GRAAFINEN SQL-LIITTYMÄ

Mikko Laakso

Opinnäytetyö
Joulukuu 2013
Tietojenkäsittelyn koulutus-
ohjelma

TAMPEREEN AMMATTIKORKEAKOULU
Tampere University of Applied Sciences

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma

LAAKSO, MIKKO
Graafinen SQL-liittymä

Opinnäytetyö 61 sivua, joista liitteitä 20 sivua
Joulukuu 2013

Opinnäytetyön aihe oli WWW-sovellus, jolla käyttäjä voi hallinnoida MySQL-tietokantoja. Tarkoituksena oli luoda helppo- ja nopeakäyttöinen sovellus, jossa olisi graafinen käyttöliittymä. Tavoitteena oli saada sovellukseen mukaan kaikki yleisimmin käytetyt toiminnot sekä luoda helpompia toimintoja nykyisten, vaikeakäyttöisten tilalle.

Opinnäytetyössä esitellään sovelluksen lisäksi muutamia tämänhetkisiä sovelluksia, joilla voidaan hallita MySQL-tietokantoja. Tämän lisäksi luodaan katsaus PHP:n ja MySQL:n historiaan. Sovelluksessa käytetään PHP:ta, HTML:ää ja CSS:ää, joten käyttäjällä ei tarvitse olla esimerkiksi javascriptiä päällä.

Opinnäytetyöstä on hyötyä niille, jotka aikovat tutustua sovelluksen käyttämiseen tai aikovat luoda samankaltaisen sovelluksen itse. Sovellus onnistui tavoitteen mukaan, ja sen toimivuus varmistettiin muutaman käyttäjäkyselyn ja jatkuvan palautteen avulla. Vaikka sovellus on nyt valmis, sen tuotekehitys jatkuu edelleen.

Asiasanat: WWW, php, mysql, graafinen käyttöliittymä

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Name of the Degree programme
Name of the Option

LAAKSO, MIKKO
Graphical SQL-interface

Bachelor's thesis 61 pages, appendices 20 pages
December 2013

The subject of this thesis was a web application that allows users to manage MySQL databases. The aim was to create an easy-to-use and fast application, which would have a graphical user interface. The goal was to get all most commonly used functions in to the application, as well as to create easier options to existing, difficult to use, functions.

This thesis also presents some of the current applications which can manage MySQL databases. In addition to this there will be quick overview on the history of PHP and MySQL. The application was made using PHP, HTML and CSS, so the user for example does not need to have javascript enabled.

This thesis will be useful to those who intend to use the application or to create similar application themselves. The application was a success, and its performance was verified with a few user inquiries and continuous feedback. Even if the application is now complete, its development will continue.

Key words: WWW, php, mysql, graphical user interface

SISÄLLYS

1 JOHDANTO

Opinnäytetyön tavoitteena on esitellä muutamia tällä hetkellä käytössä olevia ohjelmia, jotka ovat samankaltaisia kuin GSQL. Tavoitteena oli myös kertoa GSQL:n synnystä, kertoa sen suunnittelusta ja toteutuksesta, sekä kertoa ohjelman nykyisestä versiosta ja sen toiminnasta. Opinnäytetyössä perehdytään myös PHP:hen ja MySQL:ää, joiden avulla työn toteutusosa aikaansaatu sovellus on toteutettu.

Opinnäytteen käytännön toteutusosan tarkoituksena on toteuttaa graafinen käyttöliittymä MySQL-tietokantojen hallintaan palvelimella.

Opinnäytetyön toimeksiantaja on Hopeapihlaja tmi. Hopeapihlaja aloitti käsitöiden saralla, mutta on nyt kehittynyt enemmän sisällöntuotannon ja web kehityksen suuntaan. Päätuotteita ovat verkkosivut, verkkokaupat sekä erilaiset räätälöidyt ratkaisut, mutta myös graafinen suunnittelu ja valokuvaus. Hopeapihlajalla on myös käytössään aputoiminimi Riksudesign, joka on käytössä webdesignin, graafisen suunnittelun ja valokuvauksen saralla.

Yritys on aloittanut toimintansa jo vuonna 2009, ja työllistää yhden ihmisen sivutoimisesti, mutta palkkaa myös satunnaisesti esimerkiksi tietojenkäsittelyn opiskelijoita. Hopeapihlaja on myös yksi Suomen ensimmäisistä 4H-yrityksistä, ja se on toiminut kahdelle muulle 4H-yritykselle yritysohjaajana.

Työ alkoi vapaa-ajallani kehittelemästä pienestä sovelluksesta, jolla saattoi nähdä tietokantojen sisällön nopeasti, mutta joka laajeni myöhemmin pitämään sisällään myös toimintoja tietokannan hallinnoimiseksi. Koska alussa sovellus oli vain omassa käytössäni, en tullut koskaan antaneeksi sille parempaa nimeä kuin Graafinen SQL-liittymä, tai lyhennettynä GSQL.

Valitsin aiheen, koska koin siitä olevan apua kaikille, jotka joutuvat tekemisiin SQL-tietokantojen kanssa. Tällä hetkellä markkinoilla on kaksi johtavaa tapaa hallita tietokantoja: tekstipohjaiset sovellukset (kuten emacs) sekä graafiset sovellukset, joista suosituin tällä hetkellä on phpMyAdmin. Tekstipohjaisilla työkaluilla käyttäjän täytyy tietää tarkalleen, mitä hän haluaa, kun taas graafiset ohjelmat ovat hieman laajempia, ja tarjoavat enemmän tietoa ja toimintoja käyttäjälle. Tekstipohjaisten sovellusten käyttö

vie kuitenkin aikaa, eikä palvelimelle pääse välttämättä lainkaan asentamaan ohjelmia. Ohjelmien käyttöönotto voi myös olla hankalaa, ja tästä syystä suunnittelin ja toteutin oman graafisen MySQL-tietokantojen hallintatyökalun, jonka käyttö sekä käyttöönotto ovat mahdollisimman helppoja. Se pitää sisällään tavallisesti käytetyt toimenpiteet, sekä mahdollisuuden ajaa omia tietokantakyselyitä siltä varalta, että käyttäjän haluama toiminto ei ole valmiiksi käytettävissä.

Sovelluksesta on hyötyä niille käyttäjille, jotka haluavat graafisen työkalun MySQL-tietokantojen hallintaan, mutta eivät voi tai ehdi käyttää aikaa sen asentamiseen.

Tarkoituksena oli myös pitää itse ohjelman käyttäminen ja asentaminen mahdollisimman yksinkertaisina, koska on jo olemassa hankalasti käytettäviä hallintatyökaluja, joissa on suuret määrät ominaisuuksia. Minä sitä vastoin aioin pitää ohjelman helppona, nopeana ja yksinkertaisena.

Opinnäytetyötäni tehdessäni kävin monia eri lähteitä läpi, ja seuloin niistä esiin ne, jotka olivat tarpeellisia, luotettavia sekä ajan tasalla olevia. Yleisesti ottaen yritin löytää mahdollisimman tuoreita lähteitä, mutta esimerkiksi käyttöoppaiden teosta ei ollut juurikaan materiaalia saatavilla. Vaikka löytämäni lähde on vuodelta 1984, se pitää edelleen hyvin paikkansa, toisin kuin suurin osa tietoteknisten asioiden lähteistä. Tuota yhtä lukuun ottamatta kaikki lähteeni ovat 2000-luvulta. Muutamissa kohdissa olen viitannut Wikipediaan, mutta ainoastaan tapauksissa, joissa lainaamani asia on ollut melko tunnettu. Tälläisissä tapauksissa katsoin paremmaksi käyttää yhtä lähdeviitettä muutaman virkkeen takeeksi, kuin käyttää useampaa lähdettä, joissa sama asia löytyisi jaettuna useamman lähteen kesken. Aina Wikipediaa lainatessani kävin kuitenkin tarkastamassa Wikipedian lähteet kyseisessä artikkelissa, sekä artikkelin muutoshistorian. Näin saatoin varmistua siitä, että artikkelin oli kirjoittanut asiantunteva henkilö.

2 NYKYTILANNE

MySQL on ilmainen relaatiotietokantaohjelmisto, jonka kehittivät suomalainen Michael Widenius sekä ruotsalainen David Axmark, ja se julkaistiin jo vuonna 1995. Sen kehittänyt yritys on vaihtanut monesti nimeä, alkaen MySQL AB:sta, ja vaihtuen Sun Microsystemsin kautta Oracle Corporationiksi. MySQL:ää kehitetään edelleen, ja viimeisin versio (5.6.14) siitä julkaistiin 20.9.2013. MySQL on maailman suosituin avoimen lähdekoodin RDBMS (Relational database management system), ja maailman toiseksi suosituin RDBMS, jääden kakkoseksi ainoastaan Oraclelle, joka on myös Oracle Corporationin kehittämä, mutta maksullinen ohjelmisto. (DB-Engines 2013)

Nykyään MySQL-tietokantoihin törmää lähes kaikkialla Internetissä: kaikki suuremmat palvelut, kuten Google, Wikipedia ja Facebook käyttävät MySQL-tietokantoja. Tämän lisäksi maailman suosituimmat sisällönhallintajärjestelmät, kuten WordPress, Joomla, Drupal ja monet muut, ovat MySQL-pohjaisia. (Wikipedia 2013a.)

MySQL (2013) listaa sivuillaan 10 hyvää syytä valita MySQL. Yksi näistä syistä on se, että monet isot yritykset maailmalla käyttävät MySQL-kantoja, ja he tietävät, mitä tekevät. Esimerkiksi Twitterissä liikkuu tavallisesti päivässä yli 500 miljoonaa twiittiä, mikä tekee noin 5700 twiittiä per sekunti. Youtubessa on yli 6 miljardia tuntia videota, ja se käyttää MySQL:ää pohjana. PayPalilla on noin 100 Terabittiä käyttäjädataa, joka on tallennettu MySQL-kantoihin. MySQL on siis suunniteltu ja rakennettu kestämään suurtakin kuormitusta.

Tämän lisäksi MySQL on myös maailman johtavia pilvitietokantoja. Yritykset, joilla on pilvipalveluita, kuten Amazon, F-Secure ja Google, käyttävät MySQL:n palveluita pilvitietokannoissaan.

MySQL on myös siitä hyvä tietokannanhallintajärjestelmä, että sitä voidaan käyttää useista ohjelmointikielistä. Tuettuja kieliä ovat mm. C, C++, PHP, Perl ja Python. (Heinisuo & Rauta 2007, 38)

Markkinoilla on tällä hetkellä monia erilaisia ohjelmia, joita voi käyttää MySQL-tietokantojen hallitsemiseen. Tietokantojen hallintaan on olemassa monia erilaisia vaihtoehtoja, kuten koneelle asennettavat ohjelmat tai palvelimelle asennettavat web-

sovellukset. Tarjolla on sekä graafisia että tekstipohjaisia sovelluksia, ja tunnetuimpia graafisista ohjelmista ovat todennäköisesti HeidiSQL, MySQL Workbench sekä phpMyAdmin. Rajasin ohjelmani sisältämien toimintojen määrän paljon näitä pienemmäksi, sillä niitä on kehittänyt suurempi joukko ihmisiä isommalla budjetilla. GSQL tarjoaa kuitenkin suurimman osan perustyökaluista, joita tietokantojen tarkasteluun tarvitaan, kuten taulujen ja rivien lisääminen, poistaminen sekä muokkaaminen.

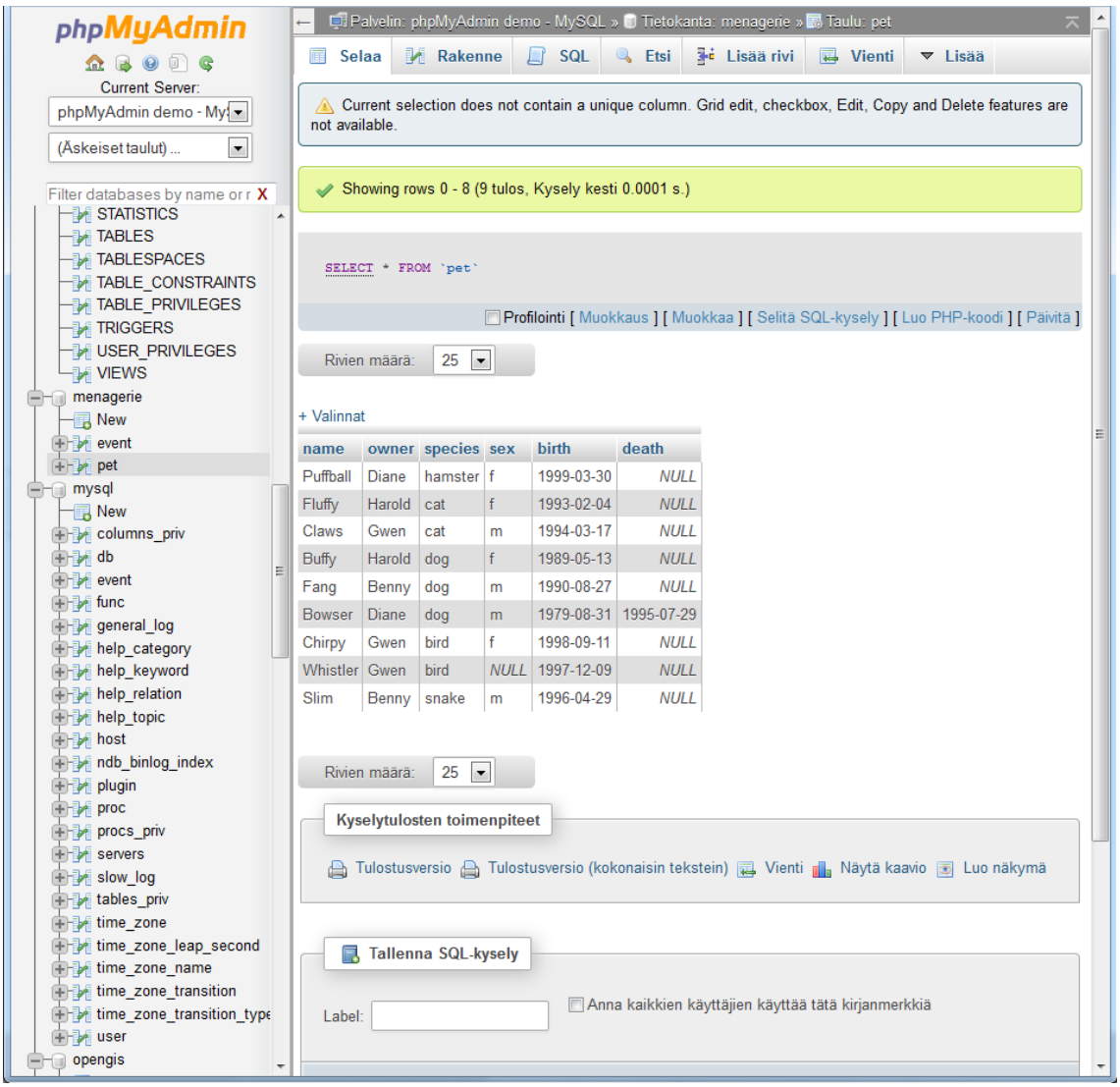
Myös PHP julkaistiin jo vuonna 1995, ja sitä on kehitetty siitä lähtien. Sen julkaisi tanskalais-grönlantilainen Rasmus Lerdorf. Uusin vakaa versio, 5.5.6 julkaistiin 14.11.2013, ja PHP on tukenut olio-ohjelmointia jo versiosta 4 alkaen. (Wikipedia 2013b)

PHP on skriptikieli, jolla pystyy suorittamaan useita toimintoja ilman, että ohjelmaa tarvitsee kääntää, ja jonka avulla voidaan tehdä dynaamisia verkkosivuja. PHP:ta voi löytää nykyään yli 244 miljoonalta verkkosivustolta. (PHP 2013)

Kuten MySQL, myös PHP on lisenssivapaa open source -tuote, eli toisin sanoen kuka tahansa saa käyttää niitä, halutessaan myös kaupalliseen käyttöön.

