

Tietokantojen perusteet

SQL – Tietokantataulu

Taulun luominen, poistaminen
Rivien lisäys, muuttaminen, poistaminen
Yksinkertaisia kyselyjä

Kati Iltanen, Harri Keto
Tietotekniikan yksikkö
Informaatioteknologian ja viestinnän tiedekunta
Tampereen yliopisto

Kalvot perustuvat kirjaan Elmasri, Navathe: Fundamentals of Database Systems.

Tietokanta

- Tietokanta (database) on kokoelma toisiinsa liittyviä tietoja.
 - Tiedot kuvaavat jotakin reaalimaailman osa-aluetta, kohdealuetta (sovellusaluetta).
 - Tiedot on talletettu jotakin käyttötarkoitusta varten.
- Tietokantaa käytetään tietokannanhallintajärjestelmän (TKHJ; database management system, DBMS) avulla.
 - Yleiskäyttöinen ohjelmisto tietokannan luomiseen ja ylläpitämiseen
 - Tarjoaa tehokkaan ja hallitun ympäristön tietojen säilyttämiseen ja hyödyntämiseen samanaikaisille käyttäjille

Tietokanta ja TKHJ

- Tietokannanhallintajärjestelmän avulla voidaan

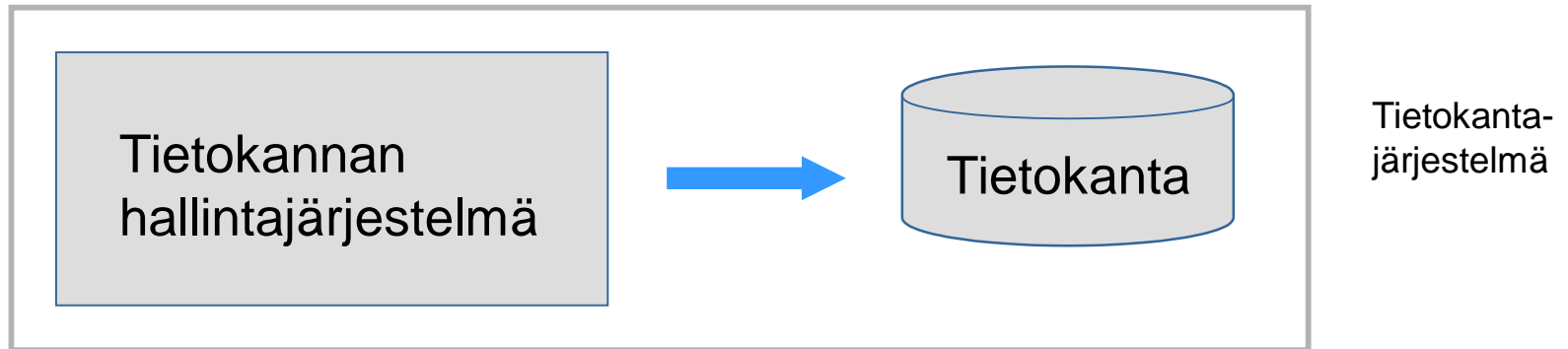
- luoda ja hallita tietokantoja

- lisätä tietoja
- muuttaa tietoja
- poistaa tietoja

- hakea tietoja

Tehdään päivityksiä

Tehdään kyselyjä



Tällä kurssilla esimerkijärjestelminä ovat SQLite ja PostgreSQL.

SQL - tietokantataulu: 3

Tietokantaa voidaan käyttää **sovellusohjelman** kautta



Tietokannan
hallintajärjestelmä

Tietokanta

SQL - tietokantataulu: 4

Tietokantaa voidaan
käyttää **komentotulkin**
(komentorivikäyttö-
liittymän) kautta

Welcome to psql 8.1.2 (server 8.3.0), the PostgreSQL interactive terminal.

