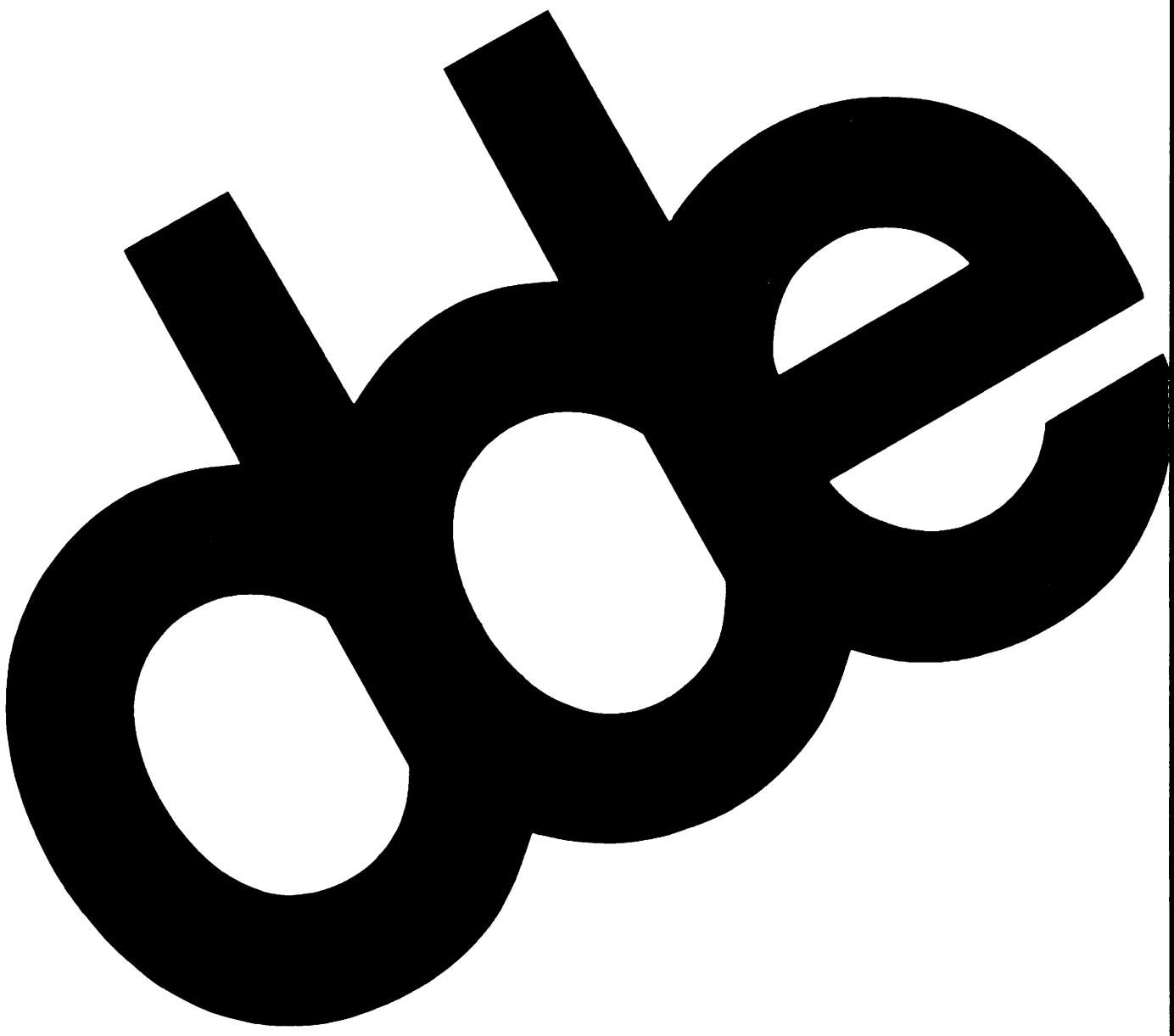
Oracle SQL*Plus



December 1994

Indholdsfortegnelse

1. Introduktion 1
1.1 SQL og SQL*Plus 2
1.2 Kald af SQL*plus-programmet 2
1.3 Tabeloversigt 3

2. SQL select 5
2.1 Søge-betingelser 6

Øvelse 1 8

3. Redigering af SQL-sætning 9

4. Udførelse af UNIX-kommandoer 10

5. Sortering 11

6. Regneudtryk 12

7. Alias 13

Øvelse 2 14

8. Rækkefunktioner 15

Øvelse 3 19

9. Gruppefunktioner 21

Øvelse 4 23

10. Join af tabeller 25
10.1 EQUI join 25
10.2 Join af én tabel med sig selv 26
10.3 NON EQUI join 26
10.4 OUTER join 27

Øvelse 5 29

11. Subquery 31
11.1 Subquery der returnerer mere end én værdi 32
11.2 Subqueries mod flere kolonner 34

Øvelse 6 35

12. Generering af SQL-sætninger 37

13. Udveksling af data 39

Øvelse 7 41

14. Manipulation af tabelrækker 43
14.1 Indsættelse af række 43
14.2 Ændring af eksisterende række 44
14.3 Sletning af eksisterende rækker 45
14.4 Commit og rollback 45
14.5 Savepoints 46

Øvelse 8 47

15. Opret/slet tabel 49
15.1 Ændring af tabel 50

16. Opret/slet view 51
16.1 Ændring af view 52

Øvelse 9 53

17. Optimering med indeks 55
17.1 Definition af indeks 55
17.2 Oracles brug af indeks 56
17.3 Sletning af indeks 56

Øvelse 10 57

18. Sequence 59

19. Comments 61

Øvelse 11 63

20. Udskrivning af rapporter 65

Øvelse 12 73

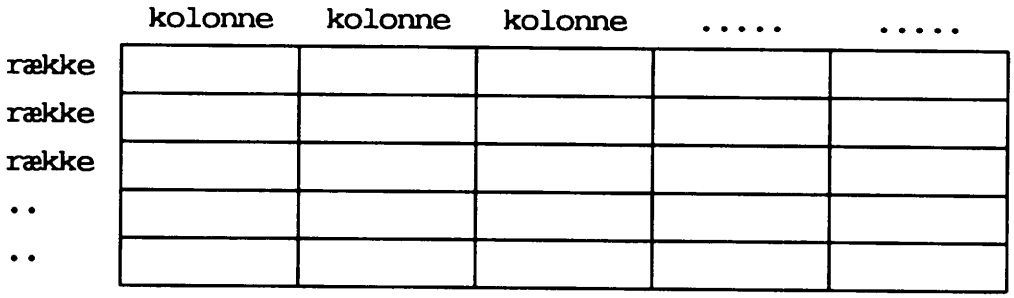
Resumé - kommandoer 75

Appendix A 76

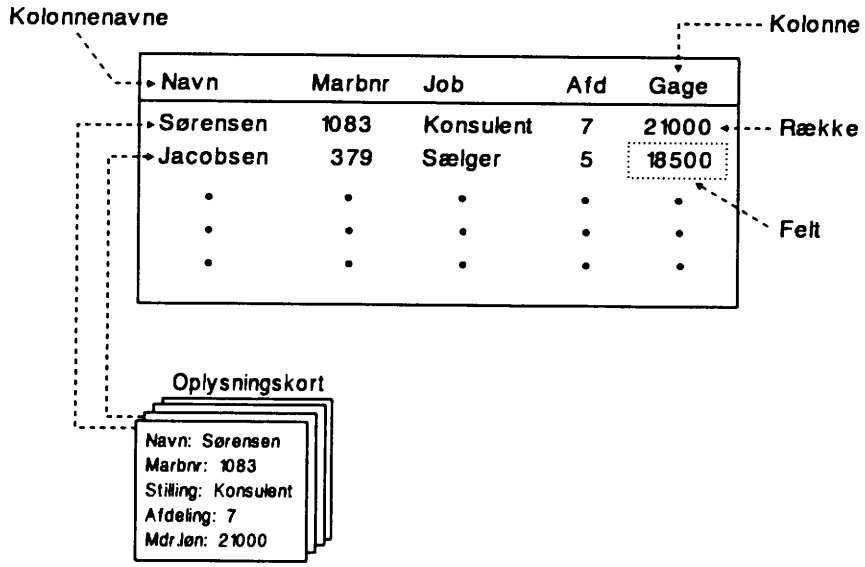
Stikordsregister 77

1. Introduktion

En database er en mængde systematiserede informationer. Oracle-databasen er en relations-database. For brugeren fremstår en relations-database som en samling af tabeller. Hver tabel består af et bestemt antal kolonner samt et antal usorterede rækker.



Informationer organiseres i tabeller som illustreret nedenfor:



* **Hver kolonne indeholder netop én type information.**
* **Hver række indeholder ét sæt af sammenhørende informationer.**
* **Hvert felt indeholder én information.**

1.1 SQL og SQL*Plus

Hvad er SQL

* SQL (Structured Query Language) er et database-sprog, der benyttes til at få vist og bearbejdet data i en relationel database.
* SQL er et programmerings-sprog, der tilstræber at ligne det engelske talesprog.
* SQL-sproget er en international standard (ISO-standard 9075).

Hvad er SQL*Plus

* SQL*Plus er den variant af SQL-standarden, der benyttes i Oracle. Ud over kommandoerne i SQL-standarden indeholder SQL*Plus en række tillægs-kommandoer.

En oversigt over SQL- og SQL*Plus-kommandoer kan findes på side 75.

1.2 Kald af SQL*plus-programmet

SQL*Plus-programmet kaldes med følgende kommando:

$ORACLE_HOME/bin/sqlplus <brugernavn>

<password>

SQL*Plus kvitterer med prompten 'SQL>', der indikerer at man er logget på Oracle-databasen.

SQL*plus-programmet afbrydes med EXIT eller CTRL D.

1.3 Tabeloversigt

Efter logon på databasen kan man få en oversigt over sine tabeller i databasen. Som alle andre informationer i databasen er tabel-oversigten placeret i en tabel. Tabellen har navnet USER_TABLES (TABS).

Kommandoen til søgning af data i tabeller hedder SELECT. Følgende select-sætning giver en oversigt over alle tabeller ejet af den aktuelle bruger.

```
SELECT table_name FROM user_tables;
```

På dette kursus vil oversigten på nuværende tidspunkt indeholde følgende tabeller:

```
TABLE_NAME
------------------------------
CUSTOMER
DEPT
EMP
ITEM
ORD
PRICE
PRODUCT
WORDS
```

Tabellerne er Oracle's demotabeller og bliver brugt i manualernes eksempler. En udskrift af tabellerne findes på løse blade.

Der er kopieret et sæt tabeller ud til hver bruger.

2. SQL select

Syntaksen for en simpel select-sætning er:

```
SELECT <kolonnenavn>,<kolonnenavn>, ... ,<kolonnenavn>
FROM   <tabelnavn>
```

I SELECT-linien angives hvilke kolonner der ønskes vist. Den rækkefølge kolonnerne angives i, er også den rækkefølge rækkefølge de udskrives i. Erstattes kolonnenavnene af en *, bliver alle kolonner udskrevet.

I FROM-linien angives hvilken tabel kolonnerne skal findes i.

EKSEMPEL

```
SELECT ename,job,deptno
FROM   emp;
```

ENAME	JOB	DEPTNO
SMITH	CLERK	20
ALLEN	SALESMAN	30
WARD	SALESMAN	30
JONES	MANAGER	20
MARTIN	SALESMAN	30
BLAKE	MANAGER	30
CLARK	MANAGER	10
SCOTT	ANALYST	20
KING	PRESIDENT	10
TURNER	SALESMAN	30
ADAMS	CLERK	20
JAMES	CLERK	30
FORD	ANALYST	20
MILLER	CLERK	10

2.1 Søge-betingelser

Syntaksen for en select-sætning med række-betingelse(r) er:

```
SELECT <kolonnenavn>,<kolonnenavn>, ... ,<kolonnenavn>
FROM   <tabelnavn>
WHERE  rækkebetingelse(r)
```

I WHERE-linien kan man opstille logiske udtryk til at udvælge rækker fra en tabel. Følgende logiske operatorer kan benyttes:

Operator	Funktion
=	Test af lighed.
!= ^= <>	Test af forskellig fra.
> >= < <=	Større end, større end eller lig med, mindre end, mindre end eller lig med.
NOT IN	Ikke indeholdt i.
IN	Indeholdt i.
ANY	Sammenholder en værdi med hver værdi der returneres fra en liste eller en underforespørgsel.
ALL	Sammenholder en værdi med alle værdier der returneres fra en liste eller en underforespørgsel.
[NOT] BETWEEN x AND y	[Ikke] større end eller lig med x og mindre end eller lig med y.
EXIST	Sand hvis en underforespørgsel returnerer mindst én række.
[NOT] LIKE	Matcher [eller matcher ikke] et søgemønster. Jokertegnene '%' og '_' kan anvendes.
IS [NOT] NULL	Er [ikke] null-værdi.
NOT	Negerer et logisk udtryk.
AND	Kombinerer logiske udtryk til værdien sand hvis begge er sande.
OR	Kombinerer logiske udtryk til værdien sand hvis mindst ét af udtrykkene er sandt.

