Database-sproget SQL

- SQL ~ SEQUEL ~ Structered English QUEry Language
- SQL-forespørgsel, generel form
-
```
SELECT A1, …, Ar
FROM R1, …, Rk
WHERE B
```
- med
  - attributter `A1, …, Ar`
  - relationer `R1, …, Rk`
  - betingelse `B` (logisk udtryk, evt. sammensat)

Datalogi
Roskilde Universitetscenter

Simple forespørgsler (enkelt relation)

PID	NAVN	VIRKSOMHED	KKODE
668	Kodein "Dak"	Nycomed Danmark	7
671	Pinex	Alpharma	1
679	Treo	Lundbeck	7
682	Kodimagnyl "Dak"	Nycomed Danmark	7
1004	Japansk encephalitisvaccine	Statens Serum Institut	7
1243	Cipramil	Lundbeck	7
1454	Treosulfan "Medac"	Medac	
2183	Pinex Comp.	Alpharma	1

"navne på præparater fra Lundbeck"

```
SELECT navn
FROM p
WHERE virksomhed='Lundbeck'
NAVN
--------------
Treo
Cipramil
```

- Variationer
  - '*' giver "alle attributter"
  - betingelser: brug af AND og OR i WHERE
  - LIKE (tegn-mønstre med '%' og '_')
  - Omdøbning af attribut-navne
  - Aritmetiske udtryk i SELECT og WHERE
  - Sortering ved ORDER BY
  - DISTINCT ...

Datalogi
Roskilde Universitetscenter

DISTINCT-operatoren

- I relationelle model
  - relation = mængde
- I SQL
  - relation = "bag"
    (bag = multimængde
    ~dubletter tilladt)
- løsning
  - dubletter kan elimineres med DISTINCT
    - "Vis Lande"

PID	NAVN	VIRKSOMHED	LAND
668	Kodein "Dak"	Nycomed Danmark	Danmark
671	Pinex	Alpharma	
679	Treo	Lundbeck	Danmark
682	Kodimagnyl "Dak"	Nycomed Danmark	Danmark
1004	Japansk encephalitis	Statens Serum Institut	Danmark
1243	Cipramil	Lundbeck	Danmark
1454	Treosulfan "Medac"	Medac	Sverige
2183	Pinex Comp.	Alpharma	

```
SELECT DISTINCT land
FROM p;

LAND
--------
Danmark
Sverige
```

- Hvad med
  - `SELECT DISTINCT Virksomhed, Land FROM p`
- og
  - `SELECT DISTINCT Land, Virksomhed FROM p`
- Forskel?

Datalogi
Roskilde Universitetscenter

Eksempel-database

Skema **p(pid, navn, virksomhed, gid, land, kkode)**
i(iid, navn)
ip(iid,pid)
g(gid, navn)

P (Præparat)

PID	NAVN	VIRKSOMHED	GID	LAND	KKODE
668	Kodein "Dak"	Nycomed Danmark	227070	Danmark	7
671	Pinex	Alpharma	229060		1
679	Treo	Lundbeck	229084	Danmark	7
682	Kodimagnyl "Dak"	Nycomed Danmark	229086	Danmark	7
1004	Japansk encephalitisvaccine	Statens Serum Institut	315323	Danmark	7
1243	Cipramil	Lundbeck	243058	Danmark	7
1454	Treosulfan "Medac"	Medac	291010	Sverige	
2183	Pinex Comp.	Alpharma	229086		1

IP (Indhold i Præp.)

IID	PID
306	668
812	671
77	679
307	679
77	682
654	682
306	682
2178	1004
283	1243
1012	1454
306	2183
812	2183

I (Indholdsstof)

IID	NAVN
77	Acetylsalicylsyre
283	Citalopram
306	Codein
307	Coffein
654	Magnesiumoxid
812	Paracetamol
1012	Treosulfan
2178	Japansk encephalitisvaccine

G (Terapeutisk gruppe)

GID	NAVN
227070	Analgetika, opioider, svagt virkende, codein,
229060	Analgetika, paracetamol,
229084	Analgetika, svagt virkende, kombinationspræparater uden codein,
229086	Analgetika, svagt virkende, kombinationspræparater med codein,
243058	Antidepressiva, selektive serotoningenoptagshæmmere,
291010	Neoplastiske sygdomme, alkylerende cytostatika,
315323	Vaccinationer ved rejse til udlandet,

Flere relationer

- Typisk introduceres flere relationer i FROM-delen
- hvorved forespørgslen bliver til en ”join”
  - "Vis navnet på gruppen for ’Treo’ "