Type: \copyright for distribution terms

\h for help with SQL commands

\? for help with psql commands

\g or terminate with semicolon to execute query

\q to quit

tkp=> SELECT *

tkp-> FROM opiskelija

tkp-> WHERE etunimi = 'Onni' and sukunimi = 'Opiskelija';



Tietokannan
hallintajärjestelmä



SQL - tietokantataulu: 5

SQL-tietokanta

- Tällä kurssilla käsitellään SQL-tietokantoja.
- Termi SQL-tietokanta tulee näiden tietokantojen käsittelyyn käytettävästä kielestä, SQL:stä (Structured Query Language).
- SQL on nykypäivän tietokantojen käsittelyssä eniten käytetty kieli.
- SQL-tietokantoja kutsutaan myös relaatorakenteisiksi tietokannoiksi tai relaatiotietokannoiksi.
 - Niiden taustalla oleva **tietomalli** on relaatiomalli.
 - Tietomalli on käsitteistö, jolla kuvataan tietokannan rakenne: tietojen tyyppi, tietojen väliset suhteet ja tietoja koskevat rajoitteet
 - Lisäksi tietomallit voivat sisältää operaatioita tietojen hakuun ja päivitykseen.
 - (Relaatiomallia käsitellään Tietokantajärjestelmät: SQL -opintojaksolla.)

SQL

- SQL:n avulla voidaan määritellä ja hallita tietokantoja sekä käsitellä niiden tietoja.
 - Data control language (DCL)
 - Data definition language (DDL)
 - Data manipulation language (DML)

SQL

- Standardoitu kieli
 - Viimeisimmät muutokset 2019
- Eri tietokannanhallintajärjestelmillä on suppeahko yhteinen standardin mukainen ydin ja lisäksi erilaisia laajennoksia.
- Tällä kurssilla pyritään keskittymään SQL-kielen ydinosaan.
 - Lisäksi käydään läpi ominaisuuksia, jotka ovat hyödyllisiä, mutteivät välttämättä löydy kaikista tietokannanhallintajärjestelmistä.

SQL-taulu

- SQL-tietokannoissa tiedot järjestetään tauluihin.

Taulun nimi

tyontekija

Sarakkeen nimi

ttnro	etunimi	sukunimi	saika	kotikunta	palkka	puhelin
88	Jukka	Susi	1957-11-10	Tampere	5500.00	444 1234
33	Ville	Viima	1975-12-08	Nokia	4000.50	444 4343
12	Pekka	Puro	1985-01-09	Tampere	3000.00	
98	Jenni	Joki	1961-06-20	Lempäälä	4300.00	444 4488
99	Alli	Kivi	1988-07-19	Nokia	2500.00	444 5555

Rivi

Sarake

Arvo

Puuttuva arvo

SQL-taulu

- Rivi
 - vastaa yhtä kohdetta (objektia l. entiteettiä), jonka tietoja halutaan tauluun tallentaa
- Sarake
 - sisältää kohteeseen liittyvää tietoa
 - tieto on tietyntyypistä
- Taulun rivien ja sarakkeiden järjestyksellä ei ole merkitystä taulun tietosisältöön
- Taulujen ja sarakkeiden nimet pyritään valitsemaan siten, että ne helpottavat riveillä olevien arvojen tulkintaa.
- Tietokantajärjestelmä sisältää
 - varsinaisia tietoja eli **dataa** (ilmentymä)
 - tietojen kuvauksen eli **metadataa** (kaavio)

Taulun luominen

- Taulu luodaan **CREATE TABLE** -lauseella, joka määrittelee
 - taulun nimen
 - taulun sarakkeiden nimet
 - Sarakkeiden nimien täytyy olla samassa taulussa toisistaan eriäviä.
 - minkä tyyppistä tietoa taulun sarakkeisiin voidaan tallettaa
 - (yleensä) pääavaimen
 - mahdollisesti avaimia ja muita rajoitteita