I select-sætninger mod emp-tabellen kan eksempelvis vælges følgende betingelser:

```
WHERE job = 'PRESIDENT'
WHERE hiredate >= '01-JAN-80'
WHERE job = 'CLERK' or sal >= 2000
WHERE sal between 1000 and 2000
WHERE ename like 'A%' or ename like '%LL%'
WHERE comm is not null
WHERE job in ('ANALYST','MANAGER','SALESMAN')
WHERE (deptno=10 or deptno=30) and job='CLERK'
```

Kommentarer:

* Tekster og datoer skal omgives af enkelt-apostroffer.
* SQL skelner mellem små og store bogstaver i WHERE CLAUSEN.
* Operatoren LIKE bruges ifm. wildcards.
  % erstatter et vilkårligt antal tegn eller slet ingen.
  _ erstatter netop ét tegn.
* AND udføres før OR. Kan ændres vha. parenteser.

Øvelse 1

Formål: - at benytte enkle søgebetingelser

1. **Udskriv navn, stilling og ansættelsesdato for alle ansatte i afdeling 20.**

2. **Udskriv en liste over de medarbejdere fra afdeling 30 der ikke har en provision.**

3. **Udskriv navn, stilling og løn for de managers og analysts der tjener mere end 2900 pr. måned.**

4. **Udskriv nummer, navn og adresse på de kunder hvor 'sport' er en del af kundenavnet.**

5. **Udskriv en liste der viser produktid og aktuel standardpris.**

6. **Udskriv produktnummer og produktbeskrivelse på de produkter hvis produktnummer ligger mellem 100870 og 102000, begge produktnumre medregnet.**

3. Redigering af SQL-sætning

Den sidst skrevne SQL-sætning gemmes i bufferen afiedt.buf. Bufferen indhold kan redigeres enten vha. en linieeditor eller vha. en skærmeditor, fx vi.

LINIE-EDITOR

Aktiv linie markeres med en *.

LIST-KOMMANDOER:	**l**	- List SQL-sætning
	3	- List 3. linie af SQL-sætning
ÆNDRINGS-KOMMANDOER:	**c/<gl tekst>/<ny tekst>**	- Erstat gl-tekst med ny-tekst
	del	- Slet aktiv linie
INDSÆT-KOMMANDOER:	**i**	- Indsæt ny linie efter aktiv linie
	a	- Tilføj til aktiv linie

SKÆRM-EDITOR

Default editoren defineres i login.sql-filen. Ønsker man fx at bruge vi, indsættes flg. linie : **def_editor=/usr/bin/vi**

eller set EDITOR=/usr/bin/vi

Indholdet af bufferen hentes ind i den definerede editor med kommandoen ED.

Indholdet af en fil hentes ind i den definerede editor med kommandoen ED <filnavn>.

FÆLLES-KOMMANDOER

AFVIKLINGS-KOMMANDOER:	**/**	- Udfør SQL-sætning
	run	- List og udfør SQL-sætning
	@<fil>	- Hent og udfør SQL-sætning
SAVE/GET-KOMMANDOER:	**save <fil>**	- Gemmer SQL-sætning i <fil>.sql
	get <fil>	- Henter <fil>.sql til buffer
		Filnavn maks. 10 tegn.

4. Udførelse af UNIX-kommandoer

UNIX-kommandoer fra SQL*Plus udføres ved at bruge '!' eller '**host**' som 'prompttegn' foran den egentlige UNIX-kommando:

EKSEMPLER:

!ls -x *.sql
Lister SQL-kommandofiler i eget katalog.

! [RETUR]
Kalder en sub-shell hvorfra UNIX-kommandoer så kan udføres.

!lp -d<printernavn> <filnavn>
Udskriver fil til printer.

5. Sortering

Rækkerne lagres usorteret i de enkelte tabeller. Ønskes rækkerne vist i en bestemt sorteringsorden, skal det derfor angives i søgningen.

Syntaksen for en select-sætning med sortering er:

```
SELECT   <kolonnenavn>,<kolonnenavn>, ... ,<kolonnenavn>
FROM     <tabelnavn>
WHERE    rækkebetingelse(r)
ORDER BY <kolonnenavn>,<kolonnenavn>, ... ,<kolonnenavn>
```

Hvis der angives flere sorteringskolonner, sorteres der primært efter den først angivne kolonne,sekundært efter den næste osv.

Sortering udføres default i stigende orden. Sortering udføres i faldende orden hvis der efter kolonnenavnet skrives DESC (decending).

EKSEMPEL
Udskriv kundenavn og kreditgrænse for alle kunder. Listen sorteres efter kreditgrænse i faldende orden og kundenavn i stigende orden:

```
SELECT   name,creditlimit
FROM     customer
ORDER BY creditlimit desc,name
```

NAME	CREDITLIMIT
EVERY MOUNTAIN	10000
TKB SPORT SHOP	10000
WOMENS SPORTS	10000
NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER	8000
VOLLYRITE	7000
SHAPE UP	6000
JOCKSPORTS	5000
K + T SPORTS	5000
JUST TENNIS	3000

Kommentarer:

* Der kan sorteres på kolonner der ikke er repræsenteret i uddata.
* Kolonner der sorteres i stigende orden vil vise null-værdier sidst.

6. Regneudtryk

Regneudtryk kan benyttes i SELECT-, WHERE- og ORDER BY-linierne. Benyttes regneudtryk i SELECT-linien vises resultatet af beregningen i uddata. Benyttes regneudtryk i WHERE-linien indgår beregningen i rækken af betingelser. Benyttes regneudtryk i ORDER BY-linien bruges resultatet i sorteringen af uddata.

EKSEMPEL

Udskriv navn, løn, provision og løn+provision for alle sælgere hvis samlede løn er større end 1700. Listen sorteres efter samlet løn:

```
SELECT   ename,sal,comm,sal+comm
FROM     emp
WHERE    job='SALESMAN' and sal+comm>1700
ORDER BY sal+comm;
```

```
ENAME          SAL       COMM   SAL+COMM
------ ---------- ---------- ----------
WARD      1,250.00     500.00   1,750.00
ALLEN     1,600.00     300.00   1,900.00
MARTIN    1,250.00   1,400.00   2,650.00
```

Indgår et felt med værdien null i beregningen, returneres værdien null. Funktionen NVL kan tildele null-felter en værdi. Syntaksen for funktionen NVL er:

NVL(<kolonnenavn>,<værdi hvis feltets værdi er null>)

EKSEMPEL

Udskriv navn,job og samlet løn for alle i afdeling 30 der tjener mere end 1800 ialt:

```
SELECT ename,job,sal+nvl(comm,0)
FROM   emp
WHERE  deptno=30 and sal+nvl(comm,0)>1800;
```

```
ENAME  JOB       SAL+NVL(COMM,0)
------ --------- ---------------
ALLEN  SALESMAN         1,900.00
BLAKE  MANAGER          2,850.00
MARTIN SALESMAN         2,650.00
```

7. Alias

Opstilling af fx regneudtryk medfører at selve regneudtrykket bliver kolonneoverskrift. Ved brug af ALIAS kan kolonneoverskriften ændres. Alias angives i umiddelbar forlængelse af kolonnenavnet.

EKSEMPEL

Udskriv navn, stilling og samlet løn for de ansatte i afdeling 30. Tilpas kolonneoverskrifterne vha. alias:

```
SELECT ename efternavn,job stilling,sal+nvl(comm,0) "SAMLET LØN"
FROM   emp
WHERE  deptno=30;
```

EFTERNAVN	STILLING	SAMLET LØN
ALLEN	SALESMAN	1900
BLAKE	MANAGER	2850
JAMES	CLERK	950
MARTIN	SALESMAN	2650
TURNER	SALESMAN	1500
WARD	SALESMAN	1750

Kommentarer:

* Uden dobbelte anførselstegn omkring vises kolonneoverskriften med store bogstaver.
* Med dobbelte anførselstegn omkring vises kolonneoverskriften præcis som angivet.
* Anvendes mellemrum og/eller specielle tegn, herunder punktum, skal alias angives med dobbelte anførselstegn omkring.

Kolonneoverskrifter kan også ændres med COLUMN-kommandoen, som omtales senere i kursusforløbet.

Øvelse 2

Formål: - *at opstille regneudtryk*
- *at benytte alias*
- *at benytte sortering*

1. **Udskriv stilling, navn og afdelingsnummer for alle ansatte. Listen sorteres efter stilling og navn. De enkelte kolonneoverskrifter tilpasses vha. alias.**

2. **Udskriv navn, stilling og samlet løn pr. kvartal for alle ansatte hvis samlede løn pr. kvartal overstiger 5000. Listen sorteres efter samlet løn pr. kvartal i faldende orden.**

3. **Udskriv en liste der viser hvor mange procent provisionen udgør af den enkelte sælgers samlede løn. Tilpas kolonneoverskrifterne vha. alias.**

4. **Udskriv en liste der for hvert produktid viser, hvor mange procent mindsteprisen er mindre end den aktuelle standardpris.**

5. **Udskriv en liste der viser hver stillingsbetegnelse netop én gang.**

   Find en passende operator s. 4-8 i SQL Language Reference Manual.

select distinct job from emp
↓
netop en gang

8. Rækkefunktioner

I SQL findes en række funktioner, der opdeles i forskellige grupper afhængig af, hvad de udfører:

1) **Aritmetiske funktioner**	Input nummerisk / output nummerisk SQL Language Reference Manual s. 4-12
2) **Karakterfunktioner**	Input tekst / output tekst, nummerisk SQL Language Reference Manual s. 4-14
3) **Konverteringsfunktioner**	Konverterer en værdi fra en datatype til en anden datatype. SQL Language Reference Manual s. 4-23
4) **Datofunktioner**	Input datoformat / output datoformat Undtaget er MONTHS_BETWEEN SQL Language Reference Manual s. 4-26
5) **Andre funktioner**	Blandede funktioner der ikke kan grupperes SQL Language Reference Manual s. 4-30

Disse funktioner kaldes rækkefunktioner, idet de returnerer én værdi for hver udvalgt række. Rækkefunktioner kan anvendes i SELECT-, WHERE- og ORDER BY-linierne.

Syntaksen for en funktion er:

```
Funktion(Argument1,[Argument2], ... )
```

EKSEMPEL1

Udskriv en liste med navn og stilling. Kun forbogstaverne skal skrives med stort.

```
SELECT initcap(ename) navn,initcap(job) stilling
FROM   emp
WHERE  lower(job)='salesman';
```

```
NAVN       STILLING
---------- ---------
Allen      Salesman
Ward       Salesman
Martin     Salesman
Turner     Salesman
```

SQL Language Reference Manual 4-15.

EKSEMPEL2

Udskriv ordrenummer og ordretotal i nærmeste hundrede for ordrer, der er afsendt i februar 1987.

```
SELECT ordid ordnr,round(total,-2) ordretotal
FROM   ord
WHERE  substr(shipdate,4,6)='FEB-87';
```

```
  ORDNR   ORDRETOTAL
------- ------------
    613         6400
    614        23900
    615          700
    616          800
    619         1300
```

SQL Language Reference Manual 4-13, 4-18.

EKSEMPEL3

Udskriv navn og dato for hver medarbejders 15 års jubilæum. Datoen udskrives i formatet '12.06.1993' og sorteres i stigende orden.

```
COLUMN "15 ÅRS JUBILÆUM" FORMAT a15
SELECT   ename navn,
         to_char(add_months(hiredate,12*15),'dd.mm.yyyy')
         "15 ÅRS JUBILÆUM"
FROM     emp
ORDER BY hiredate;

NAVN       15 ÅRS JUBILÆUM
---------- ---------------
SMITH      17.12.1995
ALLEN      20.02.1996
WARD       22.02.1996
JONES      02.04.1996
  :            :
  :            :
ADAMS      12.01.1998
```

SQL*Plus User Guide and Reference 6-17
SQL Language Reference Manual 4-25 —> 4-36, 4-26

EKSEMPEL4

Udskriv en afdelingoversigt i én kolonne. || ~ konkatenation

```
COLUMN afdelingsoversigt FORMAT a70
SELECT 'Afdeling '||deptno||': '||initcap(dname)||', '
       ||initcap(loc) afdelingsoversigt
FROM   dept;

AFDELINGSOVERSIGT
------------------------------------------------------------
Afdeling 10: Accounting, New York
Afdeling 20: Research, Dallas
Afdeling 30: Sales, Chicago
Afdeling 40: Operations, Boston
```

SQL Language Reference Manual 4-3.

EKSEMPEL5

Kunderne skal vises opdelt i kreditgrupper, der skifter for hver 4000 der må gives i kredit.

```
COLUMN name FORMAT a55
SELECT custid,name,
       decode(trunc(creditlimit/4000),0,1,1,2,2,3,3,4,5)
       kreditgruppe
FROM   customer;
```

CUSTID	NAME	KREDITGRUPPE
100	JOCKSPORTS	2
101	TKB SPORT SHOP	3
102	VOLLYRITE	2
103	JUST TENNIS	1
104	EVERY MOUNTAIN	3
105	K + T SPORTS	2
106	SHAPE UP	2
107	WOMENS SPORTS	3
108	NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER	3

SQL Language Reference Manual 4-30.

EKSEMPEL6

Udskriv kundenummer og bynavn. Bynavn skal centreres i udskriften.

```
COLUMN city FORMAT a15
COLUMN city JUS(tify) C(enter)
SELECT lpad(city,15-round((15-length(city))/2)) city
FROM   customer;
```

CUSTID	CITY
100	BELMONT
101	REDWOOD CITY
102	BURLINGAME
103	BURLINGAME
104	CUPERTINO
105	SANTA CLARA
106	PALO ALTO
107	SUNNYVALE
108	HIBBING

SQL*Plus User Guide and Reference 6-17
SQL Language Reference Manual 4-15, 4-19.

