jamk.fi

Verkkopohjaisen vuorolistan toteutus

Joni Launonen

Opinnäytetyö
Toukokuu 2015

Tietotekniikan koulutusohjelma
Tekniikan ja liikenteen ala

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

KUVAILULEHTI

Tekijä(t) Launonen, Joni	Julkaisun laji Opinnäytetyö	Päivämäärä 21.05.2015
	Sivumäärä 40 + 18	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty (X)

Työn nimi
Verkkopohjaisen vuorolistan toteutus

Koulutusohjelma
Tietotekniikan koulutusohjelma

Työn ohjaaja(t)
Antti Häkkinen
Mika Rantonen

Toimeksiantaja(t)
TeliaSonera Oyj
Control center

Tiivistelmä

Tämän opinnäytetyön toimeksiantajana toimi TeliaSonera Oyj ja sen osasto Control center. Osaston työvuoroja pidettiin Excel-sovelluksessa, jota oli automatisoitu laskentakaavioiden avulla. Vuorolistasta oli tullut ajan saatossa monimutkainen, ja siinä olevat kaavat rikkoutuivat helposti usean muokkaajan seurauksena. Opinnäytetyöni tarkoituksena oli rakentaa täysin uusi, usean rinnakkaisen käyttäjän mahdollistava järjestelmä työvuorojen ylläpitämistä varten ja suunnitella se vastaamaan osaston tarpeita.

Opinnäytetyö päätettiin toteuttaa tietokannan ja verkkosivujen avulla, ja sen tarkoituksena on helpottaa työvuoroja ylläpitävien henkilöiden työkuormaa. Opinnäytetyön suunnitteluvaiheessa otettiin huomioon vuorolistan laajentaminen erillisillä moduuleilla, jotka voidaan liittää osaksi järjestelmää. Työssä perehdyttiin tietokantoihin ja niiden rakenteisiin sekä verkkosivujen toteutukseen ja niiden eri elementteihin.

Opinnäytetyön lopputuloksena saatiin rakennettua täysin uusi järjestelmä, joka voitiin ottaa käyttöön tuotannossa. Työn vaatimusmäärittelyinä toimi aiempi vuorolista ja siinä olevat ominaisuudet, jotka uuden järjestelmän täytyi mahdollistaa. Opinnäytetyössä saatiin toteutettua vaatimusmäärittelyjä vastaava järjestelmä ja järjestelmä todettiin hyvin toimivaksi.

Avainsanat (asiasanat)
CSS, HTML, JQuery, MySQL, MySQL-injektio, PHP, tietokanta, verkkopohjainen, vuorolista

Muut tiedot

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

DESCRIPTION

Author(s)	Type of publication	Date
Launonen, Joni	Bachelor´s Thesis	21.05.2015
	Pages 40 + 18	Language Finnish
		Permission for web publication (X)

Title
Implementation of a web-based roster

Degree Programme
Information Technology

Tutor(s)
Antti Häkkinen
Mika Rantonen

Assigned by
TeliaSonera Plc.
Control Center

Abstract

This thesis was assigned by TeliaSonera Plc. and the Control center department. Work rosters of the department are saved in Excel spreadsheet automated with formulas. Excel has become very complicated and formulas are broken easily because of many editors. The thesis aims to build a whole new system, multi-user concurrent permitting system that could maintain rosters of the department and corresponds to the requirements of the department.

It was decided to implement the assignment with a database and website to make the maintenance easier and decrease the administrators' workload. Expanding of the system with extra modules was taken into account in the planning phase of the thesis. This thesis focuses on the database and its structures together with the design of the website and its different elements.

The end result of the thesis was a completely new system for rosters which can be deployed in production. The requirement specifications of the thesis included the previous Excel and its features. The thesis was to enable the features of the previous roster Excel spreadsheet as well and it succeeded in the assignment very well.