Ennen työn aloittamista ja sen aikana rajasin tutustumisen tietokantojen hallintaan emacsilla sekä phpMyAdminilla, sillä ne edustavat kahta erilaista, graafista ja komentoliittymäpohjaista, tapaa hallita MySQL-kantoja. Ne ovat myös melko suosittuja, ja suurin osa ihmisistä, jotka joutuvat tekemisiin tietokantojen kanssa ovat vähintäänkin kuulleet niistä. Katsoin siis aiheelliseksi tutustua niihin sekä pohtia, mikä niistä tekee suosittuja.

PhpMyAdmin on GPL:n alainen MySQL-tietokantojen hallintatyökalu, jota käytetään selaimen kautta. (Wikipedia 2013c). Se julkaistiin vuonna 1998, ja sitä on sen jälkeen jatkuvasti kehitetty näihin päiviin asti. Käyttäjä voi komentoliittymän kautta käyttää suoraan yleisimpiä toimintoja, kuten tietokantojen, taulujen sekä taulun rivien ja sarakkeiden lisääminen, poistaminen ja muokkaaminen. Tämän lisäksi käyttäjällä on mahdollisuus suorittaa omia kyselyitään omassa ikkunassaan. PhpMyAdmin näyttää myös käyttäjän kaikki tietokannat ja niiden taulut vasemmalla olevassa valikossa, joka on aina näkyvillä. Tämä tekee siirtymisen kantojen ja taulujen välillä erittäin helpoksi ja miellyttävän nopeaksi. Kuvassa 1 näkyy phpMyAdminin-näkymä, jossa tarkastellaan yksittäistä taulua.



KUVA 1. pet-nimisen taulun sisältö phpMyAdmin-ohjelmassa (demo.phpmyadmin.net 2013, kuvankaappaus)

Jos käyttäjä haluaa muokata riviä taulussa, voi hän painaa muokkauspainiketta, tai tuplaklikata kenttää, joka tällöin muuttuu muokattavaksi. Pitkiä tauluja selattaessa käyttäjä voi valita, montako riviä hän haluaa yhdellä rivillä nähdä.

Koska phpMyAadmin on selainpohjainen, vaatii se toimiakseen WWW-palvelimen, kuten Apache. Tämän lisäksi palvelimelta pitää löytyä php 5.2.0 tai sitä uudempi versio, ja MySQL 5.0 tai uudempi. Näiden lisäksi selaimen pitää tukea sekä Javascriptiä että evästeitä. (phpMyAdmin 2013)

Emacs on vuonna 1976 kehitetty tekstieditori, jota on helppo laajentaa omiin tarpeisiinsa, ja jossa on monia pikanäppäimiä, joita käyttäjä itsekin voi lisätä rajattomasti. Tekstieditorinsa lisäksi Emacs sisältää useita enemmän tai vähemmän tärkeitä lisäosia, kuten

kalenterin, useita pelejä, laskimia sekä lähdekoodin väritys erilaisille ohjelmointikielille. Eric Marsden jopa kirjoitti Emacsiin lisäosan, jonka avulla käyttäjä saattoi keittää kahvia, mikäli tällä oli siihen sopiva laite. (Emacswiki 2013)

Emacsilla on myös mahdollista saada yhteys palvelimella olevaan MySQL-tietokantaan. Koska Emacsin käyttöliittymä on tekstipohjainen, täytyy käyttäjän itse tietää, millä kyselyillä hän saa haluamansa tiedot tietokannasta esiin. Emacs ei esittele valmiiksi esimerkiksi tauluja, ja rivien muuttaminen tapahtuu kirjoittamalla itse kysely, joka muuttaa kyseistä riviä. Myös varmuuskopioiden ottaminen tietokannasta tai useammasta taulusta on melko hankalaa.

3 MYSQL:N HALLINTATYÖKALUN TOTEUTTAMINEN PHP:LLA

Kuten myöhemmin selviää, pohdin eri ohjelmointikieliä sovelluksen toteuttamiseen, ja päädyin PHP:en. PHP:ssa on monia tietokantafunktioita, joiden avulla tietokantojen hallinnointi tapahtuu helposti. Esimerkiksi yhteyden ottaminen MySQL-tietokantaan tapahtuu seuraavasti: käyttäjä joutuu ensimmäisenä yhdistämään MySQL:ään, koodiesimerkin 1 mukaisesti. Koodiesimerkissä 1 otetaan yhteys annetulle MySQL-palvelimelle käyttäen käyttäjän tunnusta ja salasanaa. Mikäli tämä ei onnistu, pysäytetään sivun suorittaminen ja käyttäjälle annetaan virheviesti.

```
$yhteys = mysql_connect("db.palvelin.fi", "käyttäjä",
"salasana") or die("Yh-distäminen ei onnistunut!");
```

Koodiesimerkki 1. Yhteyden muodostaminen tietokantapalvelimeen

Mikäli yhteyden muodostaminen onnistuu, pitää käyttäjän seuraavaksi ottaa yhteys haluttuun tietokantaan. Tämä tapahtuu koodiesimerkin 2 mukaisesti. Esimerkissä käytetään onnistuneesti luotua yhteyttä esimerkistä 1, ja käyttäjä yhdistetään testi-tietokantaan. Mikäli yhdistäminen ei onnistu, pysäytetään sivun suorittaminen ja käyttäjälle tulostetaan virheviesti.

```
mysql_select_db("testi", $yhteys) or die("Tietokantaa ei
löytynyt!");
```

Koodiesimerkki 2. Testi-nimisen tietokannan valitseminen

Kaikkiin toimintoihin ei kuitenkaan tarvita tietokannan valitsemista. Jos käyttäjä on vaikkapa unohtanut tietokantansa nimen, tarvitsee hänen vain suorittaa tietokantakysely *'show databases'*. Komento listaa käyttäjälle kaikki hänen tietokantansa, joista käyttäjä voi sitten valita mieleisensä. Kuitenkin suurin osa tietokantakyselyistä tapahtuu vasta käyttäjän valittua tietokantansa.

Kun käyttäjä on valinnut tietokannan, voi hän suorittaa haluamiaan kyselyitä tietokannassa. Koska käyttäjä liikkuu useilla sivuilla, täytyy tietokantatietojen liikkua hänen mukanaan. Yksinkertaisin keino tähän olisi ollut istunnon käyttäminen, jossa käyttäjän

tiedot ovat tallessa istunnon ajan. Mutta jos käyttäjä haluaa tulla takaisin myöhemmin kirjautumatta sisään, täytyy tiedot säilöä toisaalle. Päädyin käyttämään evästeitä, jonka avulla käyttäjän koneelle tallennetaan tiedosto, jossa käyttäjän tiedot ovat tallessa. Eväste on voimassa määrätyn ajan, ja tuhoutuu sen jälkeen. Koodiesimerkissä 3 on esimerkki siitä, kuinka PHP:lla tehdään 'tunnus'-niminen eväste, jolla on arvona *Nimi*, ja joka on voimassa seuraavat 30 minuuttia.

```
setcookie("tunnus", "Nimi", time() + 60 * 30);
```

Koodiesimerkki 3. Evästeen luonti

Kun käyttäjä on valmis, yhteys tietokantaan suljetaan koodiesimerkin 4 mukaisesti.

```
mysql_close($yhteys);
```

Koodiesimerkki 4. MySQL-yhteyden sulkeminen

Itse tietokantakyselyiden suorittaminen tapahtuu PHP:lla helposti. Ohjelmoija kirjoittaa MySQL-tietokantakyselyn, joka voi sisältää PHP-muuttujia, ja suorittaa sen komennolla mysql_query(). Vaikka komennot voisi kirjoittaakin samalle riville, kannattaa ne kirjoittaa erikseen selvyyden vuoksi. Koodiesimerkissä 5 on ensimmäisellä rivillä kirjoitettu MySQL-kysely, joka listaa kaikki taulut tietokannasta, joka löytyy muuttujasta $dbname. Toisella rivillä kysely suoritetaan.

```
$query = "SHOW TABLES FROM $dbname";
$result = mysql_query($query);
```

Koodiesimerkki 5. Kyselyn suorittaminen PHP:lla

Tämän jälkeen ohjelman pitää tarkistaa, onnistuiko tietokantakysely. Jos se ei onnistunut, pitää käyttäjälle ilmoittaa, että jokin meni vikaan, ja mielellään myös, mikä meni vikaan niin, että käyttäjä voi korjata asian. PHP tarjoaa esimerkiksi toiminnot *mysql_error()* ja *mysql_errno()*, jotka kertovat virheilmoituksen ja virheen numeron. Nämä eivät kuitenkaan välttämättä kerro käyttäjälle tarpeeksi selkeästi, mikä meni vikaan, ja miten sen voi korjata. Koodiesimerkissä 6 yritetään ottaa yhteyttä tietokantaan, jota ei ole olemassa, ja sen jälkeen tulostetaan virheen numero ja virheilmoitus. Käyttä-

jälle ei ole mitään hyötyä siitä, että hän tietää virheilmoituksen numeron, mutta sovelluksen ylläpitäjä voi sen avulla selvittää, missä vika on.

```
mysql_select_db("eiole", $yhteys);
echo mysql_errno($yhteys) . ": " . mysql_error($yhteys);
```

Ylläoleva koodi tulostaa seuraavan:
1049: Unknown database 'eiole'

Koodiesimerkki 6. Virheellinen kysely, ja sen virheilmoitus

Jos kysely taas onnistui, pitää sovelluksen tarkistaa, muuttiko kysely tietokantaa, jolloin käyttäjälle riittää ilmoitus kyselyn onnistumisesta. Mutta jos kysely palautti tietoja, pitää tiedot käsitellä ja tulostaa käyttäjälle. Käyttäjälle voidaan esimerkiksi tulostaa kaikki palautuvat rivit, tai palautuvien rivien tai kenttien määrä.

Sovellus on tehty mahdollisimman yksinkertaisesti, mutta yhtenä sen monimutkaisimmista osista oli vapaa koodiajo. Tämä johtui siitä, että käyttäjälle annettiin mahdollisuus suorittaa omia kyselyitään, ja ohjelman pitäisi tulostaa tulos oikein riippumatta siitä, minkälaisen kyselyn käyttäjä suoritti. Koodiesimerkissä 7 on vapaan koodiajon sisältö hieman tiivistettynä ja näennäiskoodina. Käyttäjän kysely otetaan muuttujaan, se tarkistetaan haitallisen sisällön varalta, ja ajetaan esimerkin 5 mukaan. Sen jälkeen ohjelma tarkistaa, onnistuiko kysely. Mikäli kysely ei onnistu, tarkistetaan onko käyttäjän kysely tyhjä. Jos se on, tulostetaan käyttäjälle ilmoitus tyhjästä kyselystä, ja muussa tapauksessa käyttäjälle ilmoitetaan kyselyn olevan viallinen. Mikäli kysely onnistuu, ohjelma tarkistaa, pitääkö sen tulostaa jotain. Jos kysely muuttaa tietokantaa, esimerkiksi lisäämällä tai poistamalla rivejä, tai muuttamalla rivin sisältöä, ohjelma tulostaa ainoastaan ilmoituksen kyselyn onnistumisesta. Jos käyttäjä sen sijaan yritti esimerkiksi listata taulun rivejä, ohjelma tarkistaa, montako riviä tuloksia oli. Ohjelman palauttaessa 0 riviä käyttäjälle kerrotaan, että kysely onnistui, mutta se ei palauttanut tuloksia.

```
//Onnistuiko kysely?
if($result) {
    //Kerro käyttäjälle että ajo onnistui
    if(is_resource($result)) { //Pitäisikö kyselyn tulostaa jotakin?
       if (mysql_num_rows($result)) {
            //Tulosta tulokset
        }
        else {
        //Ilmoita käyttäjälle että kysely palautti 0 tulosta
        }
    }
    else {
        //Kysely ei tulostanut mitään, vaan muutti tietokantaa (esim. lisäsi rivin)
    }
} //Ei saatu tulosta. Onko kysely tyhjä?
else if($code == "") {
    //Ilmoita käyttäjälle, että hänen kyselynsä on tyhjä
}
else { //Kysely ei ollut tyhjä, joten jokin meni pieleen
     //Tulosta virheilmoitus käyttäjälle
}
```

Koodiesimerkki 7. Vapaa koodiajo

Jos kysely sen sijaan palauttaa rivejä, pitää ne tulostaa. Koodiesimerkissä 8 on edellisen esimerkin *//Tulosta tulokset* -kohdan sisältö.

