Andmetöötlus - Andmebaasid

- Andmebaas on koht andmete hoidmiseks
- Andmebaase võib jagada neljaks tüübiks:
  - Mitte-relatsioonilised (lamedad, failidel baseeruvad: DBA)
  - Relatsioonilised (RDBMS)
  - Objekt-relatsioonilised (ORDBMS)
  - Objekt-orienteeritud (ODBMS)

Andmetöötlus – Liht Andmebaasid

- DBA (*Database Abstraction*) mudel
  - Andmeid hoitakse võti/väärtus paaridena
  - Väärtuseks võib olla suvaline string (ka. binaarset infot sisaldav) – st, kõik peab olema stringi kujule viidud
  - Võti peab olema unikaalne!
  - Baseerub failidel ega ei vaja seega lisatarkvara
  - PHP omab kõike vajalikku selliste andmebaasidega ümber käimiseks (baaside loomine, kustutamine; elemtide lisamine, otsimine, eemaldamine, jpm)

Andmebaasid - DBA

- Andmete muutmine stringiks ja tagasi – serialiseerimine / deserialiseerimine

```
// Loome massiivi
$example = array(”1,2,3”);

// Serialiseerime selle
$string = serialize($expample);


// Siin võime oma stringi kuhugi salvestada ning hiljem tagasi lugeda


// Deserialiseerime stringi
$new_example = deserialize($string);

// Nüüd on $example ja $new_example täpselt sama sisuga
```

Andmebaasid - DBA

- Andmebaasi avamine ja sulgemine

```
int dba_open(string path, string mode, string handler, [int mode]);

int dba_close(int handle);
```

Mode (teine parameeter) omab järgmisi väärtusi:

- ”r” Avada andmebaas lugemiseks
- ”w” Avada kirjutamiseks
- ”c” Luua uus andmebaas
- ”n” Kirjutada eksisteeriv üle

Handler määrab ära konkreetse andmebaasi tüübi:

- ”dbm” Originaalne Berkeley DB-stiilis andmebaas (ei soovita kasutada)
- ”ndbm” Uuem ja paindlikum, kuid siiski piiratud võimalustega
- ”gdbm” GNU andmabaasimanager
- ”db2” DB2 vahendid firmalt ”Sleepycat Software”
- ”cdb” Toetab vaid lugemisoperatsioone

Andmebaasid - DBA

- Andmete läbi käimine

```
string dba_fisrtkey(int handle);
string dba_nextkey(int handle);

string dba_fetch(string key, int handle);
```

`FirstKey` alustab esimesest sissekandest ning iga `NextKey` nihutab ühe võrra edasi. Kui rohkem sissekandeid pole, tagastab *false*.

Konkreetse sissekande kätte saamiseks kasutame `Fetch` funktsiooni

Andmebaasid - DBA

- Andmete läbi käimine (näide)

```
$db = dba_open($dbpath, ”r”, $dbtype);     // Avame andmabaase

if (!$db) {                                // DB avamine ebaõnnestus
  echo(”Andmabaasi avamine ebaõnnestus”);
  return –1;
}

$key = dba_firstkey($db);                  // Esimene võti

while ($key != false) {
  $value = dba_fetch($key, $db);           // Loeme sissekande
  $entry = deserialize($value);            // Deserialiseerime

  // Väljasta andmed, vms                  // Tegeleme andmetega

  $key = dba_nextkey($db);                 // Järgmine võti
}

dba_close($db);                            // Sulgeme andmebaasi
```

Andmebaasid - DBA

- Andmete otsimine

  Ajapikku andmebaasid muutuvad väga mahukateks ning abiks oleks võimalus piirata tagastatavate sissekannete hulka vaid meid huvitavatega

```
function search_keyword($db, $keyword) {
  $r = array();

  $key = dba_firstkey($db);

  $count = 1;
  while ($key != false) {
    $value = dba_fetch($key, $db);
    if (ereg($keyword, $value)) {
      $r[$count] = $value;
      $count++;
    }
    $key = dba_nextkey($db);
  }
  return $r;
}
```

Andmebaasid - DBA

- Andmete eemaldamine

```
int dba_delete($id, $db);

int dba_sync($db);
```

Kustutamiseks peame andmebaasi avama kirjutamise režiimis ning peale kustutamist salvestame muutused kasutades `dba_sync()` funktsiooni.

Kui meil on vaja andmeid taastada, siis on mõttekam lisada eraldi info (N: staatus) näitamaks, et sissekanne on kustutatud. Reaalne sissekanne seejuures aga säilib


```
int dba_delete($id, $db);

int dba_sync($db);
```

Andmebaasid - DBA

- Andmete muutmine

```
int dba_replace(string key, string value, int handle);
```

Muutmine on sarnane kustutamisele (DB peab olema kirjutatav ning peale täiendamist kutsuda välja `dba_sync()`)

```
dba_replace($id, serialize($data), $db);
dba_sync($db);
```

- Andmete lisamine

Lisamiseks on meil vaja teada saada unikaalne võti – lihtsaim viis on kirjed läbi käia ja kõik üle lugeda ning lõpuks 1 võrra suurendada.

Lisamine toimub sama moodi nagu muutmine – kui vastava võtmega kirjet ei esine, siis ta lisatakse, muidu muudetakse olemasoleva väärtust

Andmebaasid - DBA

- Andmete import/export

  Kõige mugavam viis andmete vahetamiseks on kasutada CSV (*Comma-Separated Values*) faile – iga kirje väli on teisest eraldatud komaga ning iga kirje on eraldi real

  PHP sisaldab spetsiaalset funktsiooni:

```
array fgetcsv(int handle, int length);
```

```
$fp = fopen($path, "r");

while (!feof($fp)) {
  $data = fgetcsv($fp, 4096);

  if (is_array($data) && count($data) > 0) {
    $new = array();
    for ($i = 0; $i < count($data); $i++) {
      $new[$vars[$i]] = $data[$i];
    }               // $vars = array("name", "email", "telno", "address", "id");
    // jne – andmete salvestamine vms
  }
}
fclose($fp);
```

Andmebaasid - DBA

- DBA Plussid ja miinused
  - Plussid
    - Lihtne kasutada
    - Ei vaja lisatarkvara
    - Vähenõudlik
  - Miinused
    - Mitme kasutaja puhul ebaefektiivne
    - Ei toeta transaktsioone
    - Otsimisvõimalused/päringud väga piiratud
    - Iga tabeli/andmebaasi jaoks eraldi fail
    - Failide avamine/sulgemine iga pöördumise puhul
    - Puudulik turvalisus (kuigi mõned variandid omavad mingit)

Andmetöötlus – Rel. Andmebaasid

- Relatsiooniline mudel (RDBMS)
  - Arvestades DBA suhteliselt suuri puudujääke oleks tore omada midagi võimsamat
  - Relatsiooniline andmebaas võib koosneda 1 või enamast omavahel seotud tabelist, mis võivad olla kas ühes või mitmes failis
  - Relatsioon e suhe – peamine, mis RDBMS süsteeme muudest eristab
  - RDBMS lisab veel võimsa andmetega manipuleerimise keele – SQL (*Structured Query Language*)

Andmebaasid - RDBMS

- Iga tabel koosneb kirjetest ehk ridadest; iga rida omakorda väljadest ehk veergudest

ID	Nimi	Sünniaasta	Isikukood	Aadress
1	Siim Susi	1960	8972376763	Vändra, Soo 1
2	Marju Mesi	1977	7297236823	Tallinn, Vana 17