Keywords
CSS, database, HTML, JQuery, MySQL, MySQL-injection, PHP, roster, web-based

Miscellaneous

Sisältö

KUVIOT

Käsitteitä

AD	Active Directory, Windows-palvelimella oleva käyttäjätilien hallinta
Apache	Verkkosivuja pyörittävä palvelu
APT	Advanced Package Tool, Debian-pohjaisten käyttöjärjestelmien pakettienhallintasovellus
Class 10	Muistikortin nopeusluokka, 10 Mt/s
CLI	Command Line Interface, tekstipohjainen käyttöliittymä
CSS	Cascading Style Sheets, käytetään WWW-sivujen tyyliohjeiden määrittämiseen
GPU	Graphic Processing Unit, grafiikkaprosessori
HTML	HyperText Markup Language, Internetsivujen koodauskieli
HTTP	HyperText Transfer Protocol, selaimen ja palvelimen välillä kulkevaan tiedonsiirtoon tehty protokolla
JQuery	JavaScript-kirjasto, joka sisältää erilaisia funktioita
LAMP	Linux, Apache, MySQL & PHP
MySQL	Tietokantasovellus
MySQLi	PHP:n laajennus, joka mahdollistaa tietokantaan yhdistämisen PHP:n avulla
PHP	HyperText Preprocessor, web-palveluiden tuottamisessa käytetty ohjelmointikieli
VPN	Virtual Private Network, julkisen verkon ylitse luotava näennäinen yksityisverkko, jossa koneet kokevat olevansa samassa verkossa.
YUM	Yellowdog Updater Modified, Red Hat -pohjaisten käyttöjärjestelmien pakettienhallintasovellus

1 Työn lähtökohdat

1.1 TeliaSonera

TeliaSonera on monikansallinen teleoperaattori ja yksi suurimmista Suomessa toimivista operaattoreista, jolla on rakennettuna omaa verkkoa. Suomalainen Sonera yhdistyi vuonna 2002 ruotsalaisen Telian kanssa ja on siitä lähtien ollut osa TeliaSonera Oyj:tä. Suomessa ovat käytössä nimikkeet TeliaSonera ja Sonera. Yhtiö työllistää noin 26 000 henkilöä maailmanlaajuisesti. (TeliaSonera in brief 2014, 2.)

1.2 Lähtökohdat

Control center -osaston työvuoroja on pidetty Excel-taulukossa, jonka hallinnointi on ollut työlästä. Excelin toimintaa on helpotettu kaavojen avulla, jolloin toimintoja on pystytty automatisoimaan. Vuorolista on myös kaavojen seurauksena mennyt monimutkaisemmaksi hallita, ja useiden ylläpitäjien takia kaavat ovat rikkoutuneet helposti ja niitä on jouduttu toistuvasti korjaamaan. Työvuorolista ei ole myöskään mahdollistanut tilastojen tekemistä työvuoroista, ja Excelin asettamien rajoitteiden vuoksi ainoastaan yksi henkilö kerrallaan voi pitää vuorolistaa muokattavassa tilassa. Myöskään työntekijöillä katselutilassa oleva Excel-tiedosto ei saa vuorolistaan tehtyjä muutoksia, vaan tiedosto pitää avata uudestaan verkosta, jotta päivitykset tulevat näkyviin.

1.3 Työn tavoitteet

Tavoitteena oli päästä eroon Excelin aiheuttamista rajoitteista, mahdollistetaan usean henkilön yhtäaikainen vuorolistan hallinnointi, päästään eroon rikkoutuvista kaavoista ja pyritään tekemään hallinnoimisesta mahdollisimman helppoa, jotta aikaa jää muihin tehtäviin. Työ toteutettiin verkkosivupohjaisena käyttöliittymänä, joka hakee tiedot palvelimella olevasta tietokannasta. Tämä mahdollistaa reaaliaikaisen tiedon siirtymisen käyttäjille, koska järjestelmä hakee tietokannasta aina ajantasaisen tiedon.

1.4 Vaatimusmäärittely

Käytössä olleessa Excel-pohjaisessa vuorolistassa olleet ominaisuudet toimivat vaatimusmäärittelyn pohjana. Työn tarkoituksena oli luoda vastaavilla ominaisuuksilla varustettu järjestelmä, jossa mahdollistetaan tulevaisuudessa uusien ominaisuuksien helppo implementointi osaksi järjestelmää. Vaikka verkkosivut suunniteltiin käytettäväksi Firefox-selaimella, se ei estä sivuston käyttämistä muilla selaimilla. Muilla selaimilla sivuston toimintojen toimivuutta ei kuitenkaan voida taata.

