Teema8.Data Manipulation Language - andmeuuendus SQL-keeles - andmete sisestamine, muutmine , kustutamine. (INSERT,UPDATE,DELETE)

- Andmete lisamine tabelitesse.
  Süntaks:

```
INSERT INTO {table | view}
    [ (column [ , column ] … ) ]
    {VALUES (expr [ , expr] … ) subquery}
```

Selle lausega on võimalik lisada ridu kas otse tabelitesse (table) või vaadete aluseks olevatesse tabelitesse (view)

- veergude arv ja avaldiste(väärtuste) arv VALUES-klauslis või alampäringus peab olema võrdne
- veerunimede loetelu võib olla

a)täielik tabeli veeriunimede loetelu - siis saavad kõik veerud ettenähtud väärtuse

b)osaline - siis saavad puuduvad veerud tabeli kirjelduses DEFAULT-klausliga määratud väärtuse või kui neid ei ole, siis NULL-väärtuse (st. tühja väärtuse)

c)täiesti tühi - siis peab VALUES-klausel andma väärtuse kõigile antud tabeli veergudele selle tabeli kirjekdusele vastavas järjekorras.

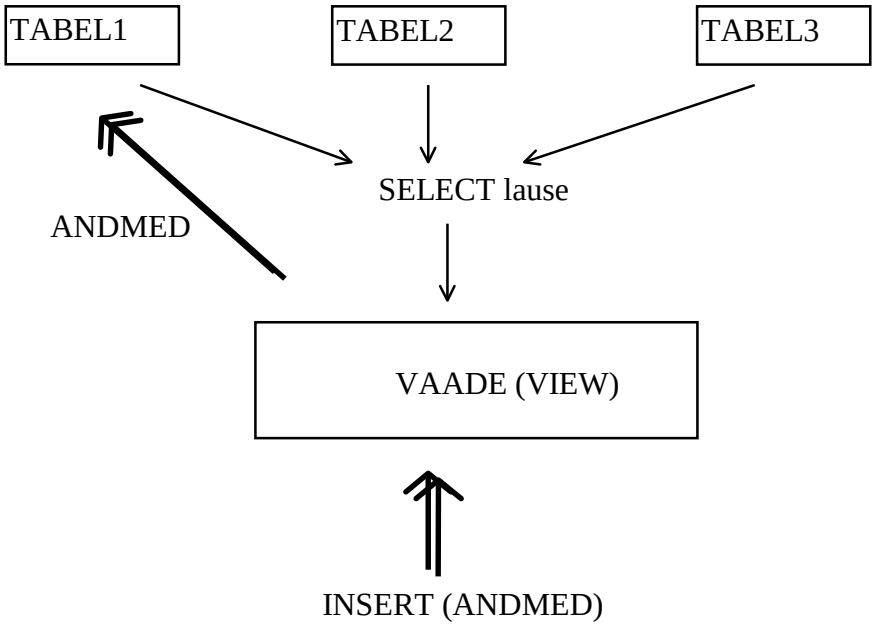
```
INSERT INTO tudeng (tudkood,pnimi,enimi,suund,kursus)
   VALUES ('123456','Tamm','Raivo',NULL,2);
```

- vajalike vääruste saamiseks INSERT-lauses võib kasutada alampäringut (SELECT-lauset) n. kõik tudengid valisid aine 'Filosoofia', konstrueerida INSERT-lause, mis uuendaks vastavalt andmeobjekti 'oppim' (õppimine)


```
INSERT INTO  oppim (tudkood,akood,regkuup)
        SELECT tudkood,akood,'28.02.1996' FROM tudeng, aine
        WHERE tudeng.kursus = 3 AND aine.nimetus = 'Filosoofia';
```

NB! Siin on alampäringus näide ristkorrutisest, kuna *joini* ei ole (*joini*-i tunnus: table1.column = table2.column2, nn. WHERE-klausel päringus). Ristkorrutise korral **korrutatakse** ühe tableli read teisest tabelist saadud ridadega.
Tegemist ei ole siiski veaga , kuna 3.kursuse tudengi read kombineeritakse **ühe** aine reaga (Tingimusel kui aine tabelis on ainult üks 'Filosoofia'). Ja veel - alampäringu korral peab veergude loetelu INSERT-lauses ja SELECT-lauses klappima, nii veergude järjekord, arv kui andmetüüp.