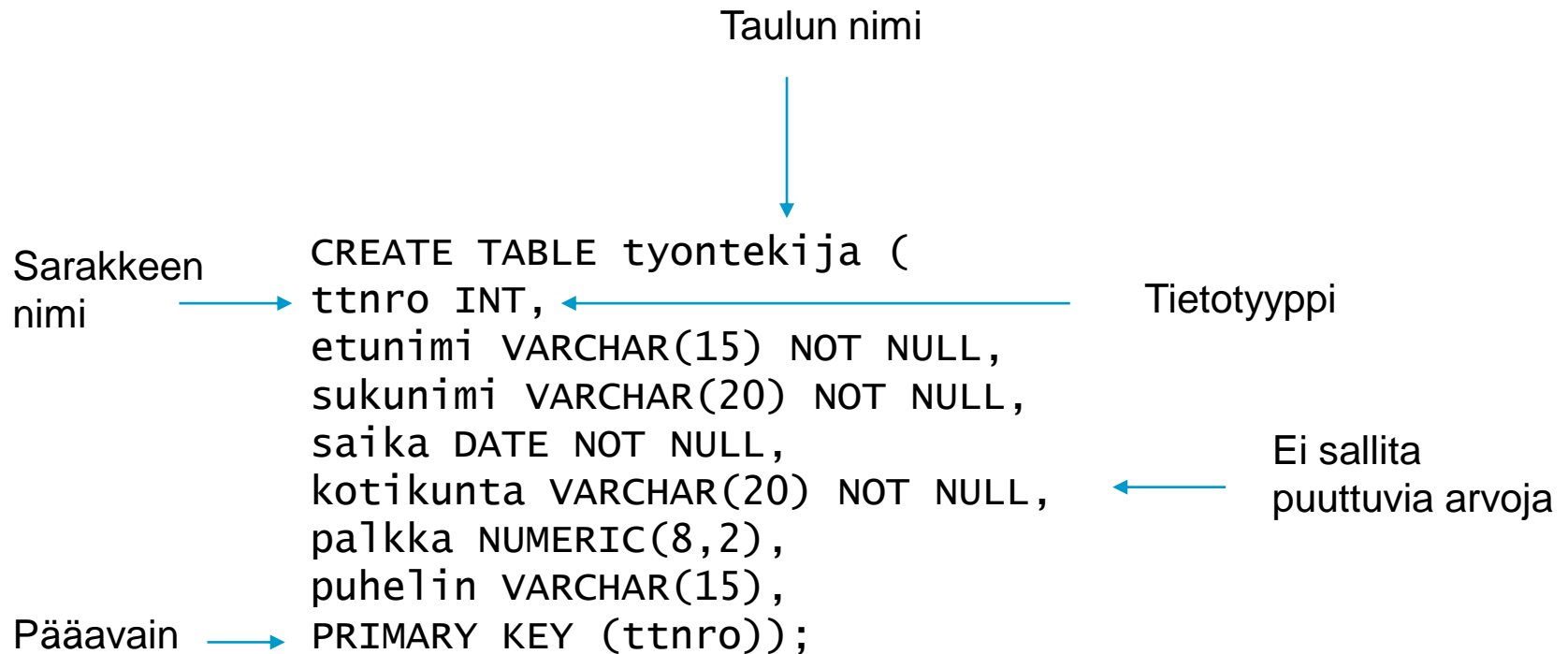
```
CREATE TABLE taulu (  
  sarake tietotyyppi [rajoite]  
  {, sarake tietotyyppi [rajoite]}  
  [rajoite {,rajoite}]);
```

[] valinnainen osa

{ } osa toistuu 0...n kertaa

Merkit [,], {, } jätetään varsinaisesta luontilauseesta pois.

Taulun luominen



Taulun luominen

SQL ei erottele isoja ja pieniä kirjaimia toisistaan SQL:n avainsanoissa (key words, esim. CREATE ja INT) eikä taulujen ja sarakkeiden nimissä.

Kirjoitustyyli vaihtelee lähteestä riippuen. Näissä kalvoissa avainsanat on kirjoitettu isolla.

```
CREATE TABLE tyontekija (  
  ttnro INT,  
  etunimi VARCHAR(15) NOT NULL,  
  sukunimi VARCHAR(20) NOT NULL,  
  saika DATE NOT NULL,  
  kotikunta VARCHAR(20) NOT NULL,  
  palkka NUMERIC(8,2),  
  puhelin VARCHAR(15),  
  PRIMARY KEY (ttnro));
```

Komento voidaan jakaa usealle riville.

Puolipiste päättää komennon.

Useimmissa järjestelmissä taulujen ja sarakkeiden nimissä voi käyttää kirjaimia a-z. ('å', 'Å', 'ä', 'Ä', 'ö' ja 'Ö' eivät siis käy.)

Lisäksi voidaan käyttää numeroita (0-9) ja alaviivaa (_).

Nimen on alettava kirjaimella tai alaviivalla.

