Kap1.Introduksjon til SQL

- Hva er en relasjonsdatabase ?
- Innlogging.
- Forskjell mellom SQL og SQL*Plus.
- Bruk av linje-editor i SQL*Plus.
- Bruk av system-editor i SQL*Plus.
- Hjelpe systemet i SQL*Plus.

Hva er en relasjonsdatabase.

- Data er representert i tabeller.
- En tabell har
  - kolonner
  - rader.
- Ingen definerte koblinger mellom tabeller når det gjelder søking.
- Ingen"synlige" pekerkjeder.
- All informasjon eksplisitt som verdier.
- Logisk representasjon uavhengig av fysisk lagring.
- Språk som kan utføre relasjonsoperasjoner og presentere resultatet som en tabell.

Innlogging

Gjennom SQL*Plus kan du bruke SQL interaktivt.

Vi skal først gjøre noen enkle øvelser for bli kjent med SQL*Plus. Først må vi logge inn på maskinen med brukernavn sql_1, sql_2 osv...og passord sql_1 , sql_2 osv... Når du er innlogget, starter du SQL*Plus ved å skrive :

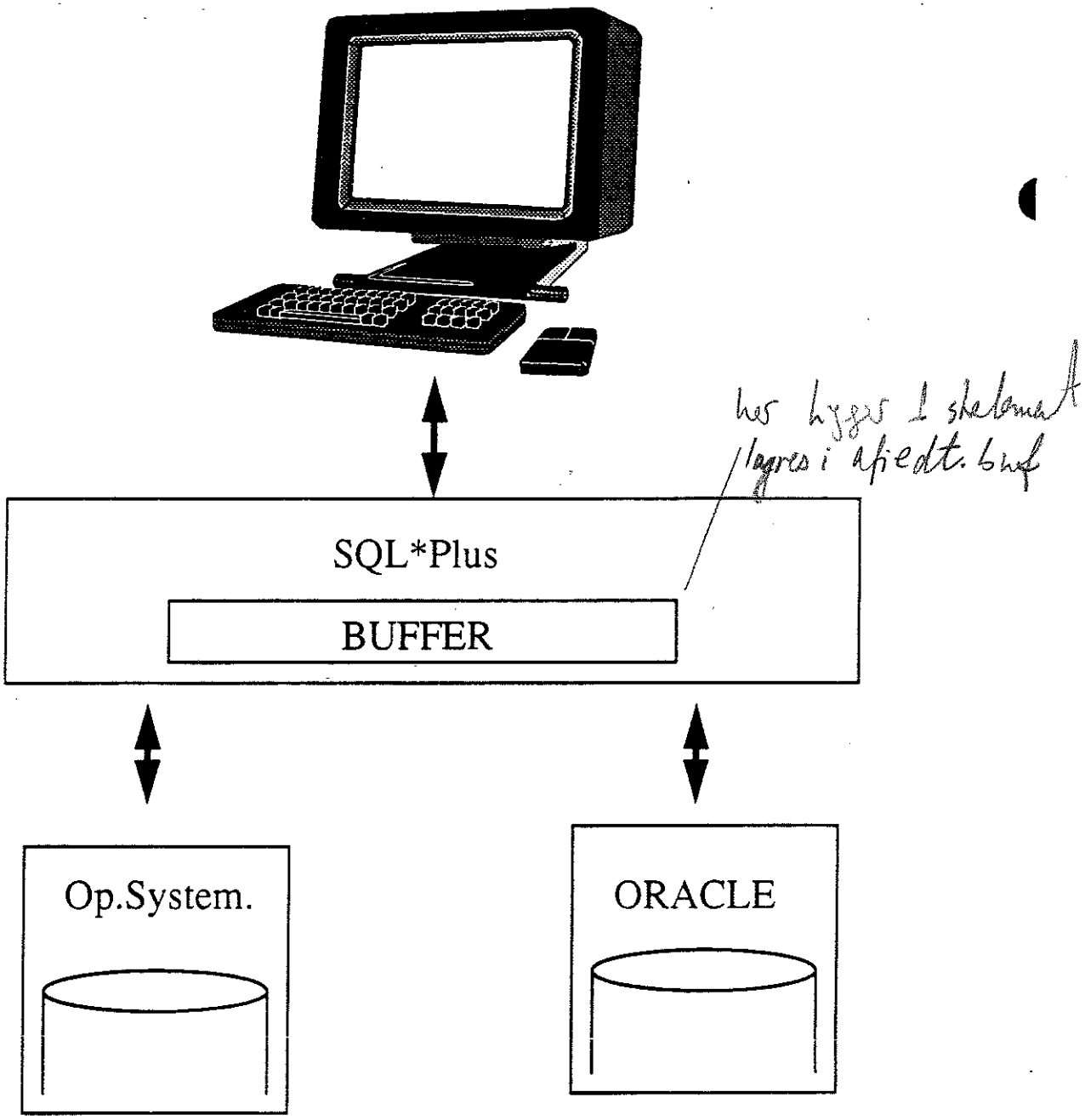
sqlplus

f.eks:

```
nosparc3:[48]% sqlplus

SQL*Plus: Version 3.0.9.1.2 - Production on Fri Jul 17 12:29:26 1992

Copyright (c) Oracle Corporation 1979, 1989.  All rights reserved.
                                    scott / tiger
Enter user-name: sql_1
Enter password: sql_1

Connected to:
ORACLE RDBMS V6.0.33.1.1, transaction processing option - Production
PL/SQL V1.0.32.3.1 - Production

SQL> exit
Disconnected from ORACLE RDBMS V6.0.33.1.1, transaction processing
option - Prod
uction
PL/SQL V1.0.32.3.1 - Production
nosparc3:[49]%
```

SQL*Plus

SQL*Plus er et grensesnitt både til databasen og til operativsystemet.