Øvelse 3

Formål: - at benytte rækkefunktioner

select sysdate from dual
SYSDATE
10-AUG-95

1. **Udskriv dagsdato i nedenstående format. Brug pseudo-kolonnen sysdate.** column "DAGSDATO"

   DAGSDATO

   Monday 19/07 1993

2. **Udskriv navn og det hele antal mdr. hver enkelt medarbejder har været ansat.** months_between(mnd1, mnd2

3. **Udskriv navn, stilling og ansættelsesdato for de medarbejdere, der har været ansat mere end 145 mdr. pr. dagsdato.** 160

4. **Udskriv medarbejdernummer og navn i én kolonne som vist nedenfor. Udskriften skal sorteres efter navn.**

   Ansatte

   7876 - Adams

5. **Uskriv navn og stillingsbetegnelse for alle ansatte. SALESMAN skal udskrives som SÆLGER, MANAGER som AFDELINGSLEDER og resten uændret.**

6. **Lønningerne skal inddeles i forskellige løngrupper og skrives ud. Løngrupperne kan fx være i spring på 1000, 2000 etc. Vælg selv.**

7. **Udskriv for alle ansatte navn, dato for det tidpunkt hvor hver enkelt har/havde været ansat i 12 år samt om det er før, på eller efter dagsdato.** *Tip: fkt. SIGN s. 4-13.*

NAVN	12 ÅR PR.	AKTUALITET
ADAMS	12-JAN-95	Efter dagsdato

   SIGN(SYSDATE -)

Fortsættes næste side.

8. **Udskriv navnene på de ansatte hvis navn enten er noget i retning af Smit, Smyte, Schmidt, Smid ... eller noget i retning af Skot, scot, Skodt ...** Soundex(enavn column) = Soundex ('Sm%')

9. **Udskriv navn, løn og provision for alle sælgere. I provisionskolonnen skal tallet 0 vises som null. Kolonnen skal beholde sit visningsformat.**

10. **Udskriv kundenummer, kundenavn og tlfnr. som vist nedenfor.**

KUNDENR	KUNDENAVN	TELEFON
100	JOCKSPORTS.....	598-6609
101	TKB SPORT SHOP.....	368-1223
:	:	:

9. Gruppefunktioner

Gruppefunktioner returnerer, i modsætning til rækkefunktioner, resultater baseret på grupper af rækker. Default opfattes alle rækker i en tabel som én gruppe. Select-sætningens GROUP BY-clause bruges til at opdele rækkerne i mindre grupper.

Alle gruppefunktioner, pånær COUNT(*), ignorerer null-værdier. Gruppefunktionerne er bla. at finde i SQL Language Reference Manual 4-20,22.

Syntaksen for en select-sætning med gruppering er:

```
SELECT   <kolonnenavn>,<kolonnenavn>, ... ,<kolonnenavn>
FROM     <tabelnavn>,<tabelnavn>, ... ,<tabelnavn>
WHERE    rækkebetingelse(r)
GROUP BY <kolonnenavn>,<kolonnenavn>, ... ,<kolonnenavn>
HAVING   gruppebetingelse(r)
ORDER BY <kolonnenavn>,<kolonnenavn>, ... ,<kolonnenavn>
```

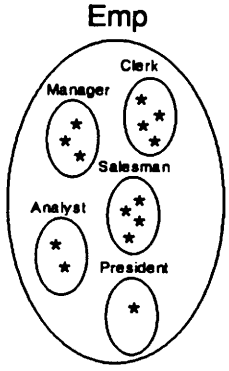
Bemærk at rækkebetingelser anbringes i WHERE-clausen og gruppebetingelser i HAVING-clausen.

EKSEMPEL1

Udskriv for hver stillingsgruppe antal medarbejdere, lønsum og løngennemsnit.

```
SELECT   job stilling,count(*) antal,sum(sal) lønsum,
         round(avg(sal)) løngns
FROM     emp
GROUP BY job;
```

STILLING	ANTAL	LØNSUM	LØNGNS
ANALYST	2	6000	3000
CLERK	4	4150	1038
MANAGER	3	8275	2758
PRESIDENT	1	5000	5000
SALESMAN	4	5600	1400



EKSEMPEL2

Udskriv vha. item-tabellen ordrenr og ordretotal. Kun totaler større end 5000 skal vises. Sammenlign totalerne med ord-tabellen.

```
SELECT   ordid ordnr,sum(itemtot) ordretotal
FROM     item
GROUP BY ordid
HAVING   sum(itemtot)>5000;
```

```
  ORDNR ORDRETOTAL
------- ----------
    605       8324
    612       5860
    613       6400
    614      23940
    617      46370
```

EKSEMPEL3

Udskriv størstelønnen pr. stillingsgruppe indenfor hver afdeling bortset fra afdeling 20. Kun lønninger over 2500 har interesse.

```
SELECT   deptno afd,job stilling,
         max(sal+nvl(comm,0)) "STØRSTE LØN"
FROM     emp
WHERE    deptno!=20
GROUP BY deptno,job
HAVING   max(sal+nvl(comm,0))>2500;
```

```
    AFD STILLING  STØRSTE LØN
------- --------- -----------
     10 PRESIDENT        5000
     30 MANAGER          2850
     30 SALESMAN         2650
```

Øvelse 4

Formål: - *at anvende gruppefunktioner*
- *at gruppere rækker vha. GROUP BY*
- *at udtrykke gruppebetingelser vha. HAVING*

1. **Udskriv alle stillingstyper. For hver stillingstype skal højeste løn, laveste løn samt gennemsnitsløn angives.**

2. **Udskriv den gennemsnitlige total-løn (løn+provision) for de stillingsgrupper, hvor gennemsnitslønnen er mindre end 2500.**

3. **Udskriv en liste over stillingsgrupper hvor den procentvise forskel på største og mindste løn er større end 20%.**

4. **Udskriv alle afdelinger med mere end 1 overordnet medarbejder (manager eller president).**

5. **Udskriv en liste over antallet af medarbejdere indenfor hver stillingsgruppe i hver enkelt afdeling.**

6. **Udskriv den årlige lønsum for hver afdeling. Provision medregnes.**

7. **Udskriv ordrenummer, antal delordrer samt ordretotal på de ordrer, der består af mere end 4 delordrer.**

8. **Opdel actualprice i intervaller på 10. Udskriv for hvert interval hyppighed, gennemsnit af qty samt gennemsnit af itemtot.**

INTERVAL	HYPPIGHED	GNS.QTY	GNS.ITEMTOT
0 - 10	33	211	775
10 - 20	2	175	2000
:	:	:	:

$$\frac{max(sal) - min(sal)}{100} * 20$$

$$\frac{max(sal) * 20}{100} = max - min$$

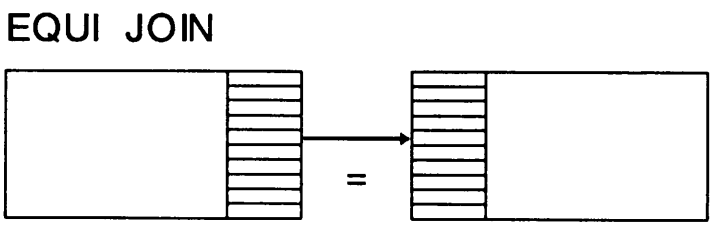
$$x = \frac{(max - min)\ 100}{max}$$

10. Join af tabeller

Et join kombinerer kolonner og data fra 2 eller flere tabeller. En tabel kan dog godt joines med sig selv. Alle tabeller skal angives i sætningens FROM-clause. Relationen mellem tabellerne skal angives i sætningens WHERE-clause.

10.1 EQUI join

Relationen mellem to tabeller er ofte en lighed som fx: a.post=b.post. Dette kaldes et equi-join, fordi det bruger et lighedstegn i WHERE-clausen.



EKSEMPEL

Udskriv navn, stilling, afdelingsnummer, afdelingsnavn og lokation for alle managers.

```
SELECT ename,job,emp.deptno,dname,loc
FROM   emp,dept
WHERE  job='MANAGER' and
       emp.deptno=dept.deptno;
```

ENAME	JOB	DEPTNO	DNAME	LOC
CLARK	MANAGER	10	ACCOUNTING	NEW YORK
JONES	MANAGER	20	RESEARCH	DALLAS
BLAKE	MANAGER	30	SALES	CHICAGO

10.2 Join af én tabel med sig selv

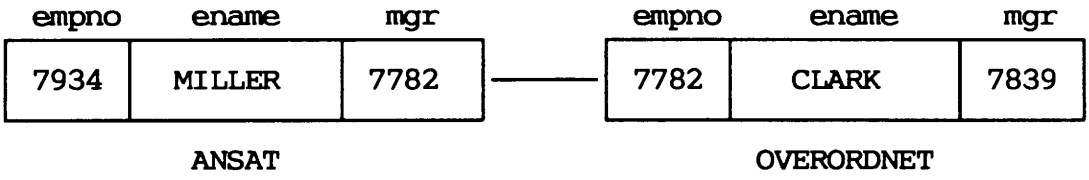
Ønskes en indbyrdes sammenligning af rækker indenfor samme tabel, kan tabellen joines med sig selv. Det gøres ved logisk at navngive 2 udgaver af den samme tabel.

EKSEMPEL

Udskriv en liste over ansatte i afdeling 10 og deres nærmeste overordnede.

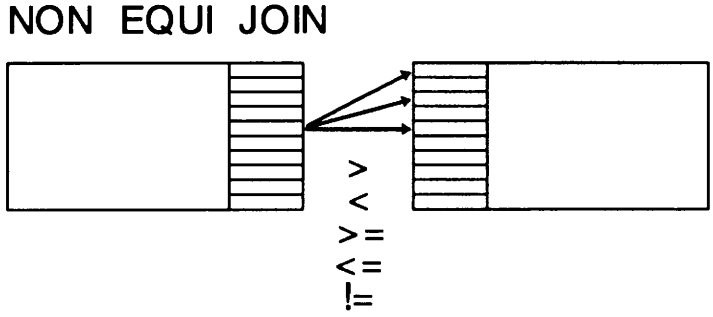
```
SELECT a.ename ansat, o.ename overordnet
FROM   emp a, emp o
WHERE  a.mgr=o.empno and a.deptno=10;
```

```
ANSAT      OVERORDNET
---------- ----------
MILLER     CLARK
CLARK      KING
```



10.3 NON EQUI join

Er relationen mellem to tabeller ikke en lighed, men en ulighed der benytter operetorer som fx <, >= og !=, taler man om et non equi-join.



EKSEMPEL

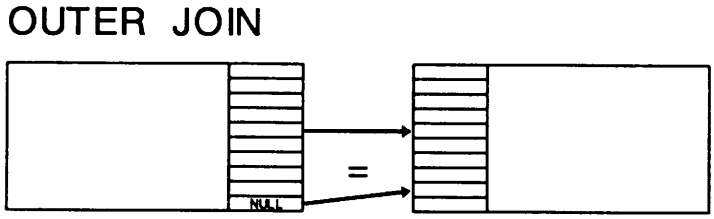
Udskriv en liste over medarbejdere fra afdeling 20, der blev ansat før King.

```
SELECT a.ename,a.hiredate
FROM   emp a,emp o
WHERE  a.hiredate<o.hiredate and
       o.ename='KING' and a.deptno=20;

ENAME      HIREDATE
---------- ---------
SMITH      17-DEC-80
JONES      02-APR-81
```

10.4 OUTER join

Outer-join er en metode til at finde rækker i én tabel, der ikke modsvares af rækker i den anden tabel.



Hvis der findes en række i den ene tabel, der ikke modsvares af en række i den anden tabel, tilføjes der en 'null-række'. Det betyder i praksis, at oplysninger der ikke har modsvarende rækker vises alligevel.

EKSEMPEL

Udskiv en liste over afdelinger uden ansatte.

```
SELECT dept.deptno,dname,loc
FROM   emp,dept
WHERE  emp.deptno(+)=dept.deptno and
       empno is null;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        40 OPERATIONS     BOSTON
```

Øvelse 5

Formål: - at joine tabeller

1. **Udskriv navn, afdelingsnavn og lokation for både Allen, Jones og Clark.**

2. **Udskriv kundenummer, kundenavn og ordretotal for alle kunder.**

3. **Udskriv en liste der viser, hvilke medarbejdere der har Jones som nærmeste overordnede.**

4. **Udskriv en liste over medarbejdere, der blev ansat før deres nærmeste overordnede.**

5. **Udskriv en liste over ansatte, der arbejder i samme afdeling som president.**

6. **Udskriv kundenavn, adresse, sælger og lokation for alle kunder.**

7. **Udskriv en oversigt der viser, hvor meget hver har sælger solgt for.**

8. **Udskriv en liste der viser produktnummer samt beskrivelse på de produkter, Martin har solgt. Listen skal udskrives uden dubletter.**

11. Subquery

Det hænder, at man ikke kan udtrykke sine søgekriterier direkte, men er afhængig af resultatet fra en anden forespørgsel. I stedet for at gemme mellemresultater kan man anvende subqueries.