- tupel-variable
  - p, g
  - opløser flertydigheder

```
  1  SELECT g.navn  Terapeutisk_gruppe
  2  FROM p,g
  3  WHERE p.gid=g.gid
  4* AND p.navn = 'Treo'
SQL> /

TERAPEUTISK_GRUPPE
------------------------------
Analgetika, svagt virkende,
kombinationspræparater uden codein,
```

Database-skema

Præparat:	**p(pid, navn, virksomhed, gid, land, kkode)**
Indholdsstof:	**i(iid, navn)**
Indhold:	**ip(iid,pid)**
Gruppe:	**g(gid, navn)**

Flere relationer

- hvad udtrykker denne:

```
SELECT DISTINCT land
FROM p,ip,i
WHERE p.pid=ip.pid AND ip.iid=i.iid
AND i.navn='Codein'
```

- og denne:

```
SELECT DISTINCT navn
FROM p,ip
WHERE p.pid=ip.pid
```

- Hvorfor "DISTINCT"?

Database-skema

Præparat:	**p(pid, navn, virksomhed, gid, land, kkode)**
Indholdsstof:	**i(iid, navn)**
Indhold:	**ip(iid,pid)**
Gruppe:	**g(gid, navn)**

Datalogi
Roskilde Universitetscenter

Flere relationer, join-betingelser

- eksempel

```
SELECT p.navn FROM p,ip,i
 WHERE p.pid=ip.pid AND ip.iid=i.iid
 AND i.navn='Codein'
```

- join-betingelse,
  - f.eks.

```
p.pid=ip.pid
```

Forenings-, fælles-, og differens-mængde

- ”UNION”, ”INTERSECT”, ”MINUS” (eller ”EXCEPT”)
- anvendes på resultater af to SELECT-sætninger

```
SELECT pid FROM ip WHERE iid=306
UNION
SELECT pid FROM ip WHERE iid=77;

SELECT pid FROM ip WHERE iid=306
INTERSECT
SELECT pid FROM ip WHERE iid=77;

SELECT pid FROM ip WHERE iid=306
MINUS
SELECT pid FROM ip WHERE iid=77;
```

- resultatet er undtagelsesvis IKKE en bag men en mængde
- bag-resultat fås ved
  ”UNION ALL”, ”INTERSECT ALL”, ”MINUS ALL”, f.eks

```
SELECT pid FROM ip WHERE iid=306
UNION ALL
SELECT pid FROM ip WHERE iid=77;
```

IP (Indhold i Præp.)

IID	PID
306	668
812	671
77	679
307	679
77	682
654	682
306	682
2178	1004
283	1243
1012	1454
306	2183
812	2183

Datalogi
Roskilde Universitetscenter

Omdøbning af tupel-variable

```
SELECT p1.navn
FROM p p1,p p2
WHERE p1.land<>p2.land
AND p2.Navn='Treo'
```

- resultat?

PID	NAVN	VIRKSOMHED	GID	LAND	KKODE
668	Kodein "Dak"	Nycomed Danmark	227070	Danmark	7
671	Pinex	Alpharma	229060		1
679	Treo	Lundbeck	229084	Danmark	7
682	Kodimagnyl "Dak"	Nycomed Danmark	229086	Danmark	7
1004	Japansk encephalitisvaccine	Statens Serum Institut	315323	Danmark	7
1243	Cipramil	Lundbeck	243058	Danmark	7
1454	Treosulfan "Medac"	Medac	291010	Sverige	
2183	Pinex Comp.	Alpharma	229086		1

Mængder i betingelser

- Mængder, f.eks.
  - ('Treo','Pinex','Cipramil')
  - (('Treo', 7), ('Pinex', 7))
  - (SELECT navn
    FROM p
    WHERE kkode=1)
- Mængde-udtryk
  - hvor A: mængdeelement og
    M: mængde og θ er en af =, <, >, …
  - A IN M (A er element i M)
  - EXISTS M (M er ikke-tom)
  - A θ M (A θ M gælder)
  - A θ ANY M (A θ M gælder for mindst et element i M)
  - A θ ALL M (A θ M gælder for alle elementer i M)

IN

- IN med direkte specificeret mgd.

```
SELECT navn
FROM p
WHERE land IN ('Danmark','Norge')
```

- IN på delforespørgsel

```
SELECT navn
FROM p
WHERE pid IN
    (SELECT pid
    FROM ip
    WHERE iid=77)
```

  - svarer til en join af p og ip
  - hvilken?

