## Tietokantojen perusteet (Avoin yliopisto), kurssikoe 16.6.2014

Vastaa tehtävät 1, 2, 3 ja 4 **erillisille** konsepteille. Kirjoita jokaiseen konseptiin kurssin nimi, kokeen päivämäärä, opiskelijanumero (tai henkilötunnus), oma nimi ja nimikirjoitus.

Vaikka jättäisit johonkin tehtävään vastaamatta, tulee vastauspaperi siinäkin tapauksessa palauttaa.

1. Miksi tietoa ei kannata aina tallentaa yksittäisiin tiedostoihin itse? Mitä tietokannat ja tietokantajärjestelmät tarjoavat tiedon käsittelyyn? (4p)

2. (5p) *Kumpulan jäätelötehdas Oy* on laajentamassa toimintaansa. Yhden jäätelökioskin lisäksi yritys suunnittelee useita uusia kioskeja Kumpulan läheisyyteen. Paperinen kirjanpito on kuitenkin käynyt toiminnalle raskaaksi, joten yritys on tilannut sähköisen järjestelmän korvaamaan paperiversiot.

   > Kumpulan jäätelötehdas Oy on avaamassa useita uusia jäätelökioskeja. Kioskeja on tarvittaessa mahdollista siirtää, mutta ne sijaitsevat aina jossain tietyssä osoitteessa. Jokaisella kioskilla on paikkaan liittyvä osuva nimi. Historiaa varten kioskeista halutaan tallentaa myös avajaispäivä.
   >
   > Kioskeissa voi työskennellä useita työntekijöitä, mutta vähintään yksi työntekijä on vastuussa aina yhdestä kioskista. Työntekijöillä voi olla myös työvuoroja muissakin kioskeissa. Työntekijästä halutaan tallentaa vähintään henkilötunnus, etunimi, sukunimi, puhelinnumero ja tilinumero. Työvuoro liittyy aina yksittäiseen päivään ja aikaväliin. Palkkausta varten työntekijöillä on yksi palkkalaji, joka sisältää tiedon työntekijän tuntipalkasta.
   >
   > Jäätelökioskit myyvät vähintään yhtä jäätelölajia, mutta isommat kioskit myyvät useita lajikkeita (kermajäätelö, gelato, pehmojäätelö, mehujää ja jugurttijäätelö). Jokaisella jäätelölajilla on lämpötila, jossa sitä tulee asianmukaisesti säilyttää.
   >
   > Jäätelölajiin liittyy useita jäätelöitä. Jäätelöillä on nimi, maku ja mahdollisesti pidempi kuvaus. Myyntiä varten on tarpeen tallentaa myyntihinta sekä pitää yllä tuotteen ostohinta. Jäätelöistä halutaan tallentaa kuvia myyntipäätettä ja muita käyttötarkoituksia varten.
   >
   > Asiakkaat voivat tehdä ostoksia kioskeilla. Ostotapahtumasta halutaan tallentaa ajankohta ja maksutapa (käteinen tai maksukortti). Ostotapahtumaan liittyy useita jäätelöitä.
   >
   > Jatkossa olisi hyvä, että jokaisen kioskin jäätelösaldot pidettäisiin myös tietokannassa — eli kuinka monta kutakin jäätelöä on kussakin kioskissa varastossa, mutta tämä ei ole vielä tässä vaiheessa vaadittua.

   Tee kuvaukselle tietosisältökartoitus. Luo kartoituksen pohjalta **tietokantakaavio** tietotarpeelle. Yhteyksien ja osallistumisrajoitteiden merkintä on oleellista, attribuuteista riittää oleellisimmat.

3. *Kesälaitumien niittokoneet Ky* myy ruohonleikkureita. Alla on suunnitelma yrityksen tietokannasta.

   Tyyppi(id, nimi)
   Voimanlähde(id, nimi)
   Malli(id, tyyppi_id → Tyyppi, voimanlähde_id → Voimanlähde, mallinumero, nimi, hevosvoimat)
   Ruohonleikkuri(id, malli_id → Malli, sarjanumero, hyllypaikka)
   Hinta(id, malli_id → Malli, myyntihinta, ostohinta)
   Myynti(id, ruohonleikkuri_id → Ruohonleikkuri, myyntihinta, päivämäärä)

   Muodosta SQL-kyselyt, joilla seuraavat käyttötapaukset voidaan toteuttaa. *Sivulla kolme on lyhyt SQL-syntaksin referenssi.*

   (a) Hae kaikki ruohonleikkurit joiden mallin nimi on ”TurboCutter 3000”. (0,5p)

