

Perusohjeita MySQL-tietokannan käyttöön Linux-palvelimella

Käyttöönotto ja peruskomennot

Käytetään ainakin ensimmäisissä harjoituksissa valmista MySQL-serveriä osoitteessa `jukkajauhiainen.ipt.oamk.fi`

Jos ollaan koulun verkon ulkopuolella, täytyy ennen ssh-yhteyden muodostamista muodostaa VPN-yhteys. Ohjeet löytyvät osoitteesta <https://it.oamk.fi/1468>

Edellä mainittuun ipt-palvelimaan pääsee kiinni Windowsissa PuTTY:llä tai Linuxissa SSH:lla. Varmuuden vuoksi PuTTY:yn kannattaa määritellä palvelimen pingaus minuutin välein, muuten palvelin saattaa katkaista pääteyhteyden.

Esimerkki SSH-yhteyden muodostamisesta Linuxissa:

```
jjauhiiai@ubu ~ $ ssh t3blaa00@jukkajauhiainen.ipt.oamk.fi
t3blaa00@jukkajauhiainen.ipt.oamk.fi's password:
Welcome to Ubuntu 12.04.5 LTS (GNU/Linux 3.2.0-70-virtual i686)

 * Documentation:  https://help.ubuntu.com/
New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Oct 14 14:22:14 2014 from addr-193-167-102-
102.dyn.oamk.fi
t3blaa00@jukkajauhiainen:~$
```

Palvelimelle on tehty jokaiselle kurssille ilmoittautuneelle opiskelijalle käyttäjätunnus, joka on saman niminen kuin ns. students-tunnus (jolla kirjaudutte joka paikkaan, siis muotoa t3blaa00). Ensimmäisellä kirjautumiskerralla salasana on sukunimi pienillä kirjaimilla. Kirjautumisen yhteydessä se vaihdetaan automaattisesti. Tämä tunnus toimii siis vain tällä palvelimella, eli se ei ole ”oikea” students-tunnukseenne.

MySQL:ään kirjautuminen tapahtuu samalla käyttäjätunnuksella. HUOM: Linuxin käyttäjätunnus ja

Tietokannat Osa 2: SQL:n peruskomentoja

MySQL:n käyttäjätunnus ovat TÄYSIN TOISISTAAN ERILLISIÄ ! Toisin sanoen MySQL:ään on mahdollista määritellä useita eri käyttäjätunnuksia yhdeltä linux-tunnukselta.

Kun on kirjaututtu Linuxiin, MySQL käynnistyy komennolla:

```
mysql -u käyttäjätunnus -p
```

käynnistuksen yhteydessä ohjelma kysyy salasanan. Ensimmäisellä kirjautumisella salasana on tyhjä. Tässä kirjautuu käyttäjä t3blaa00:

```
t3blaa00@jukkajauhiainen:~$ mysql -u t3blaa00 -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 17761
Server version: 5.5.31-0ubuntu0.12.04.1 (Ubuntu)

Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

MySQL:n salasanan (jonka siis ei tarvitse olla sama kuin Linux-salasana) vaihto (ei pakollinen, voi jättää tyhjäksiin):

```
mysql> SET PASSWORD = PASSWORD('blaablaa');
Query OK, 0 rows affected (0.00 sec)
```

Seuraava komento näyttää, mitä tietokantoja on asennettu. Komennot voi kirjoittaa joko isoilla tai pienillä kirjaimilla. Yleensä varatut sanat kirjoitetaan isolla. Huomaa puolipiste lopussa.