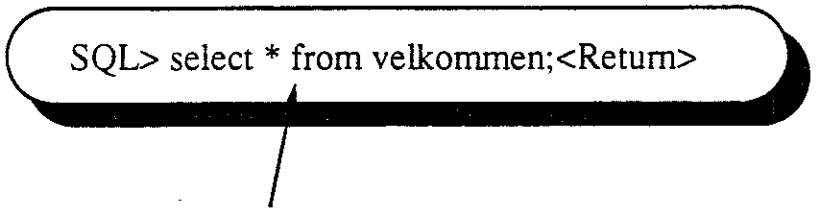


Eksempel på SQL*Plus og SQL :

Vi har en enkel tabell med to kolonner.Tabellen heter VELKOMMEN. Kolonnene heter NUMMER og TEKST.

Velkommen

Nummer	Tekst
101	Det er en stor fornøyelse
102	å ønske dere alle velkommen
103	til dette kurset
104	Lykke til med øvelsene !!!



Dette er den teksten som du skal taste inn.

Et eksempel til

```
SQL> select
  2 *
  3 from
  4 velkommen
  5
SQL>
```

Vi ser her at:

1. En SQL-setning kan gå over flere linjer. Når vi trykker på <Return> - tasten, får vi en ny linje. SQL*Plus setter selv nummer på linjen.

2. Hvis vi ikke skriver noe på linjen før vi trykker <Return> skjønner SQL*Plus at det er slutt på SQL setningen.

3. Men hvorfor skjer det ikke noe?

SQL-bufferen

SQL*Plus har en SQL-buffer. Det er et område i hukommelsen der SQL-setningen lagres.

SQL*Plus vet hvilke ord en SQL-setning kan starte med (f.eks.SELECT). Når vi skriver et slikt ord, vet SQL*Plus at det er starten på en SQL-setning. SQL*Plus vil da lagre SQL-setningen i bufferen etter hvert som vi taster det inn. Bufferen har mulighet til å lagre flere linjer, dvs. at en SQL-setning kan gå over flere linjer.

Hvordan vet SQL*Plus når SQL-setningen er slutt?

Enten ved at vi trykker return uten å skrive noe på siste linje, eller ved at vi avslutter med semikolon(;) eller skråstrek (/). Semikolonet kan skrives bakerst på en linje eller for seg selv på ny linje. Skråstrek må skrives på en ny linje.

I første tilfelle vil SQL*Plus bare lagre SQL-setningen uten å gjøre noe mer med det. I de to siste tilfellene vil det bli utført umiddelbart.

Legg merke til at ; eller / ikke lagres i SQL-bufferen. Det kan bare være en SQL-setning i bufferen om gangen.Det betyr at med en gang vi begynner på en ny setning, blir den gamle slettet fra bufferen. (Men hver bruker har sin egen SQL-buffer).

Ønsker vi å få utført SQL-setningen i bufferen, kan vi skrive RUN (eller bare R) eller / og trykke return.

Ønsker vi å få selve SQL-setningen skrevet ut på skjermen, kan vi skrive LIST(eller bare L) og trykke return.

R utfører SQL*Plus kommando RUN.

L utfører SQL*Plus kommando LIST.

SQL*Plus-kommandoer

RUN og LIST er SQL*Plus-kommandoer. Dvs. at de er IKKE en del av SQL-setningen, og de blir IKKE lagt inn i SQL-bufferen.

Du kan skrive RUN eller LIST direkte når det står SQL først på linjen, men altså ikke i linje 2,3 osv i SQL-bufferen.

Det finnes mange andre SQL*Plus-kommandoer, og vi skal se på flere av dem etter hvert.

Det er viktig å være klar over hva som er forskjellen på SQL-setninger og SQL*Plus-kommandoer.

SQL-setninger og SQL*Plus-Kommandoer.

SQL-setninger :

- ☞ Starter med SQL-ord
- ☞ Ligger i SQL-bufferen
- ☞ Er ordrer om lesing og skriving på databasen.

SQL*Plus-kommandoer.

- ☞ Ligger ikke i SQL-bufferen.
- ☞ Er ordrer til SQL*Plus om hvordan SQL-setningen skal behandles.

SQL-setninger

Nå har du lært hvordan du kan skrive en SQL-setning inn i SQL-bufferen, og hvordan du kan skrive det ut på skjermen eller utføre det.

