

SQL tietokantakieli

- n SQL:llä voidaan...
 - n määritellä ja muokata tietokantaa ja sen käyttöoikeuksia
 - n virittää tietokannan talletusrakenteita
 - n hakea tietoa tietokannasta
 - n näytölle tai tiedostoon
 - n sovellusohjelman käyttöön
 - n tehdä päivityksiä tietokantaan (muuttaa dataa)
 - n vuorovaikutteisesti
 - n sovellusohjelman kautta

1

SQL

- n SQL on standardoitu
- n viimeisin standardi vuodelta 1999
- n murteita - yhteinen suppeahko ydin

2

SQL-tietokanta

- n SQL-tietokanta muodostuu yhden tai useamman kaavion (schema) määrittelemistä tauluista (table)
- n Kullakin kaaviolla on **omistaja**, joka omistaa myös kaavion määrittelemät taulut. Taulu muodostuu riveistä (row)
- n Taulu vastaa relaatiomallin relaatiota, mutta
 - n sallii etenkin kyselyiden tuloksissa samanlaisen rivin toistumisen (duplikaatit) **matemattisesti monijoukko (multiset)**

3

SQL

```
graph TD; SQL[SQL] --- DDL[määrittelykieli (DDL)]; SQL --- DML[Käsittelykieli (DML)]; DDL --- DDL_text[käyttäjät ja oikeudet  
tietokannan rakenne  
tietokantaproseduurit]; DML --- DML_text[kyselyt  
ylläpito-operaatiot];
```

4

SQL

- n SQL-kielessä avainsanat, taulu- käyttäjä- ja sarakenimet voi kirjoittaa joko suur- tai pienaakkosina eli **select merkki** ≡ **SELECT MerKKI**
- n Tietokannassa olevan datan suhteen kieli on kuitenkin herkkä kirjainmuodolle eli
 - n **Merkki='Ford'** on eri kuin **Merkki='FORD'**
 - n joissain järjestelmissä tätä käyttäytymistä voidaan säätää asennusparametrein tai jopa kyselykohtaisesti

5

SQL tiedonmäärittelykieli

- n Tiedonmäärittelykielessä lauseita tietokantaelementtien {**user, role, schema, table, domain, procedure, function, trigger, ...**} luontiin, muokkaukseen ja poistoon
 - n **create** -luo
 - n **alter** - muokkaa
 - n **drop** - poistaa

6

SQL taulun luonti

- create table määrittelee taulun rakenteen
- create table *tablename* (
column definition 1, ...,
column definition n
 [, *constraint 1*, ...])

sarakemäärittely ::=

column_name datatype [not null]
 [default *value*] [*column constraint* ...]

7

Taulun määrittely

```
create table Ordered (  

    OrderId      integer not null,  

    WhenMade     date not null,  

    Customer     integer not null,  

    WayIssued    varchar(20),  

    PaymentBy    varchar(20) not null,  

    TotalPrice   decimal(6,2) not null,  

    constraint pk_order primary key (OrderId),  

    constraint fk_ordercustomer foreign key  

        (Customer) references Customer  

);
```

8

Taulun määrittely

```
create table Ordered (  

    OrderId      integer not null, -- päiväys (tässä on kyseessä Oracle, joten timestamp)  

    WhenMade     date not null,  

    Customer     integer not null, -- vaihtuvamittainen merkkijono  

    WayIssued    varchar(20),  

    PaymentBy    varchar(20) not null,  

    TotalPrice   decimal(6,2) not null, -- kokonaisluku  

    constraint pk_order primary key (OrderId),  

    constraint fk_ordercustomer foreign key  

        (Customer) references Customer  

);
```

kokonaisluku

desimaaliluku
kokonaispituus 6,
desimaaliosa 2

9

Taulun määrittely

```
create table Ordered (  

    OrderId      integer not null, -- pakollinen tieto  

    WhenMade     date not null, -- pakollinen tieto  

    Customer     integer not null, -- pakollinen tieto  

    WayIssued    varchar(20), -- pakollinen tieto  

    PaymentBy    varchar(20) not null, -- pakollinen tieto  

    TotalPrice   decimal(6,2) not null, -- pakollinen tieto  

    constraint pk_order primary key (OrderId), -- avain  

    constraint fk_ordercustomer foreign key  

        (Customer) references Customer -- viiteavain  

);
```

avain

viiteavain

10

SQL tiedonmäärittely

Aikoja

- Date päiväys
- Time kellonaika
- Timestamp päiväys ja kellonaika
- Interval aikaero