```
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| employees |
| sakila |
| t3blaa00 |
+-----+
4 rows in set (0.00 sec)
```

Jokaiselle käyttäjälle on luotu oma tietokanta, jossa voi vapaasti puuhastella, eli siihen voi lisätä ja poistaa tauluja jne. Kannan nimi on sama kuin käyttäjätunnus. Toistaiseksi siellä ei ole mitään.

Tietokannat employees ja sakila on MySQL:n esimerkkietokantoja, joita tullaan käyttämään SQL-

Tietokannat Osa 2: SQL:n peruskomentoja

kielen opetuksessa. Niihin teillä ei ole kuin peruskäyttöoikeudet, eli voitte lisätä ja poistaa tietoa olemassa oleviin tauluihin, mutta uusia tauluja ei voi tehdä eikä taulujen yhteyksiä muuttaa.

Information_schema sisältää yleistä tietoa järjestelmästä. Katsotaan vähän mitä siellä on. Valitaan ensin tietokanta:

```
mysql> USE information_schema;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SHOW TABLES;
```

Tähän ei kannata liittää komennon tulostusta, katsotaan harjoituksissa.

Komento DESCRIBE tai DESC näyttää yhden taulun rakenteen, eli mitä kenttiä (Field) se sisältää ja mikä kentän tietotyyppi on. Esimerkiksi tietokanta information_schema sisältää taulun ENGINES (jonka sisältönä on käytössä olevat tietokantamoottorit, siitä myöhemmin), jonka rakenne näyttää tältä:

```
mysql> DESCRIBE ENGINES;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| ENGINE         | varchar(64)   | NO   |     |          |       |
| SUPPORT        | varchar(8)    | NO   |     |          |       |
| COMMENT        | varchar(80)   | NO   |     |          |       |
| TRANSACTIONS   | varchar(3)    | YES  |     | NULL    |       |
| XA             | varchar(3)    | YES  |     | NULL    |       |
| SAVEPOINTS     | varchar(3)    | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

Katsotaanpa heti saman tien, miten tehdään SQL-kysely ENGINES-tiluun. Seuraava komento näyttää kaikki ENGINES-tiluun tallennetut tietueet:

```
mysql> SELECT * FROM ENGINES;
+-----+-----+-----+-----+-----+-----+
| ENGINE          | SUPPORT | COMMENT                                     | TRANSACTIONS | XA | SAVEPOINTS |
+-----+-----+-----+-----+-----+-----+
| MyISAM          | YES     | MyISAM storage engine                     | NO           | NO | NO         |
| CSV             | YES     | CSV storage engine                       | NO           | NO | NO         |
| MEMORY          | YES     | Hash based, stored in memory, useful for temporary tables | NO           | NO | NO         |
| BLACKHOLE       | YES     | /dev/null storage engine (anything you write to it disappears) | NO           | NO | NO         |
| FEDERATED       | NO      | Federated MySQL storage engine           | NULL         | NULL | NULL       |
| MRG_MYISAM      | YES     | Collection of identical MyISAM tables     | NO           | NO | NO         |
| ARCHIVE         | YES     | Archive storage engine                   | NO           | NO | NO         |
| InnoDB          | DEFAULT | Supports transactions, row-level locking, and foreign keys | YES          | YES | YES        |
| PERFORMANCE_SCHEMA | YES     | Performance Schema                       | NO           | NO | NO         |
+-----+-----+-----+-----+-----+-----+
```

Muutama huomautus vielä. Tässä vaiheessa opintoja teillä ei ole oikeuksia luoda uusia tietokantoja, mutta jos olisi se tapahtuisi komennolla

Tietokannat Osa 2: SQL:n peruskomentoja

```
mysql> CREATE DATABASE foobar;  
ERROR 1044 (42000): Access denied for user 't3blaa00'@'localhost' to database  
'foobar'
```

Jos komennon lopusta puuttuu puolipiste, jää MySQL:n komentotulkki odottamaan, mitä käyttäjä seuraavaksi syöttää. Tämä on itse asiassa erittäin käyttökelpoinen ominaisuus, joka mahdollistaa pitempien lauseiden jakamisen useammalle riville, mutta on ensikertalaiselle todella hämmentävä:

```
mysql> SHOW DATABASES  
->  
->  
-> ?  
-> quit  
-> end  
-> bye  
-> mitvit ?  
-> ;  
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that  
corresponds to your MySQL server version for the right syntax to use near '?  
quit  
end  
bye  
mitvit ?'
```

Eli takaisin komentotulkin perustilaan pääsee kirjoittamalla puolipisteen tyhjälle riville.