Subqueries placeres i WHERE-clausen eller HAVING-clausen eftersom de er en del af søgebetingelserne. Subqueries omsluttes af parenteser.

EKSEMPEL1

Udskriv en liste over ansatte det tjener mere end end gennemsnitslønnen for alle ansatte.

```
SELECT avg(sal+nvl(comm,0))
FROM   emp;

AVG(SAL+NVL(COMM,0))
--------------------
          2230.35714
```

```
SELECT ename,sal+nvl(comm,0) totalløn
FROM   emp
WHERE  sal+nvl(comm,0)>2230.35714;

ENAME      SAL+NVL(COMM,0)
---------- ---------------
JONES                 2975
MARTIN                2650
BLAKE                 2850
CLARK                 2450
SCOTT                 3000
KING                  5000
FORD                  3000
```

Med en subquery vil en samlet forespørgsel kunne skrives således:

```
SELECT ename navn,sal+nvl(comm,0) totalløn
FROM   emp
WHERE  sal+nvl(comm,0)>
       (SELECT avg(sal+nvl(comm,0)) FROM emp);
```

Evalueres først

EKSEMPEL2

Hvilke afdelinger har en lønsum der er større end lønsummen i afdeling 10.

```
SELECT   deptno,sum(sal+nvl(comm,0)) lønsum
FROM     emp
GROUP BY deptno
HAVING   sum(sal+nvl(comm,0))>
            (SELECT sum(sal+nvl(comm,0))
             FROM   emp
             WHERE  deptno=10);
```

```
    DEPTNO      LØNSUM
---------- ----------
        20       10875
        30       11600
```

11.1 Subquery der returnerer mere end én værdi

I de to foregående eksempler blev der kun returneret én værdi fra underforespørgslen. Hvis underforespørgslen retunerer mere end én værdi, vil værdierne blive stakket i en liste, og det er så nødvendigt at benytte 'mange-værdi-operatorer'. Det drejer sig om operatorerne IN, ALL, ANY og EXSISTS (SQL Language Reference Manual 4-3,4).

EKSEMPEL3

Hvilke stillingsbetegnelser fra afdeling 30 går igen i afdeling 10, og hvilke personer er de knyttet til?

```
SELECT ename,job
FROM   emp
WHERE  deptno=10 and job in
       (SELECT job FROM emp WHERE deptno=30);
```

```
ENAME      JOB
---------- ---------
MILLER     CLERK
CLARK      MANAGER
```

ANY eller ALL indsættes mellem sammenligningsoperatoren (=, !=, <, <=, >, >=) og underforespørgslen.

EKSEMPEL4

Udskriv navn, stilling og løn for de ansatte der tjener mere end en vilkårlig manager.

```
SELECT   ename,job,sal+nvl(comm,0) løn
FROM     emp
WHERE    sal+nvl(comm,0)> any
         (SELECT sal FROM emp WHERE job='MANAGER')
ORDER BY sal+nvl(comm,0);
```

ENAME	JOB	LØN
MARTIN	SALESMAN	2650
BLAKE	MANAGER	2850
JONES	MANAGER	2975
SCOTT	ANALYST	3000
FORD	ANALYST	3000
KING	PRESIDENT	5000

EKSEMPEL5

Udskriv en oversigt over ansatte der mindst har én ansat under sig.

```
SELECT   empno,ename,job,deptno
FROM     emp e
WHERE    exists
         (SELECT * FROM emp WHERE e.empno=mgr)
ORDER BY ename;
```

EMPNO	ENAME	JOB	DEPTNO
7698	BLAKE	MANAGER	30
7782	CLARK	MANAGER	10
7902	FORD	ANALYST	20
7566	JONES	MANAGER	20
7839	KING	PRESIDENT	10
7788	SCOTT	ANALYST	20

11.2 Subqueries mod flere kolonner

I stedet for at udføre flere subqueries mod enkelte kolonner kan man udføre én subquery mod flere kolonner. Kolonnenavnene i den primære forespørgsel angives så i parentes.

EKSEMPEL6

Udskriv en liste der viser den lavest lønnede indenfor hver stillingsgruppe.

```
SELECT   job stilling,ename navn,sal+nvl(comm,0) løn
FROM     emp
WHERE    (job,sal+nvl(comm,0)) in
         (SELECT   job,min(sal+nvl(comm,0))
          FROM     emp
          GROUP BY job)
ORDER BY job,ename;
```

STILLING	NAVN	LØN
ANALYST	FORD	3000
ANALYST	SCOTT	3000
CLERK	SMITH	800
MANAGER	CLARK	2450
PRESIDENT	KING	5000
SALESMAN	TURNER	1500

Øvelse 6

Formål: - at anvende subqueries

1. **Udskriv navn, stilling og løn for de ansatte der tjener mere end medarbejder nr. 7844.**

2. **Udskriv navn og ansættelsesdato på de ansatte der har større anciennitet end PRESIDENT.**

3. **Udskriv en liste over de CLERK's der tjener mere end den højest lønnede CLERK i afdeling 20.**

4. **Udskriv kundenummer og kundenavn for de kunder der har registreret mere end én ordre.**

5. **Udskriv navn, stilling, afdelingsnummer og løn for de ansatte der inden for hver afdeling har den højeste løn.**

Se HKS løsning

6. **Udskriv oplysninger om de ansatte der tjener mere end gennemsnittet i deres egen afdeling.**

12. Generering af SQL-sætninger

Kommandoen SPOOL lagrer resultatet af en forespørgsel i en UNIX-fil (SQL*Plus User's Guide and Reference 6-64). Det betyder at der kan genereres en SQL-kommandofil fx vha. en SELECT-sætning.

EKSEMPEL

Udskriv alle oplysninger fra samtlige tabeller under den aktuelle Oraclebruger. Opret en SQL-kommandofil med flg. indhold:

```
SET HEADING off
SET FEEDBACK off
SPOOL alletab.sql
SELECT 'select * from '||table-name||';' FROM user-tables
/
SPOOL off
SET HEADING on
@alletab
```

Eksemplet udnytter nogle af SET-kommandoens muligheder (SQL*Plus User's Guide and Reference 6-53,62). Undervejs genereres filen alletab.sql som fx kan se således ud:

```
select * from CUSTOMER;
select * from DEPT;
select * from EMP;
select * from ITEM;
select * from ORD;
select * from PRICE;
select * from PRODUCT;
```

"DROP TABLE" – Sletter tabel

TABLE_NAME
TABLESPACE_NAME
CLUSTER_NAME

desc user_tables ——
user_catalog ——

{ TABLE_NAME
TABLE_TYPE

13. Udveksling af data

Er SQL*Net TCP/IP installeret, kan der udveksles data både mellem baser på samme maskine og mellem baser på forskellige maskiner.

Grundlæggende kan data hentes fra en host-base på 2 forskellige måder. Der kan startes en session på host-basen eller data kan hentes fra en session på lokal-basen.

EKSEMPEL1

Data hentes ved at starte en session på hostmaskinens database. Hvis host-maskinen er unit2, base_id er S og bruger er user5, angives følgende programkald:

host base_id

sqlplus user5/<passwd>@T:unit2:S

↑ TCP (protokol valg)

SQL*Net TCP/IP User's Guide Kap. 2,4.

EKSEMPEL2

Skal data hentes fra en session på lokal-basen er det nødvendigt at der er oprettet et databaselink. Et link svarende til eksemplet ovenfor oprettes således:

```
CREATE DATABASE LINK unit2
CONNECT TO user5 IDENTIFIED BY <passwd>
USING 'T:unit2:S';
```

Herefter kan det oprettede link bruges til at søge data på følgende måde:

```
SELECT * FROM <tabelnavn>@unit2;
WHERE ....
```

SQL*Net TCP/IP User's Guide Kap. 2,4.

Øvelse 7

*Formål: - at udføre søgninger via SQL*Net*
at generere SQL-sætninger

1. **Start en session udfra følgende oplysninger:**
   host er unv10, sid er DEFA og bruger er scott/tiger.

2. **Hvilke tabeller er ejet af scott?**

3. **Afslut SQL*Plus**

4. **Opret et databaselink under din ora-bruger udfra oplysningerne angivet under punkt 1.**

5. **Brug dit link til at finde navn, job og samlet løn for de medarbejdere der har en samlet løn på mere end 2000. Oplysningerne skal findes i scott's emp-tabel på unv10.**

6. **Opret en sql-fil der genererer SQL-sætninger til søgning af oplysninger fra samtlige tabeller ejet af brugeren scott på unv10.**

7. **Afprøv filen**

14. Manipulation af tabelrækker

SQL-sproget rummer også mulighed for at ændre på og slette eksisterende rækker, samt at tilføje nye rækker.

14.1 Indsættelse af række

SQL*Plus kommandoen **DESC(RIBE)** viser bla. datatypen for de enkelte kolonner i en tabel.

DESC emp;

```
Name                            Null?    Type
------------------------------- -------- -----------
EMPNO                           NOT NULL NUMBER(4)
ENAME                                    CHAR(10)
JOB                                      CHAR(9)
MGR                                      NUMBER(4)
HIREDATE                                 DATE
SAL                                      NUMBER(7,2)
COMM                                     NUMBER(7,2)
DEPTNO                          NOT NULL NUMBER(2)
```

De mest anvendte datatyper er:

NUMBER - Til rene talværdier herunder decimaltal.

CHAR - Til tekst og tal. Skrives i enkelte anførselstegn.

DATE - Til datoer. Standard-format er DD-MMM-YY. Skrives i enkelte anførselstegn.

Nye rækker oprettes med kommandoen **INSERT**. SQL Language Reference Manual 5-107.

EKSEMPEL

Indsæt en ny medarbejder i emp-tabellen. På nuværende tidpunkt kendes kun medarbejdernummer, navn, ansættelsesdato og afdelingsnummer. Bemærk at 'NOT NULL'-felter skal udfyldes.

```
INSERT INTO emp VALUES
(7954,'BROWN',null,null,'1-mar-90',null,null,30);

1 row created
```

Samme medarbejder kunne indsættes på følgende måde:

```
INSERT INTO emp (empno,ename,hiredate,deptno)
VALUES          (7954,'BROWN','1-mar-90',30);
```

1 row created

Skal insert-sætningen genbruges til at indsætte flere medarbejdere, kan den ændres til at gå i dialog med brugeren.

```
INSERT INTO emp (empno,ename,hiredate,deptno)
VALUES          (&medarbnr,'&navn','&ansdato',&afdnr);
```

14.2 Ændring af eksisterende række

Eksisterende rækker ændres med kommandoen **UPDATE**. SQL Language Reference Manual 5-139.

EKSEMPEL

Giv alle clerks en lønforhøjelse på 12%.

```
UPDATE emp
SET    sal=sal*1.12
WHERE  job='CLERK';
```

4 rows updated

EKSEMPEL

Blake skal forfremmes til vicedirektør. Forfremmelsen indebærer en flytning til afdeling 10 samt en løn der skal være 1000 større end den højest lønnede manager.

```
UPDATE emp
SET    deptno=10,job='VICEPRES.',
       sal=(SELECT max(sal)+1000 FROM emp
            WHERE  job='MANAGER')
WHERE  ename='BLAKE' and job='MANAGER';
```

1 row updated

14.3 Sletning af eksisterende rækker

Eksisterende rækker slettes med kommandoen **DELETE**. SQL Language Reference Manual 5-81.

EKSEMPEL
Slet alle clerks fra afdeling 20.

```
DELETE FROM emp
WHERE       deptno=20 and job='CLERK';
```

2 rows deleted

EKSEMPEL
Slet alle ansatte hvis nærmeste overordnede har et nummer, der ikke eksisterer.

```
DELETE FROM emp
WHERE       mgr is not null
            and mgr not in (select empno from emp);
```

14.4 Commit og rollback

Ved brug af INSERT, UPDATE og DELETE gemmes de udførte ændringer ikke permanent i den forstand, at de er skrevet ned på disken.

Ændringer gøres permanente med kommandoen **COMMIT**. SQL Language Reference Manual 5-29.

```
COMMIT;
```

commit complete

Om ukommittede ændringer gælder:

* De kan ses af brugeren selv ved forespørgsler mod tabellerne.
* De kan ikke ses af andre brugere ved forespørgsler mod de samme tabeller.
* Det er muligt at fortryde ændringerne med kommandoen ROLLBACK.
(SE næste side)

Oracle udfører automatisk et COMMIT, når en af følgende kommandoer anvendes:

QUIT, EXIT, ^D
CREATE, DROP table/view
GRANT, REVOKE
CONNECT, DISCONNECT
ALTER
AUDIT, NOAUDIT

Ukommittede ændringer udført med INSERT, UPDATE eller DELETE kan fortrydes med kommandoen **ROLLBACK.**

ROLLBACK;

rollback complete

Et rollback ruller alle transaktioner udført efter sidste commit tilbage.

14.5 Savepoints

Savepoints bruges i sammenhæng med rollback til kun at rulle en del af de transaktioner, der er udført siden sidste commit tilbage. Det er fx en fordel, hvis der skal udføres en lang serie af sammenhængende opdateringer. Laves en fejl, skal man således ikke starte helt forfra.

