Oppbygging av utvalgsspørringer

- Utvalgsspørringer (**SELECT**-spørringer) mot én tabell følger dette «mønsteret»:

```
SELECT     Hvilke kolonner skal med?
FROM             Hva heter tabellen?
WHERE      Hvilke rader skal med?
GROUP BY   Gruppere på hvilken kolonne?
HAVING     Hvilke grupper skal med?
ORDER BY   Sortere på hvilken kolonne?
```

- Ikke alle delene må være med i alle spørringer, og etter hvert skal vi se at spørringene også kan være mer kompliserte.
- Blant annet kan vi lage spørringer mot flere tabeller.

- **SELECT** er én blant mange SQL-kommandoer…

Mengde-funksjoner, tidsfunksjoner

- Select max(Lønn),min(Lønn),Etternavn from Ansatt;
- Select count(*) from Ansatt;
- Select max(Lønn),Etternavn from Ansatt group by Stilling;
- Select avg(Lønn) from Ansatt;
- Select now();
- Select year(now());

Select-setninger

- **Tabellen Ansatt(AnsattNr,Etternavn,Fornavn,AnsattDato,Stilling,Lønn)**

- **List all info om alle ansatte**
- **List alle som har Fornavn Per**
- **List alle som er ansatt etter 1/1 2000 og tjener mer enn 200 000**
- **List alle ansatte alfabetisk og gruppert etter stilling**
- **Finn hvor mange som er registrert i Ansatt-tabellen**
- **Finn hvor mye lønn som utbetales hvert år**
- **Finn hvor mye lønn som utbetales til sekretærer hvert år**
- **List alle som har etternavn som slutter på 'sen'**
- **List alle som er registrert men ikke har fylt ut Stilling**
- **Finn hvem som har vært ansatt lengst**

Svar

- SELECT * FROM Ansatt;
- SELECT * FROM Ansatt WHERE Fornavn='Per';
- SELECT * FROM Ansatt WHERE AnsattDato>'2000-01-01' AND Lønn>'200000';
- SELECT * FROM Ansatt ORDER BY Etternavn GROUP BY Stilling;
- SELECT COUNT(*) FROM Ansatt;
- SELECT SUM(Lønn) FROM Ansatt;
- SELECT SUM(Lønn) FROM Ansatt WHERE Stilling='Sekretær';
- SELECT * FROM Ansatt WHERE Etternavn LIKE '%sen';
- SELECT * FROM Ansatt WHERE Stilling IS NULL;
- SELECT * FROM Ansatt WHERE AnsattDato=select MIN(AnsattDato) from Ansatt;

