SQL tietokantakieli

- Relaatiomalliin pohjautuvat tietokannat
  - Perustana relaatiomallin mukainen näkemys tietojen rakenteesta ja käsittelystä
  - Tietokantojen rakenteen määrittelyssä ja käsittelymahdollisuuksissa on paljon piirteitä, joihin abstrakti relaatiomalli ei ota kantaa, esim.
    - Miten tietokantakaavio käytännössä esitetään
    - Millaista tietoa tietokantaan voi tallettaa
    - Miten esitetään tietokantaan kohdistuvat kyselyt
    - Miten tietokantaa ylläpidetään
    - Miten hallitaan tietokannan käyttöoikeuksia
    - Miten hallitaan samanaikaisia tietojen käyttöä
    - Miten kytketään ohjelmat tietokantaan
  - Useita toteutuksia, joissa keskeisenä osana SQL-tietokantakieli

1

SQL tietokantakieli

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

2

SQL tietokantakieli

- SQL on standardoitu
- viimeisin standardi vuodelta 2003
- murteita – mutta kuitenkin yhteinen ydin
  - Vaihto toteutuksesta toiseen ei välttämättä ole kuitenkaan yksinkertaista

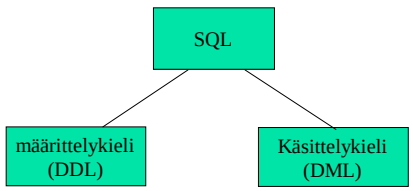
3

SQL-tietokanta

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

4

SQL



käyttäjät ja oikeudet
tietokannan rakenne
tietokantaproseduurit

kyselyt
ylläpito-operaatiot

5

SQL

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

6

SQL tiedonmäärittelykieli

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

7

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* ...]

8

Taulun määrittely

```
create table Ordered (
   OrderId       integer not null,
   WhenMade      timestamp(0) 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
);
```

9

Taulun määrittely

```
create table Ordered (
   OrderId       integer not null,
   WhenMade      timestamp(0) 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
);
```

Aikaleima, ei sekunnin osia

vaihtuvamittainen Merkkijono Maksimipituus 20

kokonaisluku

desimaaliluku kokonaispituus 6, desimaaliosa 2

10

Taulun määrittely

```
create table Ordered (
   OrderId       integer not null,
   WhenMade      timestamp(0) 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
);
```

pakollinen tieto

avain

viiteavain

11

SQL tiedonmäärittely

- Aikoja
  - Date päiväys
  - Time kellonaika
  - Timestamp(x) päiväys ja kellonaika
    - (sekunnin osat x desimaalin tarkkuudella)
  - Interval aikaero
- Aikoja voidaan verrata ja niillä voi laskea
  Olkoon this_day date-tyyppinen,
  this_day + interval 3 day
  on kolmen päivän päästä
- Huom: Harjoittelukannassa oleva date-tietotyyppi käyttäytyy kuten timestamp(0) = Oraclen vanha tapa. Päiväykseen voi lisätä aikaa desimaalilukuina, joissa kokonaisosa kertoo vuorokausien lukumäärän esim this_day+0.5 on 12 tunnin päästä this_day arvosta.

12

SQL tiedonmäärittely

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

  foreing key (*sarakkeet*) 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

13

SQL -kysely

Kyselyn yleisrakenne:

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

Kysely tuottaa nimettömän tulostaulun.

14

SQL-kysely

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

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

15

SQL -kysely

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

melkein projektio

valinta

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

16

SQL-kysely

- Tulostietomäärittelyn elementeille lasketaan normaalitapauksessa arvo jokaista valintaehdot täyttävää riviä (riviyhdistelmää) 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

17

SQL -kysely

- 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

18

SQL-kysely

- Kyselyn ehto-osassa voidaan verrata saraketta, vakiota, funktion arvoa tai lausekkeen arvoa
  - sarakkeeseen, vakioon, funktion arvoon, lausekkeen arvoon
  - arvojoukkoon
  - maskiin
- Voidaan myös tutkia sarakkeen tyhjyyttä
- Jos vertailun toisena osapuolena on tyhjäarvo on tulos 'tuntematon'. Rivi tulee valituksi tulokseen vain jos ehdon arvo on 'tosi' (true).

19

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

20

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 tyhjäarvo, muuten false

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

Sarake = null: tuottaa aina tuloksen unknown

21

SQL -kyselyt

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

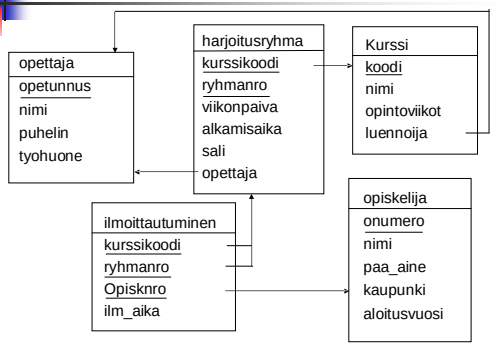
22

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 liitosehtoa (hyvin harvoin halutaan tulokseksi ristitulo)
- Jos tauluja on monta ja ehto-osassa on liitosehto, on kyseessä liitos – muista siis liitosehto

23

SQL- kyselyt esimerkkikanta



24

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';

25

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%';

26

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

27

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 Oraclessa

28

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

29

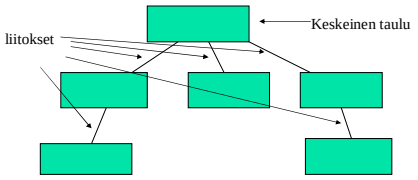
Liitokset SQL:ssä

- 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 alkeisehtojen määrä voi moninkertaistua.

30

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.



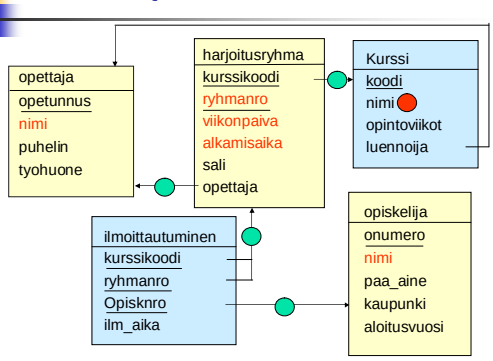
31

Liitokset SQL:ssä

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

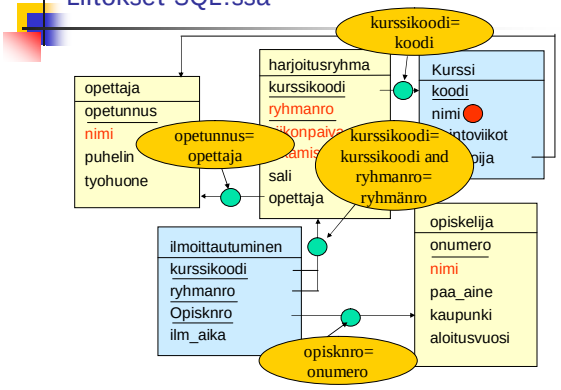
32

Liitokset SQL:ssä



33

Liitokset SQL:ssä



34

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;
```

35