- hvad med:
  - "Vis navn på præparater, der IKKE indeholder stoffet med iid=77 "

ved brug af IN?

Database-skema

Præparat:	**p(pid, navn, virksomhed, gid, land, kkode)**
Indholdsstof:	**i(iid, navn)**
Indhold:	**ip(iid,pid)**
Gruppe:	**g(gid, navn)**

Datalogi
Roskilde Universitetscenter

EXISTS

- EXISTS på delforespørgsel

```
SELECT navn
FROM p
WHERE EXISTS
    (SELECT pid
    FROM ip
    WHERE iid=77 AND pid=p.pid)
```

- Hvad står der her?

```
SELECT navn
FROM i
WHERE NOT EXISTS
    (SELECT *
    FROM p,ip
    WHERE p.pid=ip.pid
    AND iid=i.iid
    AND land='Sverige')
```

Database-skema

Præparat:	**p(pid, navn, virksomhed, gid, land, kkode)**
Indholdsstof:	**i(iid, navn)**
Indhold:	**ip(iid,pid)**
Gruppe:	**g(gid, navn)**

Datalogi
Roskilde Universitetscenter

Reference til ydre SELECT (korrelation)

- bemærk forskel imellem

```
SELECT navn
FROM p
WHERE pid IN
    (SELECT pid
    FROM ip
    WHERE iid=77)
```

- og

```
SELECT navn
FROM p
WHERE EXISTS
    (SELECT pid
    FROM ip
    WHERE iid=77 AND pid=p.pid)
```

- betingelsen
  - **pid=p.pid**
  - refererer til ydre forespørgsel, hvilket kræver gentagen evaluering af indre forespørgsel

Database-skema

Præparat:	**p(pid, navn, virksomhed, gid, land, kkode)**
Indholdsstof:	**i(iid, navn)**
Indhold:	**ip(iid,pid)**
Gruppe:	**g(gid, navn)**

Datalogi
Roskilde Universitetscenter

θ- (f.eks. <, =, >, ...), θ ALL -, θ ANY –operatorer

- "Vis navn på præparater med kkode større end kkoden for 'Pinex'"

```
SELECT navn
FROM p
WHERE kkode >
    (SELECT kkode
    FROM p
    WHERE navn='Pinex')
```

- "Vis navn på præparater med med mindste kkode"

```
SELECT navn
FROM p
WHERE kkode <=ALL
    (SELECT kkode
    FROM p
    WHERE kkode is NOT null)
```

Database-skema

Præparat: **p(pid, navn, virksomhed, gid, land, kkode)**

Indholdsstof: **i(iid, navn)**

Indhold: **ip(iid,pid)**

Gruppe: **g(gid, navn)**

Datalogi
Roskilde Universitetscenter

θ- (f.eks. <, =, >, ...), θ ALL -, θ ANY –operatorer

- Hvad står der her?

```
SELECT navn
FROM p
WHERE pid = any
   (SELECT pid
    FROM ip
    WHERE iid IN
       (SELECT iid
       FROM i
       WHERE navn='Codein'))
```

- og her

```
SELECT navn
FROM p
WHERE NOT land = 'Sverige'
AND kkode = any
     (SELECT kkode
     FROM p
     WHERE virksomhed='Lundbeck')
```

Ækvivalenser indenfor

- [NOT]{IN|EXISTS´|θ |θ ALL|θ ANY}

- IN ?
- NOT IN ?

Database-skema

Præparat:	**p(pid, navn, virksomhed, gid, land, kkode)**
Indholdsstof:	**i(iid, navn)**
Indhold:	**ip(iid,pid)**
Gruppe:	**g(gid, navn)**

Datalogi
Roskilde Universitetscenter

Tupel som element i mængde-udtryk

- ” Vis navn for præparater der ikke indeholder stoffet 306”

```
SELECT navn
FROM p
WHERE pid NOT IN
(SELECT pid FROM ip
 WHERE iid=306)
```

kan skrives

```
SELECT navn
FROM p
WHERE (pid,306) NOT IN
(SELECT pid,iid FROM ip)
```

- med tupel-element:

```
(pid,306)
```

IP (Indhold i Præp.)