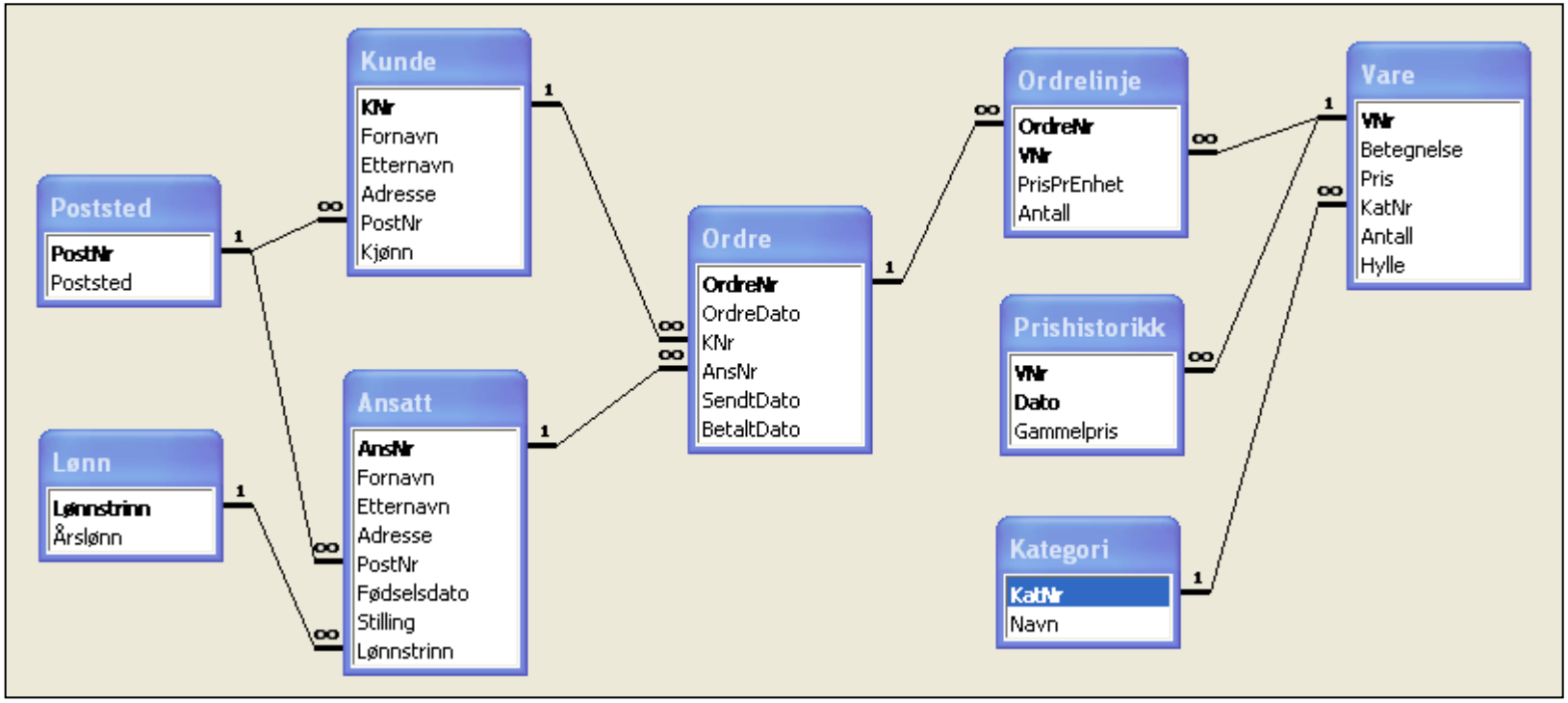
Tabelldefinisjon og datamanipulering

- Tabelldefinisjon med SQL og i utformingsvisning
  - Datatyper
  - Primærnøkler og fremmednøkler
  - Entitetsintegritet og referanseintegritet
  - Nullmerker, repetisjoner, standardverdier
  - Valideringsregler, forretningsregler
- Innsetting, oppdatering og sletting
- Import av data
- Metadatabase
- SQL-skript

Pensum: Kapittel 3

En database inneholder mange tabeller

- Tabellene til Hobbyhuset (fra Relasjonsvinduet i MS Access):



Et enklere eksempel

- Vi bruker prosjektdatabasen som eksempel:

```
Ansatt( AnsattNr, Fornavn, Etternavn,
        AnsattDato, Stilling, Lønn )
Prosjekt( ProsjektNr, Budsjett, Leder*
          Start, Slutt )
ProsjektDeltakelse( AnsattNr*, ProsjektNr*,
                    AntTimer )
```

- Notasjon:
  - Tabellnavn (kolonne, … , kolonne)
  - Primærnøkler er understreket.
  - Fremmednøkler er merket med ei stjerne *.

Data og metadata

- Et databasesystem inneholder både data og metadata.
- Metadata betyr ”data om data”.
- En metadatabase beskriver oppbyggingen av en database:
  - Hva heter tabellene?
  - Hvilke kolonner består hver tabell av?
  - Hvilken datatype har hver kolonne?
  - Hva er primærnøkkel for hver tabell?
  - Hvilke kolonner er fremmednøkler, og mot hvilken tabell?
  - Hvilke valideringsregler er definert på hvilke tabeller?

- Metadata kan lagres i tabeller.
  - Metadatabasen blir lagret i systemkatalogen.
  - DBHS ajourfører tabellene i systemkatalogen.
  - I Mysql lages en mysql database over brukere, databaser etc. Denne administreres av root.

Operasjoner på tabellstruktur

- Før vi kan legge inn og hente ut data fra en tabell må vi definere hva slags struktur tabellen skal ha.
  - Definere (opprette): `CREATE TABLE Ansatt …`
  - Endre struktur: `ALTER TABLE Ansatt …`
  - Slette: `DROP TABLE Ansatt`
- Det finnes også muligheter for å spørre:
  - Hvordan er tabellen Ansatt definert?
  - DBHS slår opp i systemkatalogen.
  - DESCRIBE Ansatt;

