125', 2312),   
 (3, 'Anders', 'Lie', 'Stangevegen 17', 2310),   
 (4, 'Johanne', 'Amundsen', 'Fossbakka 38', 2330),   
 (5, 'Arne', 'Lie', 'Grønnegata 100', 2300);  
INSERT INTO Telefon VALUES (1, 35950001),(1,35951151),  
 (2, 35950002),(2, 91523344), (3, 35950003),(3,35952249),  
 (4, 35950001),(4, 35951317),(4, 44827364), (5, 35950004);**