- Tabel moodustatakse esindamaks mingit objekti või selle osa (N: töötaja, tellimus jne)
- Vahest on objekti kujutamine lihtne, tihti aga üsna vaevaline

Andmebaasid - RDBMS

- Tabeli omadused
  - Iga tabeli veerg on mingit kindlat tüüpi (N: Sünniaasta on alati täisarv)
  - Iga tabeli rida peab olema teistest eristatav – seda siis kas ühe või enama välja abil
  - Tabeli ridu eristavat väljade kogumit nimetatakse võtmeks
  - Võti võib sisaldada ka viiteid teis(te)le tabeli(te)le
  - Ridade järjekord pole oluline – sorteerimine toimub päringutele vastamise ajal spetsiaalse konstruktsiooni abil

Andmebaasid - RDBMS

- Andmetüübid
  - **Integer** – täisarv; võib olla ka spetsiifilisematel kujudel: *SmallInt, HugeInt, ...*
  - **Float** – reaal- ehk ujupunktarv; võib olla ka täpsustatud kujudel: *Single, Double, ...*
  - **Char(size)** – muutumatu pikkusega tähejada; size määrab ära elementide arvu
  - **VarChar(size)** – muutuva pikkusega tähejada; size määrab ära elementide max arvu
  - **Date, Time** – kuupäev; võib olla ka erinevaid variatsioone: *DateTime, TimeStamp, ...*
  - **Numeric(suurus [, komakohad])** – üldine numbritüüp; suurus määrab andmeformaadi; komakohad on vaid reaalarvude puhul
  - **Blob, Text** – suured binaarsed ja tekstilõigud

Andmebaasid - RDBMS

- Andmebaasi näide (projektides osalevad inimesed)

ProjNo	ProjNimi	Töötaja#	Töötaja	Tasu kat.	Tunnitasu
1023	Web Site 1	10	Siim Susi	A	200
		11	Marju Mesi	B	175
		15	Toomas Saag	C	150
1555	Web X	10	Siim Susi	A	200
		13	Rutt Nulg	B	175

- Selleks, et taolist infot andmebaasi saaks sisestada, tuleb tabel viia 1. normaalkujule

Andmebaasid - RDBMS

- 1. normaalkuju nõuab, et:
  - Tuleb ellimineerida kõik korduvad grupid
    - Korduvad grupid paneme eraldi tabelitesse
  - Iga rida peab omama unikaalset nime (võtit) ning olema tervik

ProjNo	ProjNimi	Töötaja#	Töötaja	Tasu kat.	Tunnitasu
1023	Web Site 1	10	Siim Susi	A	200
1023	Web Site 1	11	Marju Mesi	B	175
1023	Web Sire 1	15	Toomas Saag	C	150
1555	Web X	10	Siim Susi	A	200
1555	Web X	13	Rutt Nulg	B	175

  - Kuna meil oli nõue, et iga rida oleks unikaalselt identifitseeritav, siis projekti number selleks ei sobi; sobib aga seesama koos töötaja numbriga!

Mis on sellel tabelil häda?

Andmebaasid - RDBMS

- Selline samade andmete mitmekordne sisestamine tekitab paratamatult vigu
- Nagu näha, koosneb antud tabel kahest selgelt eristuvast osast – projektist ning selles osalejast

ProjNo	ProjNimi	Töötaja#	Töötaja	Tasu kat.	Tunnitasu
1023	Web Site 1	10	Siim Susi	A	200
1023	Web Site 1	11	Marju Mesi	B	175
1023	Web Sire 1	15	Toomas Saag	C	150
1555	Web X	10	Siim Susi	A	200
1555	Web X	13	Rutt Nulg	B	175

- Jagades tabeli osadeks, saame 2 uut ->

Andmebaasid - RDBMS

Projektid

Proj#	ProjNimi
1023	Web Site 1
1555	Web X

Töötajad

Töötaja#	Töötaja	Tasu kat.	Tunnitasu
10	Siim Susi	A	200
11	Marju Mesi	B	175
13	Rutt Nulg	B	175
15	Toomas Saag	C	150

Andmebaasid - RDBMS

- Siit saame 2. normaalkuju, mis väidab, et:
  - Tabel peab olema 1. Normaalkujul
  - Tuleb luua eraldi tabel iga seostatud andmegrupi jaoks
    - Kumbki tabel on kindlale grupile – projektid ja töötajad
  - Garanteerida igas andmegrupis unikaalne primaarvõti
    - Kummagi tabeli iga rida on unikaalne – projekti puhul *Proj#*; töötaja puhul *Töötaja#*

Andmebaasid - RDBMS

- Tähelepanelikult vaadates puudub meil side nende kahe tabeli vahel!

- Et seda asja parandada, peame looma veel ühe, siduva, tabeli:

Projektides osalejad

Proj#	Töötaja#
1023	10
1023	11
1023	15
1555	11
1555	13

Andmebaasid - RDBMS

- Peale nende kahe normaalkuju on olemas veel kolmas
  - Tabel peab olema 2. Normaalkujul
  - Mitmes kirjes esinevatele väärtustele tuleb luua eraldi tabel
    Meie näite puhul on Töötajate tabelis tunnitasuga seotud info dubleeritud
  - Siduda uus tabel vanaga väliste võtmete abil
    Peame lisama viited teises tabelis olevale kirjele
  - Ellimineerida kõik võtmest mittesõltuvad väljad
    e. kui mõned väljad võivad olla olulised mitmele kirjele, siis on mõttekas nad eraldi tabelisse viia

Andmebaasid - RDBMS

Töötajad

Töötaja#	Töötaja	Tasu kat.	Tunnitasu
10	Siim Susi	A	200
11	Marju Mesi	**B**	**175**
13	Rutt Nulg	**B**	**175**
15	Toomas Saag	C	150

Töötajad

Töötaja#	Töötaja	Tasu kat.
10	Siim Susi	A
11	Marju Mesi	B
13	Rutt Nulg	B
15	Toomas Saag	C

Tunnitasud

Tasu kat.	Tunnitasu
A	200
B	175
C	150

Andmebaasid - RDBMS

- Võtmed
  - Nagu tähele võis panna, omasid meie tabelid kõik primaarseid võtmeid (kas ühte või kombineerituid)
  - Osa neist olid primaarsed (Projekt, Töötaja), teised välised (Projekti osalised, Töötaja tunnihinna klass)
  - Igal tabelil peab olema vähemalt üks võti, olgu selleks siis kas primaarne või sekundaarne (väline) võti
  - Kui mõni andmebaasisüsteem ei nõua võtmete otsest esitust, teevad nad seda taustal ikkagi

Andmebaasid - RDBMS

- Relatsioonid
  Suhteid tabelite vahel võib olla mitmeid:
  - Üks-ühele
    Suhteliselt harva kasutusel
  - Üks-mitmele
    Meie näites töötajate tunnitasu kategooriad
  - Mitu-mitmele
    Meie näite puhul Projekti personali tabel, mis sidus Projektid ja töötajad

Andmebaasid - RDBMS – SQL

- Andmebaasi struktuuri kirjeldamine kasutades DDL-i (Data Definition Language) – selleks on näiteks SQL

```
create table Projekt (
  ProjNo    Integer NOT NULL,
  ProjNimi  VarChar(20) NOT NULL DEFAULT '',

  PRIMARY KEY(ProjNo)
)

create table ProjectStaff (
  ProjNo    Integer NOT NULL,
  EmpNo     Integer NOT NULL,

  FOREIGN KEY(ProjID) REFERENCES Projekt(ProjNo),
  FOREIGN KEY(EmpNo) REFERENCES Employee(EmpNo)
)
```