Aikoja voidaan verrata ja niillä voi laskea

Olkoon *this_day date*-tyyppinen,
this_day + interval 3 day

on kolmen päivän päästä

11

SQL tiedonmäärittely

Viiteavainmäärittelyyn voidaan liittää toimintasääntö: mitä tehdään operaation rikkoessa viite-ehyden

```
foreign key (sarakkeet1) references taulu [(sarakkeet2)]  

    [ on delete {restrict | cascade | nullify} ]  

    [ on update {restrict | cascade | nullify} ]
```

Kun viitteen kohde katoaa:

- restrict* estää rikkovan operaation (*oletusarvo*)
- cascade* vyöryttää= poistaa tai muuttaa viittaavat rivit
- nullify* tyhjentää viittaukset
- valinta sen perusteella mikä on tarkoituksenmukaista

12

SQL -kysely

Kyselyn yleisrakenne:

```
select tulostietomäärittely
from taulut
[where valintaehdot]
[group by ryhmitystekijät]
[having ryhmärajoitteet]
[order by järjestysperusta]
```

Kysely tuottaa nimettömän tulostaulun.

13

SQL-kysely

```
select merkki, reknro
from auto
where vmalli=1996 and
      vari ='punainen' and merkki like
      'Fo%'
order by merkki, reknro
```

n Vuoden 1996 mallia olevien punaisten merkiltään 'Fo'-alkuisten autojen merkki ja rekisterinumero merkin ja saman merkin sisällä rekisterinumeron mukaan järjestettynä

14

SQL -kysely

```
select merkki, reknro
from auto
where vmalli=1996 and
      vari ='punainen' and merkki like 'Fo%'
order by merkki, reknro
```

melkein projektio valinta

n Vuoden 1996 mallia olevien punaisten merkiltään 'Fo'-alkuisten autojen merkki ja rekisterinumero merkin ja saman merkin sisällä rekisterinumeron mukaan järjestettynä

15

SQL-kysely

n Tulostietomäärittelyn elementeille lasketaan normaalitapauksessa arvo jokaista valintaehdot täyttävää riviä (rivyhdistelmää) kohden

```
select merkki
from auto
where vmalli=1996 and
      vari ='punainen' and merkki like 'Fo%'
order by merkki
```

Jos taulussa auto olisi 100 punaista vuoden 1996 Fordia tulisi merkki 'Ford' tulostauluun 100 kertaa.

Toimii siis toisin kuin relaatioalgebran projektio, joka poistaa tuplat

16

SQL -kysely

n Projektion kaltainen toistuvien arvojen karsinta saadaan aikaan liittämällä tulostietomäärittelyn alkuun avainsana distinct

```
select distinct merkki
from auto
where vmalli=1996 and
      vari ='punainen' and merkki like 'Fo%'
order by merkki
```

Nyt Ford tulisi tulokseen vain kerran

17

SQL-kysely

n Kyselyn ehto-osassa voidaan verrata saraketta, vakiota, funktion arvoa tai lausekeen arvoa

- n sarakkeeseen, vakioon, funktion arvoon, lausekkeen arvoon
- n arvojoukoon
- n maskiin

n Voidaan myös tutkia sarakkeen tyhjiyttä

n Jos vertailun toisena osapuolena on tyhjäarvo on tulos 'tuntematon'. Rivi tulee valituksi tulokseen vain jos ehdon arvo on 'tosi' (true).

18

SQL -kyselyt

- Totuusarvot tosi (true) ja epätosi (false) käyttäytyvät loogisissa lausekkeissa kuten ohjelmointikielten yhteydessä
- Kolmas totuusarvo 'tuntematon' käyttäytyy seuraavasti

AND	true	false	unknown
true	true	false	unknown
false	false	false	false
unknown	unknown	false	unknown

NOT	
true	false
false	true
unknown	unknown

19

SQL-kyselyt

OR	true	false	unknown
true	true	true	true
false	true	false	unknown
unknown	true	unknown	unknown

Sarake is null: tuottaa tuloksen true, jos sarakkeessa on tyhjääarvo, muuten false

Sarake is not null: tuottaa tuloksen false, jos sarakkeessa on tyhjääarvo, muuten true