Taulun luominen: Tietotyyppejä

TIETOTYYPPI	KUVAUS	ESIMERKKI
INT INTEGER	kokonaisluku	ttnro INT
NUMERIC(L,S) DECIMAL(L,S)	desimaaliluku, jossa on yhteensä L numeroa, joista desimaaliosassa on S numeroa (numeric ja decimal ovat PostgreSQL:ssä ekvivalentteja.)	palkka NUMERIC(8, 2) palkka DECIMAL(8, 2)
CHAR(N) CHARACTER(N)	vakiomittainen merkkijono, pituus N merkkiä (N:ää merkkiä lyhyempien merkkijonojen loppu täytetään tyhjämerkeillä, blankoilla l. välilyönneillä)	spuoli CHAR spuoli CHARACTER(1) henkilotunnus CHARACTER(11)
VARCHAR(N) CHARACTER VARYING(N)	vaihtuvamittainen, korkeintaan N merkkiä pitkä merkkijono	etunimi VARCHAR(15)
DATE	päivämäärä	saika DATE

- Näitä ja muita tietotyyppejä käsitellään lisää myöhemmin.

Taulun luominen: Rajoitteita

- **Primary key – pääavain (ensisijainen avain)**
 - yksilöi taulun rivit
 - voi muodostua
 - yhdestä sarakkeesta
 - sarakkeessa oltava eri arvo taulun jokaisella rivillä
 - useasta sarakkeesta
 - sarakkeissa olevien arvojen yhdistelmän oltava eri taulun jokaisella rivillä
 - puuttuvia arvoja ei sallita
 - ilmaistaan PRIMARY KEY -määreellä
PRIMARY KEY (sarake {,sarake})
 - taulussa voi olla yksi pääavain
 - tiedot löytyvät nopeasti pääavainta käyttämällä
 - tietokannanhallintajärjestelmä luo pääavainta varten erityisen hakemistorakenteen, jonka avulla tiedot löytyvät nopeasti pääavainta käyttämällä

Taulun luominen: Rajoitteita

- **Unique - avain**

- voi muodostua
 - yhdestä sarakkeesta
 - sarakkeessa oltava eri arvo taulun jokaisella rivillä
 - useasta sarakkeesta
 - sarakkeissa olevien arvojen yhdistelmän oltava eri taulun jokaisella rivillä
- PostgreSQL:ssä puuttuvat arvot sallitaan (vaihtelee järjestelmittäin)
- ilmaistaan UNIQUE -määreellä
UNIQUE (sarake {,sarake})
- taulussa voi olla monta avainta
- PostgreSQL:ssä tietokannanhallintajärjestelmä luo avainta varten erityisen hakemistorakenteen, jonka avulla tiedot löytyvät nopeasti avainta käyttämällä.

Taulun luominen: Rajoitteita

```
CREATE TABLE osasto (  
  onro INT,  
  onimi VARCHAR(15) NOT NULL,  
  PRIMARY KEY (onro),  
  UNIQUE (onimi));
```

← Pääavain
← Avain

onro	onimi
1	Pääkonttori
4	Hallinto
5	Tutkimus

PRIMARY KEY -määre sisältää NOT NULL -määreen (ei sallita puuttuvia arvoja)

PostgreSQL:ssä UNIQUE-määre ei sisällä NOT NULL -määrettä, vaan se on annettava erikseen, jos puuttuvat arvot halutaan kieltää.

Taulun luominen: Rajoitteita

- **NULL**
 - Puuttuva tieto merkitään NULL-arvolla.
 - Puhutaan NULL-arvoista tai tyhjääarvoista.
- Tieto voi puuttua sarakkeesta, koska
 - kohteella ei ole arvoa ko. sarakkeelle tai
 - kohteen arvo on tuntematon (arvoa ei tiedetä)
- Kaikki puuttuva tieto merkitään NULL-arvolla.
- **NOT NULL – ei puuttuvia arvoja**
 - Sarakkeella on aina oltava ei-tyhjä arvo
 - Kielletään tyhjääarvot taulun luontilauseessa

sarake TIETOTYYPPI NOT NULL

Taulun luominen

- CREATE TABLE -lauseen suorituksen jälkeen tietokantajärjestelmässä
 - on tyontekija-taulun määrittely (metadataa)
 - määrittely säilyy järjestelmässä niin kauan kunnes taulu käydään poistamassa DROP TABLE -lauseella
 - mutta varsinaiset tiedot (data) puuttuvat
- Taulun rakenteen tiedot näyttävät komennot vaihtelevat SQL-komentotulkeittain (komentorivikäyttöliittymittäin)
 - SQLitessa on komento .schema taulu
 - .schema tyontekija
 - PostgreSQL:ssä on komento \d taulu
 - \d tyontekija

