Tietokannan hallinta

Kurssijärjestelmämme:

Tietokantojen perusteet
- tietokannan käyttö: SQL, sovellukset

Tietokannan hallinta
- tietokannanhallintajärjestelmän ominaisuuksia:
  - tallennusrakenteet
  - kyselyjen toteutus
  - tapahtumien hallinta

Laudatur:

Tietokannan mallinnus
- tietokannan suunnittelu
- oliotietokannat

Tietokantarakenteet ja -algoritmit
- laajemmin kuin tällä kurssilla

Erikoiskursseja:

Tietovarastot
Tapahtumakäsittely
Paikkatietojärjestelmät

1. Johdanto

Tietokanta (database) = loogisesti yhtenäinen
kokoelma toisiinsa liittyviä tietoja
- fyysisellä tasolla: tiedostoja, tauluja

- tietyssä mielessä pysyvä: monen sovelluksen käytössä
- sovelluksista erillinen: suunnitellaan erikseen

Esim. yrityksen henkilöstötietokanta

EMPLOYEE(ssn, name, bdate, address, salary, superssn, dno)
DEPARTMENT(dname, dnumber, mgrssn)
PROJECT(pname, pnumber, plocation, dnum)
WORKS_ON(essn, pno, hours)
DEPT_LOCATIONS(dnumber, dlocation)
DEPENDENT(essn, dep_name, relationship)

Tietokannan kaavio (database schema) = tietokannan määrittely: ilmaisee tietokannan kuvaaman kohteen ('miniworld') jäsentelyn:
relaatiot (attribuutit, ominaisuuksia)
relaatioiden väliset suhteet (esim. viite-eheys)

- erilaisia kaavioita: tietokannan mallinnus

Henkilöstötietokannan määrittely SQL:llä:

```
create table employee
  (ssn         char(9)       not null,
   name        varchar(30)   not null,
   bdate       date,
   address     varchar(30),
   salary      decimal(10,2),
   superssn    char(9),
   dno         int           not null,
   primary key (ssn),
   foreign key (superssn) references employee(ssn),
   foreign key (dno) references department(dnumber) );

create table department
  (dname       varchar(15)   not null,
   dnumber     int           not null,
   mgrssn      char(9)       not null,
   primary key (dnumber),
   unique (dname),
   foreign key (mgrssn) references employee(ssn) );

create table project
  (pname       varchar(15)   not null,
   pnumber     int           not null,
   plocation   varchar(15),
   dnum        int           not null,
   primary key (pnumber),
   unique (pname),
   foreign key (dnum) references department(dnumber) );
```



```
create table works_on
  (essn        char(9)       not null,
   pno         int           not null,
   hours       decimal(3,1),
   primary key (essn, pno),
   foreign key (essn) references employee(ssn),
   foreign key (pno) references project(pnumber) );

create table dept_locations
  (dnumber     int           not null,
   dlocation   varchar(15)   not null,
   primary key (dnumber, dlocation),
   foreign key (dnumber) references department(dnumber) );

create table dependent
  (essn        char(9)       not null,
   dep_name    varchar(15)   not null,
   relationship varchar(8),
   primary key (essn, dep_name),
   foreign key (essn) references employee(ssn) );
```

Hieman laajempana: E&N, s. 204-205
- myös tietokannan sisältö ("tietokannan tila")

Tietokannanhallintajärjestelmä (tkhj, dbms) = yleiskäyttöinen ohjelmisto, jonka avulla luodaan, ylläpidetään ja käytetään tietokantoja.