IID	PID
306	668
812	671
77	679
307	679
77	682
654	682
306	682
2178	1004
283	1243
1012	1454
306	2183
812	2183

Relationer som operander i FROM

- Som noget nyt i SQL99-standarden:
  - Relations-argumenter (i FROM) kan være SQL-udtryk
  - også introduceret i Oracle

  - f.eks

```
  1  SELECT DISTINCT navn
  2  FROM g, (SELECT gid x
  3       FROM p,i,ip
  4       WHERE p.pid=ip.pid and ip.iid=i.iid and i.navn='Paracetamol') t
  5* WHERE x=gid
SQL> /

NAVN
------------------------------------------------------------
Analgetika, paracetamol,
Analgetika, svagt virkende, kombinationspræparater med codein,
```

  - og dette svarer til?

Datalogi
Roskilde Universitetscenter

SQL-funktioner

- Enkelt-række
- numeriske funktioner
  - **ABS**
  - **COS**
  - **SIN**
  - **LOG**
  - **SIGN**
  - **TRUNC**
  - **...**
- tegn-funktioner
  - **C0NCAT**
  - **INITCAP**
  - **L0WER**
  - **UPPER**
  - **SOUNDEX**
  - **SUBSTR**
  - **LENGTH**
  - **...**
- dato-funktioner
  - **ADD_MONTHS**
  - **LAST_DAY**
  - **MONTHS-BETWEEN**
  - **...**
- konvertering
  - **TO_CHAR**
  - **TO_DATE**
  - **TO_NUMBER**
  - **ROWIDTOCHAR(rowid)**
  - **...**

- Flere rækker / Gruppe
- aggregat -funktioner
  - **AVG**
  - **COUNT**
  - **MAX**
  - **MIN**
  - **SUM**
  - **...**

Datalogi
Roskilde Universitetscenter

Aggregering ...

- Aggregering
  - Brug af aggregat-funktioner
- F.eks.
    - "Vis antallet af præparater"
      - SELECT count(pid) FROM p;
    - eller
      - SELECT count(*) FROM p;
    - eller
      - SELECT count(virksomhed) FROM p;
    - sidste variant uklar?
-
  - kan præciseres
    - SELECT count(DISTINCT virksomhed) FROM p
    -
  - eller
    - SELECT count(ALL virksomhed) FROM p
  - ALL er default
    -

Datalogi
Roskilde Universitetscenter

Aggregering ...

- DISTINCT/ALL relevant ved:
  - COUNT !
- Hvad med
  - SUM ?
  - MAX ?
  - MIN ?
  - AVG ?

Gruppering

- Gruppering
  - Opdeling i grupper af tupler til aggregering
  - Gruppering specificeres ved:

    GROUP BY A1, …, AK
  - F.eks

```
SELECT pid, count(iid)
FROM ip
GROUP by pid
```

  - hvad med?:

```
SELECT land, count(iid)
FROM p, ip
WHERE p.pid=ip.pid
group by land
/
```

IP (Indhold i Præp.)

IID	PID
306	668
812	671
77	679
307	679
77	682
654	682
306	682
2178	1004
283	1243
1012	1454
306	2183
812	2183

Udvælgelse af grupper

- HAVING
  - virker på grupper som WHERE virker på tupler
- f.eks
  - "Vis antal producerede præparater pr. virksomhed"

```
 1  SELECT virksomhed, count(iid)
  2  FROM p, ip
  3  WHERE p.pid=ip.pid
  4* GROUP BY virksomhed
SQL> /

VIRKSOMHED                                         COUNT(IID)
-------------------------------------------------- ----------
Alpharma                                                    3
Lundbeck                                                    3
Medac                                                       1
Nycomed Danmark                                             4
Statens Serum Institut                                      1
```

  - tilføj

```
 5* HAVING count(iid)>1
```

  - Resultat?
  - Hvad med en betingelse som:?
    - WHERE `p.pid=ip.pid` AND `count(iid)` > 1

Datalogi
Roskilde Universitetscenter

SQL join-udtryk (SQL99-standard)

- i FROM kan angives join-udtryk
  - … FROM R1 **CROSS JOIN** R2 …
  - … FROM R1 **JOIN** R2 **ON** <betingelse> …
  - … FROM R1 **NATURAL JOIN** R2 …
  - … FROM R1 **NATURAL LEFT OUTER JOIN** R2 …
  - … FROM R1 **NATURAL RIGHT OUTER JOIN** R2 …
  - … FROM R1 **NATURAL FULL OUTER JOIN** R2 …

SQL join-udtryk (SQL99-standard)

```
SQL> select count(*) from p cross join g;

  COUNT(*)
----------
        56

SQL> select count(*) from p join g on p.gid=g.gid;

  COUNT(*)
----------
         8
```