   (b) Lisää uusi ruohonleikkuri, jonka mallin nimi on ”TurboSlicer 100”, sarjanumero on ”4404” ja hyllypaikka ”A1”. Voit olettaa, että järjestelmä asettaa avaimet automaattisesti ja että malli on jo olemassa. Sarjanumero ja hyllypaikka ovat merkkijonoja. (1p)

   (c) Laske myyntitulot (myyntihintojen summa) päivälle 14.6.2014. Päivämäärä on ilmaistu date-tyyppinä. (0,5p)

   (d) Laske myyntivoitto yhteensä (ostohinta vähennettynä myyntihinnasta) kaikille ruohonleikkureille joiden voimanlähde on ”ihminen” (voimanlähteen nimi). Myyntihinta pitää tarkistaa aina Myynti-taulusta, sillä asiakas on saattanut saada alennusta. (1p)

4. Selitä lyhyesti seuraavat käsitteet. Anna kustakin käsitteestä myös lyhyt esimerkki.

   (a) Relaatiomalli (1p)

   (b) Tietokantaskeema (1p)

   (c) Tietokantataulun avain, viiteavain ja yhdistelmäavain (1p)

   (d) Boyce-Codd normaalimuoto (1p)

## *Brief SQL reference*

**SELECT syntax:**

```
SELECT [ALL | DISTINCT] select-list
   FROM table-reference-list
   [WHERE search-condition]
   [GROUP BY column-name [, column-name]... ]
   [HAVING search-condition]
   [[UNION | INTERSECT | EXCEPT] [ALL ] select-statement]...
   [ORDER BY {unsigned integer | column-name}
   [ASC|DESC]]

select-list ::= * | select-sublist [ {, select-sublist} ... ]
select-sublist ::=   derived-column | [table-name | table-identifier].*
derived-column ::=  expression [ [AS] column-alias] ]

table-reference-list ::= table-reference [, table-reference ... ]
table-reference ::=  table-name [[AS] correlation-name] |
   derived-table [[AS] correlation-name [(derived-column-list )]]  | joined-
table
table-name ::= table-identifier |   schema-name.table-identifier
derived-table ::=  subquery
derived-column-list ::= column-name-list
joined-table ::= cross-join | qualified-join | ( joined-table)
cross-join ::=  table-reference CROSS JOIN table-reference
qualified-join ::= table-reference [NATURAL] [join-type] JOIN
    table-reference [join-specification]
join-type ::=  INNER | outer-join-type [OUTER] | UNION
outer-join-type ::=  LEFT | RIGHT | FULL
join-specification ::= join-condition | named-columns-join
join-condition ::= ON search-condition
named-columns-join ::= USING (column-name-list)
column-name-list ::= column-identifier [ { , column-identifier} ...]

search-condition ::=  search-item | search-item { AND | OR }
    search-item
search-item ::= [NOT] { search-test | (search-condition) }
search-test ::= comparison-test | between-test | like-test |
    null-test | set-test | quantified-test | existence-test
comparison-test ::=
    expression { = | <> | < | <= | > | >= }  { expression | subquery }
between-test ::=  column-identifier [NOT] BETWEEN expression AND
expression
like-test ::=  column-identifier [NOT] LIKE value [ESCAPE value]
null-test ::=  column-identifier IS [NOT] NULL
set-test ::=  expression [NOT] IN ( {value [,value]... | subquery} )
quantified-test ::=
   expression { = | <> | < | <= | > | >= } [ALL | ANY | SOME]
      subquery
existence-test ::= EXISTS subquery
subquery ::= (query-specification)

expression ::=    expression-item | expression-item { + | - | * | / }
      expression-item
expression-item ::=    [ + | - ] { value | column-identifier | function |
    case-expression    | cast-expression | ( expression ) }
value ::= literal | USER | variable
function ::= set-function | string-function |
    numeric-function | datetime-function | system-function |
    datatypeconversion-function
set-function ::= COUNT (*) |
    { AVG | MAX | MIN | SUM | COUNT }( { ALL | DISTINCT }
     expression )
datatypeconversion-function ::=
   CONVERT_datatype(value-exp)   where datatype is SQL datatype
case-expression ::=   case-abbreviation | case-specification
case-abbreviation ::=
   NULLIF(value-exp, value-exp) | COALESCE(value-exp {, value-exp}...)
```

**INSERT syntax:**

```
insert into tablename [{column_list)]  { values ( value_list ) | subquery }
```

**UPDATE syntax:**

```
update tablename set assignment [, assignment ...]
[where search_condition]
assignment ::=   column_name = expression
```

**DELETE syntax:**

```
delete from tablename [where search_condition]

date_literal::= date ‘yyyy-mm-dd’
```