Tietokannasta haetaan tulokset käyttäjän kyselyn perusteella, ja siitä erotellaan otsikkotiedot, eli sarakkeen nimi. Tämän jälkeen tulostetaan taulukko, jossa otsikkorivinä ovat

sarakkeiden nimet, ja niiden alle tulostetaan kaikki palautuvat rivit. Ennen rivien tulostamista täytyy palautunut tieto kuitenkin siistiä. Jos käyttäjän tietokannassa olisi esimerkiksi HTML-koodi pudotusvalikolle, se pitäisi tulostaa koodina eikä pudotusvalikkona. Myös otsikkotiedot on hyvä siistiä väärinkäytösten varalta.

```
if (mysql_num_rows($result)) {
    //Tulosta tulokset
    $data = array();
    while($row = mysql_fetch_assoc($result)) {
        $data[] = $row;
    }
    $colNames = array_keys(reset($data))

    //Tulosta otsikkorivi
    echo '<table><tr class="headrow">';
    foreach($colNames as $colName) {
        echo "<th>".htmlspecialchars($colName)."</th>";
    }
    echo '</tr>';

    //Tulosta rivit
    foreach($data as $row) {
        $class = ($class == "secondrow") ? "firstrow" :
"secondrow";
        echo "<tr class=\"$class\">";
        foreach($colNames as $colName) {
            echo
"<td>".htmlspecialchars($row[$colName])."</td>";
        }
        echo "</tr>";
    }
    echo '</table>';
}
```

Koodiesimerkki 8. Vapaa koodiajo, tulosten tulostus

4 SOVELLUKSEN TOTEUTTAMINEN

4.1 Suunnittelu

Ennen ohjelman toteuttamista jouduin ensimmäisenä harkitsemaan, millä ohjelmointikielellä sovellukseni tekisin. Tämä vaikuttaisi ohjelman sisältämiin toimintoihin ja siihen, miten sovellus kannattaisi toteuttaa.

Ensimmäinen harkitsemani ohjelmointikieli oli C++, koska se on yksinkertainen ja osaamiseni on melko hyvä sen kanssa. Toisaalta C++-ohjelma vaatisi kääntäjän, ja sovellus toimisi lähinnä tekstipohjaisena. Lisäksi tekstipohjaisen sovelluksen yhtenä suurimpana ongelmana tulisi olemaan se, että käyttäjä ei voi käyttää hiirtä. Lisäksi käyttäjän pitäisi ensin ottaa yhteys palvelimelle, kääntää ohjelma, suorittaa ohjelma ja lopuksi kirjautua sisään. Harkittuani näitä seikkoja totesin, että haluan tehdä sovelluksestani graafisen enkä tekstipohjaisen.

Seuraavaksi harkitsin PHP:ta, joka on vahvin osaamistani kielistä. Sen käyttö vaatisi palvelimen, ja sillä voisi ottaa yhteyttä vain samalla palvelimella oleviin tietokantoihin, mutta toisaalta siihen olisi helppoa suunnitella graafinen ulkoasu, jonka muokkaaminen kävisi myös nopeasti .css-tiedostojen avulla. Myös esimerkiksi phpMyAdmin on nimensä mukaisesti tehty PHP:lla. Koska PHP toimii selaimessa, käy yhteyden ottaminen palvelimelle melko yksinkertaisesti ja nopeasti. PHP:ssa on myös kasa valmiita funktioita, jotka helpottaisivat sovelluksen luomista.

Kahden edellä mainitun lisäksi mietin c#:n käyttämistä ohjelmointikielenä. Sen avulla ulkoasun tekeminen olisi helppoa, ja sen voisi kääntää suoritettavaksi ohjelmaksi, joka ottaisi yhteyden haluttuun palvelimeen. Tällöin käyttäjän ei tarvitsisi asentaa palvelimelle mitään. Toisaalta sovelluksessa voisi silloin käyttää vain melko pelkistettyä grafiikkaa, ja toisin kuin PHP:lla, C#:lla tehtyä ja käännettyä ohjelmaa ei myöskään pääse muuttamaan jälkikäteen kovinkaan helposti.

Punnittuani eri kielten etuja ja haittoja päädyin toteuttamaan sovellukseni PHP:lla. PHP (ennen lyhennys Personal Home Page, nykyään PHP: Hypertext Preprocessor) oli alun

perin kokoelma WWW-pohjaisten sovellusten tekemistä helpottavia rutiineja. (Heinisuo & Rauta 2007, 12). Koska pyrin itsekin helpottamaan rutiinitöitä, valintani oli selvä.

Jouduin myös suunnitteluvaiheessa rajaamaan useita toimintoja pois sovelluksestani. Osan toiminnoista, kuten rivien järjestäminen jonkin sarakkeen perusteella, voisin palauttaa takaisin, jos ne olisivat tarpeen ja aikaa riittäisi. Toiset toiminnot, kuten lokitiedosto kaikista suoritetuista kyselyistä tai kyselyiden muuttaminen PHP-komennoksi, jätin kokonaan pois, sillä ne olivat joko tarpeettomia, liian monimutkaisia tai liian aikaa vieviä toteuttaa.

Tarkoitukseni oli toteuttaa käytännöllinen perussovellus tietokantojen nopeaan tutkimiseen, mukaan lukien tärkeimmät perustoiminnot. Yhden miehen voimin ja nollabudjetilla en voinut lähteä tekemään mitään, mikä kilpailisi markkinoilla olevien maksullisen tai suurimpien maksuttomien sovellusten kanssa. Sovellukseni siis pitäisi sisällään vähemmän toimintoja ja vaihtoehtoja, mutta siitä kuitenkin saisi helposti selville haluamansa asiat. Tästä syystä päätin myös sisällyttää vapaan koodiajon sovellukseeni, jotta ne, jotka eivät löydä etsimäänsä, voisivat kirjoittaa ja ajaa itse oman kyselynsä.

Jouduin alussa myös hieman miettimään sovelluksen testaamista. Sovellus luonnollisesti pitäisi testata, jotta sieltä saadaan karsittua virheet pois, mutta minulla ei ollut resursseja järjestää monen ihmisen käyttäjätestausta. Sovellus pitäisi testata vähintään 15 eri ihmisellä, jotta löydettäisiin kaikki käytettävyysongelman (Nielsen Norman Group. 2013), jo viidellä testaajalla löydettäisiin 85% ongelmista. Päätin, että molemmissa kyselyissä, jotka aioin toteuttaa, tulisi olla vähintään 5 henkilöä.

4.2 Käyttäjäkysely

Järjestin kaksi epävirallista käyttäjäkyselyä opinnäytetyön aikana. Näistä toisen ajoitin heti aloittamisen jälkeen, ja toisen, kun ensimmäinen versio oli valmis. Ensimmäinen kysely tapahtui suullisesti, ja minulla oli muutamia kysymyksiä, joiden perusteella lähdin pohtimaan, mitä ominaisuuksia halusin sovellukseeni. Liitteessä 1 ovat koehenkilöille esitetyt kysymykset.

En saanut ensimmäiseen käyttäjäkyselyyn kovinkaan montaa henkilöä vastaamaan, mutta pidin kuitenkin mielessä Wiion (2004, 221) ohjeistuksen, jonka mukaan pikatestiin riittää kolme koehenkilöä, joskin suositeltava määrä on yleensä viisi tai kuusi.

Ensimmäisessä käyttäjäkyselyssä tutkin lähinnä sitä, mitä ominaisuuksia käyttäjät haluaisivat sovellukseen, mutta myös käyttäjien toiveita ohjelman ulkonäöstä.

Ensimmäisenä selvitin, mitkä ovat useimmin käytettyjä toimintoja tietokantoja hallittaessa, ja miten käyttäjät haluaisivat tehdä niitä. Kyselin myös toimintoja, joita oli vaikea tai mahdoton käyttää koulussa opetetuilla tavoilla.

Sen lisäksi minulla oli kuva sovelluksen tulevasta ulkonäöstä, ja esittelin sitä käyttäjille. Kysyin heiltä, mitä he yrittäisivät tehdä, jos heidän pitäisi suorittaa jokin tietty tehtävä. Vaikka yleensä mielipiteitä on yhtä monta kuin ihmisiäkin, kysyin lopuksi käyttäjien mielipidettä ohjelman ulkonäöstä. Alkuvaiheessa sovelluksen tekemistä sen muuttaminen olisi vielä helppoa, jos suurin osa käyttäjistä toivoisi jonkin tietyntyyppistä rakennetta.

Jo alusta alkaen oli selvää, että tärkeimpinä toimintoina tulisivat olemaan vapaa kenttä, johon käyttäjä voisi syöttää oman tietokantakyselynsä saadakseen selville juuri häntä kiinnostavat asiat, sekä taulukko, josta hyvin nopeasti vilkaisemalla saisi selville tietokannat sekä niiden sisällön. Muita tärkeitä kohtia tulisivat olemaan perustoiminnot, kuten tietojen lisääminen, poistaminen sekä muokkaaminen. Tämän lisäksi erityisesti esille nousi tietokantojen varmuuskopiointi, johon tekstipohjaisissa sovelluksissa ei ollut mitään helppoa keinoa. Useimmat graafiset sovellukset kuitenkin sallivat tietokantojen tai taulujen varmuuskopioimisen ja palauttamisen, ja osassa on tarjolla erilaisia vaihtoehtoja, mitä kaikkea varmuuskopion tulisi pitää sisällään. Ulkoasusta tuli esille monia erilaisia mielipiteitä, mutta suurin osa tuntui pitävän ajatusta yksinkertaisesta ja helposta käyttöliittymästä parhaana. Suurin osa käyttäjistä piti käyttöliittymää melko hyvänä, ja kun pyysin heitä kertomaan, missä he jonkin tietyn toiminnon suorittaisivat, meni suurin osa paikkaan, johon olin jo suunnitellut kyseisen toiminnon sijoittavani.

4.3 Ensimmäinen versio

Aloitin luomalla muutamia funktioita, jotka lyhentäisivät sivuilla esiintyvää koodia. Funktioiden, kuten printHeader ja printFooter, avulla minun tarvitsi kirjoittaa koodi vain kerran, vaikka niiden sisältö esiintyisikin joka sivulla.

Ensimmäinen versio sisälsi vain hyvin yksinkertaista grafiikkaa. Pyrin kuitenkin heti alusta pitämään ulkoasun mahdollisimman loogisena ja yksinkertaisena. Tavoitteena oli, että käyttäjä osaisi ilman kokemusta ohjelman käytöstä mennä suoraan oikeaan paikkaan suorittaakseen haluamansa toiminnon.

Käyttäjä pystyi syöttämään tietonsa erilliseen sisäänkirjautumistiedostoon, josta ohjelma luki tunnukset ja kirjasi käyttäjän automaattisesti sisään. Oli heti alusta lähtien tiedossa, että tämä ominaisuus muutettaisiin sisäänkirjautumiseksi, sillä vaikka automaattinen sisään pääseminen oli kätevää, sen aiheuttamat tietoturvariskit olivat liian suuret, ja käyttäjän oli vaivalloista käydä kirjoittamassa tunnuksensa tiedostoon sen sijaan, että olisi kirjoittanut ne sovelluksen ikkunassa.

Ensimmäisessä versiossa oli mukana myös tietokannan varmuuskopiointi, jolla käyttäjä saattoi tehdä varmuuskopiotiedoston palvelimelle. Käyttäjä saattoi ladata tiedoston itselleen, tai säilyttää sen palvelimella. Ohjelma myös palautti haluttaessa viimeisimmän varmuuskopion. Koska varmuuskopioita tulisi luultavasti olemaan paljon, ja vanhimmat niistä muuttuisivat aikanaan turhiksi, ensimmäisessä versiossa oli myös mukana toiminto, joka poisti varmuuskopiointikansiosta yli kuukauden vanhat varmuuskopiot.

Ensimmäinen vedos oli lähinnä yhdelle henkilölle suunnattu järjestelmä, ja kyseisellä henkilöllä piti olla pääsy palvelimella oleviin tiedostoihin. Tämä oli riittävästi esimerkiksi kouluolosuhteissa, mutta mikäli ohjelmaa käyttäisi useampi ihminen, pitäisi siihen tehdä muutoksia.

4.4 Työnantajan palaute

Saatuani ensimmäisen version valmiiksi, esittelin sen työnantajalleni ja sain palautetta seuraavista asioista: järjestelmä vaikutti lupaavalta, mutta se pitäisi muokata toimimaan

myös usean käyttäjän käyttäessä sitä samaan aikaan. Sillä hetkellä kaikki käyttäjät kuuluivat samaan ryhmään, ja pystyivät jakamaan ja vaihtamaan tiedostoja keskenään, mutta tämä saattaisi muuttua. Minua pyydettiin myös lisäämään jonkinlainen järjestelytyökalu monirivisiä kantoja varten. Myös vapaaseen koodiajoon toivottiin toimintoa, jolla käyttäjä pääsisi muokkaamaan ajamaansa kyselyä.

Työnantajan palautteen jälkeen tutustuin myös tarkemmin phpMyAdminin toimintaan heidän Wikinsä avulla, ja jouduin pohtimaan käyttäjäsyötteiden siistimistä. Toisaalta, koska kyseessä on tietokantasovellus, ohjelman tulee sallia tietokantakyselyt, mutta toisaalta taas käyttäjien kenttiin syöttämää php-, tai muuta koodia ei tarvitse ajaa. Myöskään käyttäjän kantaan tallentamaa tekstiä ei saa muuttaa, sillä muuten käyttäjän tietokanta (tai ohjelma tai sivu joka kantaa käyttää) saattaa mennä rikki puuttuvien tai ylimääräisten kautta- ja kenoviivojen takia. Tämän lisäksi HTTP-pyynnön mukana saapuneeseen GET-, POST-, ja Cookie-dataan lisätään kenoviivat merkkien ', ", \ ja NULL eteen, mikäli PHP:n *Magic quotes gpc* -asetus on päällä (Rantala 2005), mikä piti ottaa huomioon tietoja lähetettäessä sivujen välillä.

4.5 Toinen versio

Toisessa versiossa käyttäjä aloitti kirjautumalla sisään syöttämällä tunnuksensa, salasanansa, isännän sekä tietokannan kenttiin. Käyttäjä saattoi myös valita muutamasta vaihtoehdosta, kauanko sisäänkirjautuminen kestäisi. Tällöin käyttäjä saattoi olla kirjautuneena halutessaan vain yhden istunnon ajan, mutta jos hän käyttäisi sovellusta paljon, ei hänen tarvitsisi joka kerta kirjautua sisään uudelleen.