Työstä tuli löytyä vuoropohjakierto, joka on muokattavissa ennalta määriteltyjen vuorojen mukaan. Vuoropohjasta pystytään siirtämään uusi viikko vuorolistaan ja vuorolistan vuoroja pystytään muokkaamaan puolen tunnin tarkkuudella. Työvuorolistan piti mahdollistaa kolmivuorotyön aiheuttamat vaihtelut vuoroissa. Jokaiselle päivälle lasketaan työvuorossa olevien henkilöiden lukumäärä ja verrataan sitä oletettuun vahvuuteen. Poikkeamat vahvuuksista tuotiin esiin korostamalla vuoron vahvuus väreillä.

Yksiköllä on käytössään työvuoroissa ennalta määriteltyjä rooleja. Työvuoroihin on pystyttävä merkitsemään tarvittaessa rooli, ja jokaiselta päivältä on tarkistettava, että jokaisessa päivän vuorossa on kyseistä roolia vähintään yksi (1) kappale. Työvuorot ovat normaalisti 7,5 tai 10 tuntia pitkiä. Työviikon pituus on 37,5 tuntia, ja vuoroja tehdään kolmen (3) viikon jaksoissa, eli jakson tuntimäärä on 112,5 tuntia. Kaikki poikkeamat viikoittaisissa ja jakson tuntimäärissä piti korostaa.

Yksikkö työskentelee vuoden jokaisena päivänä, ja pyhäpäivinä pidetään eri määrä työntekijöitä töissä. Tämän takia arkipyhät pitää pystyä merkitsemään ja laskemaan vahvuudet sen mukaan oikein. Työntekijöille voi tulla arkipyhänpoistoja, jotka pyritään tasaamaan vuoden aikajaksolla tasaisesti työntekijöiden kesken. Tästä syystä tilastoinnin mahdollistaminen oli erittäin olennainen osa työn tulosta. Vuorokierrossa on myös viikon kestävä varallaolojakso, joka tulee ottaa huomioon vuorokierrossa ja vuorolistan toiminnassa.

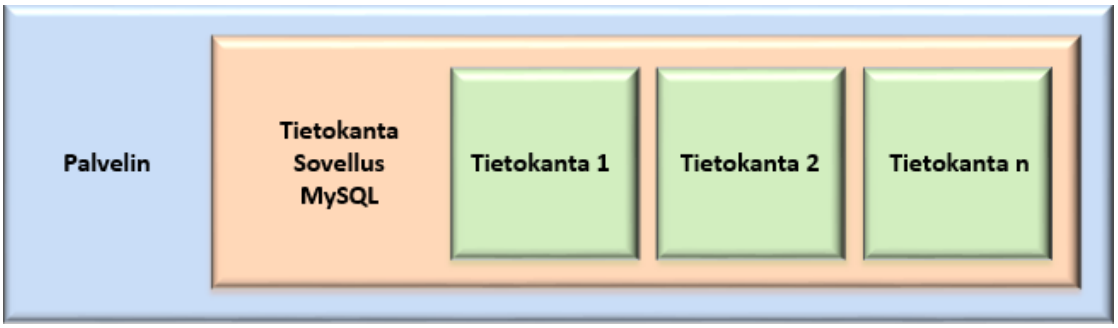
2 Tietokanta

2.1 Käyttötarkoitus

Tietokannan perimmäisenä tarkoituksen on säilyttää tietoa käyttäjälle. Käyttäjä pystyy hallinnoimaan tietokannan tietoja joko rajatusti tietystä paikasta tai vapaasti mistä tahansa. Tietokantaan pystytään tallentamaan suuria määriä tietoa ja hakemaan sieltä eri hakusanoilla aina haluttu tieto nopeasti. Tietokannassa säilytetään vain tieto, joka haetaan sieltä käyttäjän tarpeisiin ja tulostetaan käyttäjän haluamalla tavalla. (Häkkinen 2014a, 5.)