Datalogi
Roskilde Universitetscenter

Naturlig join

```
SQL> SELECT count(*) FROM p NATURAL JOIN ip;
  COUNT(*)
----------
        12

SQL> SELECT count(*) FROM ip NATURAL JOIN i
  COUNT(*)
----------
        12

SQL> SELECT count(*) FROM p NATURAL JOIN ip NATURAL JOIN I;
SQL> /

  COUNT(*)
----------
         1

 1* SELECT count(*) FROM p NATURAL JOIN ip JOIN i ON ip.iid=i.iid
SQL> /

  COUNT(*)
----------
        12
```

Datalogi
Roskilde Universitetscenter

Ydre join

```
SQL> SELECT * FROM p NATURAL JOIN i

NAVN                            PID VIRKSOMHED                     GID LAND          KKODE  IID
--------------------------- ---------- ------------------------ ---------- -------- ---------- ----
Japansk encephalitisvaccine        1004 Statens Serum Institut       315323 Danmark           7 2178

SQL> SELECT * FROM p NATURAL FULL OUTER JOIN i

NAVN                            PID VIRKSOMHED                     GID LAND          KKODE  IID
--------------------------- ---------- ------------------------ ---------- -------- ---------- ----
Japansk encephalitisvaccine        1004 Statens Serum Institut       315323 Danmark           7 2178
Pinex Comp.                        2183 Alpharma                     229086                   1
Pinex                               671 Alpharma                     229060                   1
Cipramil                           1243 Lundbeck                     243058 Danmark           7
Treosulfan "Medac"                 1454 Medac                        291010 Sverige
Kodein "Dak"                        668 Nycomed Danmark              227070 Danmark           7
Kodimagnyl "Dak"                    682 Nycomed Danmark              229086 Danmark           7
Treo                                679 Lundbeck                     229084 Danmark           7
Acetylsalicylsyre                                                                           77
Paracetamol                                                                                812
Magnesiumoxid                                                                              654
Coffein                                                                                    307
Codein                                                                                     306
Treosulfan                                                                                1012
Citalopram                                                                                 283

15 rækker er valgt.
```

Datalogi
Roskilde Universitetscenter

Ydre join

```
SQL> SELECT * FROM p NATURAL LEFT OUTER JOIN i

NAVN                          PID VIRKSOMHED                    GID LAND          KKODE  IID
--------------------------- ---------- ----------------------- ---------- -------- ---------- ----
Japansk encephalitisvaccine        1004 Statens Serum Institut       315323 Danmark           7 2178
Pinex Comp.                        2183 Alpharma                     229086                   1
Pinex                               671 Alpharma                     229060                   1
Cipramil                           1243 Lundbeck                     243058 Danmark           7
Treosulfan "Medac"                 1454 Medac                        291010 Sverige
Kodein "Dak"                        668 Nycomed Danmark              227070 Danmark           7
Kodimagnyl "Dak"                    682 Nycomed Danmark              229086 Danmark           7
Treo                                679 Lundbeck                     229084 Danmark           7

8 rækker er valgt.

SQL> SELECT * FROM p NATURAL RIGHT OUTER JOIN i

NAVN                          PID VIRKSOMHED                    GID LAND          KKODE  IID
--------------------------- ---------- ----------------------- ---------- -------- ---------- ----
Japansk encephalitisvaccine        1004 Statens Serum Institut       315323 Danmark           7 2178
Acetylsalicylsyre                                                                            77
Paracetamol                                                                                 812
Magnesiumoxid                                                                               654
Coffein                                                                                     307
Codein                                                                                      306
Treosulfan                                                                                 1012
Citalopram                                                                                  283

8 rækker er valgt.
```

Datalogi
Roskilde Universitetscenter

Opdatering i databasen

- Indsæt enkelt tupel
  - INSERT INTO R VALUES (V1, …, Vk)
- Indsæt mængde af tupler
  - INSERT INTO R (SELECT-sætning)
- Slet mængde af tupler
  - DELETE FROM R
  - WHERE <betingelse>
- Ændring af tupler
  - UPDATE R
  - SET A1=V1, …, An=Vn
  - WHERE <betingelse>