Toisessa versiossa käyttäjän oli mahdollista ottaa varmuuskopioita koko taulustaan, mutta varmuuskopiot jäivät varmuuskopio-kansioon, kunnes ne poistetaan sieltä erikseen. Vaikka yhden käyttäjän käytössä tästä ei suuresti ole haittaa, muodostaa tämä useamman käyttäjän kohdalla selkeän tietoturva-aukon.

Lisäsin järjestelmään myös työnantajan toivomuksesta vapaan koodiajoikkunan, jotta käyttäjä voisi tehdä helposti myös omia kyselyitään. Sen toteutus oli hieman haasteellista, koska sen piti selvittää, halusiko käyttäjä tietoja tietokannasta, vai yrittikö käyttäjä muuttaa kantaa, ja näyttää tulos sen mukaan. Mikäli käyttäjän kysely palauttaisi rivejä,

ne pitäisi tulostaa jokainen, eikä vain ensimmäistä, ja niiden päällä olisi hyvä näkyä sarakkeen nimi. Tämän lisäksi ohjelman pitäisi vielä erikseen ilmoittaa, mikäli käyttäjän kysely palauttaisi rivejä, mutta tietokannasta palautuisi 0 riviä. Tällöin kysely on onnistunut, mutta se ei ole löytänyt tuloksia.

4.6 Toinen kysely

Järjestin toisen kyselyn, kun olin saanut toisen version ohjelmasta valmiiksi. Koekäyttäjät olivat henkilöitä, joilla oli jonkin verran kokemusta MySQL:n käytöstä, vaihdellen välillä 'en osaa käyttää, mutta kurssitehtävät on tehtävä' ja 'olen käyttänyt hyvin paljon'. Ainoastaan yksi käyttäjä oli mukana myös ensimmäisessä kyselyssä. En järjestänyt koko kyselyä kerralla, vaan kuten Krug (2006, 138) neuvoo. Järjestin siis ensin testin pienelle ryhmälle, korjasin esiin tulleita ongelmia, ja tein sen jälkeen kyselyn uudelleen.

Käyttäjiltä kysyttiin mielipidettä ulkoasullisista asioista, kuten toimintojen sijoittelusta, ja heitä pyydettiin tekemään ohjelmalla tavanomaisia tehtäviä. Ohjelman käyttöönotto ei vaatinut suuria ponnistuksia, vaan käyttäjät osasivat mennä oikeaan paikkaan suorittamaan käsketty toiminto viimeistään toisella yrittämällä.

Kysymykset on listattu liitteessä 2. Osa kysymyksistä on samoja kuin ensimmäisessä kyselyssä, mutta siinä missä silloin esittelin lähinnä kuvia, nyt tarjolla oli koko sovellus.

Kun käyttäjät olivat tutustuneet ohjelman käyttöön, kysyttiin heiltä mielipiteitä siitä, pitäisikö toimintojen sijaita jossakin muualla, tai haluaisivatko he mahdollisesti joitain toimintoja lisää tai pois. Mielipiteet vaihtelivat luonnollisesti ihmisestä ihmiseen, ja jälkikäteen kävin kyselyn tulokset läpi Hopeapihlajan kanssa. Päädyin toteuttamaan osan pyynnöistä, kuten varmuusottamisen ainoastaan yksittäisestä taulusta, mutta osa pyynnöistä ei ollut järkevästi toteutettavissa, joten päätin jättää ne tekemättä.

4.7 Työnantajan palaute

Toteutettuni viimeisimmän käyttäjäkyselyn esiin tuomat muutokset kysyin myös työnantajan mielipidettä uudelleen.

Hänen mukaansa sovellus on toimiva ja suurimmalta osalta sulava toiminnallisuudessaan. Sen sijaan ohjelmassa on niukasti käyttäjien ohjausta, jolloin kokeneilla käyttäjillä ohjelman käyttö hidastuu, ja kokemattomilla saattaa pysähtyä kokonaan. Lisäksi sarakkeiden poistaminen tapahtuu nimen perusteella, mikä ei ole kovin käyttäjäystävällinen tapa, ja mahdollistaa monia käyttäjävirheitä. Taulun luonti, varmuuskopiointi ja muokkaus sen sijaan toimivat erinomaisesti ja sulavasti, ja erikoismerkit riveissä toimivat hyvin.

4.8 Lopputulos

Viimeiset muutokset tehtyäni sain lopulta valmiiksi lopullisen version sovelluksestani.

Lopullisessa versiossa on yläpalkki, joka kertoo ohjelman nimen, sekä käyttäjän tietokannan nimen. Näin käyttäjä on aina selvillä siitä, minkä tietokannan tauluja hän on tarkastelemassa. Lisäksi ohjelman nimeä klikkaamalla käyttäjä pääsee kätevästi takaisin yleisnäkymäsivulle.

Harkitsin placeholder-tekstien käyttöä syötekentissä, mutta valitettavasti Internet Explorer ei vielä tue tätä tekniikkaa kunnolla. Halusin ohjelmani toimivan myös hieman vanhemmilla selainversioilla, joten jätin tämän toiminnon odottamaan parempia aikoja.

Tutkittuani hieman muita samantyyppisiä ohjelmia, kuten HeidiSQL:ää, totesin sovellukseni pitävät jo sisällään ominaisuuksia, joita muihin ohjelmiin oli vasta tulossa (kuten vaihtuva rivien taustaväritys) (Heidisql 2013). Luin tarkemmin myös heidän muista ominaisuuksistaan, mutta hylkäsin osan turhina. Osa oli jo sovelluksessani, ja osa olisi ollut liian monimutkaisia toteuttaa annetussa ajassa.

4.8.1 Sisäänkirjautuminen

Kuten kuvassa 2 näkyy, sisäänkirjautuminen tapahtuu syöttämällä käyttäjänimi, salasana, isäntä sekä tietokannan nimi kenttiin ja valitsemalla voimassaoloaika. Tietokannan tulee sijaita samalla palvelimella, jotta ohjelma saa siihen yhteyden.

KUVA 2. Sisäänkirjautumisikkuna

Salasanakenttä on password-tyyppinen input-kenttä, joten käyttäjän syöttäessä salasanaansa näkyy näytöllä vain tähtiä. Tästä johtuen käyttäjän ei tarvitse huolehtia salasanansa näkymisestä selkokielellä.

Voimassaoloaika kertoo, kuinka kauan käyttäjä pysyy sisällä kirjautumatta uudelleen. *Ikuisesti*-vaihtoehdon arvona on 20 vuotta.

Ohjelma käyttää evästeitä pitääkseen kirjaa siitä, kuinka kauan käyttäjä on kirjautunut sisälle. Tästä saattaa aiheutua käyttäjälle ongelmia, mikäli hänellä ei ole evästeitä käytössä, mutta toisaalta sen ansiosta sovelluksen ei tarvitse tehdä taulua tietokantaan pitääkseen käyttäjän muistissa. Tämä myös käyttää vähemmän palvelimen resursseja sekä pitää käyttäjän tiedot tallessa, vaikka kaikki tiedostot palvelimelta tuhottaisiin ja ladattaisiin sinne uudelleen. Haittapuolina tosin on esimerkiksi rajoitetumpi tiedon saatavuus, sekä mahdollisuus siihen, että käyttäjänpuolista dataa muokattaisiin. (Lea, Buzzard, White-Cinis & Thomas 2003.)

Mikäli käyttäjä on unohtanut tietokantansa nimen, mutta muistaa muut tietonsa, voi hän syöttää tarvittavat tiedot ja painaa *Näytä tietokannat* -painiketta. Tällöin ohjelma ottaa tietokantayhteyden käyttäjän tunnuksilla ja tulostaa listan käyttäjän tietokannoista.

4.8.2 Yleisnäkymä

Kun käyttäjä pääsee kirjautumaan sisään, näkee hän suoraan tietokannassa olevat taulut, sekä painikkeet eri toimintoihin. Käyttäjä voi myös kirjautua kokonaan ulos, tai vaihtaa sisäänkirjautumistunnuksia, jolloin hän pääsee käsiksi toiseen tietokantaan.

Tietokannan taulujen ja niiden tietojen muokkaaminen tapahtuvat omilla sivuillaan, mutta niiden poistaminen tapahtuu yleisnäkymässä. Kuvassa 3 näkyy yleisnäkymä kokonaisuudessaan.

KUVA 3. Yleisnäkymä

On myös mahdollista, että käyttäjällä ei ole lainkaan tauluja tietokannassa. Tällöin yleisnäkymäsivulla käyttäjälle tulostetaan kuvan 4 mukainen ilmoitus *Tietokantasi on tyhjä*.

Kuva 4. Tyhjä tietokanta

4.8.3 Taulun sisältö

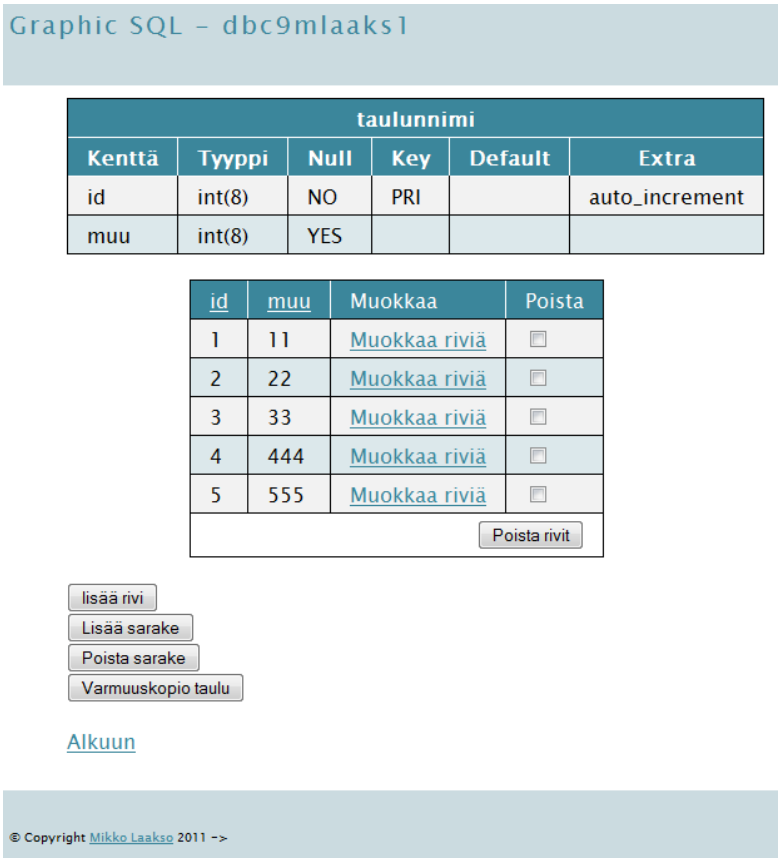
Taulun sisältö kertoo ensimmäisenä taulun rakenteen, kuten kenttien nimet, niiden tyypit sekä muita tietoja, kuten oletusavaimen.

Sen alla tulee taulukko, joka näyttää taulun sisällön. Mikäli jossain taulun kentässä on PHP- tai HTML-koodia, tulostetaan kentän sisältö näytölle, mutta sitä ei suoriteta. Tällöin esimerkiksi lihavointi tulostuu näytölle muodossa lihavoitava teksti, eikä lihavoituna tekstinä. Myöskään php-koodia ei voi ajaa *kentän sisältö* -kohdassa. Jos taulu on tyhjä, näytetään sen kenttien ominaisuudet normaalisti, mutta sen alle tulostetaan vain ilmoitus *Taulu on tyhjä*.

Taulun sisällön jälkeen tulevat taulua koskevat toiminnot, kuten rivien lisääminen ja poistaminen. Tämän lisäksi käyttäjä voi ottaa varmuuskopion ainoastaan yhdestä taulusta halutessaan.

Käyttäjä voi myös halutessaan lisätä tai poistaa taulukon sarakkeen. Jos käyttäjä poistaa sarakkeen, katoavat myös kaikki siinä olleet tiedot. Työnantajan palautteen perusteella vaihdoin sarakkeen poistamisen toiminnan, ja nyt sen sijaan, että siinä olisi kenttä, johon syötetään poistettavan sarakkeen nimi, on siinä pudotusvalikko, josta käyttäjä voi valita poistettavan sarakkeen nopeasti ja helposti. Ohjelma tarkistaa myös taulun oletusavaimen, ja jättää sen listaamatta.

Kuvassa 5 näkyy esimerkkitaulukko. Taulukossa on useita rivejä ja sarakkeita, ja käyttäjä voi järjestää taulun sisällön laskevasti tai nousevasti minkä tahansa sarakkeen perusteella. Käyttäjän on myös mahdollista poistaa useita rivejä kerralla.

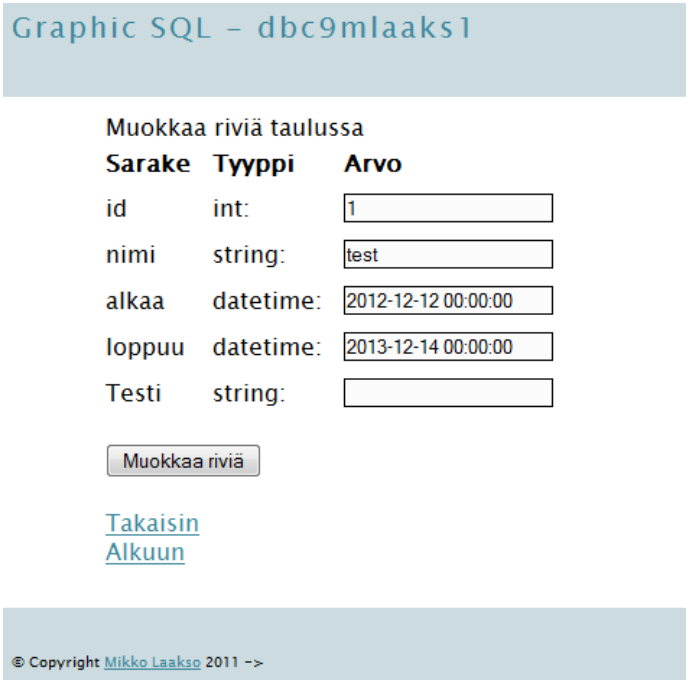


KUVA 5. Taulun sisältö

4.8.4 Rivin muokkaaminen

Kun käyttäjä haluaa muokata yksittäistä riviä tietokannassa, näyttää ohjelma eri kenttien nimet, ja tulostaa niiden jälkeen tuleviin kenttiin automaattisesti arvot, jotka niissä olivat. Käyttäjä voi muokata haluamaansa arvoa, ja klikattaessa *Muokkaa riviä* -painiketta, ohjelma tallentaa muutoksen tietokantaan. Mikäli kentän teksti on pidempi kuin 20 merkkiä, tulostaa ohjelma syöttökentän sijasta tekstialueen.