Ohjelman lopetus tapahtuu kirjoittamalla komentoriville **exit**.

```
mysql> exit  
Bye  
t3blaa00@jukkajauhiainen:~$
```

Nyt ollaan siis takaisin linuxin komentotilassa.

Lisätietoa käytössä olevista komennoista löytyy MySQL Reference Manual:ista, osoitteessa
<http://dev.mysql.com/doc/refman/5.5/en/index.html>

HUOM: MySQL:stä on eri versioita, mutta Ubuntun jakelupaketissa tulee versio 5.5.

Tietokannat Osa 2: SQL:n peruskomentoja

Tietokannan luonti ja tietojen syöttö, poisto, muuttaminen jne

Tehdään MySQL-tutorialista löytyvä lemmikkieläintietokanta **pet**.

<http://dev.mysql.com/doc/refman/5.0/en/creating-tables.html>

Tietokantataulu pitää kirjata lemmikkien nimistä, omistajista, lajista, sukupuolesta sekä syntymä- ja kuolinajoista.

Valitaan ensin tietokanta, johon taulut halutaan luoda. Ainoa vaihtoehto teillä on tietokanta, jonka nimi on sama kuin käyttäjätunnus.

```
mysql> USE t3blaa00;
mysql> CREATE TABLE pet (name VARCHAR(20), owner VARCHAR(20), species
VARCHAR(20), sex CHAR(1), birth DATE, death DATE);
Query OK, 0 rows affected (0.10 sec)
mysql> SHOW TABLES;
+-----+
| Tables_in_t3blaa00 |
+-----+
| pet                  |
+-----+
1 row in set (0.00 sec)

mysql> DESCRIBE pet;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| name  | varchar(20)   | YES  |     | NULL    |       |
| owner | varchar(20)   | YES  |     | NULL    |       |
| species | varchar(20)   | YES  |     | NULL    |       |
| sex   | char(1)       | YES  |     | NULL    |       |
| birth | date          | YES  |     | NULL    |       |
| death | date          | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.01 sec)
```

Tietueen syöttö: INSERT INTO

Täytetään taulu seuraavilla tiedoilla:

name	owner	species	sex	birth	death
Fluffy	Harold	cat	f	1993-02-04	
Claws	Gwen	cat	m	1994-03-17	
Buffy	Harold	dog	f	1989-05-13	
Fang	Benny	dog	m	1990-08-27	
Bowser	Diane	dog	m	1979-08-31	1995-07-29
Chirpy	Gwen	bird	f	1998-09-11	
Whistler	Gwen	bird		1997-12-09	
Slim	Benny	snake	m	1996-04-29	

Tietokannat Osa 2: SQL:n peruskomentoja

```
mysql> INSERT INTO pet VALUES('Fluffy','Harold','cat','f','1993-02-04',NULL);
```

Ja samalla periaatteella muut rivit. MySQL Tutorialissa kikkaillaan tiedot tauluun tekstitiedostosta. Ei lähdetä sitä nyt pohtimaan vaan syötetään käsin :)

Tarkistetaan lopuksi, mitä tauluun on syötetty.

```
mysql> select * FROM pet;
```

name	owner	species	sex	birth	death
Puffball	Diane	hamster	f	1999-03-30	NULL
Fluffy	Harold	cat	f	1993-02-04	NULL
Claws	Gwen	cat	m	1994-03-17	NULL
Buffy	Harold	dog	f	1989-05-13	NULL
Fang	Benny	dog	m	1990-08-27	NULL
Bowser	Diane	dog	m	1979-08-31	1995-07-29
Chirpy	Gwen	bird	f	1998-09-11	NULL
Whistler	Gwen	bird	NULL	1997-12-09	NULL
Slim	Benny	snake	m	1996-04-29	NULL

