6 Data Manipulation Language

6.1 Gegevens invoeren

6.2 Gegevens wijzigen

6.3 Gegevens verwijderen

6.4 Transactieverwerking

6.5 Oefeningen

6 Data Manipulation Language

- Productieomgeving: datamanipulatie via applicaties
- Applicaties over het algemeen gebouwd m.b.v. tools zoals Oracle Forms, …
- Soms handig datamanipulatie uit te voeren in SQL als het gaat over globale updates

6.1 Gegevens invoeren

```
INSERT INTO tabelnaam [(kolomnaam,…)]
VALUES (expressie,…)

INSERT INTO tabelnaam [(kolomnaam,…)]
SELECT [kolomnaam,…]
FROM tabelnaam
```

Gegevens op 2 manieren toevoegen

- Via VALUES-component → verzameling kolomwaarden opgeven
- Via SUBQUERY → gebruik maken van bestaande gegevens

6.1 Gegevens invoeren

Voorbeelden

- *Fysieke kolomvolgorde*

  INSERT INTO medewerkers
  VALUES ('NIJS', 7955 , 'PETER', 'TRAINER', 7566, date '1967-02-16', 2660, NULL, 20);

- *Willekeurige volgorde (onvolledig)*

  INSERT INTO medewerkers (naam,voorn,mnr,maandsal,gbdatum)
  VALUES ('VOS', 'GEERT', 7956, 2660, date '1963-05-22');

6.1 Gegevens invoeren

- *Gegevens toevoegen met VALUES*

INSERT INTO medewerkers

VALUES(7369,'CASPERS','JANA' ,'TRAINER', 7902 , date '1965-12-17' , 1600 , NULL, 20);

- *Gegevens toevoegen dmv een substitutievariabele*

  INSERT INTO personeel VALUES (&mnr,'&naam','&voorn',&functie,&chef,
  ,to_date('&gebdatum','dd-mm-yyyy'),&maansal,&comm,&afd);

```
Enter value for mnr: 7666
Enter value for naam: NIJS
Enter value for voorn: P
Enter value for functie: null
Enter value for chef: 7696
Enter value for gbdatum: 24-09-1997
Enter value for maansal: 2960
Enter value for comm: 200
Enter value for afd: 30
old   2: values ( &mnr,'&naam','&voorn',&functie,&chef,to_date('&gebdatum','dd-mm-yyyy'),&maansal,&comm, &afd)
new   2: values ( 7666, 'NIJS', 'P', null, 7696, to_date('24-09-1997','dd-mm-yyyy'), 2960, 200, 30)
```

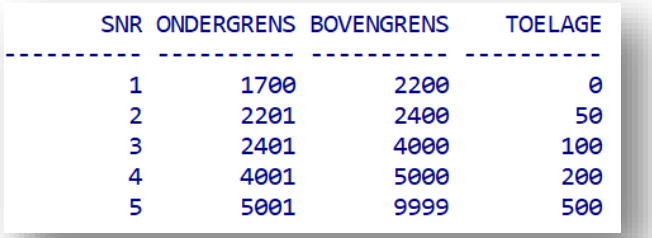
Merk op: alfanumerieke waarden en datums geplaatst tussen ' '

6.1 Gegevens invoeren

- Op basis van een query worden 4 rijen toegevoegd

```
INSERT INTO schalen
SELECT snr+5, ondergrens + 2300, bovengrens + 2300, 500
FROM schalen
WHERE snr <= 4;
```

SNR	ONDERGRENS	BOVENGRENS	TOELAGE
1	1700	2200	0
2	2201	2400	50
3	2401	4000	100
4	4001	5000	200
5	5001	9999	500
6	4000	4500	500
7	4501	4700	500
8	4701	6300	500
9	6301	7300	500



SNR	ONDERGRENS	BOVENGRENS	TOELAGE
1	1700	2200	0
2	2201	2400	50
3	2401	4000	100
4	4001	5000	200
5	5001	9999	500

Rijen 6 tem 9 worden toegevoegd

6.2 Gegevens wijzigen

- Meerdere kolommen van rij 7666 in de tabel wijzigen
  - UPDATE medewerkers
    SET functie = 'TRAINER'
    , maandsal = 5030
    , afd = 60
    WHERE mnr = 7666

- Alle medewerkers krijgen een loonsverhoging van 10%
  - UPDATE medewerkers
    SET maandsal = maandsal *1.1;

6.2 Gegevens wijzigen

- De werknemers van het hoofdkantoor krijgen een loonsverhoging van 10%.

```
UPDATE medewerkers
SET maandsal = maandsal *1.1
WHERE afd = ( SELECT anr
              FROM afdelingen
              WHERE naam = 'HOOFDKANTOOR');
```

6.2 Gegevens wijzigen

UPDATE tabelnaam

SET kolomnaam = waarde, ….