Taulun poistaminen

- Taulun voi poistaa **DROP TABLE** -lauseella.
 - poistaa taulun määrittelyn ja varsinaisen tietosisällön

```
DROP TABLE taulu;
```

```
DROP TABLE tyontekija;
```

Taulun rakenteen muuttaminen

- Taulun rakennetta voi muuttaa **ALTER TABLE** -lauseella.
 - lisätä tai poistaa sarakkeita
 - vaihtaa sarakkeiden nimiä ja tietotyypppejä
 - muuttaa taulun rajoitteita
- Mahdollisuudet muuttaa taulun rakennetta vaihtelevat tietokannanhallintajärjestelmittäin.
- ALTER TABLE -lausetta käydään läpi myöhemmin kurssilla.

Rivien lisääminen tauluun

- Rivejä (varsinaisia tietoja, dataa) lisätään tauluun **INSERT**-lauseella.
- INSERT-lause lisää tauluun yhden rivin.

```
INSERT INTO taulu [(sarake {, sarake} )]  
VALUES (vakioarvo {,vakioarvo});
```

```
INSERT INTO tyontekija  
VALUES (88, 'Jukka', 'Susi', '1957-11-10', 'Tampere', 5500.00, '444 1234');
```

- Merkkijonovakiot ja päivämäärät esitetään yksinkertaisissa lainausmerkkeissä (l. "hipsuissa")
- Numeeriset vakiot esitetään ilman lainausmerkkejä.

Varsinaisessa datassa pienet ja isot kirjaimet erotellaan toisistaan.

'Susi' ei ole siis sama asia kuin 'SUSI' tai 'SuSi'

Rivien lisääminen

- Luettelemalla sarakkeet voidaan uudelleenkäytettävissä lisäyslauseissa varautua taulun rakenteen muutoksiin.
 - Sarakkeiden järjestyksen vaihtumiseen
 - Uusien sarakkeiden lisäämiseen
- Lisäyslause on tässä muodossa myös informatiivisempi: se kertoo sarakkeet, joihin tietoja lisätään.

```
INSERT INTO  
tyontekija(ttnro, etunimi, sukunimi, saika, kotikunta, palkka, puhelin)  
VALUES (88, 'Jukka', 'Susi', '1957-11-10', 'Tampere', 5500.00, '444 1234');
```

Rivien lisääminen

- Rivi, jolla on puuttuva arvo, lisätään tauluun
 - ilmaisemalla puuttuva arvo NULL-merkinnällä tai
 - luettelemalla ne sarakkeet, joiden arvo annetaan

```
INSERT INTO tyontekija  
VALUES (12, 'Pekka', 'Puro', '1985-01-09', 'Tampere', 3000.00, NULL);
```

```
INSERT INTO  
tyontekija(ttnro, etunimi, sukunimi, saika, kotikunta, palkka)  
VALUES (12, 'Pekka', 'Puro', '1985-01-09', 'Tampere', 3000.00);
```

- Tauluun lisätyt rivit säilyvät taulussa siihen asti, kunnes ne käydään poistamassa DELETE-lauseella tai koko taulu poistetaan DROP TABLE -lauseella.

Rivien lisääminen

- Tauluun lisättävien tietojen on noudatettava taulun määrittelyn yhteydessä annettuja vaatimuksia.
 - SQL-komentotulkki ja tietokannanhallintajärjestelmä tarkastavat lisättävät tiedot.
 - Jos lisättävät tiedot eivät vastaa taulun luontilauseetta, niitä ei lisätä tietokantaan, vaan komentotulkki antaa virheilmoituksen.

Kyselyt

- Tietoja haetaan tauluista **SELECT**-lauseella.

```
SELECT sarake {, sarake}  
FROM taulu {,taulu}  
[WHERE ehto];
```

mistä sarakkeista
mistä tauluista
miltä riveiltä

- Kyselyn tuloksena saadaan nimetön tulostaulu.
 - Jos kyselyssä ei ole WHERE-osaa, on tulostaulussa yksi rivi kutakin FROM-osan taulun (tai taulujen karteesisen tulon) riviä kohden.
 - Karteesinen tulo käsitellään myöhemmin.
- SQL on deklarativinen kyselykieli: kyselyssä määritellään, mitä haetaan (eikä sitä miten haetaan).