9 rows in set (0.00 sec)

Tietueen poistaminen: DELETE FROM

```
mysql> DELETE FROM pet where name='Puffball';
```

Query OK, 1 row affected (0.03 sec)

```
mysql> select * FROM pet;
```

name	owner	species	sex	birth	death
Fluffy	Harold	cat	f	1993-02-04	NULL
Claws	Gwen	cat	m	1994-03-17	NULL
Buffy	Harold	dog	f	1989-05-13	NULL
Fang	Benny	dog	m	1990-08-27	NULL
Bowser	Diane	dog	m	1979-08-31	1995-07-29
Chirpy	Gwen	bird	f	1998-09-11	NULL
Whistler	Gwen	bird	NULL	1997-12-09	NULL
Slim	Benny	snake	m	1996-04-29	NULL

8 rows in set (0.00 sec)

Tietueen tietojen muuttaminen: UPDATE

```
mysql> UPDATE pet SET birth = '1989-08-31' WHERE name = 'Bowser';
```

Query OK, 1 row affected (0.02 sec)

Rows matched: 1 Changed: 1 Warnings: 0

Tietokannat Osa 2: SQL:n peruskomentoja

Tiedonhaku: SELECT

Katsotaan muutama esimerkki, miten taulusta voidaan hakea haluttuja tietoja käyttäen SELECT-lausetta. Käytössä on siis SQL-kyselykieli (Structured Query Language).

```
mysql> SELECT * FROM pet WHERE name = 'Bowser';
```

name	owner	species	sex	birth	death
Bowser	Diane	dog	m	1989-08-31	1995-07-29

```
mysql> SELECT * FROM pet WHERE birth >= '1998-1-1';
```

name	owner	species	sex	birth	death
Chirpy	Gwen	bird	f	1998-09-11	NULL

```
mysql> SELECT * FROM pet WHERE species = 'snake' OR species = 'bird';
```

name	owner	species	sex	birth	death
Chirpy	Gwen	bird	f	1998-09-11	NULL
Whistler	Gwen	bird	NULL	1997-12-09	NULL
Slim	Benny	snake	m	1996-04-29	NULL

```
mysql> SELECT * FROM pet WHERE (species = 'cat' AND sex = 'm')
-> OR (species = 'dog' AND sex = 'f');
```

name	owner	species	sex	birth	death
Claws	Gwen	cat	m	1994-03-17	NULL
Buffy	Harold	dog	f	1989-05-13	NULL

```
mysql> SELECT name, birth FROM pet;
```

name	birth
Fluffy	1993-02-04
Claws	1994-03-17
Buffy	1989-05-13
Fang	1990-08-27
Bowser	1989-08-31
Chirpy	1998-09-11
Whistler	1997-12-09
Slim	1996-04-29

```
mysql> SELECT owner FROM pet;
```

Tietokannat Osa 2: SQL:n peruskomentoja

```
+-----+
| owner |
+-----+
| Harold |
| Gwen  |
| Harold |
| Benny  |
| Diane  |
| Gwen   |
| Gwen   |
| Benny  |
+-----+
```

DISTINCT-lauseen avulla ehdon täyttävä tieto tulostuu vain kerran:

```
mysql> SELECT DISTINCT owner FROM pet;
```

```
+-----+
| owner |
+-----+
| Harold |
| Gwen  |
| Benny  |
| Diane  |
+-----+
```