Andmebaasid - RDBMS – SQL

- Indeksid
  - Igale mitte-võtmele võib lisada indeksi
  - Indekseid kasutatakse andmete poole pöördumise kiirendamiseks
  - Indeksite liigne hulk võib aga aeglustada andmete sisestust

```
create table Projekt (
  ProjNo    Integer NOT NULL,
  ProjNimi  VarChar(20) NOT NULL DEFAULT '',

  PRIMARY KEY(ProjNo)
)

create index IdxProjNimi ON Projekt(ProjNimi);
```

Andmebaasid - RDBMS – SQL

- Andmetega manipuleerimiseks on DML (Data Manipulation Language)
- SQL sisaldab ka selle taoks mõeldud käske
  - Andmete sisestamine
  - Andmete kustutamine
  - Andmete muutmine
- Lõpuks on ka spetsiaalne päringukeel, mis lubab andmeid mitmetest tabelitest kokku võtta ning sorteerida, grupeerida jne

Andmebaasid - RDBMS – SQL

- Andmete lisamine - `insert`

```
insert into Projekt (ProjNo, ProjNimi)
  values (1023, ”Web Site 1”);
insert into Projekt (ProjNo, ProjNimi)
  values (1555, ”Web X”);

insert into Employee (EmpNo, EmpName, SalaryCategory)
  values (10, ”Siim Susi”, ”A”);
insert into Employee (EmpNo, EmpName, SalaryCategory)
  values (11, ”Marju Mesi”, ”B”);
...

insert into ProjektStaff (ProjNo, EmpNo)
  values (1023, 10);
insert into ProjektStaff (ProjNo, EmpNo)
  values (1023, 11);
...
```

Andmebaasid - RDBMS – SQL

- Andmete parandamine – `update`

```
// Muudame kõigi B-grupi tunnitasuga töötajatel selle A-grupiks
update Employee set (SalaryCategory = ”A”)
  where (SalaryCategory = ”B”)

// Muudame töötaja 10 projekti 1024-ks.
// NB! Sellist projekti pole
update ProjektStaff set (ProjNo = 1024)
  where (EmpNo = 10)
```

- Andmete kustutamine - `delete`

```
// Kustutame kõik projektid
delete from Projekt;

// Kustutame kõik Siimud töötajate hulgast
delete from Employee where (EmpName like ”Siim%”)
```

Andmebaasid - RDBMS – SQL

- Andmete pärimine - `select`

```
// Kõik Projekt tabeli elemendid
select * from Projekt;

// Tagastab projektide arvu
select sum(*) from Projekt;

// Tagastab töötaja numbri ja nime
select EmpNo, EmpName from Employee;

// Tagastab kõik projektiga 1023 seotud töötajate numbrid
select EmpNo from ProjectStaff where (ProjID = 1023);

// Trükib projekti ja töötaja nime paarid
select Proj.ProjName, Emp.EmpName
  from ProjectStaff PS, Projekt Proj, Employee Emp
  where ((PS.ProjNo = Proj.ProjNo) and
         (PS.EmpNo = Emp.EmpNo));
```

Andmebaasid - RDBMS – SQL

- SQL omab väga laialdasi võimalusi andmebaasi ning selles olevate andmetega ümber käimiseks
- Igale tabelile saab ära määrata ka kasutajate õigused – kes ja milliseid käske sooritada saavad

Andmebaasid – RDBMS – PHP

```
$dbhost = ”localhost”;
$dbuser = ”php”;
$dbpassword = ”php”;

mysql_connect($dbhost, $dbuser, $dbpassword);
mysql_select_db($database);
$query = stripSlashes($query);
$result = mysql_query($query);

echo(”Results of query: $query<BR>\n”);

if ($result == 0):
  echo(”<B>Error ” . mysql_errno() . ”: ” . mysql_error() . ”</B>”);
elseif (mysql_num_rows($result) == 0):
  echo(”<B>Query executed successfully</B>”);
else:
  echo(”<TABLE><TR>”);
  for ($i = 0; $i < mysql_num_fields($result); $i++) {
    echo(”<TH>” . mysql_field_name($result, $i) . ”</TH>”);
  }
  for ($i = 0; $i < mysql_num_rows($result); $i++) {
    echo(”<TR>”);
    $row_array = mysql_fetch_row($result);
    for ($j = 0; $j < mysql_num_fields($result); $j++) {
      echo(”<TD>” . $row_array[$j] . ”</TD>”);
    echo(”</TR>”);
  }
endif
```

Andmebaasid - RDBMS – PHP

- Ühendamine andmebaasiga

  Kui ühendumine õnnestus, siis tagasi ühenduse identifikaator, muidu false

```
int mysql_connect(string [hostname [:port]], string [username],
string [password]);
```

- Jääva ühenduse loomiseks

```
int mysql_pconnect(string [hostname [:port]], string [username],
string [password]);
```

  Selle funktsiooni erinevus eelmisest seisneb selles, et ühendust ei katkestata ei skripti töö lõppedes, ega mysql_close() väljakutsel. Kui ühendus tehakse, vaatab skripti parser kas juba leidub selliste parameetritega ühendus – kui leidub, võetakse see; kui ei, luuakse uus

- Ühenduse katkestamine

```
int mysql_close(int [link_identifier]);
```

  Seda ei pea tingimata välja kutsuma – PHP peab ise ka arvet

Andmebaasid - RDBMS – PHP

- Andmebaasi loomine

  Selle käsu asemel võib kasutada ka vastavaid SQL lauseid

```
int mysql_create_db(string name, int [link_identifier]);
```

- Andmebaasi eemaldamine

```
int mysql_drop_db(string name, int [link_identifier]);
```

- Jooksva andmebaasi valik

```
int mysql_select_db(string database_name, int [link_identifier]);
```

Andmebaasid - RDBMS – PHP

- Päringu esitamine

  See käsk võimaldab anda suvalisi SQL-keelseid käske

```
int mysql_query(string query, int [link_identifier]);
```

- Päringu esitamine koos andmebaasi määramisega

```
int mysql_db_query(string name, string query, int [link_identifier]);
```

- Andmebaaside ja tabeli nimistu saamine

```
int mysql_list_dbs(int [link_identifier]);
int mysql_list_tables(string database, int [link_identifier]);
```

  Läbi nimistu käimiseks tuleb kasutada mysql_tablename() funktsiooni

Andmebaasid - RDBMS – PHP

- Ridade arvu teadasaamine

```
int mysql_num_rows(int link_identifier);
```

- Tabeli väljade info tagastamine

```
int mysql_list_fields(string database_name, string table_name, int
[link_identifier]);
```

Tulemusi saab kasutada mysql_field_flags(), _field_len(), _field_name(), _field_type() funktsioonide abil

- Väljade arvu teada saamine

```
int mysql_num_fields(int result_identifier);
```

Andmebaasid - RDBMS – PHP

- Tabeli rea saamine

```
array mysql_fetch_row(int result_identifier);
```

- Terve päringu tagastamine

```
int mysql_fetch_array(int result_identifier, int [result_type]);
```

```
$result = mysql_query("SELECT (...)");
$row = mysql_fetch_array($result);

echo($row["id"] . ", " . $row[0]);
```

- Info vigade kohta

```
int mysql_errno(int [link_identifier]);
string mysql_error(int [link_identifier]);
```