2.2 Rakenne

Tietokantasovellus sijaitsee palvelimella, ja palvelimella voi olla useita tietokantoja käytössä eri tarkoituksiin. Tietokantojen määrää palvelimella ei ole rajoitettu ja jokaiselle tietokannalle pystytään määrittämään oikeudet erikseen. Tietokanta rakentuu kuvion 1 mukaisesti.



Kuvio 1 Tietokannat palvelimella

2.3 Oikeudet

Oletuksena tietokantapalvelimelle pystyy kirjautumaan vain pääkäyttäjä paikallisesti. Tietokantaan pystytään luomaan erillisiä käyttäjätunnuksia, joilla on pääsy koko tietokantaan tai vain tiettyyn osaan sitä. Oikeuksia voidaan määrittää kolmella eri tasolla palvelimella. Eri tasot ovat tietokantapalvelimen hallinnolliset oikeudet, tietokantaoikeudet ja osittaiset oikeudet tietokantaan. Tietokantapalvelimen hallinnollisilla oikeuksilla pystytään hallinnoimaan koko palvelimen tietokannan toimintaa. Tieto-

kantaoikeuksilla hallinnoidaan palvelimella olevan tietyn tietokannan toimintaa ja oikeuksia. Osittaisilla oikeuksilla voidaan myöntää oikeus vain tiettyyn osaan tietokantaa, esimerkiksi tauluihin tai näkymiin. (Privileges provided by MySQL n.d.)

Käyttäjän autentikointi

MySQL-tietokantaa käyttävän täytyy tunnistautua yhteyden muodostamisen yhteydessä palvelimelle, jolloin palvelin tarkistaa, löytyivätkö käyttäjän syöttämät tiedot palvelimen käyttäjistä ja tunnistautuiko käyttäjä sallitulta yhteydeltä. Käyttäjäoikeuden luonnin yhteydessä pystytään määrittämään kohde, josta käyttäjä saa kirjautua järjestelmään. Käyttäjän kirjautuminen pystytään sallimaan mistä tahansa osoitteesta, rajoittaa tiettyyn osoitteeseen tai sallia vain paikallinen kirjautuminen. Mikäli käyttäjän tietoja ei löydy tai käyttäjä kirjautuu sellaisesta paikasta, joka ei ole sallittu, tietokannan hallintajärjestelmä evää käyttäjältä pääsyn tietokantaan. Käyttöoikeuksia määritettäessä on tärkeää rajata käyttäjän pääsy tarpeiden mukaan. (The MySQL Access Privilege System n.d.)

2.4 Taulut

Kenttä ja tietue

Tietokannan taulu on kaksiulotteinen kanta, jossa sarakkeet muodostavat taulun kentät ja rivit muodostavat tietueet. Kentät määrittävät tallennettavan tiedon tyypin ja kenttien riveille tallennettavat tiedot ovat tietueita. Varsinainen tieto, jota tietokannasta haetaan, ovat taulun tietueiden arvot. (Relaatiotietokantojen peruskäsitteet 2004.)

Rajoitteet

Kentälle voidaan määrittää ominaisuuksia, jotka tarkentavat kentän toiminteita ja sallittavia arvoja. Ominaisuuksia ovat, että tietue ei voi olla tyhjä, tietueella on oletusarvo, tietueen arvon täytyy olla yksilöllinen, sarake määräytyy perus- ja/tai viiteavaimeksi. (Relaatiotietokantojen peruskäsitteet 2004.)

Eheys

Toimivan tietokannan kuuluu olla eheä. Eheydelle on kolme vaatimusta, jotka ovat avaineheys, viite-eheys ja attribuuttieheys. Avaineheys saavutetaan, kun taulussa on pääavain ja jokainen pääavaimen arvo on yksilöllinen. Viite-eheydellä tarkoitetaan pääavaimen ja viiteavaimen välistä viittausta. Viiteavaimelle täytyy löytyä vastinpari toisen taulun pääavaimesta. Attribuuttieheys saavutetaan, kun tietueen arvot vastaavat kentän parametreja eli ovat sallittuja tietoja. (Häkkinen 2014c, 9.)