Tietokantajärjestelmä = tietokanta + sen käyttöön liittyvät ohjelmat

Tietokantaperiaatteen ominaisuuksia:

- tieto tallennetaan vain kerran (yleensä)
- tallennus pysyvää (ohjelman tietoihin verrattuna)
- tietojen loogisia suhteita voidaan määritellä: rajoitteet; suhteiden voimassaolon valvonta
- tietokanta(järjestelmä) sisältää datan lisäksi myös kuvaustietoa (metadataa, tietohakemisto)
- tietokannan ja sovellusohjelmien riippumattomuus
- tietokannan eri käyttäjiä varten erilaisia näkemyksiä, erilaisia käyttötapoja
- tietojen kontrolloitu samanaikainen käyttö
- häiriötilanteiden hallinta (elvytys)

- erilaisia toimijoita:
  - tietokannan hoitaja
  - tietokannan suunnittelija
  - peruskäyttäjät
  - sovellusohjelmoijat
  - tkhj:n ja apuohjelmien suunnittelijat (taustalla)

Tietohakemiston merkitys:

- kaavion tallennuspaikka
- kyselyä toteutettaessa saadaan tarvittavat tiedot taulujen ominaisuuksista
- relaatiotietokannassa taulujen muodossa
  → voidaan kohdistaa kyselyjä

Esimerkki. Oraclen kaaviotauluja

USER_CATALOG: käyttäjän omistamat taulut, näkymät ym.
USER_TAB_COLUMNS: käyttäjän omistamien taulujen ja näkymien sarakkeet
USER_VIEWS: käyttääjän näkymät
ALL_CATALOG: kaikki oliot, joihin käyttäjällä on pääsy
ALL_TABLES, ALL_VIEWS jne

Käyttö SQL-lauseilla: esim.

```
select * from user_catalog;
```

TABLE_NAME	TABLE_TYPE
DEPARTMENT	TABLE
EMPLOYEE	TABLE
. . .	


```
select column_name, data_type
from user_tab_columns
where table_name='EMPLOYEE;
```

COLUMN_NAME	DATA_TYPE
SSN	CHAR
NAME	VARCHAR2
BDATE	DATE
ADDRESS	VARCHAR2
SALARY	NUMBER
SUPERSSN	CHAR
DNO	NUMBER

```
desc employee;
```

NAME	NULL?	TYPE
SSN	NOT NULL	CHAR(9)
NAME	NOT NULL	VARCHAR2(30)
BDATE		DATE
ADDRESS		VARCHAR2(30)
SALARY		NUMBER
SUPERSSN		CHAR(9)
DNO	NOT NULL	NUMBER

Kolmitasoarkkitehtuuri:

ulkoinen taso: käyttäjänäkymät, sovellusohjelmat

käsitetaso: tietokannan looginen kaavio (yleinen rakenne, ”kaikille”)
sisäinen taso: tietokannan säilytyskaavio (tiedostorakenteet, saantipolut)

Tietoriippumattomuus:

- looginen: tietokannan loogisen kaavion muutos ei välttämättä vaikuta sovellusohjelmiin (jos ei muuteta ohjelman näkymän käsittelemiä tietoja)

- fyysinen: säilytystason (tallennustason, tiedostotason) ratkaisut eivät vaikuta loogisen tason ratkaisuihin (esim. kyselyn muoto, erilaiset saantipolut; kuitenkin vaikuttavat suorituskykyyn)

Tietokannanhallintajärjestelmän osat:

- tietohakemisto
- tietokantamoottori
  - pääsyn valvonta (saantioikeudet)
  - kyselyn optimoija
  - tapahtuman hallinta (samanaikaiset käyttäjät, elvytys häiriöistä)
  - eheyden valvonta (tietokannan osien väliset suhteet, oikeellisuus)
  - puskurien hallinta ja saantimenetelmät (muistiliitäntä: siirto-operaatiot käyttöjärjestelmän avulla)
  - ajonaikainen tk-prosessori toteuttaa kyselyt ja hoitaa toimintaan liittyvän kontrollin

- kääntäjiä/liittymiä:
  tiedonmäärittelykielelle
  kyselyille
  sovellusohjelmien käyttämille tietokantakielille (ohjelmointirajapinnoille)

- apuohjelmia
  - tietokannan lataukseen (tiedostosta tai toisesta tietokannasta)
  - tietokannan varmistuksiin (toiselle levylle, nauhalle)
  - tietokannan uudelleenorganisointiin
  - suorituskyvyn ym. toiminnan seurantaan