WHERE voorwaarde

Opgelet: Indien de WHERE-component wordt weggelaten → wijzigingen worden op alle rijen uitgevoerd

```
update afdelingen set naam = naam || 'X';

select naam, substr(naam, 0, length(naam)-1) from afdelingen;
update afdelingen set naam = substr(naam, 0, length(naam)-1);
```

6.3 Gegevens verwijderen

- DELETE FROM medewerkers

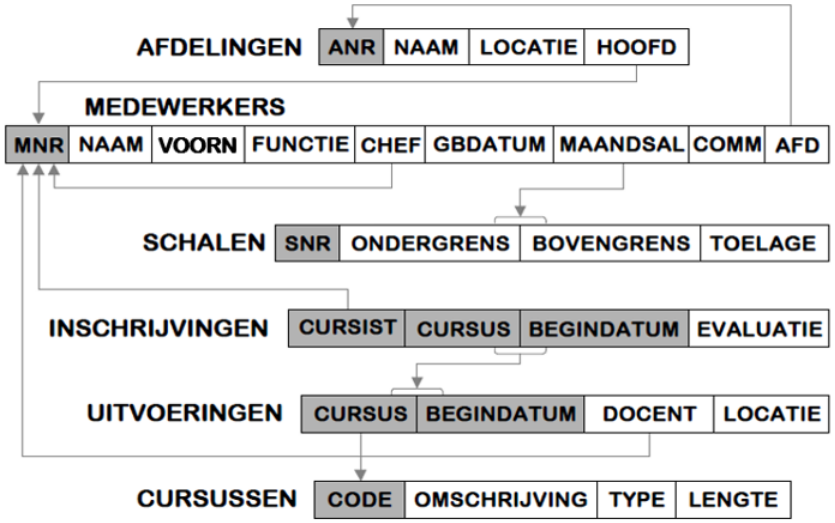
  WHERE mnr = 7499;

- Opgelet:
  - Indien het WHERE-commando achterwege gelaten wordt, is het resultaat een LEGE tabel.
  - DROP TABLE <-> DELETE
    - DROP: DDL-commando → dit verwijdert niet alleen inhoud, maar ook de gegevens en de daarmee samenhangende structuren (indexen, privileges, vreemde sleutels,…)
    - DELETE: DML-commando → dit wijzigt de structuur niet, maar verwijdert enkel de inhoud van de tabel. Beter TRUNCATE gebruiken (vlugger)

6.3 Gegevens verwijderen

- TIP
  - Constrains niet vergeten bij toevoegen, verwijderen of wijzigen. Er kunnen geen gegevens toegevoegd worden die niet voldoen aan voorwaarden. Gegevens kunnen ook niet verwijderd worden die verwijzen naar een andere tabel.

    ALTER TABLE med
    DISABLE constraint med_afd_FK



6.4 Transactieverwerking

- Wijzigingen in de inhoud krijgen in eerste instantie een voorlopig karakter → wijzigingen worden definitief gemaakt na het COMMIT-commando

COMMIT

ROLLBACK [(TO SAVEPOINT savepointnaam)]

SAVEPOINT naam

- ROLLBACK-commando: wijzigingen ongedaan maken
- SAVEPOINT: markeringspunten plaatsen tussen verschillende mutaties

6.4 Transactieverwerking

- Voorbeeld:

DML-commando 1	
Commit	*mutatie 1 wordt definitief*
DML-commando 2	
DML-commando 3	
Savepoint EEN	
DML-commando 4	
DML-commando 5	
Savepoint TWEE	
DML-commando 6	
DML-commando 7	
Rollback to savepoint TWEE	*mutaties 6 en 7 worden ongedaan gemaakt*
Commit	*mutaties 2, 3, 4 en 5 worden definitief*

6.4 Transactieverwerking

- Opmerkingen:
  - Mutaties vergeten te 'committen' → risico bij systeemstoring om werk te verliezen. Mutaties zijn nog niet zichtbaar voor andere gebruikers.
  - Oracle hanteert een impliciete commit bij het beëindigen van SQL*Plus.
  - Alle DDL-Commando's en DCL-commando's hanteren een impliciete commit → transacties onmiddellijk definitief.
  - Oracle kent ook een autocommit om DML-commando's onmiddellijk te laten 'committen' (SET AUTOCOMMIT ON). → geen rollback meer mogelijk!