Vi skal nå se på noen eksempler på SQL-setninger. Du husker sikkert SQL-setningen:

```
SQL> select * from velkommen;
```

Prøv også disse:

```
SQL> select nummer,tekst from velkommen;
SQL> select nummer from velkommen;
SQL> select tekst from velkommen;
SQL> select tekst,nummer from velkommen;
```

SQL*Plus editor

Det er ikke nødvendig å skrive hele SQL-setningen på nytt når det skal gjøres endringer. SQL*Plus har en egen linje editor som gjør at du kan forandre på SQL-setningen som ligger i SQL-bufferen.

```
1* select tekst,nummer from velkommen
```

Stjernen etter linjenummeret viser at linjen er AKTIV. Siden vi har med en linje-editor å gjøre så betyr det at det er kun denne aktive linjen som kan editeres.

Skriv en SQL-setning som går over flere linjer:

```
SQL> select
  2  *
  3  from
  4  velkommen
  5
SQL> l
  1  select
  2  *
  3  from
  4* velkommen
SQL>
```

Etter ha gjort RUN eller LIST, ser du at den siste linjen er AKTIV.

Vi kan liste en enkelt linje av gangen, ved å angi linjenummeret sammen med liste kommandoen slik:

```
 1  select
 2  *
 3  from
 4* velkommen
SQL> l1
 1* select
SQL>
```

Den linjen du lister, blir AKTIV. Prøv!!

Dette er nyttig fordi SQL*Plus editoren opererer kun på linjen som er AKTIV.Vi har hittil lært R - Run og L - List SQL*Plus kommandoer. Andre SQL*Plus kommandoer er :

- I - Input (Åpner opp nye linjer som legger seg bak den aktive linjen.)
- A - Append (Føyer til tekst bakerst på den aktive linjen).
- C.gammeltekst.nytekst. - Change (Endrer første forekomst av gammeltekst på aktive linjen med nytekst).
- Del - Delete line (Sletter aktiv linje)

Eksempler

La oss ta vårt gamle eksempel.

```
SQL> select
  2  *
  3  from
  4  velkommen;
```

Og prøv disse :

```
SQL> l2
  2* *
SQL> c.*.tekst.
SQL> r
```

```
SQL> l2
  2* tekst
SQL> a,nummer
  2* tekst,nummer
SQL> r
```

```
SQL> del
SQL> l
  1  select
  2  tekst,nummer
  3* from
SQL> i
  4  velkommen
  5
SQL> r
```

Hva skjer ?

Bruk av systemeditor

Det går raskt å gjøre små endringer med SQL*Plus editoren, men til større endringer er en kraftigere editor å foretrekke. SQL*Plus har en åpning til maskinens/operativsystemets egen editor.

☞ ED eller EDIT kommando

bruker systemeditoren til å editere SQL-bufferen.

Det som skjer er følgende:

1. SQL-bufferen lagres i en fil som heter afiedt.buf.
2. Systemeditoren startes, filen afiedt.buf leses automatisk inn i systemeditoren slik at du kan gjøre dine endringer med vanlige systemeditor kommandoer.
3. Når systemeditoren forlates, blir den endrede afiedt.buf lest inn i SQL-bufferen igjen.

Du kan velge hvilken editor som skal benyttes ved å skrive

define _editor = (editor_navn)

f.eks:

```
SQL> define _editor=emacs
```

Fil-kommandoer

- EDIT filnavn - Editere en fil.
- SAVE filnavn - Kopierer SQL-bufferen til en fil.
- GET filnavn - Kopierer en fil inn i SQL-bufferen
- START filnavn - Leser fra en fil, og utfører de kommandoer som filen inneholder, som om de ble tastet inn på tastaturet.

OBS ! Default filnavn er: filnavn.sql.

Vær oppmerksom på følgende:

- ☞ EDIT-komandoen vil, når filnavn oppgis, kun påvirke den angitte fil, og ikke endre SQL-bufferen.
- ☞ SAVE-kommandoen endrer ikke innholdet i SQL-bufferen.
- ☞ SAVE-kommandoen utfører ikke SQL-setningen.
- ☞ GET-kommandoen er bare meningsfylt dersom filen inneholder en og bare en SQL-setning.
- ☞ START-kommandoen kan brukes på en fil som inneholder flere SQL*Plus-kommandoer og/eller SQL-setninger. Disse blir utført etterhvert som de leses fra filen.
- ☞ SQL-setninger må avsluttes med ; eller / og blir da utført umiddelbart.
- ☞ Siste SQL-setning blir liggende i SQL-bufferen.

Andre SQL*Plus kommandoer.

- ? el. HELP kommandonavn - Gir hjelpeinformasjon om angitte SQL*Plus kommando el. SQL-ord.
- DESC el. DESCRIBE tabellnavn - Lister opp kolonnenavnene i den angitte tabellen.

Disse kommandoene påvirker ikke SQL-bufferen. De er derfor nyttige hvis du står fast på noe du ikke husker, mens du holder på å skrive en SQL-setning. Du kan da avbryte skrivingen av setningen og få hjelp av disse kommandoene.

Deretter kan du fortsette på SQL-setningen.