Andmebaasid – RDBMS

- RDBMS andmebaasidel on teatavad piirangud
  - Nõuavad, et me oma andmed viiksime tabeli kujule
  - Nõuavad, et me looksime kaks eraldi mudelit
    - Üks programmis kasutatavate andmestruktuuride jaoks
    - Teine andmebaasi kirjutatava loogika jaoks
  - Võivad olla aeglased, sest meile vajaliku objekti moodustamiseks võib vaja olla andmeid mitmest eri tabelist

Andmebaasid - ODBMS

- Objekt-orienteeritud andmebaasides vaadeldakse igat kirjet kui objekti
- Objektid on lähtekeele jaoks ”omad”, st. mingeid otseseid andmebaasikäske tihti vaja polegi
- Baseerib atribuutidel mis määravad ära, millised objekti muutujad salvestada ja kuidas
- Defineerib oma päringukeele OQL (Object Query Language)
- Eriti kasutusel CAD (Computer-Aided Design) vahendites (N: võiks sellisteks olla AutoCAD, ArchiCAD; Cadence, Synopsys)
- Eksisteerivad peamiselt C++ ja Java jaoks, kuid on ka teisi

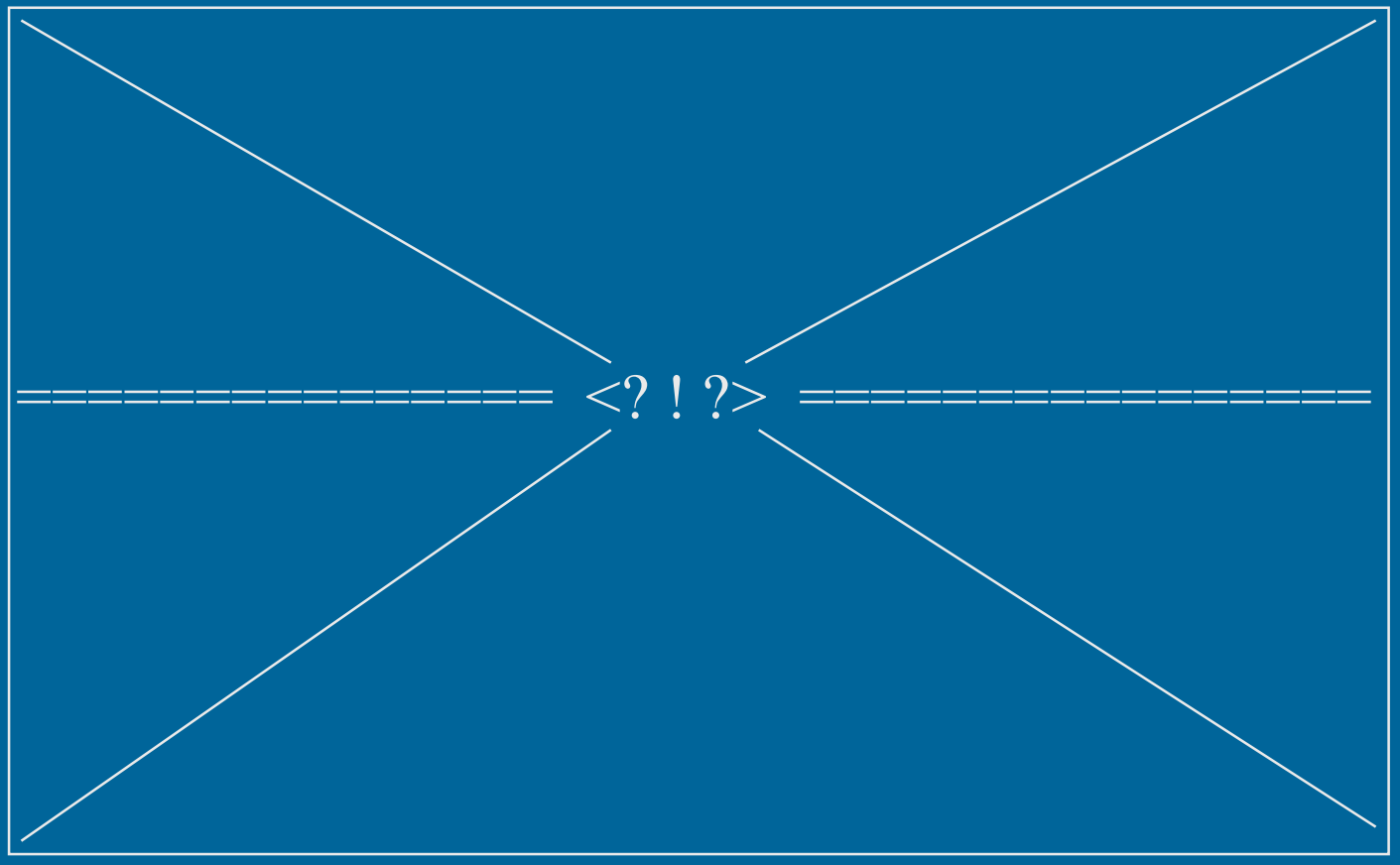
Andmebaasid - ORDBMS

- Objekt-relatsioonilised andmebaasid on RDBMS, millele on juurde lisatud teatav objektide defineerimise võimalus
- Näiteks Oracle, millel võib väli sisaldada teisi tabeleid
- Näieks PostgreSQL, mis võimaldab vaadelda tabeleid kui objekte

<Viited/>

- Andmebaasid
  - ODBMS
    - ObjectStore – http://www.odi.com/odilive/
    - Versant – http://www.versant.com/
    - GemStone – http://www.gemstone.com
  - ORDBMS
    - Oracle – http://www.oracle.com
    - PostgreSQL – http://www.postgresql.org
  - RDBMS
    - Interbase – http://www.interbase.com
  - Teised
    - MySQL – http://www.mysql.com
    - MSQL – http://www.hughes.com.au/

Küsimused?



Andmetöötlus – Kataloogiteenused

- Kataloogiteenused on sisuliselt samuti andmebaasid
- Erinevuseks tavalistest:
  - Hoiavad peamiselt vähemuutuvat infot
  - Orienteeritud päringute efektiivsusele
  - Sarnasus igapäevaste kataloogidega (telefoniraamatud jne)
- Kataloogiteenuste standardid
  - X.500
  - LDAP (Light-weight Directory Access Protocol)
  - Active Directory

Kataloogiteenused

- LDAP võimalused
  - Ideaalsel juhul garanteerib andmete globaalse unikaalsuse
  - Avatud standard
  - Laiendatav
  - Homogeenne andmete hoidla
    - Andmed võivad taustal pesitseda ühes või mitmes andmebaasis (nii samasugustes kui erinevates)
  - Turvaline
    - SASL (Simple Authentication Security Layer)
      - Kerberos
      - CRAM MD5
    - SSL (Secure Socket Layer)
    - TLS (Transport Layer Security)