Datatyper

- SQL definerer et sett av innebygde datatyper, bl.a.:
  - `CHAR(n)` - for eksempel `CHAR(30)`
  - `VARCHAR(n)` - for eksempel `VARCHAR(30)`
  - `INTEGER`
  - `SMALLINT`
  - `NUMERIC(p,s)` - for eksempel `NUMERIC(5,2)`
  - `REAL`
  - `DATE`
  - `TIME`
- Noen DBHS tilbyr flere datatyper. Eksempel:
  - `CURRENCY` i Access.
- Til hver datatype hører det et sett av operatorer og funksjoner.
- Gå på www.mysql.com og finn datatyper som benyttes i Mysql

Hensikten med datatyper

- Følgende uttrykk er meningsløse:
  - `250000 > "Andersen"`
  - `Lønn + AnsattDato`
  - `AnsattNr + Lønn`
  - `Etternavn < Stilling`
- Ved å bestemme en datatype for hver kolonne vil DBHS sjekke at vi ikke ”sammenligner epler med pærer”.
- Hvilke av uttrykkene over fører til feilmelding?
- Det finnes funksjoner for eksplisitt konvertering mellom datatyper.
- Cast() as datatype

Tabelldefinisjon i SQL (forsøk 1)

- Tabellnavn + navn og datatype for hver kolonne:

```
CREATE TABLE Ansatt
( AnsattNr INTEGER,
  Fornavn CHAR(15),
  Etternavn CHAR(40),
  AnsattDato DATE,
  Stilling CHAR(20),
  Lønn CURRENCY )
```

- Valg av datatype er ikke alltid opplagt.
  - Hva med telefonnummer?

Definisjon av primærnøkkel i SQL

- Alle tabeller skal ha en primærnøkkel:

```
CREATE TABLE Ansatt
( AnsattNr INTEGER,
  Fornavn CHAR(15),
  Etternavn CHAR(40),
  AnsattDato DATE,
  Stilling CHAR(20),
  Lønn CURRENCY,
  CONSTRAINT AnsattPN
     PRIMARY KEY (AnsattNr) )
```

- AnsattPN er navnet på primærnøkkeldefinisjonen.
  - Navnet velges fritt.
  - Kan for eksempel bruke tabellnavn + PN (for PrimærNøkkel).

Oppgave

- Bruk Google. Finn putty.exe og installer.
- Kjør putty mot studssh.cs.hioa.no
- Logg inn med vanlig brukernavn og passord.
- Kjør mysql-klient fra kommando-promptet på Unix-maskinen med:
- mysql –u <studentnr> -h student.cs.hioa.no <studentnr>
- Lag tabellen Ansatt:

```
CREATE TABLE Ansatt
( AnsattNr INTEGER primary key
  auto_increment,
   Fornavn CHAR(15),
   Etternavn CHAR(40));
```

- Gi kommandoene show tables; og describe Ansatt;

Unionskompatible kolonner

- Informasjonen i to tabeller kan ”kobles” når de inneholder kolonner med ”samme slags verdier”.
  - Ansattnummer forekommer i Ansatt.AnsattNr, Prosjekt.Leder og ProsjektDeltakelse.AnsattNr.
  - Prosjektnummer forekommer i Prosjekt.ProsjektNr og ProsjektDeltakelse.ProsjektNr.
  - Beløp forekommer i Ansatt.Lønn og Prosjekt.Budsjett.

- To kolonner er unionskompatible hvis de har samme domene (datatype).
  - Både Etternavn og Stilling inneholder tekststrenger, men det gir ikke mening å sammenligne verdier fra de to domenene.

Fremmednøkler

- Eksempel: Vi registrerer kun timer (i ProsjektDeltakelse) på ansatte som allerede er lagt inn i tabellen Ansatt!

- En (samling) kolonne(r) X i tabell T1 er en fremmednøkkel med hensyn på tabell T2 hvis følgende alltid gjelder:
  - En verdi i X er enten NULL, eller identisk med en verdi i primærnøkkelen til T2.
  - Ingen rader i T1 er delvis NULL i kolonnene X.
- Fremmednøkkelen skal være unionskompatibel med tilhørende primærnøkkel.
- Beskjed til DBHS: Sjekk at verdiene i fremmednøkkelen er en delmengde av verdiene i tilhørende primærnøkkel.