Lajittelu: ORDER BY

Lajitellaan syntymäajan mukaiseen kasvavaan järjestykseen. Alenevaan järjestykseen lajittelu saadaan lisäämällä määre DESC (engl. descending):

```
mysql> SELECT name, birth FROM pet ORDER BY birth;
```

```
+-----+-----+
| name   | birth   |
+-----+-----+
| Buffy  | 1989-05-13 |
| Bowser | 1989-08-31 |
| Fang   | 1990-08-27 |
| Fluffy | 1993-02-04 |
| Claws  | 1994-03-17 |
| Slim   | 1996-04-29 |
| Whistler | 1997-12-09 |
| Chirpy | 1998-09-11 |
+-----+-----+
```

```
mysql> SELECT name, birth FROM pet ORDER BY birth DESC;
```

```
+-----+-----+
| name   | birth   |
+-----+-----+
| Chirpy | 1998-09-11 |
| Whistler | 1997-12-09 |
| Slim   | 1996-04-29 |
| Claws  | 1994-03-17 |
+-----+-----+
```


Tietokannat Osa 2: SQL:n peruskomentoja

Fluffy	1993-02-04	
Fang	1990-08-27	
Bowser	1989-08-31	
Buffy	1989-05-13	
+-----+	+-----+	+

Päivämäärän avulla laskeminen: TIMESTAMPDIFF, YEAR(), MONTH(), ja DAYOFMONTH().MONTH()

Lasketaan lemmikkien ikä vuosina. CURDATE() on MySQL:n valmisfunktio, joka palauttaa tämän päivän päivämäärän:

```
mysql> SELECT CURDATE();
```

+-----+	
CURDATE()	
+-----+	
2014-08-29	
+-----+	

```
mysql> SELECT name, birth, CURDATE(), TIMESTAMPDIFF(YEAR,birth,CURDATE()) AS age
FROM pet;
```

+-----+	+-----+	+-----+	+-----+
name	birth	CURDATE()	age
+-----+	+-----+	+-----+	+-----+
Fluffy	1993-02-04	2014-08-29	21
Claws	1994-03-17	2014-08-29	20
Buffy	1989-05-13	2014-08-29	25
Fang	1990-08-27	2014-08-29	24
Bowser	1989-08-31	2014-08-29	24
Chirpy	1998-09-11	2014-08-29	15
Whistler	1997-12-09	2014-08-29	16
Slim	1996-04-29	2014-08-29	18
+-----+	+-----+	+-----+	+-----+

Edellisessä lauseessa on toinen uusi avainsana **AS**. Sen avulla voidaan luoda uusi kenttä tulostukseen (tässä **age**). Toisin sanoen funktio TIMESTAMPDIFF ottaa syötteen tiedon siitä, missä muodossa tulos lasketaan (tässä vuosina, **YEAR**), sitten päivämäärät joiden välinen erotus lasketaan (vuosina). Tulos tallentuu uuteen väliaikaiseen kenttään **age**. Tietokannan rakenteeseen

Tietokannat Osa 2: SQL:n peruskomentoja

sillä ei ole vaikutusta (pet-tauluun ei synny uutta age-kenttää).

Pari esimerkkiä, jossa edellinen lause on yhdistetty lajitteluun:

```
mysql> SELECT name, birth, CURDATE(),TIMESTAMPDIFF(YEAR,birth,CURDATE()) AS age
FROM pet ORDER BY name;
```

name	birth	CURDATE()	age
Bowser	1989-08-31	2014-08-29	24
Buffy	1989-05-13	2014-08-29	25
Chirpy	1998-09-11	2014-08-29	15
Claws	1994-03-17	2014-08-29	20
Fang	1990-08-27	2014-08-29	24
Fluffy	1993-02-04	2014-08-29	21
Slim	1996-04-29	2014-08-29	18
Whistler	1997-12-09	2014-08-29	16

```
mysql> SELECT name, birth, CURDATE(),TIMESTAMPDIFF(YEAR,birth,CURDATE()) AS age
FROM pet ORDER BY age;
```

name	birth	CURDATE()	age
Chirpy	1998-09-11	2014-08-29	15
Whistler	1997-12-09	2014-08-29	16
Slim	1996-04-29	2014-08-29	18
Claws	1994-03-17	2014-08-29	20
Fluffy	1993-02-04	2014-08-29	21
Fang	1990-08-27	2014-08-29	24
Bowser	1989-08-31	2014-08-29	24
Buffy	1989-05-13	2014-08-29	25