Kuvassa 6 on kuvattu esimerkkirivin lisääminen tietokantaan. Käyttäjän ei ole pakko täyttää kaikkia kenttiä, vaan hän voi myös jättää osan kentistä tyhjiksi.

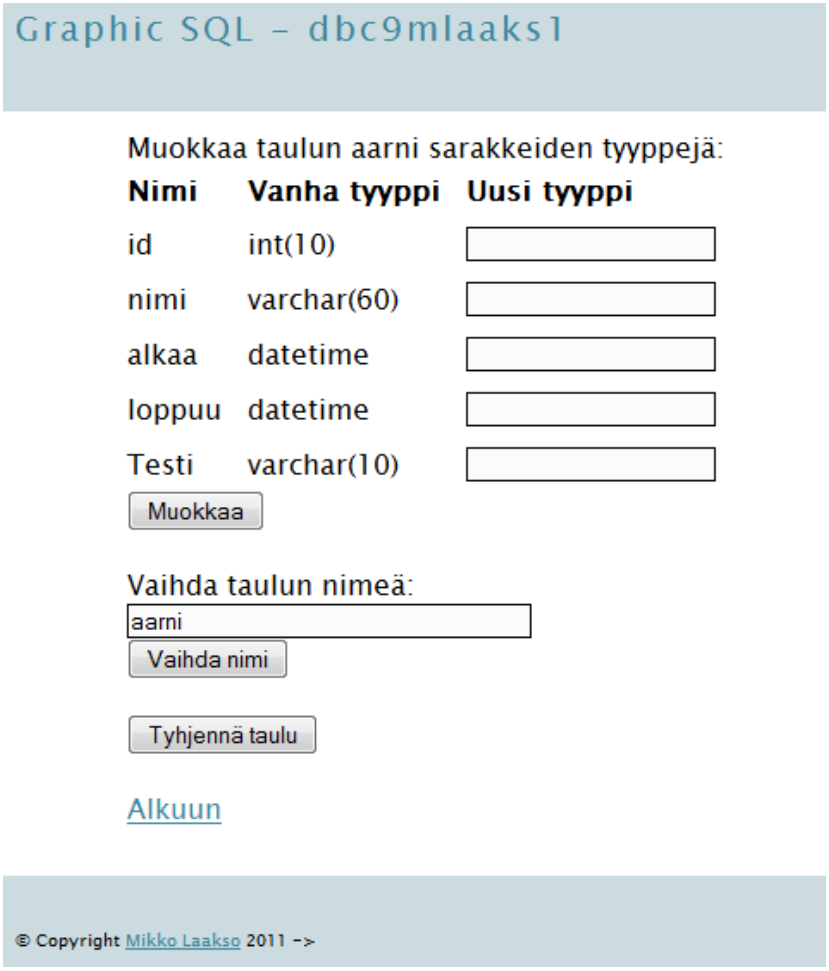


KUVA 6. Rivin muokkaaminen

4.8.5 Taulun muokkaaminen

Kuvassa 7 näkyvät kaikki taulun muokkaamiseen liittyvät toiminnot. Käyttäjä voi vaihtaa taulun nimen, tyhjentää taulun kokonaan tai vaihtaa taulun rivien tyyppejä.

Mikäli käyttäjä vaihtaa rivin tyypin, ei ohjelma tarkista, että kaikki kentissä olleet tiedot ovat sopivia myös uusiin kenttiin. Käyttäjän on siis itse tarkistettava, että kenttien tiedot ovat oikein myös muutoksen jälkeen. Käyttäjältä edellytetään myös perustietämystä MySQL-tietotyypeistä, sillä ohjelma ei tarjoa valmista listaa eri tietotyypeistä.



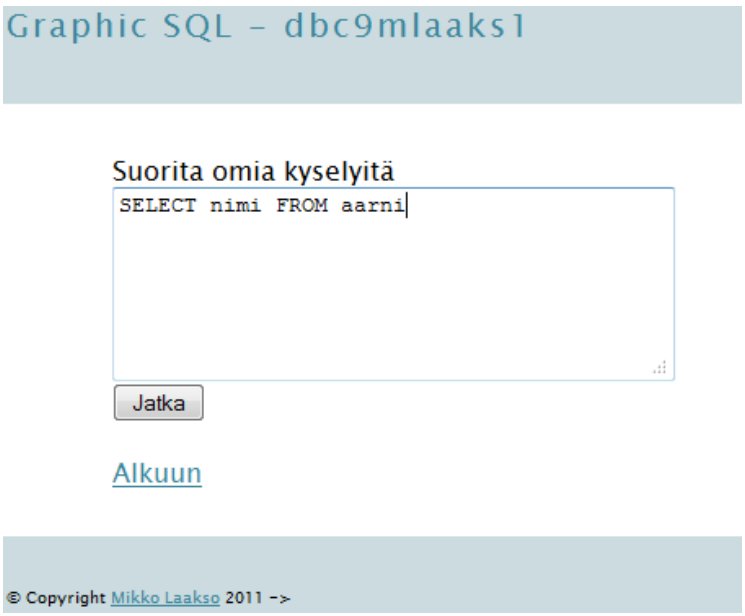
KUVA 7. Taulun muokkaaminen

4.8.6 Omien kyselyiden suorittaminen

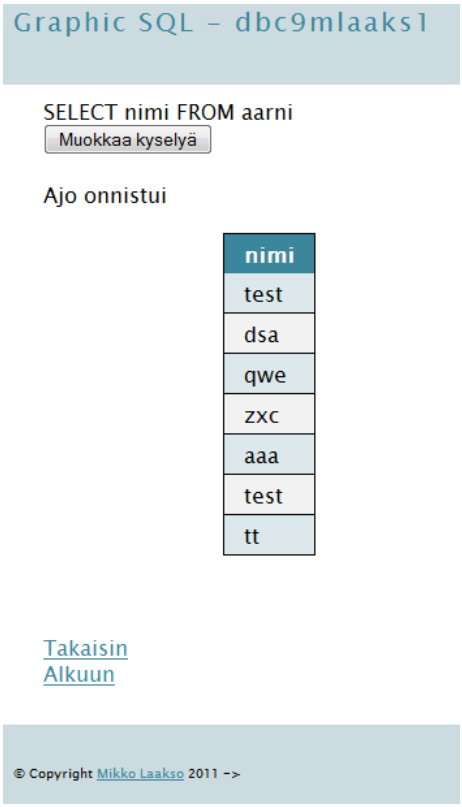
Vapaassa koodiajossa on mahdollista joko tulostaa tietoja tietokannasta, tai tehdä muutoksia tietokantaan.

Mikäli käyttäjä haluaa tulostaa tietokannan tietoja, hakee ohjelma halutut tiedot ja tulostaa ne näytöllä näytettyään ensin käyttäjän tietokantakyselyn.

Kuvissa 8 ja 9 käyttäjä hakee tietoa tietokannasta, ja ohjelma palauttaa kyselyn tuloksen näytölle.



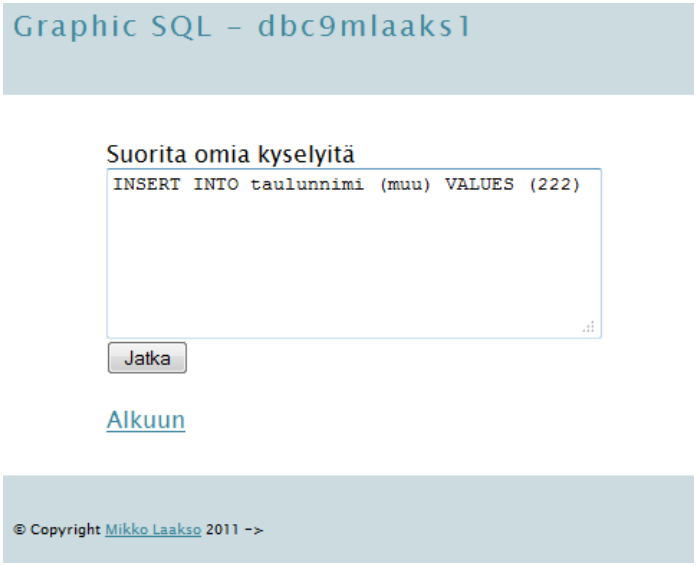
KUVA 8. Vapaan koodiajon kyselyikkuna



KUVA 9. Tietokantakyselyn tulos

Käyttäjä voi myös tehdä muutoksia tietokantaan, jolloin ohjelma tulostaa tietokanta-kyselyn, ja kertoo sen jälkeen, onnistuiko kannan muuttaminen.

Kuvissa 10 ja 11 käyttäjä yrittää lisätä uuden rivin tietokantaan, ja ohjelma kertoo ajon onnistuneen.



KUVA 10. Uuden rivin lisääminen tietokantaan

KUVA 11. Rivin lisääminen onnistui

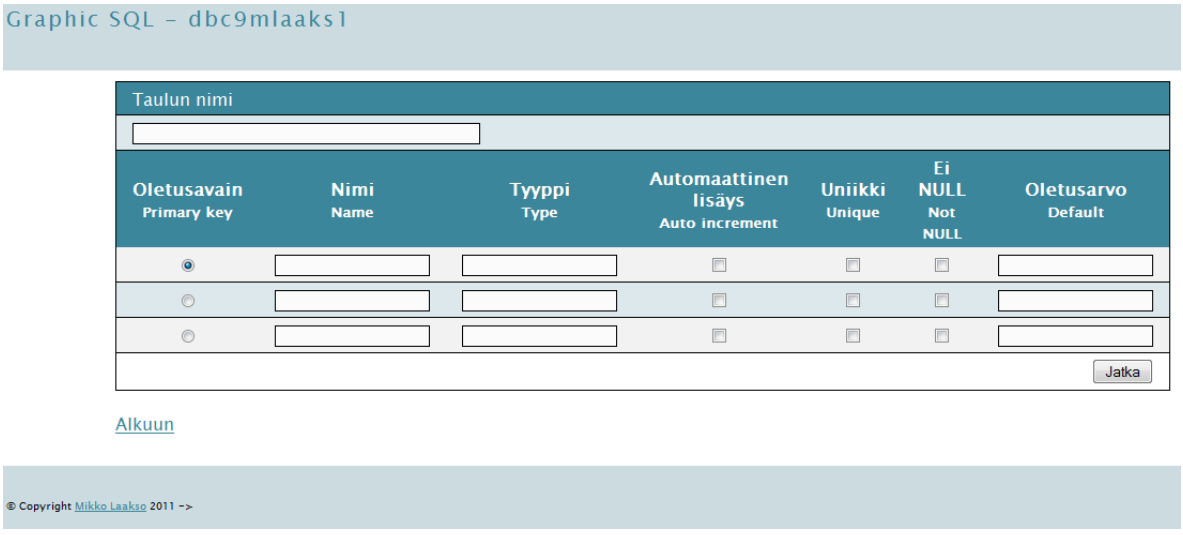
Mikäli käyttäjä tekee virheellisen kyselyn tai muuten haluaa palata takaisin koodiajoikkunaan, voi hän painaa *Takaisin*-painiketta, jolloin hänelle näytetään tyhjä koodikenttä. Mikäli hän sen sijaan painaa *Muokkaa kyselyä* -painiketta, siirtää ohjelma kyselyn takaisin koodiajoikkunaan, joten käyttäjän on helppo tehdä muutoksia kyselyyn, mikäli hän on tehnyt esimerkiksi huolimattomuusvirheen.

4.8.7 Taulun lisääminen

Lisättäessä uutta taulua, syöttää käyttäjä ensin haluamansa kenttien lukumäärän. Tämän jälkeen hän syöttää taulun nimen, kenttien nimet ja tiedot, sekä määrittää oletusavaimen (Primary Key). Käyttäjä voi myös valita oletusarvon kentän sisällölle. Taulukolla on aina oltava valittuna oletusavain.

Kuvassa 12 näkyy perusnäkymä uutta taulua lisättäessä, ja kuvassa 13 näkyy tyhjä 3 kentän taulu, johon käyttäjä voi syöttää haluamansa määritykset.

KUVA 12. Taulun lisääminen, osa 1



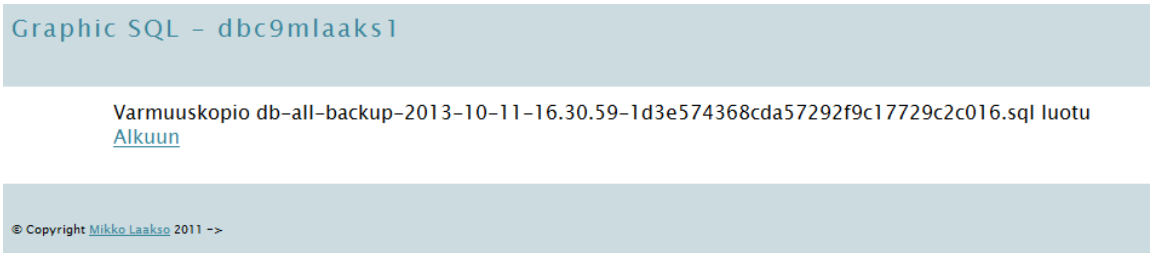
KUVA 13. Taulun lisääminen, osa 2

4.8.8 Taulujen varmuuskopioiminen

Halutessaan käyttäjä voi myös varmuuskopioida kaikki taulunsa kerralla. Kun hän painaa yleisnäkymässä painiketta *Varmuuskopioi taulut*, päätyy hän varmuuskopiointisivulle, joka luo automaattisesti varmuuskopion kaikista tauluista. Tämän jälkeen ohjelma avaa latausikkunan ja poistaa tiedoston palvelimelta. Käyttäjä voi joko ladata tai peruuttaa tiedoston latauksen, mutta jos hän peruuttaa sen, täytyy hänen tehdä uusi varmuuskopio ennen sen lataamista.