Fremmednøkkel $\subseteq$ Primærnøkkel

Definisjon av fremmednøkler i SQL

- Vi må angi navn på kolonne(r) og navn på referert tabell:

```
CREATE TABLE ProsjektDeltakelse
( AnsattNr INTEGER,
  ProsjektNr INTEGER,
  AntTimer INTEGER,
  CONSTRAINT ProsjektDeltakelsePN
       PRIMARY KEY ( AnsattNr, ProsjektNr ),
  CONSTRAINT ProsjektDeltakelseAnsattFN
       FOREIGN KEY ( AnsattNr )
       REFERENCES Ansatt,
  CONSTRAINT ProsjektDeltakelseProsjektFN
       FOREIGN KEY ( ProsjektNr )
       REFERENCES Prosjekt )
```

- Selve regelen kan for eksempel navngis ved:

  FraTabell + TilTabell + FN (for FremmedNøkkel).

I MySQL

```
CREATE TABLE innodb_child
( iparent_id INT NOT NULL,
ichild_id INT NOT NULL,
PRIMARY KEY (iparent_id, ichild_id),
FOREIGN KEY (iparent_id) REFERENCES innodb_parent
(iparent_id) )
 ENGINE = INNODB;
```

Så vi ser at det går helt fint å definere primær og fremmednøkler uten å benytte navngitte beskrankninger.

Egenskaper ved fremmednøkler

- Ved sletting av ansatt: Sett nullmerke i Prosjekt.Leder for prosjekter der vedkommende er prosjektleder.
- Ved endring av ansattnummer: Endre ansattnummer i Prosjekt.Leder tilsvarende.

```
CONSTRAINT AnsattProsjektFN
  FOREIGN KEY (Leder) REFERENCES Ansatt
  ON DELETE SET NULL,
  ON UPDATE CASCADE
```

- Andre muligheter:
  - `RESTRICT` (gir feilmelding)
  - `SET DEFAULT` (setter inn standardverdi)

Integritetsregler

- Entitetsintegritet: En primærnøkkel skal ikke inneholde nullmerker eller repetisjoner.
  - Motivasjon: Rader representerer «ting» eller «entiteter». Vi lagrer ikke informasjon om noe vi ikke kan identifisere.
- Referanseintegritet: Alle verdier i en fremmednøkkel (som ikke er NULL) skal finnes i primærtabellen.
  - Motivasjon: Fremmednøkler er «referanser» eller «pekere» til entiteter. Hvis vi peker på noe skal det eksistere, men det er lov ikke å peke!

Kontroll med integritetsregler

- Entitetsintegritet:
  - Beskjed til DBHS: Ansatt.AnsattNr skal ikke inneholde repetisjoner eller nullmerker!
  - I hvilke situasjoner må DBHS sjekke?
    - Når en ny rad blir satt inn.
    - Når en verdi i primærnøkkel blir endret.
- Referanseintegritet:
  - Beskjed til DBHS: ProsjektDeltakelse.AnsattNr skal kun inneholde verdier som også finnes i Ansatt.AnsattNr!
  - I hvilke situasjoner må DBHS sjekke?

Nullmerker, repetisjoner og standardverdier

- NOT NULL betyr at kolonnen må fylles ut, UNIQUE hindrer to like verdier, og DEFAULT brukes for å definere standardverdier.

```
CREATE TABLE Student
( StudNr CHAR( 6 ),
  PersonNr CHAR( 11 ) UNIQUE NOT NULL,
  Fornavn CHAR(15) NOT NULL,
  Etternavn CHAR(40) NOT NULL,
  PrivatEpost VARCHAR( 40 ) UNIQUE,
  Status CHAR( 1 ) DEFAULT 1,
  CONSTRAINT StudentPK PRIMARY KEY ( StudNr ))
```

Valideringsregler

- Valideringsregler øker datakvaliteten:
  - `Lønn < 1000000`
  - `Fødselsdato < AnsattDato`

> ➢ DBHS sjekker at valideringsreglene blir overholdt.

- Mer kompliserte ”forretningsregler”:
  - Summerte lønnsutgifter for et prosjekt skal ikke overstige budsjettet.
  - Kontroll krever ”programmering” (kan gjøres med såkalte triggere, se kap. 13).

SQL-kommandoer for ajourhold

`INSERT`	- setter inn nye rader
`UPDATE`	- endrer i eksisterende rader
`DELETE`	- sletter utvalgte rader

- `INSERT` kommer i to varianter: Legg til 1 rad / legg til mange.
- `UPDATE` og `DELETE` virker i utgangspunktet på alle rader, og må begrenses med en betingelse.
- Kommandoene brukes oftere av programmerere enn av sluttbrukere.