Taulujen välisiä liitoksia kuvaavia suhteita on kolmea erilaista tyyppiä. Ensimmäisenä on yksi-yhteen, joka tarkoittaa että taulun A tieto voi löytyä vain kerran taulun B tiedoista. Käytännön esimerkkinä tämä olisi esimerkiksi aviopuolisot, eli kummallakin voi olla vain yksi aviopuoliso. Seuraava ja yleisin on yksi-moneen-suhde, jossa yksi taulun A tieto voi ilmetä useassa taulun B tietueessa. Käytännön esimerkkinä toimii tilanne, jossa yhdellä henkilöllä voi olla useita puhelinnumeroita omistuksessa ja yhdellä puhelinnumerolla voi olla vain yksi omistaja. Viimeisimpänä on monesta-moneen-suhde, joita pyritään välttämään. Mikäli tilanne vaatii monesta-moneen-suhteen on suositeltavaa tehdä taulujen välille uusi taulu, johon tulee yksi-moneen- ja monta-yhteen-suhteet. Tällaisena tapauksena toimii verkkokauppa Internetissä, jossa useat käyttäjät voivat tilata useita ja samoja tuotteita. Tilanteeseen rakennetaan välitaulu, jossa pidetään tilauksen tietoja. Eli silloin suhde muuttuu yksi-moneen- ja monta-yhteen-suhteeksi. Sanallisesti selitettynä tilanne on niin, että yhdellä henkilöllä voi olla useita tilauksia ja yksi tuote voi kuulua useampaan tilaukseen. (Relaatiotietokannat luento 3 2011.)

Liitokset

Tietokanta mahdollistaa tietojen yhdistämisen useammasta eri tietokannan taulusta ja tulostaa ne liitoksen avulla yhdessä tulosjoukossa. Liitoksen muodostamisessa käytetään yleensä perus- ja viiteavaimia eli tietoa, joka löytyy molemmista tauluista. Liitosehto muodostetaan MySQL-lausekkeen WHERE-lauseessa, jossa määritetään kahden taulun sarakkeet vastaamaan toisiaan. (Häkkinen 2014b, 2-4.)

2.5 Tietueiden käsittely

MySQL tietokantaa käsitellään komentojen avulla. Komennoissa määritetään tehtävä toimenpide, viitattava taulu, mitä kenttiä asia koskee ja arvojoukkoa määrittäviä parametreja, joilla voidaan rajata arvojoukkoa. Muutoksessa täytyy aina olla taulun pääavain mukana, muutoin tietokannan oma suojaus estää arvojen muuttamisen. MySQL-lausekkeet päättyvät aina puolipisteeseen.

Lisäys

Tietojen lisääminen tietokantaan tapahtuu INSERT INTO -lausekkeella. Lausekkeen rakenne on seuraava:

```
insert into EMPL (tunnus, etunimi, sukunimi) values ('f1217', 'Joni', 'Launonen');
```

Lausekkeessa arvo *EMPL* viittaa tauluun, johon lisäys tehdään. Ensimmäisissä suluissa määritellään kentät, joihin arvoja tullaan lisäämään. Tässä tapauksessa kentät ovat *tunnus*, *etunimi* ja *sukunimi*. Sen jälkeen tulee *values* eli arvot, jotka syötetään samassa järjestyksessä, kuin kentät on määritelty aiemmissa suluissa. Tässä tapauksessa arvot ovat *f1217*, *Joni* ja *Launonen*. EMPL-taulussa tunnusta suositellaan käytettävän pääavaimena, koska tunnus on kaikille työntekijöille yksilöllinen arvo.