Koska varmuuskopiot ladataan välittömästi niiden tekemisen jälkeen, ei palvelimelle jää tilaa vieviä ja mahdollisesti arkaluonteisia varmuuskopioita. Automaattinen lataamisen aloittaminen oli hieman haastavaa, sillä tiedoston automaattinen lataaminen edellyttää header-tietojen muokkaamista ja lähettämistä, ja tätä ei voi tehdä, jos sivulla on jo lähetetty sisältöä. Ratkaisin ongelman tekemällä uuden sivun, mikä ensimmäisenä lataa tiedoston, ja ohjaamalla käyttäjän sinne.

Kuvassa 14 näkyy, mitä käyttäjälle näytetään, mikäli hän luo varmuuskopion onnistuneesti. Varmuuskopio nimetään mallilla db-[taulun nimi tai all]-backup-[päivämäärä]-[aika]-[salattu yhteenveto tauluista].sql.



KUVA 14. Taulujen varmuuskopiointi

4.8.9 Varmuuskopion palauttaminen

Jos käyttäjä haluaa palauttaa varmuuskopionsa, voi hän syöttää varmuuskopion sisällön vapaaseen koodiin, mutta helpommin se käy varmuuskopion palauttamisen kautta. Käyttäjä lataa tiedoston palvelimelle, ja mikäli se on oikean tyyppinen, alkaa ohjelma palauttamaan varmuuskopiota. Ohjelma tulostaa näytölle varmuuskopion sisällön ja ajaa

sen sisältämät komennot kantaan yksitellen. Kuvassa 15 näkyy varmuuskopion palautusikkuna.

KUVA 15. Varmuuskopion palautus

5 KÄYTTÖOHJEET

Tämän opinnäytetyöraportin liitteenä 3 on laatimani käyttöohje GSQL:n käyttöön.

Hyvä käyttöohje sisältää esimerkiksi sisällysluettelon, vianmääritysosan, kuvitusta sekä selkeät väliotsikot (Tietotekniikan projektityö 2013). Käyttöoppaalle on tarvetta, mikäli käyttäjä ei ole perehtynyt MySQL-tietokantojen hallintaan, tai mikäli hän ei löydä etsimäänsä toimintoa sovelluksesta. Sen suunnittelussa tulee kuitenkin huomioida, että käyttäjä ei välttämättä lue käyttöohjetta ollenkaan läpi, tai vain selailee sieltä muutamia kohtia.

Tavoitteenani oli tehdä järjestelmän käyttämisestä mahdollisimman yksinkertaista, helppoa ja intuitiivista, joten uskoakseni käyttöohjeen lukeminen ei ole tarpeen, mikäli käyttäjä on perehtynyt MySQL-tietokantojen käyttämiseen ennen sovellukseeni tutustumista. Siitä huolimatta ohjeesta on hyötyä kokeneemmallekin käyttäjälle, sillä se sisältää muun muassa taulukon, josta löytyvät MySQL-tietokantoihin sopivat tietotyypit.

Vaikka tohtori Hodgson (2013) suositteleekin tekemään fyysisen manuaalin, tulee käyttöohjeeni olemaan PDF-tiedosto, sillä myös koko sovellukseni on digitaalinen. Yleisten ohjeiden lisäksi Hodgson (2013) neuvoo, kuinka ohjekirja tulisi muotoilla. Hänen mukaansa käyttöoppaan teossa kannattaa käyttää mahdollisimman vähän erilaisia fontteja, ja niiden tulisi olla selkeitä, hyvin luettavia san-serif -fontteja. Hän suosittelee myös käyttämään kuvia, sillä kuten sananlaskukin sanoo, kuva kertoo enemmän kuin tuhat sanaa.

Hyvän käyttöohjeen pääkriteerit ovat sisällön paikkansapitävyys ja riittävyys, esitystavan ymmärrettävyys ja rakenteen sopiva jäsentyneisyys (Pöyhönen 1984). Pitää siis varmistaa, että ohje on ajan tasalla, ja tarkistaa, muuttuuko ohje kun sovellus muuttuu. Pitää myös varmistua siitä, että oppaassa olevat asiat ovat loogisessa järjestyksessä, ja että käyttäjä voi intuitiivisesti selata käyttöohjetta.

Kuten Ttietotekniikan projektityö (2013), myös Pöyhönen (1984), neuvovat tekemään käyttöohjeesta kuvitetun, sillä käyttäjän on helpompi ja miellyttävämpi lukea kuvitettua tekstiä. Koska kuvilla on myös helpompi havainnollistaa sovelluksen eri toimintoja ja näkymiä, tein käyttöohjeestani kuvitetun. Tavoitteena on saada käyttöohjeesta niin yk-

sinkertainen, että käyttäjä pystyy käyttämään sovellusta, vaikka hänellä olisikin vain perustaidot MySQL-tietokantojen hallinnassa, minkä vuoksi siinä on käsitelty päällisin puolin ohjelman sisältämät toiminnot. Tarkoituksena on myös se, että mikäli käyttäjä ei löydä etsimäänsä sovelluksesta, voi hän tarkistaa käyttöohjeesta, mistä kyseinen toiminto löytyy.

Käyttöohjeessa on myös lista mahdollisista tietotyypeistä, sillä tällä hetkellä sovellus ei anna vaihtoehtoja tyyppi-kenttiin, vaan käyttäjän on tiedettävä, minkä tyyppisiä sarakkeita MySQL tukee. Tämä auttaa sekä uusia käyttäjiä kertomalla heille mahdolliset arvot kenttään syötettäviksi että vanhoja käyttäjiä, jotka eivät välttämättä muista kaikkia tyyppejä ulkoa.

Kuten itse sovelluskin, tulee myös käyttöohje testata käyttäjillä (Pöyhönen. 1984). Testasin siis käyttöohjeen muutamalla käyttäjällä. Heidän annettiin lukea käyttöohje läpi, ja heiltä kysyttiin sen jälkeen, osaisivatko he mielestään käyttää sovellusta ohjeiden perusteella. Jos he olettivat kykenevänsä, heitä pyydettiin suorittamaan muutamia toimintoja samoin kuin toisessa käyttäjäkyselyssä.

6 POHDINTA

Opinnäytteen käytännön toteutusosan tarkoituksena oli toteuttaa graafinen MySQL-käyttöliittymä, ja pitää sovelluksen asentaminen ja käyttäminen mahdollisimman yksinkertaisina. Katsoessani GSQL:ää jälkikäteen tunnen päässeeni tavoitteeseeni. Alkuperäisessä suunnitelmassa oli vain osa siinä nyt olevista ominaisuuksista, sillä todettuani riittävän monta kertaa, miksi tätä toimintoa ei ole sovelluksessa, olen lisännyt siihen monia asioita, jotka ovat enemmän tai vähemmän tarpeellisia.

Opinnäytetyöraportin kirjoittaminen sen sijaan oli vaikeampaa. Vaikka sovellusta kirjoittaessa ajauduin useasti umpikujaan miettiessäni, miten jokin toiminto pitäisi toteuttaa, etenemään pääsi kuitenkin suhteellisen nopeasti. Opinnäytetyön kirjoittaminen sen sijaan tuntui jatkuvalta suossa tarpomiselta alusta loppuun.

Tunnen kuitenkin onnistuneeni opinnäytetyöni kirjallisen osuuden tavoitteessa. Tavoitteena oli esitellä GSQL, kertoa sen toteuttamisesta, sekä kertoa tällä hetkellä käytettävistä, muista samankaltaisista ohjelmista. Tarkoituksena oli myös kertoa PHP:sta ja MySQL:stä. Vaikka jälkimmäiset käytiinkin hyvin pintapuolisesti läpi, esitteltiin kuitenkin molempien historiaa, ja kerrottiin, kuinka käytin niitä opinnäytetyöni käytännön osassa.

Olisin toivonut voivani järjestää myös laajamittaisemman käyttäjätestauksen, mutta ikävä kyllä minulla ei ollut resursseja siihen. Voi olla, että joskus tulevaisuudessa, kun pystyn järjestämään kattavamman käyttäjätestauksen, puran sovellukseni osiksi ja kokoan sen uudelleen, toivottavasti entistä parempana. Olen myös tässä versiossa kokeillut hieman erilaisia tapoja saavuttaa sama asia, joten osa niistä saattaa vaikuttaa hieman tyhmiltä tavoilta toteuttaa jokin. Jos tulevaisuudessa teen sovellukseni uusiksi, vaihdan nämä asiat toimimaan kaikkialla samoin, käyttäen parasta tarjolla olevaa tapaa. Tärkein uudistus tuotekehityksen puitteissa tulee kuitenkin olemaan uusi tapa antaa ohjelmalle tietotyypit. Tällä hetkellä, mikäli käyttäjä vaikkapa haluaisi tauluunsa INT-tyyppisen sarakkeen, tulee hänen syöttää INT(pituus) tekstinä. Tämä häiritsee sekä uusia, että kokeneempia käyttäjiä, ja tarkoituksena onkin vaihtaa se helpompaan tapaan, kuten pudotusvalikkoon.

Vaikka ohjelma onkin opinnäytetyön puitteissa valmis, on siinä vielä paljon lisäiltävää ja paranneltavaa. Jatkokehityksessä on asioita kahdella listalla: niitä, jotka aion lisätä sovellukseen, kun aikaa riittää, ja niitä, joita en vielä osaa toteuttaa. Kuitenkin ennen kuin lisään mitään toimintoja sovellukseen, pitää niiden olla sekä hyödyllisiä, että mutkattomia. Aion vastaisuudessakin pitää sovelluksen mahdollisimman yksinkertaisena käyttää, joten osa liian monimutkaisista ideoista on pakko jättää pois.

LÄHTEET

DB-Engines. 2013. Luettu 16.11.2013
http://db-engines.com/en/system/MySQL%3BOracle

Emacswiki: Coffee mode. 2013. Luettu 27.11.2013
http://www.emacswiki.org/emacs/CoffeeMode

HeidiSQL. 2013. Luettu 11.7.2013
http://www.heidisql.com/

Heinisuo, R & Rauta, I. 2007. PHP ja MySQL : tietokantapohjaiset verkkopalvelut. Jyväskylä: Gummerus.

Krug, S. 2006. Älä pakota minua ajattelemaan. Jyväskylä: Gummerus.

Lea, C., Buzzard, M., White-Cinis, J & Thomas, D. 2003. PHP MYSQL Website Programming. Birmingham: Wrox Press.

MySQL. 2013. Luettu 15.11.2013
http://www.mysql.com/

Nielsen Norman Group. 2000. Luettu 6.11.2013
http://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/

PHP. 2013. Luettu 19.11.2013
http://www.php.net/usage.php

phpMyAdmin. 2013. Luettu 22.3.2013
www.phpMyAdmin.net

Pöyhönen, M. 1984. Käyttöohje on osa tuotetta. Käyttöohjeen laatijan opas. Helsinki: Pohjola-Yhtiöt

Rantala, A. 2005. Web-ohjelmointi. Jyväskylä : Docendo.

Tietotekniikan projektityö. 2013. Tampereen Teknillinen yliopisto. Luettu 31.10.2013
http://www.cs.tut.fi/~projekti/dokumentit/sunnumv_kayttoohje2005.pdf

Hodgson, P. 2013. Userfocus. Luettu 19.11.2013
http://www.userfocus.co.uk/articles/usermanuals.html

Wiio, A. 2004. Käyttäjäystävällisen sovelluksen suunnittelu. Helsinki : IT Press.

Wikipedia. 2013a. Luettu 24.5.2013
http://en.wikipedia.org/wiki/MySQL

Wikipedia. 2013b. Luettu 24.5.2013
http://en.wikipedia.org/wiki/Php

Wikipedia. 2013c. Luettu 24.5.2013
http://en.wikipedia.org/wiki/PhpMyAdmin

LIITTEET

Liite 1. Ensimmäisen käyttäjäkyselyn kysymykset

- Kuinka paljon sinulla on kokemusta MySQL-tietokantojen käytöstä?
- Miten tällä hetkellä otat yhteyden tietokantaan?
- Mitkä ovat tämän menetelmän hyvät ja huonot puolet?
- Mitä toimintoja käytät eniten, kun olet tekemisissä tietokantasi kanssa?
- Minkä toiminnot toivoisit olevan helpommin käytettävä, kuin mitä se nyt on?
- Miltä GSQL näyttää ensi silmäyksellä?
- Selvitys siitä, kuinka intuitiivista ohjelman käyttö on:
    - Jos sinun pitäisi lisätä rivi tietokantaan, mitä tekisit?
    - Jos haluaisit vaihtaa kentän arvoa rivissä 2, miten toimisit?
    - Miten varmuuskopioisit tietokantasi?
    - Haluat saada selville montako riviä taulussasi on. Mitä teet?
    - Kuinka lisäisit uuden taulun?

Liite 2. Toisen käyttäjäkyselyn kysymykset

- Kuinka paljon sinulla on kokemusta MySQL-tietokantojen käytöstä?
- Miltä GSQL näyttää ensi silmäyksellä?
- Jos sinun pitäisi lisätä rivi tietokantaan, mitä tekisit?
- Jos haluaisit vaihtaa kentän arvoa rivissä 2, miten toimisit?
- Miten varmuuskopioisit tietokantasi?
- Haluat saada selville montako riviä taulussasi on. Mitä teet?
- Kuinka lisäisit uuden taulun?
- Miten toiminnot ovat mielestäsi sijoitetut? Pitäisikö jonkin toiminnon sijaita jossakin muualla?
- Löysitkö helposti etsimäsi toiminnot?
- Onko ohjelmassa turhia toimintoja, jotka voisi ottaa pois?
- Puuttuuko ohjelmasta toimintoja, joiden toivoisit siellä olevan?

Liite 3. Käyttöohje

Käyttöohjeet GSQL:n käyttöön

SISÄLLYS

1 Asennus

Asenna ohjelma purkamalla paketti palvelimella kansioon, jonne saa Internet-yhteyden. Varmista, että kansioiden sekä tiedostojen käyttöoikeudet ovat oikein. Sen jälkeen navigoi internetselaimella kansioon.