EKSEMPEL
Fra konto-A skal overføres hhv. 2000 til konto-B og 3500 til konto-C.

```
UPDATE kontoA SET saldo=saldo-5500;
SAVEPOINT a;
UPDATE kontoB EST saldo=saldo+2000;
SAVEPOINT b;
UPDATE kontoC SET saldo=saldo+4500;
ROLLBACK TO SAVEPOINT b;
UPDATE kontoC SET saldo=saldo+3500;
```

Øvelse 8

Formål: - at indsætte, slette og opdatere rækker

1. **Opret en medarbejder i emp-tabellen med følgende oplysninger:**

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7912	OLSEN	TEGNER		01-NOV-93			50

2. **Ret navnet på den nye medarbejder til Olesen.**

3. **Giv alle ansatte i afdeling 10, bortset fra King, en lønforhøjelse på 10%.**

4. **Indsæt dig selv i emp-tabellen med en højere løn end president.**

5. **Udfør et commit.**

6. **Slet alle rækker fra dept-tabellen.**
   Udskriv alle oplysninger fra dept-tabellen.
   Udfør et rollback.
   Udskriv alle oplysninger fra dept-tabellen.

7. **Slet alle medarbejdere der tjener mere end president.**

8. **Foretag en opdatering af emp-tabellen, således at alle Jones's umiddelbart underordnede bliver underordnet Clark i stedet.**

9. **Slet 'JUST TENNIS' fra customer-tabellen. De detailoplysninger der ligger i ord-tabellen og item-tabellen skal også slettes. Brug savepoints mellem hver sletning.**

15. Opret/slet tabel

Tabeller oprettes med kommandoen **CREATE TABLE**. SQL Language Reference Manual 5-66.

Ved oprettelse af en tabel bestemmes hvilke kolonner den skal indeholde. Endvidere bestemmes datatype og længde for hver kolonne.

EKSEMPEL

Dept-tabellen er oprettet således:

```
CREATE TABLE dept
      (deptno number(2) not null, dname char(14), loc char(13));
Table created
```

Hele tabeller eller dele af tabeller inkl. indhold kan kopieres vha. kommandoen **CREATE TABLE**.

EKSEMPEL

Kopier tabellen dept til tabellen kopidept.

```
CREATE TABLE kopidept
AS SELECT * FROM dept;
Table created
```

Tabeller slettes med kommandoen **DROP TABLE**. SQL Language Reference Manual 5-91.

EKSEMPEL

```
DROP TABLE kopidept;
Table dropped
```

DROP TABLE-kommandoen sletter både tabelindhold og tabeldefinition på én gang.

BEMÆRK: *Drop table kan ikke fortrydes med rollback!!*

15.1 Ændring af tabel

Ændringer i tabeldefinitionen udføres med kommandoen **ALTER TABLE.**

EKSEMPEL

I dept-tabellen ønskes mulighed for at skrive afdelingsnavne på op til 20 tegn.

```
ALTER TABLE dept
MODIFY      (dname char(20));
Table altered
```

Kommentarer:

* Formatet kan kun gøres mindre, hvis alle værdier i kolonnen er null.
* Data-typen kan kun ændres, hvis alle værdier i kolonnen er null.

EKSEMPEL

I dept-tabellen ønskes en kolonne, til hver afdelings samlede budget, tilføjet.

```
ALTER TABLE dept
ADD         (budget number(8,2));
Table altered
```

Ændring af tabelnavn udføres med kommandoen **RENAME TO.** SQL Language Reference Manual 5-118.

EKSEMPEL

Omdøb tabellen ord til orders.

```
RENAME ord TO orders;
Table renamed
```

16. Opret/slet view

Et view er en logisk tabel der kan behandle data fra en eller flere eksisterende tabeller. For brugeren er der ingen synlig forskel på et view og en tabel.

Views bruges fx til 'pre-queries'. Det skal forstås sådan, at et view bruges som et mellemresultat, der så kan udføres forspørgsler imod.

Et view oprettes med kommandoen **CREATE VIEW**. SQL Language Reference Manual 5-74.

EKSEMPEL

Hvis der ofte skal udskrives en liste over kunder med mere end XXXX i ordrer, kan følgende view forenkle den endelige forespørgsel:

```
CREATE VIEW ordretotal (kundenavn,ordretotal)
AS SELECT   name,sum(total)
   FROM     customer c,ord o
   WHERE    c.custid=o.custid
   GROUP BY name,o.custid;
```

Det 'grove' arbejde er ny en gang for alle gjort i viewet ordretotal. Den endelige søgning er således forenklet til følgende:

```
SELECT * FROM ordretotal
WHERE  ordretotal>5000;
```

KUNDENAVN	ORDRETOTAL
JOCKSPORTS	5280.90
VOLLYRITE	27775.50
EVERY MOUNTAIN	7160.80
K + T SPORTS	46370.00
SHAPE UP	9024.40
NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER	6400.00

Et view slettes med kommandoen **DROP VIEW**. SQL Language Reference Manual 5-93.

EKSEMPEL

DROP VIEW ordretotal;
View dropped

16.1 Ændring af view

Kommandoen ALTER VIEW findes ikke. Ønskes et view ændret, droppes det og oprettes påny i den ønskede form. Dette er en meget simpel procedure hvis oprettelseskommandoen er gemt i en fil.

En ændring af viewnavn kan klares med kommandoen **RENAME TO**.

EKSEMPEL

RENAME ordretotal TO ordtot;
Table renamed

Øvelse 9

Formål: - *at oprette, slette og ændre tabeller*
- *at kopiere tabeller*
- *at oprette og slette views*

1. **Opret en projekt-tabel med kolonnerne projektnr, projektbeskrivelse, projektleders efternavn og personalenr.**
2. **Indsæt 3-4 rækker i projekt-tabellen.**
3. **Kopier projekt-tabellen til en tabel med navnet nyprojekt.**
4. **Udfør en ændring i nyprojekt-tabellen således at projektledernavnet udvides med 10 tegn.**
5. **Nyprojekt-tabellen skal have tilføjet en kolonne til projektledernes fornavne på op til 12 tegn.**
6. **Opret et view til forenkling af følgende opgave:**
   Udskriv navn, samlet løn, afdelingsnummer, afdelingsnavn for de medarbejdere der tjener mere end XXXX.
7. **Udskriv vha. viewet de medarbejdere hvis samlede løn overstiger 2000.**
8. **Omdøb tabellen nyprojekt til projekt1.**
9. **Slet tabellen projekt1.**

17. Optimering med indeks

Et indeks bruges til følgende opgaver:

- At forbedre søgetiden
- At sortere uddata
- At sikre unikke værdier (kun unikt indeks)

Et indeks kan defineres på en eller flere kolonner (sammensat indeks). Begge typer indeks kan være unikke eller ikke-unikke.

17.1 Definition af indeks

Et indeks oprettes med kommandoen **CREATE INDEX**. SQL Language Refence Manual 5-52.

EKSEMPEL

Opret et unikt indeks på emp-tabellens empno- og deptno-kolonne.

```
CREATE UNIQUE INDEX i_emp
ON emp(empno,deptno);
```

Kommentarer:

* Et sammensat indeks (concatenated index) defineres ved at nævne de aktuelle kolonner adskilt af komma (se eksempel).
* Hvis indekset ikke eksplicit defineres som unikt, antages det at være ikke-unikt.

17.2 Oracles brug af indeks

Det er ikke altid en god ide at definere et indeks. Et indeks kræver plads på disken, og det tager procestid, når det skal opdateres ved brug af kommandoerne INSERT, UPDATE og DELETE.

Der kan betale sig at oprette indeks når:

* Der er stor distribution af indeksværdier. Ideelt set skal indeksværdierne være unikke.
* Tabellen indeholder mindst et par hundrede rækker.
* Der skal udvælges mindre end 10-15% af en tabel.
* Kolonnen indgår i et join (dvs. en tendens til at indeksere primær- og fremmednøglen).
* Kolonnen bruges til direkte opslag.
* Der ikke oprettes mere end 3-4 indices pr. tabel. Hvis tabelindholdet er meget statisk dog gerne flere.

Følgende er også værd at vide om Oracles brug af indeks i en SQL-sætning:

* Indeks bliver ikke brugt, hvis kolonnen indgår i en beregning eller er modificeret af en funktion.
* Indeks bliver ikke brugt i en group by sætning. Ej heller i forbindelse med distinct.
* Indeks bruges ikke, hvis der ikke findes en where-betingelse i SQL-sætningen.

17.3 Sletning af indeks

Et indeks slettes med kommandoen **DROP INDEX**. SQL Language Reference Manual 5-87.

EKSEMPEL

Slet indekset i_emp_ename.

```
DROP INDEX i_emp_ename;
```

Øvelse 10

Formål: - at oprette indeks

1. **Givet at emp- og dept-tabellerne havde en størrelse, der gjorde definitionen af indeks fornuftig, hvilke kolonner ville du pege på som mulige indekskolonner og hvilke(n) kolonne(r) ville det være uhensigtmæssigt at definere indeks på?**
   emp.empno emp.mgr
   dept.deptno
2. **Definer et unikt index på empno-kolonnen i emp-tabellen.**
3. **Prøv at dublere et medarbejdernummer vha. INSERT.**
   ⇒ "dublet nøgle i indeks"
4. **Definer et indeks på ename-kolonnen i emp-tabellen.**
5. **Udfør følgende følgende søgninger:**
   a) **SELECT ename FROM emp;**
   b) **SELECT ename FROM emp WHERE ename>' ';**
   c) **SELECT ename FROM emp WHERE initcap(ename)>' ';**
6. **Udfør SQL*Plus-kommandoen: SET TIMIMG ON**
7. **Udfør følgende søgning:**

   ```
   SELECT * FROM words WHERE word like 'zo%';
   ```

8. **Definer et indeks på word-kolonnen i words-tabellen.**
9. **Udfør igen søningen fra punkt 7.**

18. Sequence

Sequences bruges til at tildele unikke numre, fx kundenumre. Når en sequence er oprettet kan der hentes værdier fra den i forbindelse med INSERT, UPDATE og i sjældne tilfælde SELECT.

En sequence oprettes med kommandoen **CREATE SEQUENCE**. SQL Language Reference Manual 5-58,61.

EKSEMPEL

Opret en sequence der starter med vædien 9000 og vokser med 1.

```
CREATE SEQUENCE medarbnr INCREMENT BY 1 START WITH 9000;
```

For at generere et nyt medarbejdernummer gennem medarbnr bruges pseudo-kolonnen NEXTVAL.

EKSEMPEL

En ny sælger ansættes og skal indsættes i emp-tabellen.

```
INSERT INTO emp (empno,ename,ansdato,deptno)
VALUES       (medarbr.NEXTVAL,'THOMSEN',sysdate,30);
```

Ønskes det aktuelle serienummer genanvendt bruges pseudo-kolonnen CURRVAL.

EKSEMPEL

Den nye sælger skal være betjene kunden SHAPE UP.

```
UDATE CUSTOMER
SET   repid=medarbnr.CURRVAL
WHERE name='SHAPE UP';
```

NEXTVAL og CURRVAL må bruges i:

* SELECT-delen af en SELECT-sætning (undtagen i views)
* Værdilisten i en INSERT-kommando
* SET-delen af en UPDATE-kommando

NEXTVAL og CURRVAL må ikke bruges:

* I en subquery
* Sammen med DISTINCT
* Sammen med operatorerne UNION, INTERSECT og MINUS

En sequence slettes med kommandoen **DROP SEQUENCE**. SQL Language Reference Manual 5-89.

EKSEMPEL

```
DROP SEQUENCE medarbnr;
```

19. Comments

I oracle er det muligt at skrive kommentarer til både den enkelte tabel og til de enkelte kolonner i tabellerne. Kommentarerne kan bruges som en del af dukumentationen.

EKSEMPEL

Kommentar til dept-tabellen.

```
COMMENT ON TABLE dept IS 'Oversigt over afdelinger';
```

EKSEMPEL

Kommentar til dname-kolonnen i dept-tabellen.

```
COMMENT ON COLUMN dept.dname IS 'Afdelingens navn';
```

Øvelse 11

Formål: - at oprette sequence
at oprette comments

1. **Opret en tabel med en talkolonne og en tekstkolonne.**

2. **Opret en sequence efter eget valg.**
   create sequence segno increment by 1 start with 10;
3. **Indsæt nogle rækker i tabellen. Talværdierne skal indsættes vha. sequence.**
   insert into tab (tal, tekst) values (segno.nextval, 'tekst' || segno.nextval);
4. **Kontroller talværdierne. Er de som forventet?**

5. **Opret en passende kommentar til customer-tabellen og til udvalgte kollonner i den.**

6. **Find dine kommentarer ved at søge i datakataloget - se Appendix A.** select * from user_tab_comments where table_name =)
   ⇒ TABLE_NAME TABLE_TYPE COMMENTS
   CUSTOMER TABLE 'Kunde oversigt'
7. **Giv et forslag til en søgning der kun finder kolonner, hvortil der eksisterer en kommentar.**

LASTNAME ',

20. Udskrivning af rapporter

Det er muligt at generere simple rapporter med SQL*plus. En SQL-rapport laves på baggrund af en almindelig forespørgsel, hvor uddata blot formateres.