Kyselyt: SELECT-osa

- SELECT-osassa luetellaan sarakkeet, jotka halutaan tulostauluun mukaan.

ttnro	etunimi	sukunimi	saika	kotikunta	palkka	puhelin
88	Jukka	Susi	1957-11-10	Tampere	5500.00	444 1234
33	Ville	Viima	1975-12-08	Nokia	4000.50	444 4343
12	Pekka	Puro	1985-01-09	Tampere	3000.00	
98	Jenni	Joki	1961-06-20	Lempäälä	4300.00	444 4488
99	Alli	Kivi	1988-07-19	Nokia	2500.00	444 5555

```
SELECT etunimi, sukunimi, kotikunta  
FROM tyontekija;
```

```
etunimi | sukunimi | kotikunta  
-----+-----+-----  
Jukka   | Susi     | Tampere  
Ville   | Viima    | Nokia  
Pekka   | Puro     | Tampere  
Jenni   | Joki     | Lempäälä  
Alli    | Kivi     | Nokia
```

tulostaulu

Kyselyt: SELECT-osa

- * tarkoittaa ”kaikki sarakkeet”

```
SELECT ttnro, etunimi, sukunimi, saika,  
       kotikunta, palkka, puhelin  
FROM tyontekija;
```

```
SELECT *  
FROM tyontekija;
```

ttnro	etunimi	sukunimi	saika	kotikunta	palkka	puhelin
88	Jukka	Susi	1957-11-10	Tampere	5500.00	444 1234
33	Ville	Viima	1975-12-08	Nokia	4000.50	444 4343
12	Pekka	Puro	1985-01-09	Tampere	3000.00	
98	Jenni	Joki	1961-06-20	Lempäälä	4300.00	444 4488
99	Alli	Kivi	1988-07-19	Nokia	2500.00	444 5555

Kyselyt: SELECT-osa

- SQL sallii duplikaattien (ts. samojen rivien) esiintymisen sekä tauluissa että tulostauluissa.
 - Oletusarvoisesti duplikaatit säilytetään tulostauluissa.
 - Voidaan käyttää myös ALL-määrettä.
- Duplikaatit voidaan poistaa **DISTINCT**-määreellä.

```
SELECT kotikunta  
FROM tyontekija;
```

```
SELECT ALL kotikunta  
FROM tyontekija;
```

```
kotikunta  
-----  
Tampere  
Nokia  
Tampere  
Lempäälä  
Nokia
```

```
SELECT DISTINCT kotikunta  
FROM tyontekija;
```

```
kotikunta  
-----  
Lempäälä  
Nokia  
Tampere
```

Kyselyt: WHERE-osa

- Tulokseen tulevia rivejä rajataan WHERE-osassa annettavalla totuusarvoisella ehdolla.
 - Ehdossa tehdään yleensä vertailu(ja).
 - Tulokseen otetaan mukaan rivit, joille ehdon arvo on tosi (true).
 - SQL:ssä on kolme totuusarvoa
 - true - tosi
 - false - epätosi
 - unknown - tuntematon (tyhjäarvojen vuoksi)

Kyselyt: WHERE-osa

Haetaan työntekijät, joiden kotikunta on Tampere.

ttnro	etunimi	sukunimi	saika	kotikunta	palkka	puhelin
88	Jukka	Susi	1957-11-10	Tampere	5500.00	444 1234
33	Ville	Viima	1975-12-08	Nokia	4000.50	444 4343
12	Pekka	Puro	1985-01-09	Tampere	3000.00	
98	Jenni	Joki	1961-06-20	Lempäälä	4300.00	444 4488
99	Alli	Kivi	1988-07-19	Nokia	2500.00	444 5555

```
SELECT etunimi, sukunimi  
FROM tyontekija  
WHERE kotikunta = 'Tampere';
```

verrataan sarakkeen arvoa
merkkijonovakioon

```
etunimi | sukunimi  
-----+-----  
Jukka   | Susi  
Pekka   | Puro
```

Kyselyt: WHERE-osa

```
SELECT etunimi, sukunimi  
FROM tyontekija  
WHERE etunimi = kotikunta;
```

Verrataan sarakkeen arvoa
toisen sarakkeen arvoon.