Sarake = null: tuottaa aina tuloksen unknown

20

SQL -kyselyt

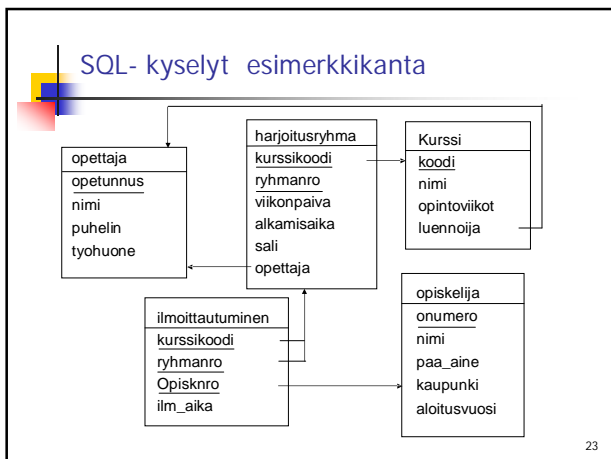
- Tietotyyppistä riippuen sarakearvoilla voi tulostietomäärittelyssä ja ehdoissa suorittaa laskentaoperaatioita (+, -, *, /, merkkijonoille katenaatio || (peräkkäin laittaminen))
 - pituus*paino, etunimi||' '||sukunimi
- Merkkijonovakiot 'hispuihin', numeeriset ilman
- Päiväykset muodossa date '2005-11-24'
- On mahdollista käyttää myös skalaarifunktioita – esim. merkkijonon pituus
- length(Sarake), funktiovalikoima vaihtelee

21

SQL -kyselyt

- Kyselyn from -osassa yksi tai useampi taulu (tai alikysely)
 - Jos tauluja on vain yksi, on kyseessä valinta
 - Jos tauluja on monta, on kyseessä ristitulo ellei ehto-osassa ole liittosehtoa (hyvin harvoin halutaan tulokseksi ristitulo)
 - Jos tauluja on monta ja ehto-osassa on liittosehto, on kyseessä liitos – muista siis liittosehto

22



SQL -kyselyt

- Opettajien nimet:


```
select nimi from opettaja;
```
- Opiskelijoiden pääaineet


```
select distinct paa_aine from opiskelija;
```
- Tietojenkäsittelytieteen pääaineopiskelijoiden nimet


```
select nimi from opiskelija where paa_aine = 'TKT';
```
- Espoossa asuvat matematiikan opiskelijat


```
select * from opiskelija where paa_aine = 'MAT' and kaupunki = 'Espoo';
```

24

SQL -kyselyt

- Opiskelijat joiden sukunimi on Tele
 - `select * from opiskelija where nimi like 'Tele %';`
 - (esimerkkitaulussa nimet ovat muodossa sukunimi+'space'+etunimi)
- Opiskelijat, joiden etunimi alkaa L:llä
 - `select * from opiskelija where nimi like '% L%';`

25

Liitokset SQL:ssä

- Kyselyn from-osassa voi olla useita tauluja
- Kaikki ne taulut, joiden dataa halutaan mukaan tulokseen on annettava from-osassa
- Arto Wiklan luennoimat kurssit
 - `Select kurssi.nimi`
 - `from kurssi, opettaja`
 - `where opettaja.nimi='Arto Wikla' and`
 - `kurssi.luennoija=opettaja.opetunnus;`

täytyy täsmentää koska sama sarake kahdessa taulussa

liitosehto

26

Liitokset SQL:ssä

- Tauluille voidaan from-osassa antaa tilapäinen kyselyn sisäinen nimi (alias, correlation name)
 - `from taulu [AS] alias`
- liitettävillä tauluilla on usein samannimisiä sarakkeita, joten taulunimeä on käytettävä tarkenteena - alias voi olla lyhenne, joka vähentää kirjoitusvaivaa

as ei käy Oracllessa

27

Liitokset SQL:ssä

- Jos sama taulu esiintyy from -osassa useaan kertaan, on taulun esiintymät erotettava käyttämällä aliasta
- Esim.: Kurssiparit, joilla on sama luennoija
 - `select A.nimi, B.nimi`
 - `from kurssi A, kurssi B`
 - `where A.luennoija=B.luennoija and`
 - `A.koodi<B.koodi`
 - `order by A.nimi, B.nimi`
- ehto `A.koodi<B.koodi` estää saman parin toistumisen eri järjestyksissä

28

Liitokset SQL:ssä

- Tyypillinen virhe liitoksissa on jättää jokin liitosehto pois, jolloin tuloksen rivijoukko tulee huomattavasti suuremmaksi kuin pitäisi
- jos from-osassa on n kpl liitettäviä tauluja tarvitaan vähintään n-1 liitosehtoa. Taulujen liittäminen voi perustua useaan sarakkeeseen, jolloin ehtolausekkeessa tarvittavien alkusehtojen määrä voi moninkertaistua.