Funktio MONTH kaivaa päivämäärästä kuukauden kokonaislukuna:

```
mysql> SELECT name, birth, MONTH(birth) FROM pet;
```

name	birth	MONTH(birth)
Fluffy	1993-02-04	2
Claws	1994-03-17	3
Buffy	1989-05-13	5
Fang	1990-08-27	8
Bowser	1989-08-31	8
Chirpy	1998-09-11	9
Whistler	1997-12-09	12
Slim	1996-04-29	4

Miten saataisiin selville, kenellä lemmikillä on syntymäpäivä ensi kuussa ?

Haetaan MONTH-funktiolla CURDATE:sta tämän hetkinen kuukausi kokonaislukuna (tätä kirjoitettaessa 8). Asetetaan (SET) se oman itse määritellyn muuttujan (*user-defined variable*)

Tietokannat Osa 2: SQL:n peruskomentoja

@kuu arvoksi. Muuttuja tunnustetaan @-merkillä.

```
SET @kuu=MONTH(CURDATE());
```

Tarkistetaan vielä, mikä muuttujan arvo on:

```
mysql> select @kuu;
+-----+
| @kuu |
+-----+
|     8 |
+-----+
```

Muuttujan arvoa voi käyttää seuraavassa SQL-lauseessa.

```
SELECT name, birth FROM pet WHERE MONTH(birth) = @kuu + 1;
+-----+-----+
| name   | birth      |
+-----+-----+
| Chirpy | 1998-09-11 |
+-----+-----+
```

Palataan muuttujiin tarkemmin myöhemmin kurssilla.

NULL-arvo ja sen testaaminen

NULL on tyhjä tieto. Se **ei ole numeroarvo 0**. Tässä esimerkissä niillä lemmikeillä, jotka ovat hengissä, ei ole kuolinaikaa. Seuraava lause etsii kuolleet lemmikit ja ilmoittaa niiden iän vuosina kuolinhetkellä.

```
mysql> SELECT name, birth, death, TIMESTAMPDIFF(YEAR,birth,death) AS age FROM
pet WHERE death IS NOT NULL ORDER BY age;
+-----+-----+-----+-----+
| name   | birth      | death      | age |
+-----+-----+-----+-----+
| Bowser | 1989-08-31 | 1995-07-29 |    5 |
+-----+-----+-----+-----+
```

Kyselyssä on käytetty avainsanaa NOT NULL. Toisin sanoen tulostetaan niiden tietueiden tiedot, joissa kenttä death **ei ole tyhjä** (eli kuolinaika on olemassa).

Tietokannat Osa 2: SQL:n peruskomentoja

Aggregate functions eli koontifunktiot

Tässä on lueteltu MySQL:n valmisfunktiot. Funktio ottaa syötteen parametreja ja palauttaa return-lauseella yhden arvon. Sitä voidaan käyttää SQL-lauseiden sisällä. Toinen samantyyppinen rakenne on aliohjelma (stored procedure), johon palataan myöhemmin. Valmisfunktioiden lisäksi on mahdollista kirjoittaa omia funktioita.

Osoitteesta <http://dev.mysql.com/doc/refman/5.0/en/group-by-functions.html>

Name	Description
<u>AVG()</u>	Return the average value of the argument
<u>BIT_AND()</u>	Return bitwise and
<u>BIT_OR()</u>	Return bitwise or
<u>BIT_XOR()</u>	Return bitwise xor
<u>COUNT(DISTINCT)</u>	Return the count of a number of different values
<u>COUNT()</u>	Return a count of the number of rows returned
<u>GROUP_CONCAT()</u>	Return a concatenated string
<u>MAX()</u>	Return the maximum value
<u>MIN()</u>	Return the minimum value
<u>STD()</u>	Return the population standard deviation
<u>STDDEV_POP()</u>	Return the population standard deviation
<u>STDDEV_SAMP()</u>	Return the sample standard deviation
<u>STDDEV()</u>	Return the population standard deviation
<u>SUM()</u>	Return the sum
<u>VAR_POP()</u>	Return the population standard variance
<u>VAR_SAMP()</u>	Return the sample variance
<u>VARIANCE()</u>	Return the population standard variance

Tietokannat Osa 2: SQL:n peruskomentoja

Ei tässä lähetä kaikkia käymään läpi. Muutama esimerkki.