RAPPORT-EKSEMPEL

A C M E W I D G E T
SALARY ANALYSIS

DEPARTMENT NAME	EMPLOYEE NAME	MONTHLY SALARY	PERCENT DEPARTMENT	PERCENT COMPANY
ACCOUNTING	CLARK	$2,572.50	25.29	5.58
	KING	$5,500.00	54.07	11.93
	MILLER	$1,300.00	12.78	2.82
	BROWN	$800.00	7.86	1.74
**********		---------	----------	-------
sum		$10,172.50	100.00	22.07
RESEARCH	SMITH	$920.00	3.96	2.00
	JONES	$3,123.75	3.44	6.78
	SCOTT	$3,795.00	16.32	8.23
	ADAMS	$1,265.00	5.44	2.74
	FORD	$3,450.00	14.84	7.49
	MASON	$3,910.00	16.82	8.48
	CHAN	$3,450.00	14.84	7.49
	FOX	$3,335.00	14.34	7.24
**********		---------	----------	-------
sum		$23,248.75	100.00	50.44
SALES	ALLEN	$1,600.00	12.63	3.47
	WARD	$1,312.50	10.36	2.85
	MARTIN	$1,312.50	10.36	2.85
	BLAKE	$2,992.50	23.62	6.49
	TURNER	$1,500.00	11.84	3.25
	JAMES	$950.00	7.50	2.06
	CARTER	$1,000.00	7.89	2.17
	WILSON	$2,000.00	15.79	4.34
**********		---------	----------	-------
sum		$12,667.50	100.00	27.49
		---------		-------
sum		$46,088.75		100.00

I det følgende gennemgås nogle faciliteter til at udskrive en rapport.

Kolonneoverskrifter

Kolonner kan tildeles overskrifter med kommandoen **COLUMN HEADING**. SQL*Plus User's Guide and Reference 6-17,21. Disse overskrifter bliver gældende for alle kolonner med samme navn. Alias er kun gældende for den ene søgning, hvori det angives.

EKSEMPEL

Udskriv navn, løn og afdelingsnummer fra emp-tabellen med passende overskrifter.

```
COLUMN ename  HEADING Efter-¦navn JUSTIFY CENTER
COLUMN sal    HEADING "  Måneds-¦  løn" JUSTIFY LEFT
COLUMN deptno HEADING Afdelings-¦nummer JUS L
select ename,sal,deptno from emp;

Efter-   Måneds-  Afdelings-
navn     løn      nummer
------ ---------- ----------
ALLEN    1,600.00         30
JONES    2,975.00         20
  :          :              :
```

Formater

Formater ændres med kommandoen **COLUMN FORMAT**. SQL*Plus User's Guide and Reference 6-17,19. Undtaget er datoformater, som ændres med funktionen TO_CHAR.

EKSEMPEL

Udskriv navn og løn fra emp-tabellen. Lønnen skal vises uden decimaler.

```
COLUMN ename HEADING NAVN FORMAT a10
COLUMN sal   HEADING LØN FORMAT 99,990 JUS C
SELECT ename,sal FROM emp;

NAVN         LØN
---------- -------
ALLEN        1,600
WARD         1,250
 :             :
```

DATATYPE	JUSTERING	DEFAULT FORMAT
CHAR	Venstre	Størst af tabeldefinition og heading
NUMBER	Højre	Bredde=numwidth, alle betydende cifre
DATE	Venstre	Bredde=9, format: DD-MMM-YY

Formatet for alle talkolonner der ikke er omfattet af et specifikt tildelt format, kan tildeles med kommandoen **SET NUMF(ORMAT)**. SQL*Plus User's Guide and Reference 6-53,57.

EKSEMPEL

SET NUMF 99990.99

Sideformattering

Følgende kommandoer har med sideformattering at gøre:

SET PAGES(IZE) xx **sætter sidestørrelse i udskrift**
SET NEWP(AGE) xx **sætter antal blanke linier øverst på side**
SET LIN(ESIZE) xx **sætter linielængde i udskrift**
SET SPA(CE) xx **sætter antal blanke mellem kolonner**
*SQL*Plus User's Guide and Reference 6-53,58.*

En sideoverskrift kan indsættes med kommandoen **TTITLE**. SQL*Plus User's Guide and Reference 6-71,73. Overskriftens placering er bla. afhængig af linesize og newpage.

En bundtekst kan indsættes med kommandoen **BTITLE**. SQL*Plus User's Guide and Reference 6-13. Bundtekstens placering er bla. afhængig af linesize og pagesize.

EKSEMPEL

```
TTITLE KUNDEOVERSIGT
SET LINESIZE 66
SET PAGESIZE 18
BTITLE 'EKSISTERENDE KUNDER'
SELECT name,address FROM customer;
```

```
Tue Oct 26                                                    page  1
                              KUNDEOVERSIGT

NAME                                               ADDRESS
-------------------------------------------------- --------------------
JOCKSPORTS                                         345 VIEWRIDGE
TKB SPORT SHOP                                     490 BOLI RD.
VOLLYRITE                                          9722 HAMILTON
JUST TENNIS                                        HILLVIEW MALL
EVERY MOUNTAIN                                     574 SURRY RD.
K + T SPORTS                                       3476 EL PASEO
SHAPE UP                                           908 SEQUOIA
WOMENS SPORTS                                      VALCO VILLAGE
NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER       98 LONE PINE WAY


                           EKSISTERENDE KUNDER
```

Opbrydning i grupper

En rapport kan opbrydes i grupper som der derefter kan foretages statistiske beregninger på. Det betyder at enkeltrækker kan vises samtidig med resultatet af gruppefunktioner.

Opbrydning i grupper udføres med kommandoen **BREAK**. SQL*Plus User's Guide and Reference 6-9.

Statistiske beregninger udføres med kommandoen **COMPUTE**. SQL*Plus User's Guide and Reference 6-25.

EKSEMPEL

Udskriv en rapport over kunder og deres ordrer. Rapporten skal vise de enkelte ordrer, ordretotal pr. kunde og samlet ordretotal.

```
SET PAGESIZE 40
COLUMN ordid FORMAT 9990
BREAK ON report ON name SKIP 1
COMPUTE SUM OF total ON name report
SELECT   name,ordid,total
FROM     customer c,ord o
WHERE    c.custid=o.custid
ORDER BY name;
```

NAME	ORDID	TOTAL
EVERY MOUNTAIN	612	5860.00
	619	1260.00
	607	5.60
	608	35.20
**********		----------
sum		7160.80
JOCKSPORTS	606	3.40
	609	97.50
	620	4450.00
	621	730.00
**********		----------
sum		5280.90
NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER	613	6400.00
**********		----------
sum		6400.00
SHAPE UP	601	2.40
	605	8324.00
	604	698.00
**********		----------
sum		9024.40
VOLLYRITE	611	45.00
	618	3510.50
	602	56.00
	614	23940.00
	603	224.00
**********		----------
sum		27775.50
		----------
sum		103587.00

Variable i titler

Variable kan indlæses i toptitler og bundtitler vha. kommandoen **COLUMN NEW_VALUE**. SQL*Plus User's Guide and Reference 6-21.

EKSEMPEL

Udskriv en rapport over ansatte og deres løn. Rapporten skal udskive afdelingerne sidevis og toptitlen skal vise det aktuelle afdelingsnummer og det aktuelle sidenummer.

```
SET LINESIZE 60
COLUMN deptno NEW_VALUE xdn NOPRINT
TTITLE 'AFDELING:' FORMAT 99 xdn R SIDE FORMAT 99 SQL.PNO SKIP 2
BREAK ON deptno SKIP page
COMPUTE SUM OF sal ON deptno
SELECT deptno,ename,job,sal FROM emp ORDER by deptno;
```

```
Afdeling: 10                                          Side  1

ENAME  JOB              SAL
------ --------- ----------
BLAKE  VICEPRES.   3,975.00
CLARK  MANAGER     2,450.00
KING   PRESIDENT   5,000.00
MILLER CLERK       1,300.00
                  ----------
                  12,725.00

Afdeling: 20                                          Side  2

ENAME  JOB              SAL
------ --------- ----------
JONES  MANAGER     2,975.00
SCOTT  ANALYST     3,000.00
FORD   ANALYST     3,000.00
                  ----------
                   8,975.00

Afdeling: 30                                          Side  3

ENAME  JOB              SAL
------ --------- ----------
ALLEN  SALESMAN    1,600.00
MARTIN SALESMAN    1,250.00
TURNER SALESMAN    1,500.00
JAMES  CLERK         950.00
WARD   SALESMAN    1,250.00
                  ----------
                   6,550.00
```

Udskrivning af rapporter

Udtræk fra databasen kan gemmes i en ascii-fil med kommandoen **SPOOL**. SQL*Plus User's Guide and Reference 6-64. Sålænge SPOOL-kommandoen er aktiv skrives al skærminformation til den angivne fil.

EKSEMPEL

Udskriv rapporten rapport1 til filen kvartal1.

```
SPOOL kvartal1
@rapport1
```

Filen får default extention LST og gemmes altså som kvartal1.lst.

EKSEMPEL

Filen kvartal1.lst lukkes således:

```
SPOOL OFF
```

EKSEMPEL

Luk filen kvartal1.lst og udskriv til default printer.

```
SPOOL OUT
```

Øvelse 12

Formål: - at udskrive SQL-rapporter

1. **Udskriv en rapport der viser produktnummer, beskrivelse af varen samt standardprisen. Gennemsnitsprisen skal vises for hvert produktnummer. Giv rapporten passende overskrifter og titler.**

2. **Udskriv en rapport over de ansattes navn, afdeling og løn fordelt efter stillingsgruppe med subtotaler og en samlet total. Inkluder i rapporten provision for dem der har en sådan. Brug passende overskrifter, formater og titler.**

   Rapporten kan evt. udskrives med en stillingsgruppe pr. side og stillingsbetegnelsen som en del af toptitlen.

3. **Udskriv en rapport efter eget ønske.**

Resumé - kommandoer

SQL-KOMMANDOER

ALTER TABLE	Tilføjer eller omdefinerer en kolonne i en tabel.
COMMIT	Ændringer siden sidste COMMIT gøres permanente.
CREATE TABLE	Opretter en tabel.
CREATE VIEW	Opretter et view.
DELETE	Fjerner rækker fra en tabel.
DROP	Fjerner tabeller og view's.
INSERT	Indsætter rækker i en tabel.
ROLLBACK	Kasserer ændringer foretaget efter sidste COMMIT.
SELECT	Udvælger rækker og kolonner i tabeller.
UPDATE	Ændrer feltværdier i en tabel.

SQL*Plus-KOMMANDOER

/	Udfører kommandoerne i SQL-bufferen.
A(PPEND)	Føjer tekst til indholdet af aktuel buffer-linie.
BREAK	Grupperer rækker efter angivet kolonne.
BTITLE	Indsætter tekst nederst på siden i rapporter.
C(HANGE)	Ændrer indholdet af aktuel buffer-linie.
CLEAR	Fjerner bl.a. compute- og column-definitioner.
COLUMN	Definerer kolonne-format i rapporter.
COMP(UTE)	Udfører beregninger i rapporter.
DEL	Sletter aktuel buffer-linie.
DESC(RIBE)	Udskriver tabel-definition.
ED(IT)	Kalder editor.
EXIT	Afslutter SQL*Plus.
GET	Henter SQL-fil ind i aktuel buffer.
HOST	Udfører UNIX-kommando.
I(NPUT)	Tilføjer nye linier til aktuel buffer.
LIST	Udskriver indhold af aktuel buffer.
RUN	Viser og udfører indholdet af SQL-bufferen.
SAVE	Gemmer indholdet af den aktuelle buffer i en SQL-fil.
SET	Definerer en række SQL-parametre.
SPOOL	Aktiverer udskrivning til fil.
START	Udfører SQL-fil.
TTITLE	Indsætter overskrift i rapporter..