7 Kirjautuminen

Kirjaudu syöttämällä käyttäjätunnuksesi (esim. c9kkayttaj), salasanasi (esim. Salas4na), host eli isäntä (esim. mydb.tamk.fi) sekä tietokantasi nimi (esim. mydatabase1). Valitse voimassaoloaika kirjautumisellesi. Paina *Lähetä*, ja mikäli tunnuksesi ovat oikein, siirryt päänäkymään. Kuvassa 1 on näkymä siitä, miltä sisäänkirjautumisikkunan pitäisi näyttää.

Kuva 1. Sisäänkirjautumisikkuna.

Mikäli olet unohtanut tietokantasi nimen, mutta muistat muut tiedot, voit syöttää tarvittavat tiedot ja painaa *Näytä tietokannat* -painiketta. Tällöin ohjelma ottaa tietokantayhteyden tunnuksillasi ja tulostaa listan tietokannoistasi.

8 Tietokannan näyttäminen

Päänäkymässä voit taulua klikkaamalla tarkastella sen sisältöä ja *Muokkaa*-painiketta painamalla muokata sitä. Laittamalla ruksin *Poista*-valintalaatikkoon ja painamalla *Poista valitut*, voit poistaa yhden tai useamman taulun tietokannasta.
Suorita koodia -painiketta painamalla pääset ajamaan oman tietokantakyselysi.
Lisää taulu antaa sinun luoda uuden taulun.
Varmuuskopioi taulut -linkistä pääsee ottamaan varmuuskopion kaikista tietokannan tauluista.
Vaihda kirjautumistunnuksia antaa kirjautua uudelleen uusilla tunnuksilla tai toiseen tietokantaan. Ohjelman voi sulkea *Kirjaudu ulos* – linkkiä painamalla.

Kuvassa 2 näkyy esimerkkitietokannan taulut. Mikäli tietokannassasi ei ole tauluja, ruudulla näytetään ilmoitus *Tietokantasi on tyhjä*.

Kuva 2. Tietokannan sisältö

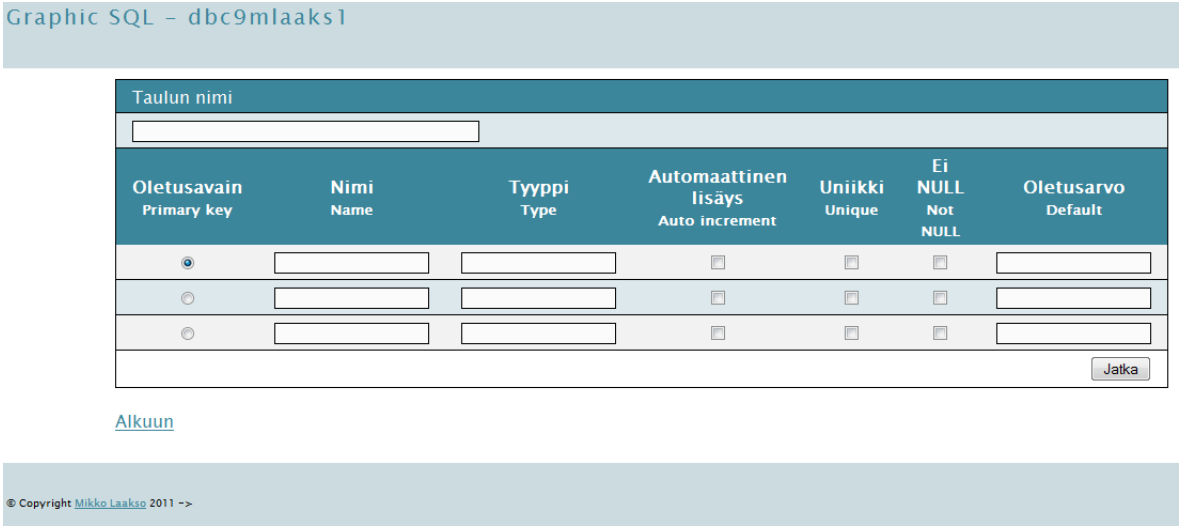
8.1 Taulun lisääminen

Taulun lisääminen aloitetaan syöttämällä, montako saraketta tauluun on tulossa. Syötä numero (esim. 3) ja paina *Jatka* -painiketta. Kuvassa 3 on esimerkki siitä, miltä ikkunan kuuluisi näyttää.

Kuva 3. Taulun lisääminen, osa 1

Tämän jälkeen jatka syöttämällä tauluun nimi, ja sen jälkeen jokaiselle taulun sarakkeelle nimi, tyyppi sekä muut arvot. Taululla pitää olla oletusavain.

Kuvassa 4 näkyy tyhjä taulun lisäämisikkuna, jossa haluttiin olevan 3 saraketta.



Kuva 4. Taulun lisääminen, osa 2

Tyyppi -kenttään syötetään joko tietokentän tyyppi kuten Date, tai tyyppi, joka vaatii muuttuvan arvon (esim. varchar) muodossa muuttuja(arvo), kuten Varchar(10).

Alla olevassa taulukossa on osittain suomeksi käännetyt tietokenttien tyypit. Lihavoidut tyypit vaativat muuttujan, kun taas lihavoimattomia voi käyttää sellaisenaan.

	Yleiset
INT	4-tavuinen kokonaisluku, etumerkillisenä arvot ovat -2147483648 - 2147483647, etumerkittämänä arvot ovat 0 - 4294967295.
VARCHAR	A variable-length (0-65,535) string, the effective maximum length is subject to the maximum row size
TEXT	A TEXT column with a maximum length of 65,535 (2^16 - 1) characters, stored with a two-byte prefix indicating the length of the value in bytes
DATE	Päiväys, pitää olla välillä 1000-01-01 - 9999-12-31

	Numeeriset
TINYINT	1-tavuinen kokonaisluku, etumerkillisenä arvot välillä -128 - 127, etumerkittömänä arvot välillä 0 - 255
SMALLINT	2-tavuinen kokonaisluku, etumerkillisenä arvot olvat -32768 - 32767, etumerkittömänä arvot ovat 0 - 65535
MEDIUMINT	3-tavuinen kokonaisluku, etumerkillisenä arvot ovat -8388608 - 8388607, etumerkittömänä arvot ovat 0 - 16777215
INT	4-tavuinen kokonaisluku, etumerkillisenä arvot ovat -2147483648 - 2147483647, etumerkittämänä arvot ovat 0 - 4294967295.
BIGINT	8-tavuinen kokonaisluku, etumerkillisenä arvot ovat -9223372036854775808 - 9223372036854775807, etumerkittämänä arvot ovat 0 - 18446744073709551615
DECIMAL	Desimaaliluku (M,D) - kokonaislukuosan numeroiden maksimäärä (M) on 65 (oletuksena 10), desimaaliosan numeroiden maksimäärä (D) on 30 (oletuksena 0)
FLOAT	Pieni liukuluku, mahdolliset arvot ovat välillä -3.402823466E+38 - -1.175494351E-38, 0 ja 1.175494351E-38 - 3.402823466E+38
DOUBLE	Kaksoistarkkuuksinen liukuluku, mahdolliset arvot ovat välillä -1.7976931348623157E+308 - -2.2250738585072014E-308, 0, ja 2.2250738585072014E-308 - 1.7976931348623157E+308
REAL	Synonyymi DOUBLE:lle (poikkeus: REAL_AS_FLOAT SQL moodissa se on synonyymi FLOAT:lle)
BIT	Bittikenttä tyyppi (M), sisältää M verran bittejä arvoa kohti (oletuksena on 1, maksimi on 64)
BOOLEAN	A synonym for TINYINT(1), a value of zero is considered false, nonzero values are considered true
SERIAL	An alias for BIGINT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE

	Päiväys ja aika
DATE	Päiväys, pitää olla välillä 1000-01-01 - 9999-12-31
DATETIME	A date and time combination, supported range is 1000-01-01 00:00:00 to 9999-12-31 23:59:59
TIMESTAMP	A timestamp, range is 1970-01-01 00:00:01 UTC to 2038-01-09 03:14:07 UTC, stored as the number of seconds since the epoch (1970-01-01 00:00:00 UTC)
TIME	Aika, mahdollinen välillä -838:59:59 - 838:59:59
YEAR	A year in four-digit (4, default) or two-digit (2) format, the allowable values are 70 (1970) to 69 (2069) or 1901 to 2155 and 0000
	Merkkijono
CHAR	A fixed-length (0-255, default 1) string that is always right-padded with spaces to the specified length when stored
VARCHAR	A variable-length (0-65,535) string, the effective maximum length is subject to the maximum row size
TINYTEXT	A TEXT column with a maximum length of 255 (2^8 - 1) characters, stored with a one-byte prefix indicating the length of the value in bytes
TEXT	A TEXT column with a maximum length of 65,535 (2^16 - 1) characters, stored with a two-byte prefix indicating the length of the value in bytes
MEDIUMTEXT	A TEXT column with a maximum length of 16,777,215 (2^24 - 1) characters, stored with a three-byte prefix indicating the length of the value in bytes
LONGTEXT	A TEXT column with a maximum length of 4,294,967,295 or 4GiB (2^32 - 1) characters, stored with a four-byte prefix indicating the length of the value in bytes
BINARY	Similar to the CHAR type, but stores binary byte strings rather than non-binary character strings
VARBINARY	Similar to the VARCHAR type, but stores binary byte strings rather than non-binary character strings
TINYBLOB	A BLOB column with a maximum length of 255 (2^8 - 1) bytes, stored with a one-byte prefix indicating the length of the value
MEDIUMBLOB	A BLOB column with a maximum length of 16,777,215 (2^24 - 1) bytes, stored with a three-byte prefix indicating the length of the value
BLOB	A BLOB column with a maximum length of 65,535 (2^16 - 1) bytes, stored with a two-byte prefix indicating the length of the value

LONGBLOB	A BLOB column with a maximum length of 4,294,967,295 or 4GiB (2^32 - 1) bytes, stored with a four-byte prefix indicating the length of the value
ENUM	An enumeration, chosen from the list of up to 65,535 values or the special '' error value
SET	A single value chosen from a set of up to 64 members

	Geometrinen
GEOMETRY	A type that can store a geometry of any type
POINT	A point in 2-dimensional space
LINESTRING	A curve with linear interpolation between points
POLYGON	Monikulmio
MULTIPOINT	A collection of points
MULTILINESTRING	A collection of curves with linear interpolation between points
MULTIPOLYGON	A collection of polygons
GEOMETRYCOLLECTION	A collection of geometry objects of any type

8.2 Taulujen poistaminen

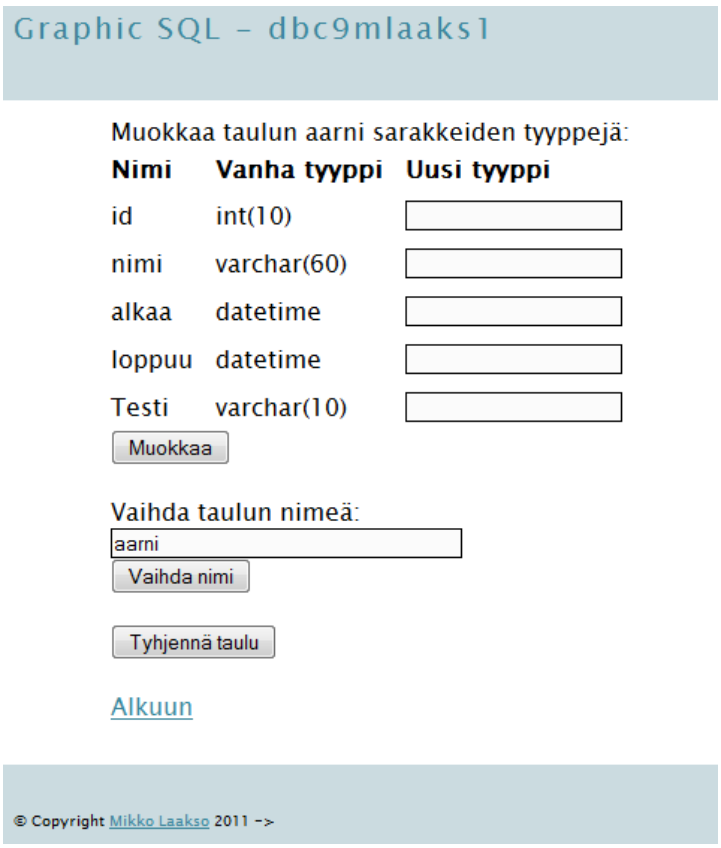
Taulujen poistaminen tapahtuu päänäkymässä valitsemalla poistettavien taulujen valintaruudut ja klikkaamalla *Poista valitut* -painiketta. Ohjelman kysyessä, haluatko varmasti poistaa taulut, paina OK-painiketta.

8.3 Taulun muokkaaminen

Taulun muokkauksessa voit vaihtaa taulun sarakkeiden tyyppiä toiseen. Katso tietokantatyypit kohdasta *Taulun lisääminen*. Ohjelma ei tarkista, sopivatko rivien arvot sarakkeisiin muutoksen jälkeen, vaan käyttäjän on huolehdittava tästä itse.

Taulun muokkaamisessa voit myös vaihtaa taulun nimeä (se ei kuitenkaan saa olla jo tietokannassa esiintyvä taulun nimi) sekä tyhjentää taulun sisällön.