Kataloogiteenused

- Rääkides LDAPist, räägime kolmest komponendist
  - LDAP andmete orgenisatsioon

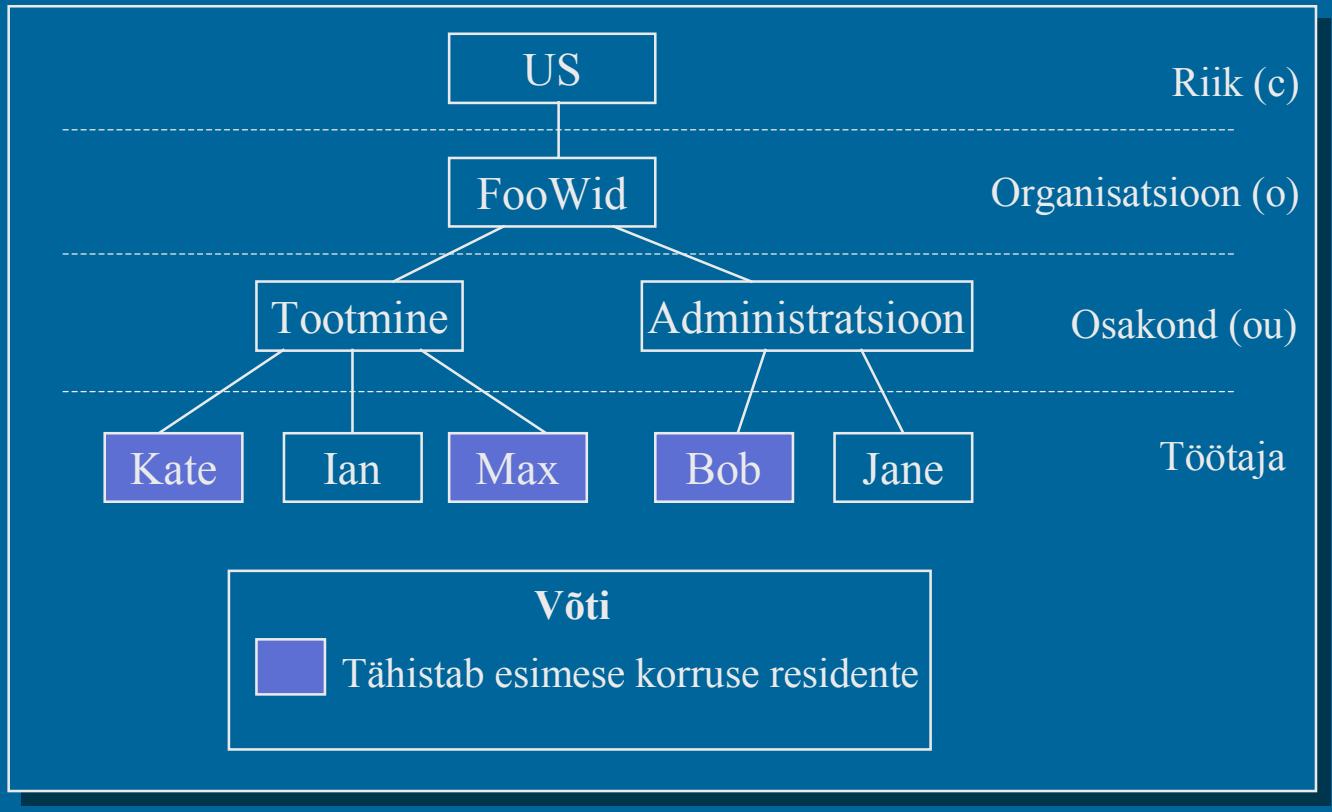
    Kuidas ja kuhu andmed salvestatakse
  - LDAP protokoll

    Ühine keel serveri ja vastava kliendi vahel; LDAP Võimaldab ka serveritevahelist sidepidamist
  - LDAP kliendid

    Kasutajale nähtav osa LDAP tehnoloogiast

Kataloogiteenused

- Andmete organisatsioon



Kataloogiteenused

- Andmete organisatsioon
  - Andmed on esitatud hierarhiliselt
  - Igal tüübil on oma tähis – C = Country; O = Organization; OU = Organizational Unit; CN = Common Name (töötaja nimi)

    CN=Kate Li, OU=Manufacturing, O=FooWid.com, C=US
  - See näide lubab meil globaalsel tasandil unikaalselt ära määrata, millise Kate Li'ga meil tegemist on

Kataloogiteenused

- LDAP mõisted
  - **Entry** – DB mõistes kirje tabelis
  - **Attributes** – DB mõstes kirje väljad (N: CN on sissekande omaniku nimi)
  - **Objects** – DB mõistes tabeli analoog. Nii nagu tabelis on samalaadne info, nii ka objekt esindab mingit samade attribuutidega kooslust
  - **Distinguished Name (DN)** – Nimi, mis unikaalselt määrab ära mingi sissekande (meie Kate Li näide)
  - **Relative Distiguished Name (RDN)** – iga hierarhia tase määrab ära komponendi igas tipus
  - **Directory Information Tree (DIT)** – kogu puu
  - **Schema** – määrab temas sisalduva informatsiooni jaotuse (layout)

Kataloogiteenused

- LDAP vs Database
  - Andmebaasis on tavaliselt igas tabelis vaid unikaalse võtmega kirjed;
    LDAP lubab ühele attribuudile ka mitu väärtust anda (näiteks mitu telefoninumbrit)
  - Kataloogiteenused esitavad informatsiooni hierarhiliselt; lisaks hierarhiale võib neid ka grupeerida (N: esimese korruse inimesed), moodustades niiviisi objektiklassi
  - Objekt on üsna sarnane tabelitele – sama tüüpi objektidel on samad atribuudid. Objekte saab aga ka laiendada objekt-orienteeritud viisil

Kataloogiteenused

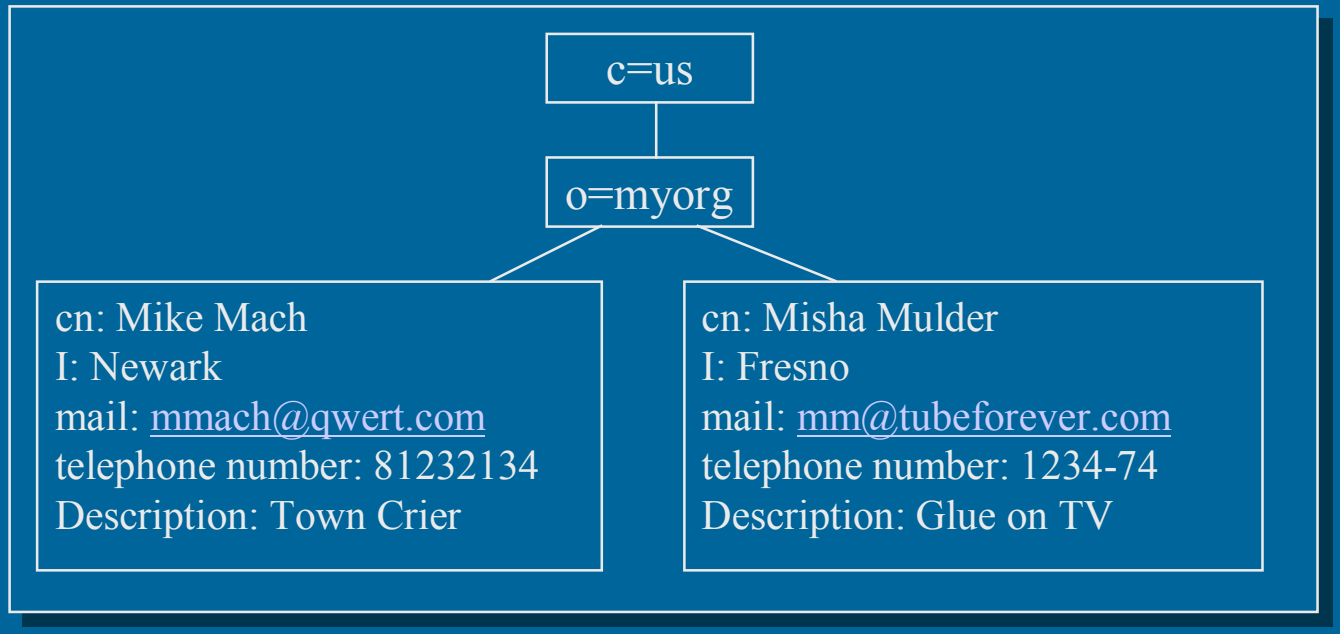
- LDAP operatsioonid
  - Otsing
    - Otsimiseks tuleb määrata kriteerium (otsingu filter), mis on regulaaravaldis
      N:
      CN=*an* - kõik töötajad, millel nimes 'an'
      (&(CN=*a*) (OU=manufacturing)) – kõik töötajad, millel nimes 'a' ja mis asuvad Manufacturing divisionis
  - Lisamine
    - Lisada saab suvalise arvu uusi objekte kui nad vastavad skeemis defineeritud reeglitele. Ette tuleb anda DN, et teada kuhu see objekt lisada
  - Kustutamine
    - Kustutamine on ka üsna otsekohene – vaja on vaja teada objekti DN-i
  - Modifitseerimine
    - Modifitseerida võib suvalise objekti atribuute
- Loomulikult operatsioonid lubatud vaid vastavaid õigusi omavatele isikutele