Appendix A

Brugers egne objekter	(Synonym)	Udvidet bruger-view	DBA's globale view
Generelle informationer:			
	(DICT)	DICTIONARY	
		DICT_COLUMNS	
Bruger objekter:			
USER_CATALOG	(CAT)	ALL_CATALOG	DBA_CATALOG
USER_OBJECTS	(OBJ)	ALL_OBJECTS	DBA_OBJECTS
USER_TABLES	(TABS)	ALL_TABLES	DBA_TABLES
USER_IND_COLUMNS	(COLS)	ALL_IND_COLUMNS	DBA_IND_COLUMNS
USER_INDEXES	(IND)	ALL_INDEXES	DBA_INDEXES
USER_COL_COMMENTS		ALL_COL_COMMENTS	DBA_COL_COMMENTS
USER_TAB_COMMENTS		ALL_TAB_COMMENTS	DBA_TAB_COMMENTS
USER_TAB_COLUMNS		ALL_TAB_COLUMNS	DBA_TAB_COLUMNS
USER_VIEWS		ALL_VIEWS	DBA_VIEWS
USER_SYNONYMS	(SYN)	ALL_SYNONYMS	DBA_SYNONYMS
Brugere:			
USER_USERS		ALL_USERS	DBA_USERS
USER_TS_QUOTAS			DBA_TS_QUOTAS
System objekter:			
			DBA_DATA_FILES
USER_TABLESPACES			DBA_TABLESPACES
			DBA_ROLLBACK_SEGS
Lagring af data:			
USER_SEGMENTS			DBA_SEGMENTS
USER_EXTENTS			DBA_EXTENTS
USER_FREE_SPACE			DBA_FREE_SPACE
Rettigheder:			
USER_TAB_GRANTS		ALL_TAB_GRANTS	DBA_TAB_GRANTS
USER_COL_GRANTS		ALL_COL_GRANTS	DBA_COL_GRANTS
USER_TAB_GRANTS_MADE		ALL_TAB_GRANTS_MADE	
USER_COL_GRANTS_MADE		ALL_COL_GRANTS_MADE	
USER_TAB_GRANTS_RECD		ALL_TAB_GRANTS_RECD	
USER_COL_GRANTS_RECD		ALL_COL_GRANTS_RECD	
Export/Import			
			DBA EXP FILES
			DBA EXP VERSION

Stikordsregister

A

add 50
add_months 17
afviklings-kommandoer 9
alias 13, 66
all 6
all 33
alter table 50
and 6
andre funktioner 15
any 6
any 33
aritmetiske funktioner 15
automatisk commit 46
avg 31

B

between 6
break 68
btitle 67
bundtitler 70

C

char 43, 67
column 13
column format 66
column heading 66
column new_value 70
comments 61
commit 45
compute 68
count 21
create index 55
create sequence 59
create table 49
create view 51
currval 59

D

database 1
databaselink 39
datatype 43, 49
date 43, 67
datoformat 66
datofunktioner 15
decending 11
decode 18
def_editor 9
default format 67
definition af index 55
delete 45
demotabeller 3
desc(ribe) 43
dialog 44
drop index 56
drop sequence 60
drop table 49
drop view 52

E

ed 9
equi join 25
exist 6
exists 33

F

formater 66
funktion 15

G

generer fil 37
group by 21
gruppebetingelser 21
gruppefunktioner 21, 68
grupper 21, 68

H

having 21
heading 66
host-base 39

I

in 6, 32
indeks 55
indsæt række 43
indsæt-kommandoer 9
initcap 17
insert 43
intersect 60

J

join 25, 26
justering 67
justify 66

K

karakterfunktioner 15
kolonneoverskrifter 13, 66
kommandofil 37
konverteringsfunktioner 15

L

like 6, 7
linesize 67
linieeditor 9
link 39
list-kommandoer 9
login.sql 9
lokal-base 39
lower 16

M

manipulation af tabelrækker 42
max 22
minus 60
modify 50

N

new value 70
newpage 67
nextval 59
non equi join 26
not 6
null 6
null 12
null-række 27
number 43
number 67
numwidth 67
nvl 12, 31

O

opbrydning i grupper 68
operatorer 6
opret tabel 49
opret view 51
optimering med index 55
or 6
order by 11
outer join 27

P

pagesize 67

R

rapporter 65, 71
redigering 9
regneudtryk 12
relations-database 1
rename 50
rollback 45, 49
rækkebetingelser 21
rækkefunktioner 15

S

sammensat index 55
save/get-kommandoer 9
savepoints 46
sequence 59
session 39
set lin(esize) 67
set newp(age) 67
set numf(ormat) 67
set pages(ize) 67
set spa(ce) 67
sideformattering 67
skærmeditor 9
slet index 56
slet række 45
slet sequence 60
slet tabel 49
slet view 51
sortering 11
spool 37, 71
sql 2
sql select 5
sql*plus 2
statistiske beregninger 68
subquery 31
substr 16
sum 22
søgebetingelser 6

T

tabel 1
tabeloversigt 3
to char 17
toptitler 70
ttitle 67

U

udveksling af data 39
unikt index 55
union 60
unix-kommandoer 10
update 44

V

variable 70
variable i titler 70
view 51

W

where 6

Æ

ændring af række 44
ændring af tabel 50
ændring af view 52
ændrings-kommandoer 9

Løsningsforslag

Opgave 1

```
1  select hiredate,ename,job from emp
   where deptno=30;

2. select * from emp
   where job!='SALESMAN'
   and deptno = 30;

3. select ename,job,sal
   where job in ('MANAGER','ANALYST')
   and sal > 2900;

4. select ename,job from emp
   where hiredate='3-DEC-81';

5. Select ename, empno from emp
   where ename like '_A%' and sal > 1000;

6. select ename,job,sal from emp
   where sal between 900 and 1400 or job='MANAGER';
```

Opgave 2

```
1. select ename navn,job stilling, deptno "AFD. NR." from emp
   order by job,ename;

2. select ename,job,sal,hiredate from emp
   where deptno=30
   order by sal desc;

3. select ename,job from emp
   where 3*(sal+nvl(comm,0)) >5000;

4. select ename navn, sal "LØN", comm provision,
   comm*100/sal "PROV. I % AF LØN"
   from emp
   where (comm*100/sal)>30 and job='SALESMAN';

5. select distinct sal from emp
   order by sal desc;
```

Opgave 3

```
1. select ename, empno from emp
   where (sysdate-hiredate) > 90;

2. select empno||'-'||ename "Ansatte" from emp;

3. select empno "M.NR.",ename navn,initcap(substr(job,1,3)) job
   from emp;

4. select ename navn,
   trunc(months_between(sysdate,hiredate)) "Antal hele mdr."
   from emp where job='CLERK';

5. select ename navn, add_months(hiredate,120) "10 års jubilæum",
   add_months(hiredate,25*12) "25 års jubilæum",
   trunc(months_between(add_months(hiredate,120),sysdate))
   from emp;

6. select ename navn,round(comm*100/sal,2) "Prov. i % af løn"
   from emp where job='SALESMAN';
```

Opgave 4

```
1. select ename navn,to-char(hiredate,'dy dd/mm-yy') anstid from emp;

2. select ename navn,
   to_char(add_months(hiredate,18),'yyddmm(dy)') "Ansat 1½ år pr."
   from emp;

3. select distinct sal,
   decode(sal,800,1,950,1,1100,2,1250,2,1300,2,3)
   from emp;
```

Opgave 5

1.
```
select job stilling,max(sal) "Maks.løn",min(sal) "Min.løn" ,
avg(sal) "Gns. Løn" from emp
group by job;
```

2.
```
select job stilling,round(avg(sal+nvl(comm,0))) "GNSN.TOT"
from emp group by job
having avg(sal + nvl(comm,0)) < 2500;
```

3.
```
select job stilling,max(sal) "Maks.løn",min(sal) "Min.løn",
max(sal)-min(sal) "Forskel",
round((max(sal)-min(sal))*100/max(sal),2) "Forskel i %"
from emp group by job
having round((max(sal)-min(sal))*100/max(sal),2)>20;
```

4.
```
select deptno, COUNT(*) from emp
where job in('MANAGER','PRESIDENT')
group by deptno
having count(*) >1;
```

5.
```
select job, count(*) from emp
where deptno=10
group by job;
```

6.
```
select deptno,12*sum(sal+nvl(comm,0)) from emp
group by deptno;
```

Opgave 6

1.
```
select ename,dname,loc
from emp,dept
where emp.deptno=dept.deptno
and ename in ('ALLEN','JONES','CLARK');
```

2.
```
select ansat.ename ansat,overordnet.ename overordnet
from emp ansat,emp overordnet
where ansat.mgr=overordnet.empno
and ansat.job='SALESMAN';
```

3.
```
select ansat.ename ansat,overordnet.ename overordnet
from emp ansat,emp overordnet
where ansat.mgr=overordnet.empno
and overordnet.ename='JONES';
```

4.
```
select a.ename ansat,o.ename overordnet,
a.hiredate "A. ansat",o.hiredate "O. ansat"
from emp a,emp o
where a.mgr=o.empno
and a.hiredate<o.hiredate;
```

5.
```
select distinct dept.deptno "afd.",dname "afd. navn", loc sted
from dept, emp
where dept.deptno = emp.deptno(+)
and empno is null;
```

6.
```
select a.ename, a.job, a.sal
from emp a, emp B
where B.ename='BLAKE'
and a.sal > B.sal;
```

7.
```
select a.* from emp a, emp o
where a.job='SALESMAN'
and a.mgr=o.empno
and o.job!='MANAGER';
```

8.
```
select a.* from emp a, emp P
where P.job='PRESIDENT'
and a.deptno=P.deptno
and a.job!='PRESIDENT';
```

Opgave 7

1.
```
select ename,sal from emp
where sal=(select sal from emp where ename='WARD')
and ename!='WARD';
```

2.
```
select ename,sal from emp
where sal>(select sal from emp where ename='TURNER')
and ename!='TURNER'
order by sal;
```

3.
```
select ename,sal,deptno from emp
where deptno = 30
and sal > (select avg(sal) from emp,dept
           where emp.deptno=dept.deptno
           and dept.dname='SALES');
```

4.
```
select * from emp
where job='CLERK'
and sal > (select max(sal) from emp
           where deptno = 20
           and job='CLERK');
```

5.
```
select * from emp
where job='CLERK'
and length(ename)=5
and sal > (select sal from emp
           where job='CLERK'
           and deptno=30);
```

6.
```
select ename,sal+nvl(comm,0) "Totalløn",deptno from emp
where deptno=30 and
sal+nvl(comm,0)>(select avg(sal+nvl(comm,0)) from emp);
```

7.
```
select ename,job,deptno,sal from emp
where (sal,deptno) in (select max(sal),deptno from emp
                       group by deptno);
```

8.
```
select * from emp
where deptno!=30
and sal > any (select sal from emp
               where deptno=30);
```

Opgave 8

1. insert into emp values
   (7912,'OLSEN','TEGNER',null,'1-dec-89', null,null,50);

2. update emp
   set ename='OLESEN'
   where empno=7912;

3. update emp
   set sal=sal*1.1
   where deptno=10 and ename!='KING';

4. delete from dept
   where deptno=40;

5. commit

6. delete from dept;
   select * from dept;
   rollback
   select * from dept;

7. delete from emp
    where sal>
   (select sal from emp where job='PRESIDENT');

8. update emp
   set mgr=(select empno from emp where ename='CLARK')
   where mgr=(select empno from emp where ename='JONES');

Opgave 9

1. Mulige kolonner i emp-tabellen: empno, deptno, ename, (sal)
   Mulige kolonner i dept-tabellen: deptno, dname, loc

   Umulig kolonne i emp-tabellen: comm

2. Eksempel:
   create unique index (emp_no)
   on emp (empno);

3. Unik indeks dannes på comm.
   Unik indeks kan derimod ikke dannes på f.eks. sal.

Opgave 10

1. create table projekt
   (nr number(2) not null,
    besk char(30),
    ledernr number(4));

2. insert into projekt values
   (45,'organisationsplan',7734);
   1 record created

3. create table ny-projekt
   as select * from projekt;

4. alter table ny-projekt
   modify(ename char(40));

5. alter table ny-projekt
   add (fname char(12));

6. create view afd1020 (navn,mnr,afd)
   as select ename,empno,deptno from emp
   where deptno in (10,20);

7. select * from afd1020
   where mnr between 7400 and 7900;

8. drop view afd1020;

9. drop table ny-projekt;

Opgave 11

1. Lægges rapporten i en fil kan indholdet være:

```
set pagesize 60
column deptno heading AFD
column ename heading NAVN format a20
column job heading STILLING format a20
column sal+nvl(comm,0) heading 'TOTALLØN' format 99,999.99
column comm heading 'HERAF PROV.' format 9,999.99
ttitle col 22 'Personale-oversigt' skip -
col 24 'Afdelingopdelt' skip 2
break on job skip
compute sum of sal+nvl(comm,0) on job
select deptno,ename,sal+nvl(comm,0),job from emp
order by job
/
```

2. Lægges rapporten i en fil kan indholdet være:

```
set pagesize 60
column deptno heading AFD
column ename heading NAVN format a20
column trunc(sysdate-hiredate) heading 'ANTAL DG. ANSAT'
column to_char(hiredate,'dd/mm - yy')heading 'ANSÆTTELSESDATO'
ttitle col 22 'Personale-oversigt' skip -
col 24 'Anciennitet' skip 2
select ename,deptno,
to_char(hiredate,'dd/mm - yy'),trunc(sysdate-hiredate)
order by hiredate
/
```

CUSTOMER

CUSTID	NAME	ADDRESS	CITY	STATE	ZIP	AREA	PHONE	REPID	CREDITLIMIT
100	JOCKSPORTS	345 VIEWRIDGE	BELMONT	CA	96711	415	598-6609	7844	5000
101	TKB SPORT SHOP	490 BOLI RD.	REDWOOD CITY	CA	94061	415	368-1223	7521	10000
102	VOLLYRITE	9722 HAMILTON	BURLINGAME	CA	95133	415	644-3341	7654	7000
103	JUST TENNIS	HILLVIEW MALL	BURLINGAME	CA	97544	415	677-9312	7521	3000
104	EVERY MOUNTAIN	574 SURRY RD.	CUPERTINO	CA	93301	408	996-2323	7499	10000
105	K + T SPORTS	3476 EL PASEO	SANTA CLARA	CA	91003	408	376-9966	7844	5000
106	SHAPE UP	908 SEQUOIA	PALO ALTO	CA	94301	415	364-9777	7521	6000
107	WOMENS SPORTS	VALCO VILLAGE	SUNNYVALE	CA	93301	408	967-4398	7499	10000
108	NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER	98 LONE PINE WAY	HIBBING	MN	55649	612	566-9123	7844	8000