- Ehdoissa käytettäviä vertailuoperaattoreita

=	yhtä suuri kuin
<>	eri suuri kuin
<	pienempi kuin
<=	pienempi tai yhtä suuri kuin
>	suurempi kuin
>=	suurempi tai yhtä suuri kuin

- Muita operaattoreita esitellään myöhemmin.

Kyselyt: WHERE-osa

- Operaattorit <, <=, >, >= sopivat tietotyypeille, joille järjestyksen määrittäminen on järkevää

```
SELECT sukunimi, etunimi, saika  
FROM tyontekija  
WHERE saika < '1985-01-01';
```

Haetaan ennen vuotta 1985
syntyneet työntekijät

sukunimi	etunimi	saika
Susi	Jukka	1957-11-10
Viima	Ville	1975-12-08
Joki	Jenni	1961-06-20

WHERE-osa: tyhjäarvot

- Tyhjäarvot ovat keskenään eri suuria.
- Tyhjäarvon (NULL-arvon) vertailu antaa AINA arvoksi **tuntematon**.
 - Vertailuoperaattorit = ja <> EIVÄT siis sovi tyhjäarvon olemassaolon testaamiseen!!
- Tyhjäarvon olemassaoloa testataan operaattoreilla IS NULL ja IS NOT NULL
 - sarake IS NULL
 - tosi silloin, kun sarakkeessa on tyhjäarvo
 - sarake IS NOT NULL
 - tosi silloin, kun sarakkeessa on ei-tyhjä arvo

WHERE-osa: tyhjäarvot

ttnro	etunimi	sukunimi	saika	kotikunta	palkka	puhelin
88	Jukka	Susi	1957-11-10	Tampere	5500.00	444 1234
33	Ville	Viima	1975-12-08	Nokia	4000.50	444 4343
12	Pekka	Puro	1985-01-09	Tampere	3000.00	
98	Jenni	Joki	1961-06-20	Lempäälä	4300.00	444 4488
99	Alli	Kivi	1988-07-19	Nokia	2500.00	444 5555

Kuinka tyhjäarvot käyttäytyvät?

a)
`SELECT ttnro`
`FROM tyontekija`
`WHERE puhelin = '444 1234';`

```
ttnro
-----
    88
(1 row)
```

b)
`SELECT ttnro`
`FROM tyontekija`
`WHERE puhelin <> '444 1234';`

```
ttnro
-----
    33
    98
    99
(3 rows)
```

WHERE-osa: tyhjäarvot

ttnro	etunimi	sukunimi	saika	kotikunta	palkka	puhelin
88	Jukka	Susi	1957-11-10	Tampere	5500.00	444 1234
33	Ville	Viima	1975-12-08	Nokia	4000.50	444 4343
12	Pekka	Puro	1985-01-09	Tampere	3000.00	
98	Jenni	Joki	1961-06-20	Lempäälä	4300.00	444 4488
99	Alli	Kivi	1988-07-19	Nokia	2500.00	444 5555

Kuinka tyhjäarvot käyttäytyvät?

c)
SELECT ttnro
FROM tyontekija
WHERE puhelin = NULL;

ttnro

(0 rows)

d)
SELECT ttnro
FROM tyontekija
WHERE puhelin <> NULL;

ttnro

(0 rows)

TYHJÄARVOJA EI TESTATA NÄIN

WHERE-osa: tyhjäarvot

ttnro	etunimi	sukunimi	saika	kotikunta	palkka	puhelin
88	Jukka	Susi	1957-11-10	Tampere	5500.00	444 1234
33	Ville	Viima	1975-12-08	Nokia	4000.50	444 4343
12	Pekka	Puro	1985-01-09	Tampere	3000.00	
98	Jenni	Joki	1961-06-20	Lempäälä	4300.00	444 4488
99	Alli	Kivi	1988-07-19	Nokia	2500.00	444 5555

Kuinka tyhjäarvot käyttäytyvät?

e)

```
SELECT ttnro
FROM tyontekija
WHERE puhelin IS NULL;
```

```
ttnro
-----
    12
(1 row)
```

TYHJÄARVOJEN TESTAUS OK

f)

```
SELECT ttnro
FROM tyontekija
WHERE puhelin IS NOT NULL;
```

```
ttnro
-----
    88
    33
    98
    99
(4 rows)
```

Kyselyt: Tulosrivien järjestäminen

- Tulostaulun rivit voidaan järjestää ORDER BY -määreellä.

```
SELECT [DISTINCT] sarake {, sarake}  
FROM taulu {,taulu}  
[WHERE ehto]  
[ORDER BY sarake [ASC|DESC] {, sarake [ASC|DESC]}]
```

|-merkki ilmaisee vaihtoehtoiset avainsanat

- ASC nouseva järjestys
 - Lajittelujärjestys on oletusarvoisesti nouseva; ASC-määrettä ei tarvitse välttämättä käyttää.
- DESC laskeva järjestys
- Lajittelu suoritetaan sisäkkäin ORDER BY -listan mukaisessa järjestyksessä.