Kataloogiteenused

- LDAPiga suhtlemiseks võib kasutada
  - LDIF (LDAP Data Interchange Format) faile
  - komadega eraldatud stringe
  - XML dokumendiga DSML (Directory Services Markup Language) formaadis

Kataloogiteenused

- Näiteks on meil järgmine hierarhia



Kataloogiteenused

- Sellele vastav LDIF faili osa

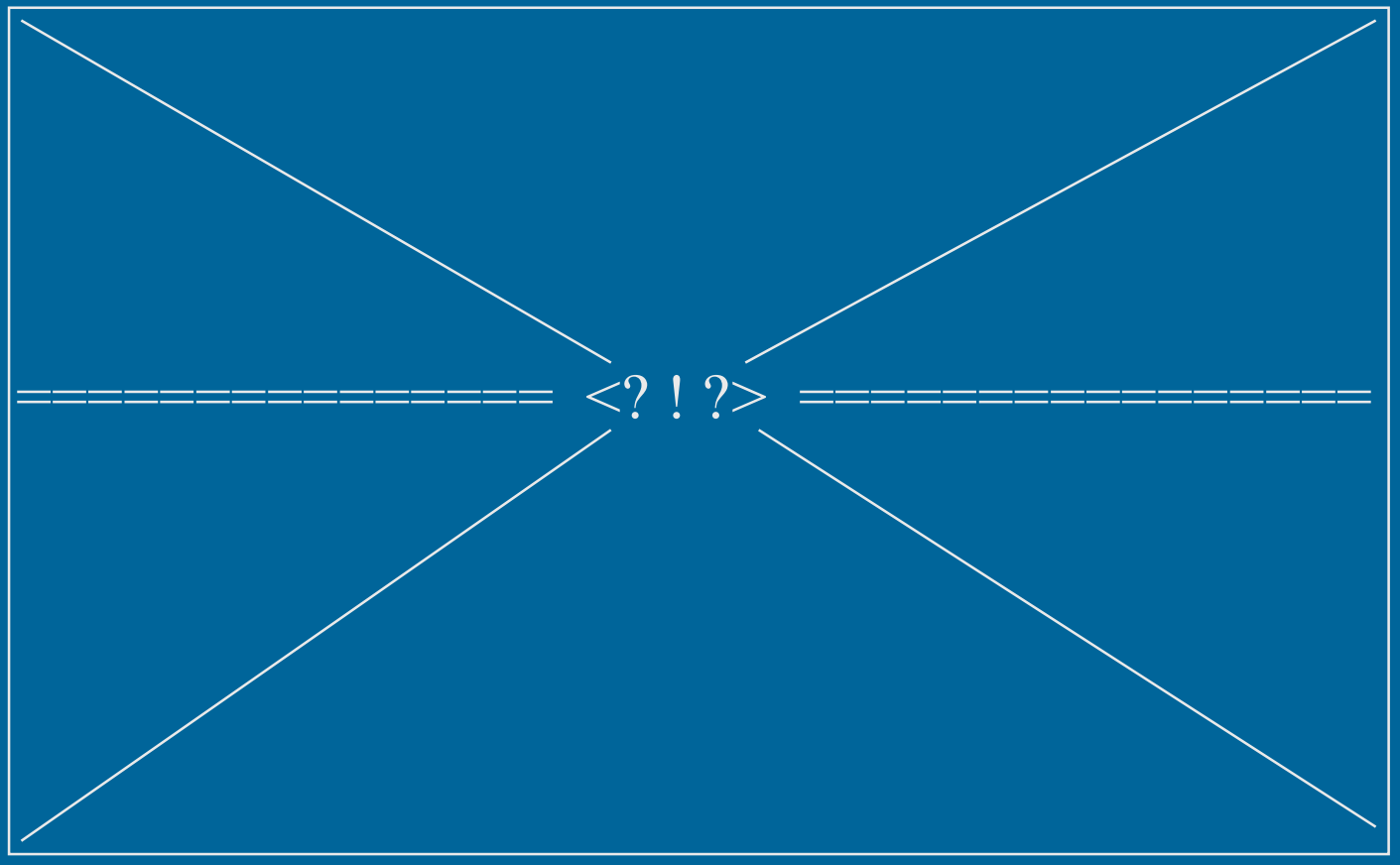
```
dn: o=myorg, c=us
objectclass: top
objectclass: organization
o: myorg
dn: mail=mmach@qwerty.com, o=myorg, c=us
cn: Mike Mach
objectclass: top
objectclass: organizationalPerson
objectclass: inetOrgPerson
l: Newark
mail: mmach@qwerty.com
telephonenumber: 828229-2
Description: Town crier

...
```

<Viited/>

- Kataloogiteenused
  - DSML (Directory Service Markup Language) http://www.dsml.org
  - OpenLDAP Linuxi jaoks http://www.openldap.org
- Andmebaasid ja SQL
  - http://w3.one.net/~jhoffman/sqltut.htm
  - http://www.contrib.andrew.cmu.edu/~shadow/sql.html
  - http://sqlcourse.com/
  - http://www.lap.ttu.ee/mart1n/sql/ (Eesti keeles)
  - http://www.programmingtutorials.com/main.asp
  - http://www.pcslink.com/~ej/dbweb.html
  - http://www.stars.com/Authoring/DB/

Küsimused?