Definition af databasen

```
CREATE TABLE p (
  pid           INTEGER,
  navn          VARCHAR2 (70),
  virksomhed    VARCHAR2 (70),
  gid           INTEGER,
  land   VARCHAR2 (8),
  kkode         INTEGER,
  UNIQUE (navn),
  PRIMARY KEY(pid),
  FOREIGN KEY (gid) REFERENCES g (gid))
/
```

denne er vigtig


Datalogi
Roskilde Universitetscenter

Data definition i SQL – i øvrigt

- Data definition i SQL
  - primært ved
    - CREATE X <navn>
    - ALTER X <navn>
    - DROP X <navn>
  - hvor X er
    - DATABASE
    - TABLE (*)
    - VIEW (*)
    - INDEX (*)
    - FUNCTION
    - PROCEDURE
    - TRIGGER
    - CONSTRAINT
    - ...
  - bemærk ALTER er speciel, strengt taget overflødig og mangler på enkelte steder
- Der er mange flere, primært "administrative" operationer til styring af
  - skema og database detaljer
  - forespørgsels-optimering
  - brugere og rettigheder
  - pladsforbrug
  - sikkerhed

Datalogi
Roskilde Universitetscenter

SQL-DDL: Flere eksempler

- ALTER TABLE

```
ALTER TABLE p ADD (nykode smallint)
ALTER TABLE p ADD (unique (navn))
ALTER TABLE p DROP (unique (navn))
```

- DROP TABLE

```
DROP TABLE ip;
DROP TABLE p;
DROP TABLE g;
DROP TABLE i;
```

  - hvorfor denne rækkefølge?
  - Alternativ:

```
DROP TABLE i CASCADE CONSTRAINTS;
DROP TABLE ip CASCADE CONSTRAINTS;
DROP TABLE g CASCADE CONSTRAINTS;
DROP TABLE p CASCADE CONSTRAINTS;
```

SQL-DDL: Flere eksempler

- default-værdier

```
CREATE TABLE test(i int);

INSERT INTO test VALUES (1);

ALTER TABLE test ADD t char(6) DEFAULT 'ukendt';

INSERT INTO test VALUES (2,'flot');

SELECT * FROM test;
          I T
---------- ------
          1 ukendt
          2 flot
```

SQL-DDL: Flere eksempler

CREATE INDEX

- opret indeks på navn i p

```
CREATE INDEX navn_index ON p(navn)
```

- opret indeks på kkode,navn i p

```
CREATE INDEX kkodenavn_index ON p(kkode,navn)
```

View

- Et view er en virtuel relation
  - Når der forespørges på et view, sørger databasesystemet for at modificere forespørgslen til en forespørgsel på "ægte" relationer
- Eksempel

- Opret view
  - pliste(pnavn,inavn)
  - over kombinationer af præparater og indholdsstoffer
- view-definition?

```
1  CREATE VIEW p306 AS
  2  SELECT navn,land,kkode
  3  FROM p NATURAL JOIN ip
  4* WHERE iid=306
SQL> /
SQL> SELECT * FROM p306;

NAVN                           LAND            KKODE
------------------------------ -------- ----------
Kodein "Dak"                   Danmark              7
Kodimagnyl "Dak"               Danmark              7
Pinex Comp.                                         1
```

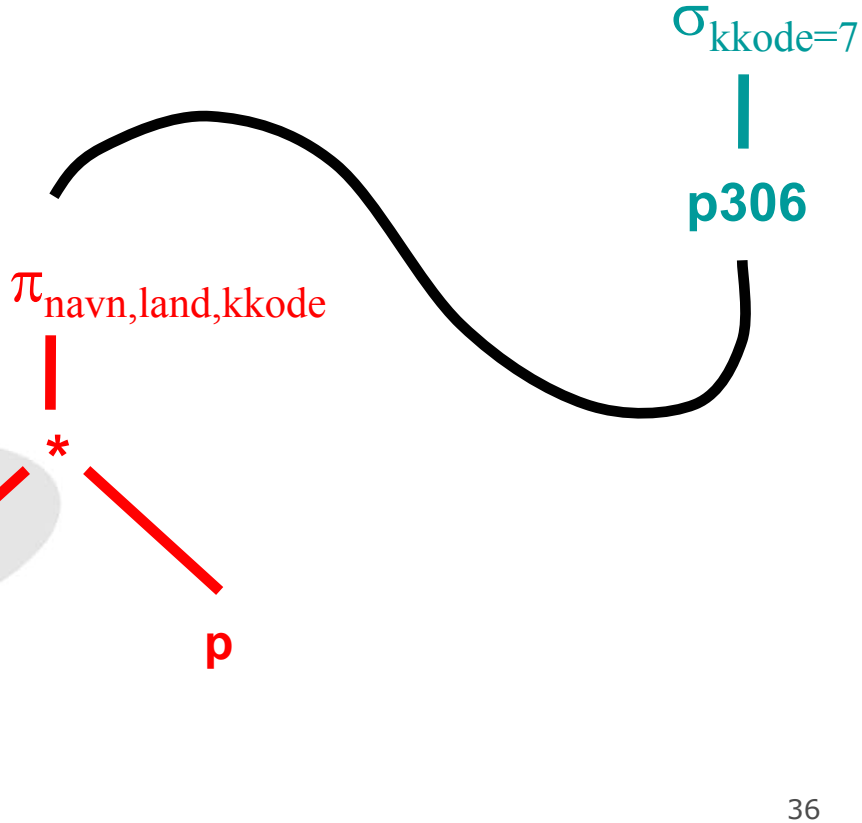
View ...