ORD

ORDID	ORDERDATE	C	CUSTID	SHIPDATE	TOTAL
610	07-JAN-87	A	101	08-JAN-87	101.4
611	11-JAN-87	B	102	11-JAN-87	45
612	15-JAN-87	C	104	20-JAN-87	5860
601	01-MAJ-86	A	106	30-MAJ-86	2.4
602	05-JUN-86	B	102	20-JUN-86	56
604	15-JUN-86	A	106	30-JUN-86	698
605	14-JUL-86	A	106	30-JUL-86	8324
606	14-JUL-86	A	100	30-JUL-86	3.4
609	01-AUG-86	B	100	15-AUG-86	97.5
607	18-JUL-86	C	104	18-JUL-86	5.6
608	25-JUL-86	C	104	25-JUL-86	35.2
603	05-JUN-86		102	05-JUN-86	224
620	12-MAR-87		100	12-MAR-87	4450
613	01-FEB-87		108	01-FEB-87	6400
614	01-FEB-87		102	05-FEB-87	23940
616	03-FEB-87		103	10-FEB-87	764
619	22-FEB-87		104	04-FEB-87	1260
617	05-FEB-87		105	03-MAR-87	46370
615	01-FEB-87		107	06-FEB-87	710
618	15-FEB-87	A	102	06-MAR-87	3510.5
621	15-MAR-87	A	100	01-JAN-87	730

ITEM

ORDID	ITEMID	PRODID	ACTUALPRICE	QTY	ITEMTOT
610	3	100890	58	1	58
611	1	100861	45	1	45
612	1	100860	30	100	3000
601	1	200376	2.4	1	2.4
602	1	100870	2.8	20	56
604	1	100890	58	3	174
604	2	100861	42	2	84
604	3	100860	44	10	440
603	2	100860	56	4	224
610	1	100860	35	1	35
610	2	100870	2.8	3	8.4
613	4	200376	2.2	200	440
614	1	100860	35	444	15540
614	2	100870	2.8	1000	2800
612	2	100861	40.5	20	810
612	3	101863	10	150	1500
620	1	100860	35	10	350
620	2	200376	2.4	1000	2400
620	3	102130	3.4	500	1700
613	1	100871	5.6	100	560
613	2	101860	24	200	4800
613	3	200380	4	150	600
619	3	102130	3.4	100	340
617	1	100860	35	50	1750
617	2	100861	45	100	4500
614	3	100871	5.6	1000	5600
616	1	100861	45	10	450
616	2	100870	2.8	50	140
616	3	100890	58	2	116
616	4	102130	3.4	10	34
616	5	200376	2.4	10	24
619	1	200380	4	100	400
619	2	200376	2.4	100	240
615	1	100861	45	4	180
607	1	100871	5.6	1	5.6
615	2	100870	2.8	100	280
617	3	100870	2.8	500	1400
617	4	100871	5.6	500	2800
617	5	100890	58	500	29000
617	6	101860	24	100	2400
617	7	101863	12.5	200	2500
617	8	102130	3.4	100	340
617	9	200376	2.4	200	480
617	10	200380	4	300	1200
609	2	100870	2.5	5	12.5
609	3	100890	50	1	50
618	1	100860	35	23	805
618	2	100861	45.11	50	2255.5
618	3	100870	45	10	450
621	1	100861	45	10	450
621	2	100870	2.8	100	280
615	3	100871	5	50	250
608	1	101860	24	1	24
608	2	100871	5.6	2	11.2
609	1	100861	35	1	35
606	1	102130	3.4	1	3.4
605	1	100861	45	100	4500
605	2	100870	2.8	500	1400
605	3	100890	58	5	290
605	4	101860	24	50	1200
605	5	101863	9	100	900
605	6	102130	3.4	10	34
612	4	100871	5.5	100	550
619	4	100871	5.6	50	280

PRODUCT

PRODID	DESCRIP
100860	ACE TENNIS RACKET I
100861	ACE TENNIS RACKET II
100870	ACE TENNIS BALLS-3 PACK
100871	ACE TENNIS BALLS-6 PACK
100890	ACE TENNIS NET
101860	SP TENNIS RACKET
101863	SP JUNIOR RACKET
102130	RH: "GUIDE TO TENNIS"
200376	SB ENERGY BAR-6 PACK
200380	SB VITA SNACK-6 PACK

PRICE

PRODID	STDPRICE	MINPRICE	STARTDATE	ENDDATE
100871	4.8	3.2	01-JAN-85	01-DEC-85
100890	58	46.4	01-JAN-85	
100890	54	40.5	01-JUN-84	31-MAJ-84
100860	35	28	01-JUN-86	
100860	32	25.6	01-JAN-86	31-MAJ-86
100860	30	24	01-JAN-85	31-DEC-85
100861	45	36	01-JUN-86	
100861	42	33.6	01-JAN-86	31-MAJ-86
100861	39	31.2	01-JAN-85	31-DEC-85
100870	2.8	2.4	01-JAN-86	
100870	2.4	1.9	01-JAN-85	01-DEC-85
100871	5.6	4.8	01-JAN-86	
101860	24	18	15-FEB-85	
101863	12.5	9.4	15-FEB-85	
102130	3.4	2.8	18-AUG-85	
200376	2.4	1.75	15-NOV-86	
200380	4	3.2	15-NOV-86	

DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAJ-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	09-DEC-82	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	12-JAN-83	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

dde Dansk Data Elektronik A/S

SQL-Plus Kursus	Initials GR	Page
	Date 10+11/8-95	Project

c UNV2 , ora1/ , printer lg3 ,

DML Data Manipulation Language
DDL Data Definition Language

Desc. <tablenm>

types : NUMBER
CHAR
DATE

'NOT NULL' - key primary key / foreign key ref. til ande tabel

---

Tabel DUAL har en colom SYSDATE der indeholder dags dato

SELECT
FROM
WHERE

ORDER BY
GROUP BY · HAVING

Group func :

avg (n) gennemsnit
count (n) antal af rækker
max (n)
min
sum

George Koch "Oracle 7, the complete ref"
ISBN 0-07-88 1919-9

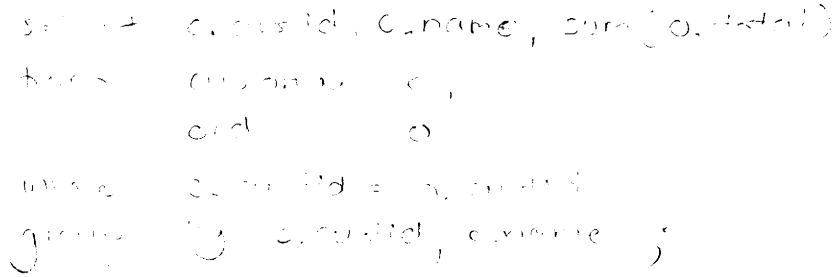
Oracle's database for Windows 25 Aug ; kurs afd for PROSA kursister (Bente BFN)

CUSTOMER

CUSTID	NAME	ADDRESS	CITY	STATE	ZIP	AREA	PHONE	REPID	CREDITLIMIT
100	JOCKSPORTS	345 VIEWRIDGE	BELMONT	CA	96711	415	598-6609	7844	5000
101	TKB SPORT SHOP	490 BOLI RD.	REDWOOD CITY	CA	94061	415	368-1223	7521	10000
102	VOLLYRITE	9722 HAMILTON	BURLINGAME	CA	95133	415	644-3341	7654	7000
103	JUST TENNIS	HILLVIEW MALL	BURLINGAME	CA	97544	415	677-9312	7521	3000
104	EVERY MOUNTAIN	574 SURRY RD.	CUPERTINO	CA	93301	408	996-2323	7499	10000
105	K + T SPORTS	3476 EL PASEO	SANTA CLARA	CA	91003	408	376-9966	7844	5000
106	SHAPE UP	908 SEQUOIA	PALO ALTO	CA	94301	415	364-9777	7521	6000
107	WOMENS SPORTS	VALCO VILLAGE	SUNNYVALE	CA	93301	408	967-4398	7499	10000
108	NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER	98 LONE PINE WAY	HIBBING	MN	55649	612	566-9123	7844	8000

ORD

ORDID	ORDERDATE	C	CUSTID	SHIPDATE	TOTAL
610	07-JAN-87	A	101	08-JAN-87	101.4
611	11-JAN-87	B	102	11-JAN-87	45
612	15-JAN-87	C	104	20-JAN-87	5860
601	01-MAJ-86	A	106	30-MAJ-86	2.4
602	05-JUN-86	B	102	20-JUN-86	56
604	15-JUN-86	A	106	30-JUN-86	698
605	14-JUL-86	A	106	30-JUL-86	8324
606	14-JUL-86	A	100	30-JUL-86	3.4
609	01-AUG-86	B	100	15-AUG-86	97.5
607	18-JUL-86	C	104	18-JUL-86	5.6
608	25-JUL-86	C	104	25-JUL-86	35.2
603	05-JUN-86		102	05-JUN-86	224
620	12-MAR-87		100	12-MAR-87	4450
613	01-FEB-87		108	01-FEB-87	6400
614	01-FEB-87		102	05-FEB-87	23940
616	03-FEB-87		103	10-FEB-87	764
619	22-FEB-87		104	04-FEB-87	1260
617	05-FEB-87		105	03-MAR-87	46370
615	01-FEB-87		107	06-FEB-87	710
618	15-FEB-87	A	102	06-MAR-87	3510.5
621	15-MAR-87	A	100	01-JAN-87	730



ITEM

listepris

ORDID	ITEMID	PRODID	ACTUALPRICE	QTY	ITEMTOT
610	3	100890	58	1	58
611	1	100861	45	1	45
612	1	100860	30	100	3000
601	1	200376	2.4	1	2.4
602	1	100870	2.8	20	56
604	1	100890	58	3	174
604	2	100861	42	2	84
604	3	100860	44	10	440
603	2	100860	56	4	224
610	1	100860	35	1	35
610	2	100870	2.8	3	8.4
613	4	200376	2.2	200	440
614	1	100860	35	444	15540
614	2	100870	2.8	1000	2800
612	2	100861	40.5	20	810
612	3	101863	10	150	1500
620	1	100860	35	10	350
620	2	200376	2.4	1000	2400
620	3	102130	3.4	500	1700
613	1	100871	5.6	100	560
613	2	101860	24	200	4800
613	3	200380	4	150	600
619	3	102130	3.4	100	340
617	1	100860	35	50	1750
617	2	100861	45	100	4500
614	3	100871	5.6	1000	5600
616	1	100861	45	10	450
616	2	100870	2.8	50	140
616	3	100890	58	2	116
616	4	102130	3.4	10	34
616	5	200376	2.4	10	24
619	1	200380	4	100	400
619	2	200376	2.4	100	240
615	1	100861	45	4	180
607	1	100871	5.6	1	5.6
615	2	100870	2.8	100	280
617	3	100870	2.8	500	1400
617	4	100871	5.6	500	2800
617	5	100890	58	500	29000
617	6	101860	24	100	2400
617	7	101863	12.5	200	2500
617	8	102130	3.4	100	340
617	9	200376	2.4	200	480
617	10	200380	4	300	1200
609	2	100870	2.5	5	12.5
609	3	100890	50	1	50
618	1	100860	35	23	805
618	2	100861	45.11	50	2255.5
618	3	100870	45	10	450
621	1	100861	45	10	450
621	2	100870	2.8	100	280
615	3	100871	5	50	250
608	1	101860	24	1	24
608	2	100871	5.6	2	11.2
609	1	100861	35	1	35
606	1	102130	3.4	1	3.4
605	1	100861	45	100	4500
605	2	100870	2.8	500	1400
605	3	100890	58	5	290
605	4	101860	24	50	1200
605	5	101863	9	100	900
605	6	102130	3.4	10	34
612	4	100871	5.5	100	550
619	4	100871	5.6	50	280

PRODUCT

PRODID	DESCRIP
100860	ACE TENNIS RACKET I
100861	ACE TENNIS RACKET II
100870	ACE TENNIS BALLS-3 PACK
100871	ACE TENNIS BALLS-6 PACK
100890	ACE TENNIS NET
101860	SP TENNIS RACKET
101863	SP JUNIOR RACKET
102130	RH: "GUIDE TO TENNIS"
200376	SB ENERGY BAR-6 PACK
200380	SB VITA SNACK-6 PACK

PRICE

PRODID	STDPRICE	MINPRICE	STARTDATE	ENDDATE
100871	4.8	3.2	01-JAN-85	01-DEC-85
100890	58	46.4	01-JAN-85	
100890	54	40.5	01-JUN-84	31-MAJ-84
100860	35	28	01-JUN-86	
100860	32	25.6	01-JAN-86	31-MAJ-86
100860	30	24	01-JAN-85	31-DEC-85
100861	45	36	01-JUN-86	
100861	42	33.6	01-JAN-86	31-MAJ-86
100861	39	31.2	01-JAN-85	31-DEC-85
100870	2.8	2.4	01-JAN-86	
100870	2.4	1.9	01-JAN-85	01-DEC-85
100871	5.6	4.8	01-JAN-86	
101860	24	18	15-FEB-85	
101863	12.5	9.4	15-FEB-85	
102130	3.4	2.8	18-AUG-85	
200376	2.4	1.75	15-NOV-86	
200380	4	3.2	15-NOV-86	

$\frac{STD \times K}{100} = MIN$

DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

$\frac{SAL \times K}{100} = COMM$

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAJ-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	09-DEC-82	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	12-JAN-83	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

Data Display Deutschland GmbH
Hansaallee 203
D-40549 Düsseldorf
Tel. +49 (0)211/52 61 01
Fax +49 (0)211/52 61 09