Andmebaaside loomine

- Ülesande püstitus ja realisatsioon lihtsa näite varal
- ERD diagrammid
- Individuaalne töö

Näide ülesandepüstitusest

- Oletame, et meil on tarvis luua rakendus toodete müügiprotsessi haldamiseks
  - Toodete nimekiri
  - Klientide nimekiri
  - Hinnakirjad (mingitele klientidele/klienditüüpidele erihindadega)
  - Tellimuste nimekiri (ia tellimus võib sisaldada suvalist arvu tooteid)
- Luua andmebaasi struktuur selliste andmete hoidmiseks
- Luua veebilihed toodete hinnakirjade vaatamiseks (sõltuvalt kliendist võib olla eri hinnaga)
- Kliendid võivad oma parooliga sisse logida ja nii saavad enda kohta käiva hinnakirja

Näide ülesandepüstitusest

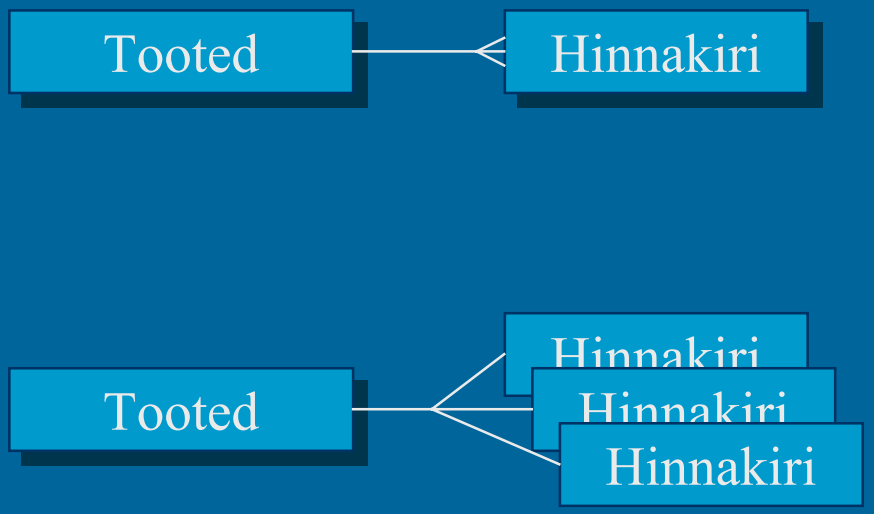
- Meil on vaja järgnevaid andmehoidlaid:
  - Tooted
  - Kliendid
  - Klienditüübid (seotud klienditüübiga)
  - Hinnakirjad (seotud toote ja klienditüübiga)
  - Tellimused (sisaldab palju tooteid ning seotud kliendiga ning tootega)

Näide ülesandepüstitusest (jätkub)

- Andmed
  - Tooted – Nimi, Pilt, Seletus
  - Kliendid – Nimi, Login, Password, Klienditüüp, Telefon, Fax, Riik, Linn, Tänav
  - Klienditüüp – Nimi
  - Hinnakiri – Toode, Klienditüüp, Hind
  - Tellimus – Klient, Staatus, Tellitud tooted
  - Tellitud tooted – Toode, Hind, Kogus

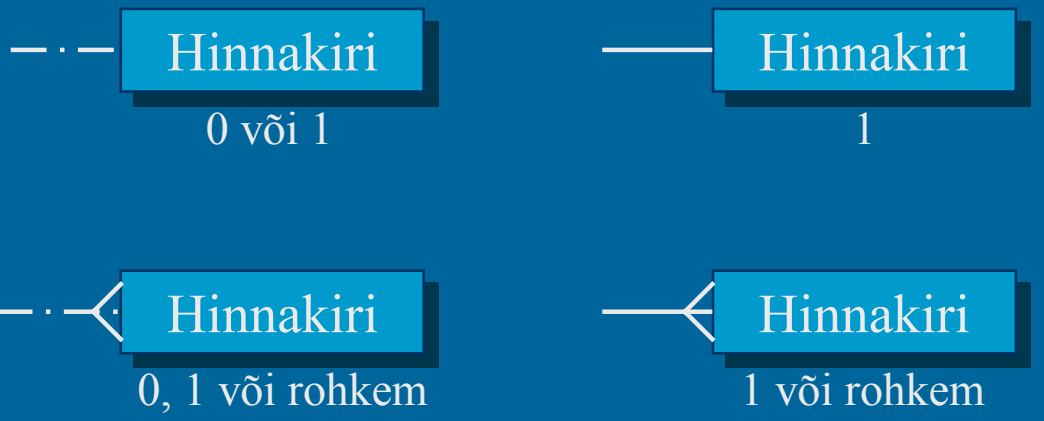
Näide ülesandepüstitusest (jätkub)

- ERD (*Entity Relationship Diagram*)
  - Üks-mitmele relatsioon



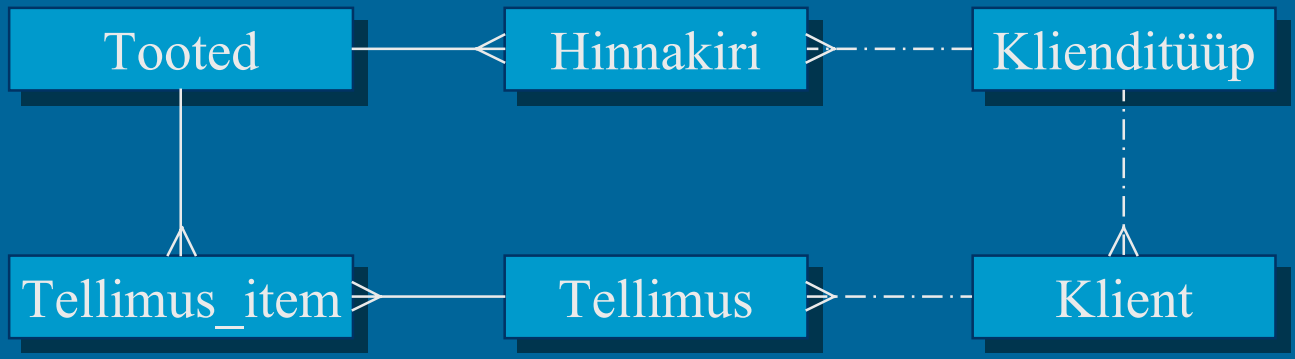
Näide ülesandepüstitusest (jätkub)

- ERD (*Entity Relationship Diagram*)
  - Erinevad relatsioonide vormid



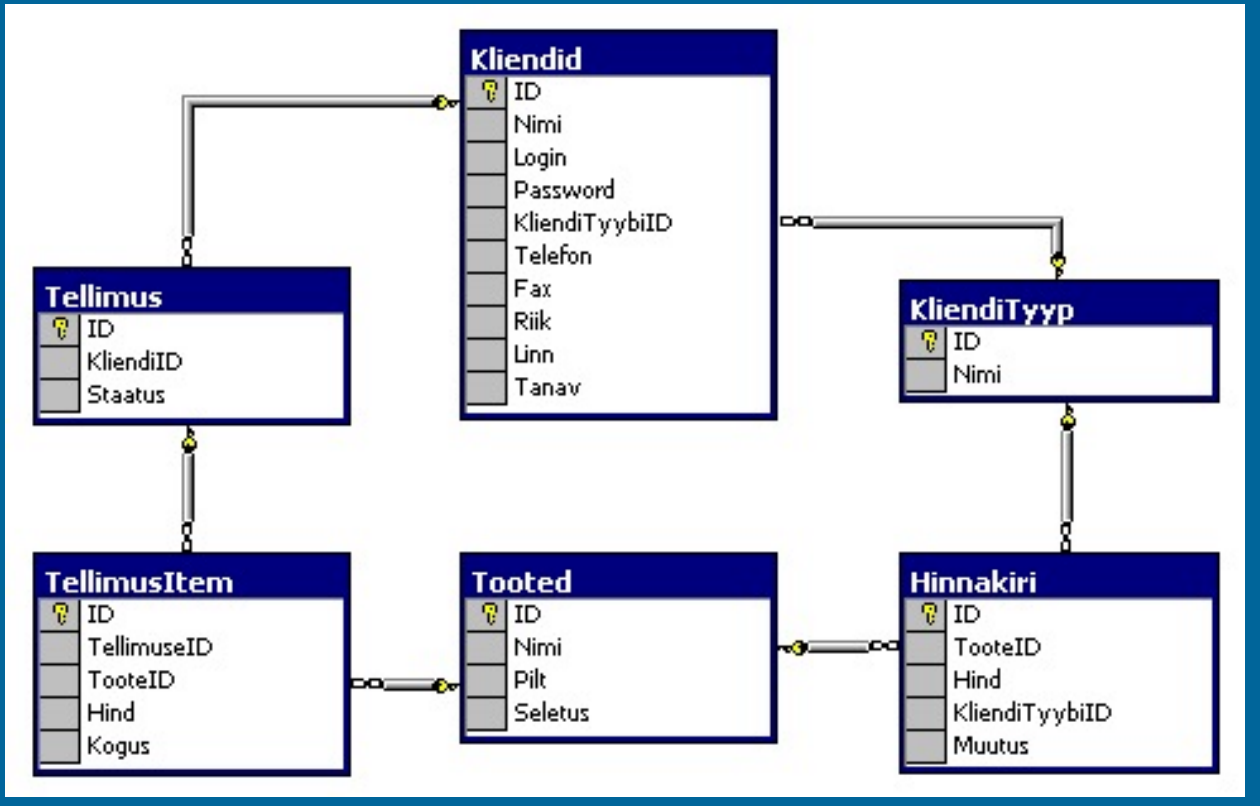
Näide ülesandepüstitusest (jätkub)