- Forespørgsler der involverer views
  - forspørgslen ”flettes” med view-definitionen
  - princippet er simpelt og illustreres bedst ved omformning til algebra
- Eksempel

```
CREATE VIEW p306 as
SELECT navn,land,kkode
FROM p NATURAL JOIN
ip
WHERE iid=306
```

- forespørgsel

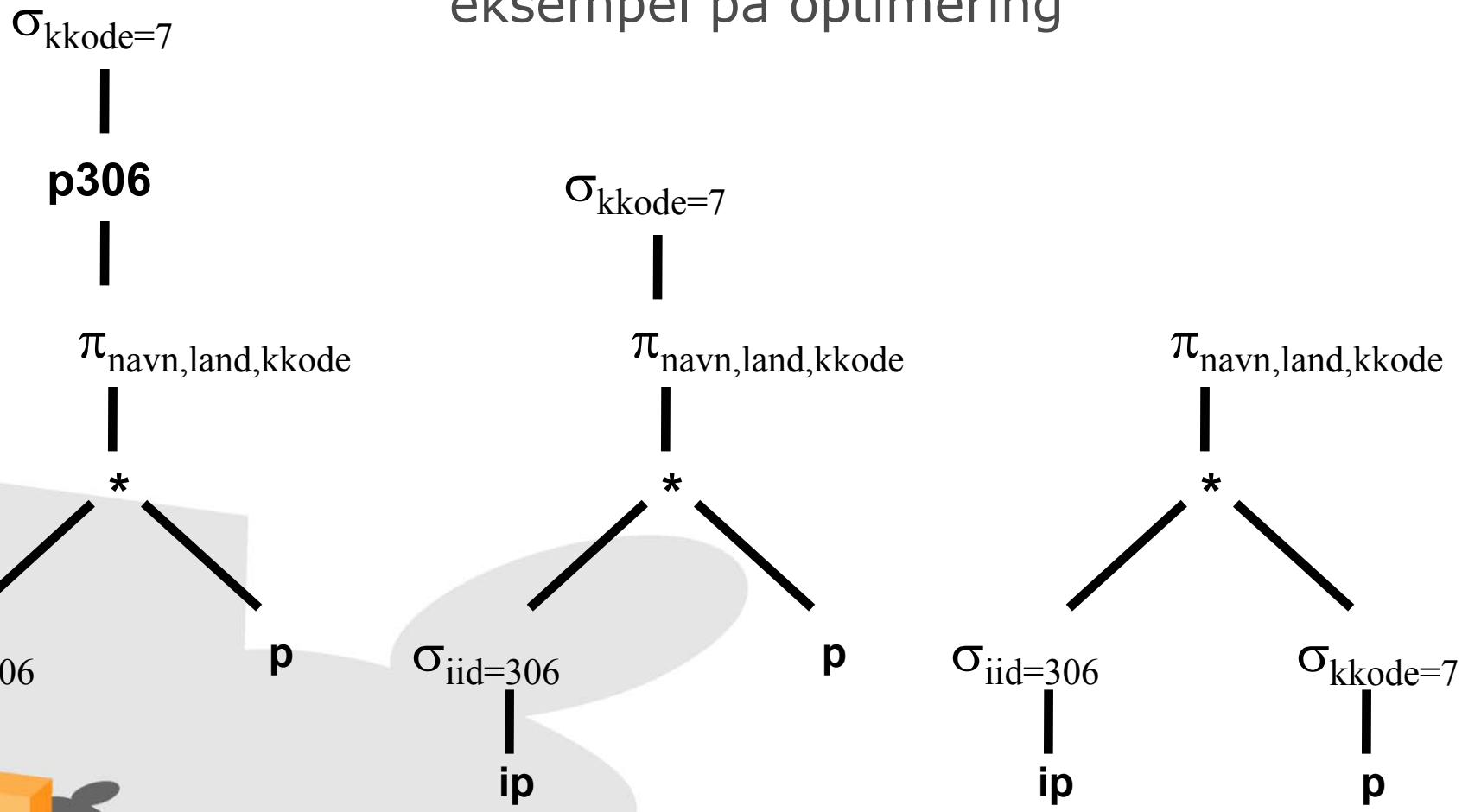
```
SELECT navn
FROM p306
WHERE kkode=7
```



* er her naturlig join

View ...