- INSERT lausega saab andmeid lisada ka vaatesse (VIEW).Vaatesse sisestatud andmed lähevad tegelikult vaate aluseks olnud tabelitesse.



Kui vaate loomisel on kasutatud WITH CHECK OPTION-i , siis saab lisada ainult selliseid ridu, mis rahuldavad vaadet loonud alampäringu tingimusi:

```
CREATE VIEW repsid
  AS SELECT pnimi,enimi,suund,kursus
  FROM tudeng WHERE kursus = 1
  WITH CHECK OPTION;
```

INSERT INTO repsid
VALUES (‘Maasikas’,’Mart’,LD,3);

- see ülaltoodud lause ei lähe läbi, sest läbi antud vaate lubatakse lisada vaid 1. Kursuse tudengeid. Kui 3 asendataks 1-ga, siis lisatakse tudengi tabelisse uus rida (mitte vaatesse, mida füüsiliselt üldse ei eksisteerigi). Vaates esindamata veerud saavad DEFAULT-väärtuse või NULL-i.

NB! Kui vaates ei oleks WITH CHECK OPTION-i , siis saaks lisada ka 3. Kursuse tudengit, kui hiljem ei oleks seda enam selles vaates näha (näiteks **SELECT * FROM** repsid - lisatud ridasid ei näita)

- Olemasolevate andmete muutmine tabelis - UPDATE-lause.

UPDATE {table|view}
SET column = expr [, column = expr]...
[**WHERE** condition]

Kui WHERE-klauslit pole, siis uuendatakse kôiki antud tabeli ridu.
Näiteks - uue ôppeaasta alguses said kôik tudengid ühe kursuse vôrra edasi:
UPDATE tudeng SET kursus = kursus+1

Reformi tulemusena muutus TI-suund TY-suunaks:

UPDATE tudeng **SET** suund = 'TY' **WHERE** suund =' TI';

Kati Karu abiellus ja tema uus nimi on Rebane:

UPDATE tudeng **SET** pnimi='Rebane'
WHERE enimi='Kati' **AND** pnimi='Karu'

NB! condition vôib sisaldada ka alampäringut. Näiteks: Jaan Juurikas abiellus oma kursuseôe Mallega - see oli ainus Malle tema kursusel:

UPDATE tudeng **SET** pnimi 'Juurikas'
WHERE enimi='Kati' **AND** kursus =
(**SELECT** kursus **FROM** tudeng **WHERE** pnimi='Juurikas' **AND**
enimi = 'Jaan')

Vaate uuendamine on analoogne lisamisega - kui on WITH CHECK OPTION, siis ei saa rida uuendada selliseks, et ta enam pärast vaatesse ei satuks.

- Ridade kustutamine tabelist - DELETE -lause.

DELETE FROM {table|view}
  [**WHERE** condition]

Kui WHERE-klauslit ei ole, siis tehakse tabel täiesti tühjaks. Kui tegemist on vaatega, siis eemaldatakse ainult need read, mis sisalduvad ka vaates.

Näit. - kustutada uue õppeaasta alguses kõik eelmise aasta diplomandid (kursus=5)

DELETE FROM tudeng **WHERE** kursus=5;

Ridade lisamine, parandamine ja kustutamine ei tohi olla vastuolus tabelitele defineeritud piirangutega, vastasel juhul antud operatsiooni ei toimu.

Näit. kui tabelitele teha sellised piirangud:
ALTER TABLE oppim **ADD** (**CONSTRAINT** tudvoti
  FOREIGN KEY (tudkood) **REFERENCES** tudeng (tkood))
ALTER TABLE tudeng
  ADD (CONSTRAINT mn **CHECK** (sugu **IN** ('M','N')),
    CONSTRAINT tkood **PRIMARY KEY** (tkood))

siis alljärgnevad tegevused lõpevad veateatega:

UPDATE oppim **SET** tudkood = '112233'
 WHERE tudkood = '123456';

- Kui tudengit koodiga 112233 pole olemas

INSERT INTO tudeng (pnimi,enimi,sugu) **VALUES** ('Kask',NULL,'X');

- sellepärast, et primaarvõti (tkood) ei saa olla tühi
- sugu ei saa olla 'X'

DELETE tudeng **WHERE** suund = 'TI';

- juhul, kui mõni TI tudeng on ennast mõne aine jaoks registreerinud, siis ei saa seda lauset täita, sest siis tekiksid oppim-tabelisse olematud välisvõtmed.