- ERD (*Entity Relationship Diagram*)



Näide ülesandepüstitusest (jätkub)

- Andmebaasi diagramm koos tabeli struktuuriga



Näide ülesandepüstitusest (jätkub)

- SQL käsud tabelite loomiseks

```
CREATE TABLE Tooted (
  ID              int           NOT NULL ,
  Nimi            varchar(50)   NULL ,
  Pilt            image         NULL ,
  Seletus         text          NULL

  PRIMARY KEY(ID)
);

CREATE TABLE KliendiTyyp (
  ID              int           NOT NULL,
  Nimi            varchar(50)   NULL,

  PRIMARY KEY(ID)
);
```

Näide ülesandepüstitusest (jätkub)

- SQL käsud tabelite loomiseks

```
CREATE TABLE Hinnakiri (
  ID              int          NOT NULL,
  TooteID          int          NOT NULL,
  Hind            money        NOT NULL    DEFAULT 0,
  KliendiTyybiID  int          NULL        DEFAULT NULL,

  PRIMARY KEY(ID),
  FOREIGN KEY(TooteID) REFERENCES Tooted(ID),
  FOREIGN KEY(KliendiTyybiID) REFERENCES KliendiTyyp(ID)
);
```

Näide ülesandepüstitusest (jätkub)

- SQL käsud tabelite loomiseks

```
CREATE TABLE Kliendid (
  ID              int           NOT NULL ,
  Nimi            varchar(50)   NULL ,
  Login           varchar(50)   NULL ,
  Password        varchar(50)   NULL ,
  KliendiTyybiID  int           NULL ,
  Telefon         varchar(50)   NULL ,
  Fax             varchar(50)   NULL ,
  Riik            varchar(50)   NULL ,
  Linn            varchar(50)   NULL ,
  Tanav           varchar(50)   NULL ,

  PRIMARY KEY(ID),
  FOREIGN KEY(KliendiTyybiID) REFERENCES KliendiTyyp(ID)
);
```

Näide ülesandepüstitusest (jätkub)

- SQL käsud tabelite loomiseks

```
CREATE TABLE Tellimus (
  ID              int           NOT NULL ,
  KliendiID       int           NULL ,
  Staatus         varchar(10)   NULL      DEFAULT 'Uus'

  PRIMARY KEY(ID),
  FOREIGN KEY(KliendiID) REFERENCES Kliendid(ID)
);
```

Näide ülesandepüstitusest (jätkub)

- SQL käsud tabelite loomiseks

```
CREATE TABLE TellimusItem (
  ID              int           NOT NULL ,
  TellimuseID     int           NULL ,
  TooteID         int           NULL ,
  Hind            money         NOT NULL ,
  Kogus           int           NOT NULL

  PRIMARY KEY(ID),
  FOREIGN KEY(TellimuseID) REFERENCES Tellimus(ID),
  FOREIGN KEY(TooteID) REFERENCES Tooted(ID)
);
```

Näide ülesandepüstitusest (jätkub)

```
<?php
  $query = "select Kliendid.Nimi, KliendiTyyp.Nimi from Kliendid,
→    KliendiTyyp where Kliendid.KliendiTyybiID = KliendiTyyp.ID";


  $conn_id = odbc_connect("Tooted", "test", "test");
  $qry_id = odbc_do($conn_id, $query);


  $Fields = odbc_num_fields($qry_id);
  print "<table border='1' width='100%'><tr>";


  // Build Column Headers
  for ($i=1; $i <= $Fields; $i++){
    printf("<th bgcolor='silver'>%s</th>",
           odbc_field_name($qry_id,$i));
  }

  ...
```

Näide ülesandepüstitusest (jätkub)

```
  ...


  // Table Body
  $Outer=0;
  while( odbc_fetch_row( $qry_id )){
    $Outer++;
    print "<tr>";
    for($i=1; $i <= $Fields; $i++){
      printf("<td>%s</td>", odbc_result( $qry_id, $i ));
    }
    print "</tr>";
  }

  print "</table>";
  print "<b> Your request returned $Outer rows!</b>";
  odbc_close($conn_id);

?>
```

Individuaalne töö

- Analüüsida, millised andmed ülesanne vajab
- Vaadata kuidas eri andmed kõige paremini grupeeruvad (nende baasil luuakse hiljem tabelid), pidades silmas 3 normaalkuju nõudeid
- Luua andmebaasi relatsioonide diagramm
- Välja tuua tabeli struktuur (millised andmeväljad ning primaarsed/sekundaarsed võtmed vajalikud on)
- Luua PHP webileh(-t/-ed) põhiandmete vaatamiseks

Individuaalne töö (jätkub)

- Ülesandeid
  - Raamatukogu

    Kajastada tuleb raamatuid, autoreid, asukohta riiulitel, kirjastusi, laenutajaid, laenutusaegu (millal laenutas ja millal peab tagastama)

  - Muusika andmebaas

    Koosneb kogumikust, esitajast, lugudest, meediast (CD, DVD, Kassett), plaadifirmast

  - Kokaraamat

    Toiduained, retseptid, toidu tüübid (maiustud, magustoit, supp, ...)

Individuaalne töö (jätkub)

- Ülesandeid
  - Saatekava

    Jaam, tüüp (TV, raadio, ...), teema, saade, saateaeg

  - Hindamissüsteem

    Aine, õpilane, õpetaja, töö tüüp (arvestus, eksam), tulemus

  - Sõidugraafik

    (Bussi)liin, peatused, kellaajad

  - Ja nii edasi

<Viited/>

- SQL & Database design
  - http://w3.one.net/~jhoffman/sqltut.htm
  - http://www.sqlmag.com (SQL Magazine)
  - http://databases.about.com/compute/databases/cs/specificproducts/
- ERD
  - http://www.arbortext.com/wp.html
  - http://wwwflag.wr.usgs.gov/GLIMS/erd_desc.html (kasutusnäide)
  - http://www.embarcadero.com/products/Design/erdatasheet.htm (ER Studio)
  - http://www.cai.com/products/alm/erwin.htm (ER*Win*)
  - http://www.datanamic.com/dezign/index.html (DeZign for Databases)

Küsimused?