Kyselyt: Tulosrivien järjestäminen

Järjestetään tulosrivit sukunimen mukaan laskevasti.

```
SELECT sukunimi, kotikunta  
FROM tyontekija  
ORDER BY sukunimi DESC;
```

sukunimi		kotikunta
Viima		Nokia
Susi		Tampere
Puro		Tampere
Kivi		Nokia
Joki		Lempäälä

Järjestetään tulosrivit nousevasti ensin kotikunnan mukaan ja sitten sukunimen mukaan.

```
SELECT kotikunta, sukunimi  
FROM tyontekija  
ORDER BY kotikunta, sukunimi;
```

kotikunta		sukunimi
Lempäälä		Joki
Nokia		Kivi
Nokia		Viima
Tampere		Puro
Tampere		Susi

Kyselyt: Tulosrivien järjestäminen

```
SELECT puhelin  
FROM tyontekija  
ORDER BY puhelin;
```

puhelin

444 1234
444 4343
444 4488
444 5555

(5 rows)

```
SELECT puhelin  
FROM tyontekija  
ORDER BY puhelin DESC;
```

puhelin

444 5555
444 4488
444 4343
444 1234

(5 rows)

tyhjäarvo

Kyselyn evaluointialgoritmi

- Yksinkertaisen SQL-kyselyn rakenne

SELECT sarakeluettelo

FROM taulunnimi

[WHERE ehto]

[ORDER BY];

Yksinkertainen kyselyn evaluointialgoritmi: kyselyn osien suoritusjärjestys

1. FROM-osa

- Kopioidaan FROM-osassa annettu taulu tulostauluksi.

2. WHERE-osa

- Poistetaan edellisen vaiheen tulostaulusta ne rivit, jotka eivät täytä ehtoa.

3. SELECT-osa

- Poistetaan kaikki sarakkeet, jotka eivät esiinny SELECT-listassa.

4. ORDER BY -osa

- Järjestetään rivit järjestystekijöiden (sarakkeiden) mukaisesti.

Tietokannanhallintajärjestelmä ei välttämättä evaluoi kyselyä tällä tavoin –

algoritmin tarkoituksena on auttaa ymmärtämään, miten kyselyn tulos muodostuu.

Rivien muuttaminen

- Rivejä muutetaan UPDATE-lauseella.

UPDATE taulu

SET sarake = ilmaus {, sarake = ilmaus}

[WHERE ehto];

- Muutettavat rivit rajataan WHERE-osan ehdolla.
 - Jos WHERE-osan ehto ei anneta, päivitetään kaikkia rivejä.
- Ilmauksessa voidaan käyttää operaatioita ja funktioita.
 - Näistä lisää myöhemmin.

Rivien muuttaminen

```
UPDATE tyontekija  
SET sukunimi = 'Myrsky', kotikunta = 'Orivesi'  
WHERE ttnro = 99;
```

Muutokset riville, jolla
ttnro-sarakkeen arvo
on 99.

```
UPDATE tyontekija  
SET palkka = 4050.00  
WHERE palkka = 4000.50;
```

← 2
← 1 Ehdon totuusarvo
lasketaan ensin.

Tämän jälkeen
päivitetään ehdossa
testatun sarakkeen
arvoa.

Rivien poistaminen

- Rivien poistaminen tapahtuu DELETE-lauseella.

```
DELETE FROM taulu  
[WHERE ehto];
```

- Poistettavat rivit rajataan WHERE-osan ehdolla.
 - Jos WHERE-osan ehtoa ei anneta, **poistetaan kaikki rivit!**
 - Taulun määrittely säilyy kuitenkin järjestelmässä.

```
DELETE FROM tyontekija;
```

```
DELETE FROM tyontekija  
WHERE ttnro > 20;
```