29

Liitokset SQL:ssä

- Yleensä kyselyt rakentuvat siten, että niissä on jokin keskeinen taulu, johon muita liitetään. Voi olla, ettei tuosta keskeisestä taulusta tule mitään dataa tulokseksi.

Keskeinen taulu

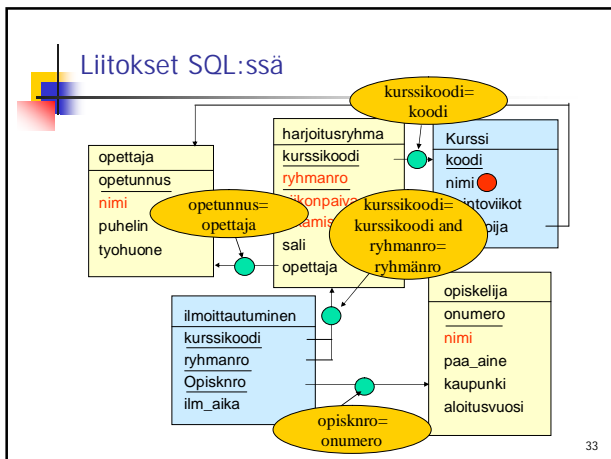
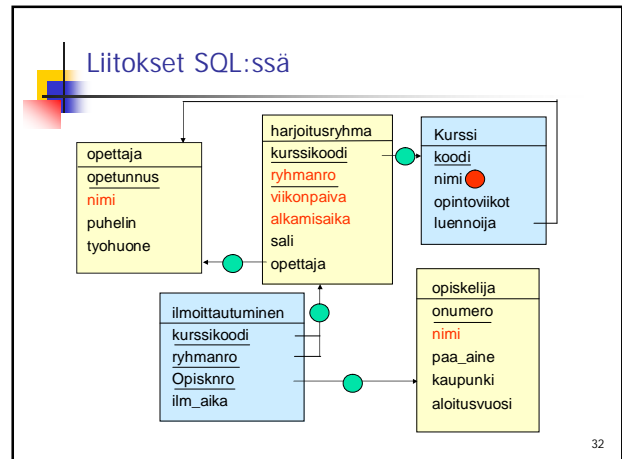
liitokset

30

Liitokset SQL:ssä

- n Laadi raportti kurssin Java-ohjelmointi harjoitusryhmistä
- n Mitä halutaan tulokseen:
 - n Ryhmän numero (taulussa harjoitusryhmä)
 - n Ohjaajan nimi (taulussa opettaja)
 - n kokoontumispaiva (taulussa harjoitusryhmä)
 - n alkamisaika (taulussa harjoitusryhmä)
 - n opiskelijan nimi (taulussa opiskelija)
- n Taulut **opettaja**, **harjoitusryhmä** ja **opiskelija** on välttämättä otettava kyselyn from osaan
- n Taulu **ilmoittautuminen** tarvitaan opiskelijoiden kytkemiseksi ryhmiin ja taulu **kurssi**, jotta saataisiin selville Java ohjelmoinnin kurssikoodi

31



Liitokset SQL:ssä

```

select H.ryhmanro rno, Ope.nimi ope, H.viikonpaiva,
H.alkamisaika, O. Nimi opiskelija
from Harjoitusryhma H, opettaja Ope, opiskelija O,
ilmoittautuminen I, kurssi K
where
H.kurssikoodi=K.koodi and
I.kurssikoodi=H.kurssikoodi and
I.ryhmanro=H.ryhmanro and
Ope.Opetunnus=H.Opettaja and
I.Opisknro=O.onumero and K.nimi='Java ohjelmointi'
order by H.ryhmanro, O.nimi;
    
```

34