COUNT laskee lukumääriä. AVG laskee keskiarvon numeerista tietotyyppiä olevasta sarakkeesta.

COUNT (*) laskee kaikki tietueet (rivit).

```
mysql> SELECT COUNT(*) FROM employees;
```

```
+-----+  
| COUNT(*) |  
+-----+  
|    300026 |  
+-----+
```

COUNT(last_name) laskee montako sukunimeä tietokannassa esiintyy kaikkiaan.

```
mysql> SELECT COUNT(last_name) FROM employees;
```

```
+-----+  
| COUNT(last_name) |  
+-----+  
|          300026 |  
+-----+
```

Määre DISTINCT laskee kunkin sukunimen vain kerran.

```
mysql> SELECT COUNT(DISTINCT last_name) FROM employees;
```

```
+-----+  
| COUNT(DISTINCT last_name) |  
+-----+  
|                1638 |  
+-----+
```

Palkkojen keskiarvo:

```
mysql> SELECT AVG(salary) FROM salaries;
```

```
+-----+  
| AVG(salary) |  
+-----+  
|  63810.7448 |  
+-----+
```

Luokittelu GROUP BY

GROUP BY-lauseella voidaan luokitella yleensä jonkin funktion (tässä COUNT(*)) tulostus halutulla tavalla. Esimerkiksi, jos halutaan tietää montako miestä tai naista employees-kannassa on:

```
mysql> SELECT gender, count(*) FROM employees GROUP BY gender;
```

```
+-----+-----+  
| gender | count(*) |  
+-----+-----+
```

Tietokannat Osa 2: SQL:n peruskomentoja

M	179975
F	120051

HAVING

Jos GROUP BY:llä tehdyn luokittelun tulosta joudutaan rajaamaan jollain ehdolla, käytetään HAVING-lausetta. Se on hyvin pitkälle sama asia kuin WHERE. Nyrkkisääntö on, että *WHERE tulee ennen GROUP BY-ta ja HAVING GROUP BY:n jälkeen.*

Tässä haetaan employees-kannasta sellaiset nimikkeet (title), joita on enemmän kuin 90 000 kappaletta.

```
mysql> SELECT title,count(*) AS lkm FROM titles GROUP BY title HAVING lkm > 90000
```

title	lkm
Engineer	115003
Senior Engineer	97750
Senior Staff	92853
Staff	107391

Kannattaa huomata, että count-funktion tulos on talletettu AS-lauseella lkm-nimiseen muuttujaan. Sitä käytetään sitten myöhemmin HAVING-ehdon kanssa.

Merkkien ja merkkijonojen haku

Jokerimerkkinä toimii prosenttimerkki (%). Alaviivalla (_) haetaan mitä tahansa merkkiä niin monta kuin alaviivoja on. Kolmannessa esimerkissä on siis 5 alaviivaa peräkkäin, jolloin haetaan kaikki sellaiset osastot, joiden nimessä on 5 merkkiä (Sales).

```
mysql> SELECT * FROM departments WHERE dept_name LIKE "M%";
```

dept_no	dept_name
d001	Marketing

```
mysql> SELECT * FROM departments WHERE dept_name LIKE '%duct%';
```

dept_no	dept_name
d004	Production

```
mysql> SELECT * FROM departments WHERE dept_name LIKE '_____';
```

dept_no	dept_name
d007	Sales

Tietokannat Osa 2: SQL:n peruskomentoja