Kap3.Oppdateringsspråk

- INSERT
- UPDATE
- DELETE

OPPDATERINGSSPRÅK(Data Manipulation Language)

Del av SQL-språket

☞ INSERT

- Legger nye rader inn i en tabell.

☞ UPDATE

- Endrer innholdet i eksisterende rader.

☞ DELETE

- Sletter rader fra en tabell.

INSERT

Legger nye rader inn i en tabell.

SYNTAKS:

```
INSERT INTO tabellnavn  VALUES (verdi,...,verdi)
```

Krever en verdi for alle kolonnene som er definert i tabellen, i den rekkefølgen de er definert.

```
INSERT INTO tabellnavn (kolonnenavn,...,kolonnenavn)
       VALUES (verdi,...,verdi)
```

Benyttes hvis bare noen av kolonnene skal ha verdi, eller hvis man vil angi kolonnene i en annen rekkefølge.

Pass på følgende:

- ☞ Alle kolonner som er definert som NOT NULL, må ha verdi.
- ☞ Alfanummeriske verdier skal stå mellom enkelt-apostrofer.
- ☞ Ordet NULL uten apostrofer betyr NULL VALUE.

EKSEMPEL:

```
INSERT INTO ANSATT VALUES (8011,'LI','SJEF',NULL,1000);
```

Insert into ansatt (ANSNR, ENAVN)
values
(8011, 'LI');

INSERT med kopiering

Vi kan hente verdier fra andre (eller samme) tabell ved å skrive en SELECT-setning i stedet for VALUES.

Det kan være en hvilken som helst gyldig SELECT- setning, så lenge det returnerer riktig antall kolonner og oppfyller de øvrige kravene som INSERT stiller.

Det blir lagt inn en rad for hver rad SELECT-setningen returnerer.

SYNTAKS:

INSERT INTO tabellnavn SELECT ...

INSERT INTO tabellnavn (kolonnenavn,...,kolonnenavn) SELECT ...

MORTEN OMVIK

KURS ORACLE

Eksempel

Ansatt

Ans No	Navn	Lonn	fdato

Pensj

P_NO	NAVN	P_LONN

```
insert into pnsj (p-no, navn, p_lonn)
    select from ansatt ans no, navn, lonn * 0,8
        where f_dato < 01-JAN-29;

delete from ansatt
    where F_DATO < 01-JAN-29;
```

UPDATE

Endrer innholdet i eksisterende rader.

SYNTAKS:

```
UPDATE tabellnavn SET kolonnenavn = uttrykk, kolonnenavn = uttrykk
WHERE ...
```

Man kan oppdatere en eller flere kolonner.

Kolonnene kan settes lik en konstant eller et uttrykk der samme kolonne eller en annen kolonne benyttes.

EKSEMPEL:

```
UPDATE ANSATT SET LONN = LONN * 1.1 WHERE ANSNR = 7934
```

WHERE-betingelsen styrer hvilke rader som blir endret.

Uten WHERE endres alle radene.

UPDATE med SELECT

UPDATE-setningen kan bruke en SELECT-setning til å finne den verdien som skal inn i oppdateringen.

SYNTAKS:

```
UPDATE tabellnavn SET kolonnenavn = (SELECT ....) WHERE ...
```

SELECT-setningen må returnere 1 rad.

EKSEMPEL:

```
UPDATE ANSATT SET LONN = (SELECT LONN FROM ANSATT
WHERE ANSNR= 7654)
WHERE ANSNR=7876
```

DELETE

Sletter rader fra tabellen.

SYNTAKS:

```
DELETE FROM tabellnavn WHERE ...
```

WHERE-betingelsen styrer hvilke rader som blir slettet.

Uten WHERE slettes alle radene.

COMMIT og ROLLBACK

COMMIT eller COMMIT WORK sørger for at utførte oppdatering (INSERT, UPDATE og DELETE) blir gjort gjeldende permanent.

Inntil COMMIT er gjort, kan oppdateringene fjernes med ROLLBACK eller ROLLBACK WORK. Databasen vil da bli satt tilbake til slik den var ved siste COMMIT. (Angrefrist)

Har vi f.eks. hatt en feil i WHERE-betingelsen (eller kanskje glemt den) i en UPDATE eller DELETE-setning, er denne angrefristen god ha.

Endringer som ikke er COMMIT-et, er usynlig for andre brukere enn den som har gjort endringen.

Ved kontrollert utgang fra SQL*Plus skjer COMMIT automatisk, ved ukontrollert utgang skjer ROLLBACK.

DDL-setninger gir automatisk COMMIT både før og etter.

SQL*Plus-kommandoen SET AUTOCOMMIT IMM gir automatisk COMMIT umiddelbart etter alle oppdateringer. Slås av igjen med SET AUTOCOMMIT OFF. SHOW AUTOCOMMIT viser hva som gjelder.