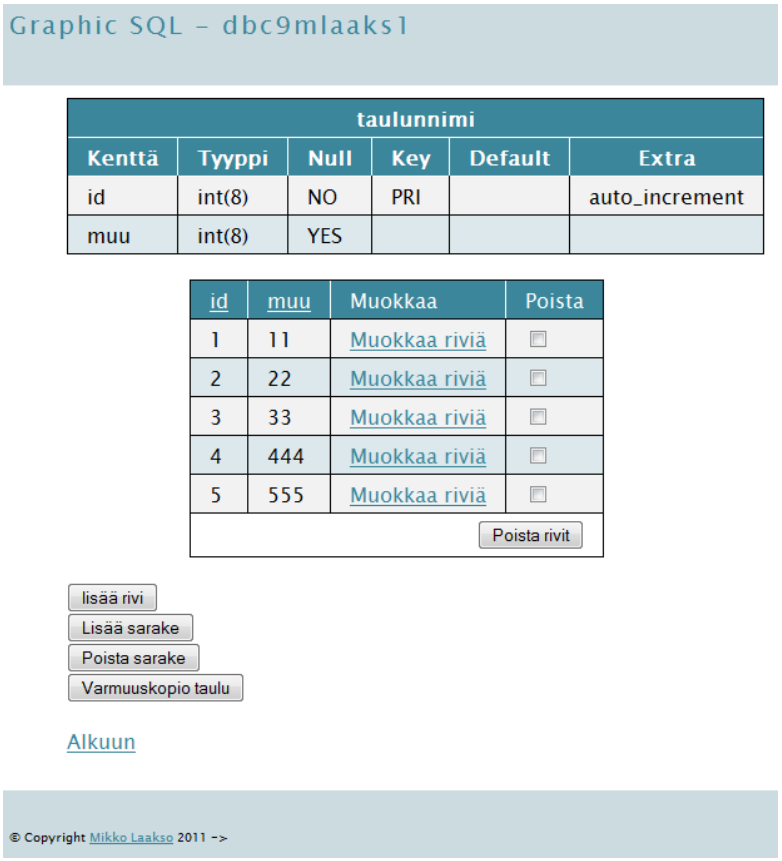
Kuvassa 5 on esimerkki taulun muokkaussivusta.



Kuva 5. Taulun muokkaaminen

9 Taulun sisällön näyttäminen

Yksittäisestä taulusta näytetään ensin sen sarakkeiden nimet, tyypit ja muut tiedot, ja sen alla taulun rivit. Kuvassa 6 näkyy esimerkkitaulu nimeltä taulunnimi, ja sen sisältö.



Kuva 6. Taulun sisältö

Voit järjestää rivit uudelleen klikkaamalla sen sarakkeen otsikkoa, jonka perusteella haluat rivit järjestää. Ensimmäinen kerta järjestää rivit nousevasti ja toinen laskevasti.
Voit myös ottaa varmuuskopion yksittäisestä taulusta painamalla *Varmuuskopioi taulu* -painiketta.

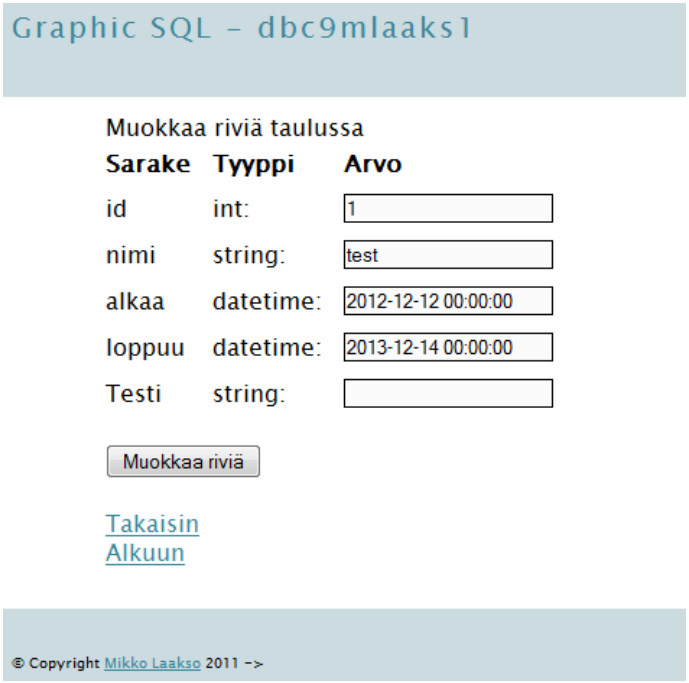
9.1 Rivin lisääminen tauluun

Painamalla L*isää rivi* -painiketta voit lisätä uuden rivin. Syötä arvot haluamiisi kohtiin (kaikkia kohtia ei ole pakko täyttää ellet ole taulua toisin määrittänyt) ja paina *lisää rivi*-painiketta.

9.2 Taulun rivien muokkaaminen

Muokkaa riviä syöttämällä haluamasi arvot kenttiin ja paina *Muokkaa riviä* -painiketta. Oletuksena jokainen syöttökenttä on input-tyyppinen, mutta mikäli tekstin pituus on yli 20 merkkiä, muuttuu syöttökenttä tekstialueeksi.

Kuvassa 7 muokataan yhtä riviä taulussa.



Kuva 7. Taulun rivin muokkaaminen

9.3 Taulun rivien poistaminen

Taulun rivit poistetaan valitsemalla poistettavien rivien valintalaatikot ja painamalla *Poista rivit*-painiketta. Ohjelma näyttää poistettavat rivit, ja mikäli olet varma siitä, että haluat poistaa ne, paina *Haluan varmasti poistaa rivit* -painiketta.

9.4 Sarakkeen lisääminen

Sarake lisätään syöttämällä sarakkeen nimi ja tyyppi. Katso sopivat tyypit kohdasta *Taulun lisääminen*.

9.5 Sarakkeen poistaminen

Sarake poistetaan painamalla *Poista sarake* -painiketta ja syöttämällä sarakkeen nimi. Ohjelma ei anna poistaa oletusavainsaraketta, sillä sitä tarvitaan muun muassa taulun rivien poistamiseen. Sen lisäksi on suositeltu, että taululla olisi aina yksi oletusavain.

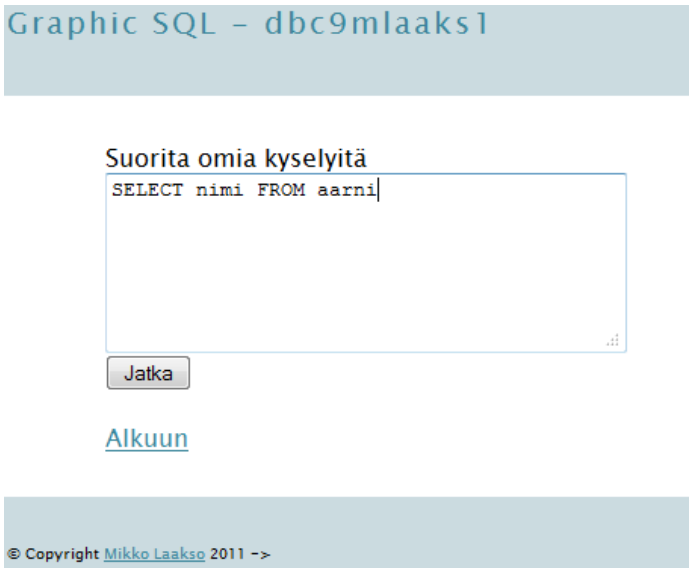
Mikäli kuitenkin haluat poistaa sen, mene vapaaseen koodiajoon (*Suorita koodia* päänäkymässä) ja kirjoita kenttään kysely ALTER TABLE taulu DROP sarake.

9.6 Taulun varmuuskopiointi

Painamalla *Varmuuskopioi taulu* -painiketta ohjelma luo kyseisestä taulusta varmuuskopion ja käynnistää sen latauksen. Kun lataus on valmis tai se peruutetaan, poistuu varmuuskopio palvelimelta.

10 Omien kyselyiden suorittaminen eli vapaa koodiajo

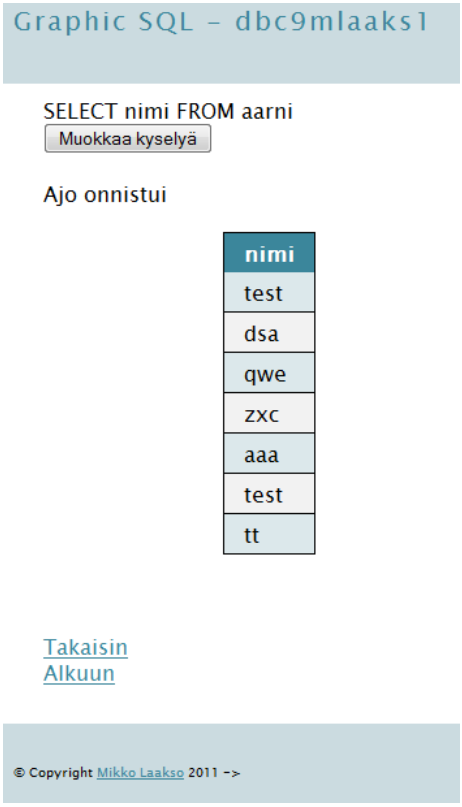
Päänäkymän *Suorita koodia* -painiketta painamalla voit suorittaa omia tietokanta-kyselyitä. Kirjoita kyselysi laatikkoon ja paina *Jatka*-painiketta. Kuvassa 8 on esimerkkikysely, joka palauttaa kaikken rivien nimi-sarakkeen tiedot taulusta aarni.



Kuva 8. Vapaa koodiajo

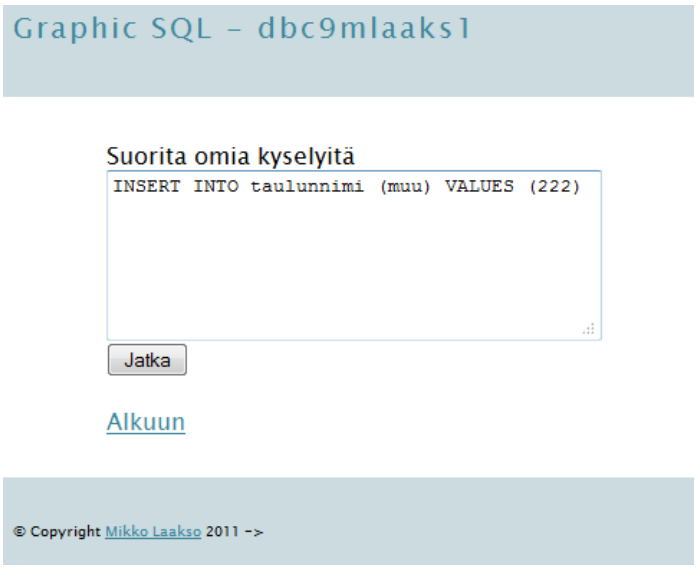
Mikäli kyselysi haki tietoa kannasta, tulostaa sovellus alla olevan kuvan mukaisesti taulukon, jossa on hakemasi tiedot.

Kuvassa 9 näkyvät ohjelman palauttamat rivit, kun kysely onnistuneesti suoritettiin.



Kuva 9. Vapaan koodiajon tulokset.

Voit myös suorittaa kyselyitä, jotka eivät hae tietokannasta tietoja, vaan muokkaavat sitä. Kuvassa 10 on esimerkki tälläisestä kyselystä. Kuvan 10 esimerkki lisää tauluun taulunnimi uuden rivin, jossa on vain sarakkeessa muu arvo 222.



Kuva 10. Muokkaava tietokantakysely

Tällöin ohjelma tulostaa kyselysi ja kertoo, onnistuiko tietokannan muokkaaminen. Kuvassa 11 suoritettiin kuvan 10 kysely, ja ohjelma palautti ilmoituksen kyselyn onnistumisesta.

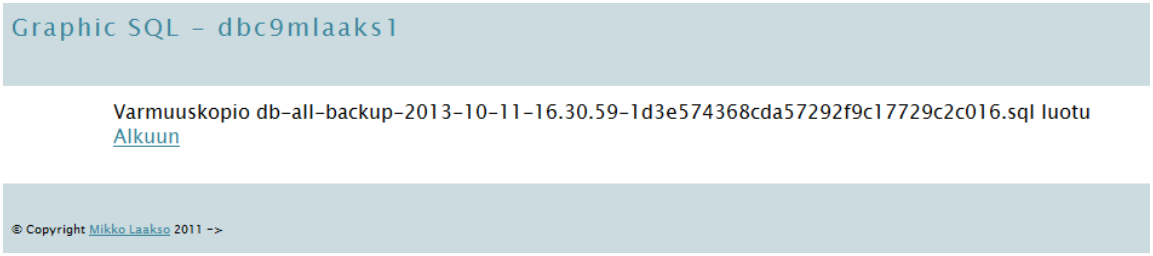
Kuva 11. Muokkaavan kyselyn tulos.

Mikäli kyselysi ei onnistunut, tai haluat jostain muusta syystä palata takaisin kyselyikkunaan niin, että nykyinen kyselysi tulee mukaan, paina painiketta *Muokkaa kyselyä*. Sovellus ohjaa sinut takaisin edelliseen ikkunaan ja tulostaa kyselysi valmiiksi tekstilaatikkoon.

11 Tietokannan varmuuskopiointi

Painamalla päänäkymässä *Varmuuskopioi taulut* –painiketta ottaa ohjelma kaikista tauluistasi varmuuskopion, ja käynnistää automaattisesti varmuuskopion lataamisen. Kun olet ladannut varmuuskopion tai peruuttanut sen lataamisen, poistetaan varmuuskopio palvelimelta.

Kuvassa 12 näkyy käyttäjälle tulostuva tieto tietokannan luomisen onnistumisesta.



Kuva 12. Tietokantavarmuuskopion ottaminen.

12 Varmuuskopion palauttaminen

Palauta varmuuskopio -kohdassa klikkaa painiketta *Selaa*, ja navigoi kansioon, johon tallensit sovelluksen kautta otetun varmuuskopiotiedoston. Valitse tiedosto, paina *Avaa*-painiketta ja klikkaa sen jälkeen painiketta *Lähetä*, niin ohjelma palauttaa varmuuskopiosi. Kuvassa 13 on varmuuskopion palauttamisikkuna ennen tiedoston valitsemista.

Kuva 13. Varmuuskopion palauttaminen

13 Uloskirjautuminen ja tunnuksien vaihtaminen

Painamalla päänäkymässä linkkiä *Vaihda kirjautumistunnuksia* pääset takaisin kirjautumisikkunaan, ja voit syöttää uudet tunnuksesi (kuten kirjautua toisena käyttäjänä tai toiseen tietokantaan) kirjautuaksesi uudelleen sisään.

Mikäli vaihdat kirjautumistunnuksia, vanhat tunnuksesi ovat edelleen voimassa ennen kuin syötät onnistuneesti seuraavat tunnukset.

Jos haluat poistua sovelluksesta turvallisesti, paina päänäkymässä linkkiä *Kirjaudu ulos*.

14 Lähteet

Osittain suomennettu lista sopivista tietotyypeistä - phpMyAdmin. 2013. www.phpmyadmin.net/