- Fletning af view og forespørgsel + eksempel på optimering



Datalogi
Roskilde Universitetscenter

View ...

- Opdatering på views
  - generelt et ”svært” problem
  - derfor ofte ulovligt, f.eks

```
SQL> CREATE VIEW opdateringikkeok AS
  SELECT pid,sum(iid) antal FROM ip GROUP BY pid;
SQL> INSERT INTO opdateringikkeok VALUES(812,17);
INSERT INTO opdateringikkeok VALUES(812,17)
            *
FEJL i linie 1:
ORA-01732: datamanipulation er ikke tilladt på dette view
```

  - men lovligt hvis opdatering entydig, f.eks

```
SQL> CREATE VIEW opdateringok AS SELECT pid FROM p;
SQL> INSERT INTO opdateringok VALUES(813);
1 række er oprettet.
SQL>
```

Datalogi
Roskilde Universitetscenter

Databaseskemaet gemmes i databasen

- En SQL-database indeholder sin egen definition
- Man kan derfor spørge på skema som man spørger på forekomst
- I ORACLE er relationers skemaer repræsenteret (bl.a.) i:
    - USER_TABLES
    - USER_TAB_COLUMNS

```
SELECT table_name
FROM user_tables

SELECT table_name,column_name
FROM user_tab_columns
```

Datalogi
Roskilde Universitetscenter

Databaseskemaet gemmes i databasen ...

```
SQL> SELECT table_name
  2  FROM user_tables
  3  /
TABLE_NAME
-------------------------
G
I
IP
P
4 rækker er valgt.
```

```
SQL> SELECT table_name,column_name
  2  FROM user_tab_columns
  3  /
TABLE_NAME                     COLUMN_NAME
------------------------------ -------
G                              GID
G                              NAVN
I                              IID
I                              NAVN
IP                             IID
IP                             PID
P                              PID
P                              NAVN
P                              VIRKSOMHED
P                              GID
P                              LAND
P                              KKODE
12 rækker er valgt.
SQL>
```

Datalogi
Roskilde Universitetscenter

Null-værdier og 3-værdi logik

- **Null-værdi**
  - værdi der ikke er specificeret
  - værdi der ikke giver mening
- **Regning med Null**
  - hvis X er Null
    - 1+X er Null
    - X=3 har sandhedsværdi ”ukendt” (eller ”muligvis”)

```
SQL> SELECT * FROM t1;
         A          B
---------- ----------
         1          2
         1          3
         1
         1          4
SQL> SELECT * FROM t1 WHERE B is NULL;
         A          B
---------- ----------
         1
```

```
SQL> SELECT COUNT(A) FROM t1;
  COUNT(A)
----------
         4

SQL> SELECT COUNT(B) FROM t1;
  COUNT(B)
----------
         3

SQL> SELECT COUNT(*) FROM t1;
  COUNT(*)
----------
         4

SQL> SELECT COUNT(*) FROM t1 WHERE
B<=2 or B>2;
  COUNT(*)
----------
         3
```

•OK?

Datalogi
Roskilde Universitetscenter

3-værdi logik

- Sammenligning med Null giver
  - ”ukendt” / ”muligvis”,
  - altså en 3’die sandhedsværdi
- Evaluering af logiske udtryk
  - kræver afklaring af logik for de 3 værdier:
    - sand, falsk, muligvis
- 3-værdi logik
  - sæt
    - 0 = falsk, 1 = sand og 0,5 = muligvis
  - X og Y er 3-værdi logiske udtryk
    - X **and** Y = **min**(X,Y)
    - X **or** Y = **max**(X,Y)
    - **not** X = **1 - X**

- Afledt sandhedstabel

X	Y	X and Y	X or Y	not X
1	1	1	1	0
1	0,5	0,5	1	0
1	0	0	1	0
0,5	1	0,5	1	0,5
0,5	0,5	0,5	0,5	0,5
0,5	0	0	0,5	0,5
0	1	0	1	1
0	0,5	0	0,5	1
0	0	0	0	1

- Bemærk
  - dette minder om (og kunne være starten på)
  - Fuzzy Logik
    - en ”mange-værdi” logik
    - der graduerer flydende imellem 0 og 1