Innsettingsspørringer

- Vi ansetter en ny selger:

```
INSERT INTO Ansatt
       (AnsattNr, Etternavn, Stilling)
VALUES ( 14, 'Hansen', 'Selger')
```

- For kolonnenavn som sløyfes fra listen blir det satt inn nullmerker / standardverdier. Autonummer-kolonner skal ikke føres opp. Visse kolonner må føres opp. Hvilke?

- Kopierer gamle prosjekter til en hjelpetabell:

```
INSERT INTO GamleProsjekter
  SELECT *
  FROM Prosjekt
  WHERE SluttDato <= '31/12/1999'
```

Oppgave

- Legg inn 4 poster i Ansatt-tabellen deres:

```
INSERT INTO Ansatt
       (Fornavn, Etternavn)
VALUES ('Ole','Hansen');
```

Oppdateringsspørringer

- Vi øker lønn for alle ansatte med 10%:

```
UPDATE Ansatt
SET Lønn=Lønn*1.1
```

- Vi øker lønn og endrer tittel for sekretærene:

```
UPDATE Ansatt
SET Lønn=Lønn*1.5, Stilling='Sjefssekretær'
WHERE Stilling='Sekretær'
```

- UPDATE uten WHERE oppdaterer alle radene!

Slettespørringer

- Slett alle varer i kategori Kjøtt:

```
DELETE FROM Vare
WHERE Kategori='Kjøtt'
```

- Hva gjør denne?

```
DELETE FROM Vare
```

- Hvis vi skal slette i flere tabeller må vi skrive flere DELETE-kommandoer.

Oppgave

- Slett en av postene dine.

- Oppdater en av posten dine,

Skriptfiler

- Et SQL-skript er en sekvens av SQL-setninger lagret på en fil.
  - Nyttig ved utvikling av nye databaser:

```
CREATE TABLE Vare …
CREATE TABLE Kunde …
CREATE TABLE Ordre …
INSERT INTO Vare …
INSERT INTO Vare …
…
```

- Ved å starte skriptet med å fjerne tabellene kan vi kjøre skriptet flere ganger (endre litt – kjøre på nytt …)

```
DROP TABLE Vare
DROP TABLE Kunde
…
```

Eksempel på script

```
create table DEPT (DEPTNO int primary key, DNAME varchar(20),LOC
    varchar(20));

create table EMP (EMPNO int PRIMARY KEY, ENAME
    varchar(20),JOB varchar(20),MGR int references EMP(EMPNO),
    HIREDATE date,SAL int ,COMM int, DEPTNO int references
    DEPT(DEPTNO));

insert into DEPT values ('10','ACCOUNTING','NEW YORK'),
    ('20','RESEARCH','DALLAS'),('30','SALES','CHICAGO'),
    ('40','OPERATIONS','BOSTON');

insert into EMP values
    ('7369','SMITH','CLERK','7902','1980-12-17','800',NULL,'20'),
('7499','ALLEN','SALESMAN','7698','1981-02-18','1600',300,'30'),
('7521','WARD','SALESMAN','7698','1981-02-22','1250','500','30'),
('7566','JONES','MANAGER','7839','1981-04-02','2975',NULL,'20'),
('7654','MARTIN','SALESMAN','7698','1981-09-28','1250','1400','30');
```

Bruk av mysql:

Putty til studssh.cs.hioa.no

mysql –u torunngj –h student.cs.hioa.no torunngj

Det er ikke satt passord i databasen

Kan settes selv med SET PASSWORD

Trenger da en –p også ved oppkobling

Kommer inn i mysql i databasen torunngj og kan gi kommandoer:

Mysql>show tables;

Mysql>create table (…..);

For å kjøre et ferdig script (gjøres fra kommandopromptet i UNIX):

Må ha scriptet liggende på studssh.iu.hio.no kan overføre det ditt ved hjelp av secure FTP (f.eks Winscp)

studssh$ mysql –u torunngj –p –h student.cs.hioa.no torunngj<ansatt.txt

Oppgaver

- **Tabellen Ansatt(AnsattNr,Etternavn,Fornavn,AnsattDato,Stilling,Lønn)**

- **Lag et script som oppretter tabellen og legger inn to poster**
- **Slett alle med Fornavn ' Per'**
- **Endr Lønn til 300 000 for alle som tjener 250 000**
- **Slett alle poster i tabellen**
- **Slett tabellen**