Valinta

Tietojen valinta tietokannan tauluista tapahtuu SELECT-lausekkeella. Lausekkeessa määritetään, mitkä kentät halutaan tauluista tulostaa ja tämän jälkeen joukkoa voidaan rajoittaa tarkentavilla parametreilla. Yksinkertainen lauseke on seuraavanlainen:

```
select * from EMPL where tunnus = 'f1217';
```

Lausekkeessa valitaan tähtimerkin vuoksi kaikki kentät taulusta EMPL ja tulostetaan arvot, joissa täyttyy määre, että tunnus on *f1217*. Eli tässä tapauksessa juuri lisätty kenttä EMPL-tauluun. Tietoja pystytään hakemaan useasta eri taulusta, ja haettujen kenttien määrää voidaan rajoittaa korvaamalla tähtimerkki haluttujen kenttien nimillä. Seuraavassa esimerkissä haetaan kahdesta taulusta tietoja, ja mikäli tauluissa on samannimisiä kenttiä, tulee ne tarkentaa taulun nimellä.

```
select etunimi, sukunimi, joukkue from EMPL, TEAM where EMPL.tunnus =
TEAM.pelaajatunnus;
```

Esimerkissä haetaan etunimi ja sukunimi EMPL-taulusta ja joukkue TEAM-taulusta. Lopullisessa tulosteessa näkyvät etunimi, sukunimi sekä joukkue, jossa henkilö pelaa. Henkilön tunnus löytyy TEAM-taulun pelaajatunnuskentästä, ja lausekkeessa määritellään, että pelaajatunnuksen ja henkilön tunnuksen tulee olla samat. Kun kahdessa eri taulussa olevia tietoja halutaan yhdistää, tulee tiedon löytyä molemmista tauluista. EMPL-taulun tunnus toimii taulun pääavaimena ja TEAM-taulun pelaajatunnus on viiteavaimena.

Muutos

Tietojen muuttaminen tietokannan taulussa tapahtuu UPDATE-lausekkeella. Sen rakenteesta löytyy jo aiemmin tutut komponentit eli taulu, kentät, uusi arvo kentälle ja tarkentava määre muutosta koskevista tietueista. Lauseke on seuraavanlainen:

```
update TEAM set joukkue='Palloveikot' where pelaajatunnus = 'f1217';
```

Lausekkeessa päivitetään TEAM-taulun joukkuekenttään uusi arvo Palloveikot. Muutos tehdään sellaisille tietueriveille, joissa pelaajatunnus on *f1217*.

Poisto

Tietojen poistaminen taulusta tapahtuu DELETE-lausekkeella. Lauseke on yksinkertainen ja sisältää tiedon, mistä taulusta poistetaan, ja tarkentavien parametrien avulla määritetään, mitkä rivit kuuluvat poistettavien listalle.

```
delete from EMPL where tunnus = 'f1217';
```

Lauseke poistaa EMPL-taulusta aiemmin luodun henkilön tiedot. Tietueita poistettaessa on aina tärkeää lisätä pääavain tarkentaviin parametreihin, koska tietokannan suojaus vaatii avaimen lausekkeeseen. Mikäli poistettavien tietueiden pääavaimia ei voida yksilöidä, mutta poistettavat tietueet määritetään muilla parametreilla, voidaan käyttää seuraavanlaista komentoa:

```
delete from VUORO where viikko <= '12' and indeksi <> '';
```

Lauseke poistaa VUORO-taulusta kaikki tietueet, joissa viikon arvo on 12 tai alle ja indeksi on jotain muuta kuin tyhjä. Vuorotaulussa käytetään juoksevaa indeksinumerointia pääavaimena, eikä sitä voida helposti määrittää poistolauseessa määrittäväksi parametriksi.

2.6 Näkymät

Näkymien tarkoituksena on helpottaa toistuvien hakujen tekemistä tietokantaan samoilla tiedoilla. Näkymä toimii viittauksena siihen tallennettuun lausekkeeseen, ja näkymiä pystytään luomaan useita samanaikaisesti. Tämä toteutetaan kuitenkin niin, että jokaisella näkymällä on yksilöllinen nimi. Näkymä tallennetaan CREATE VIEW – lausekkeella, ja sen osana on normaali MySQL-valintalauseke.

```
create view suomalaiset as select * from EMPL where maa = 'Suomi';
```

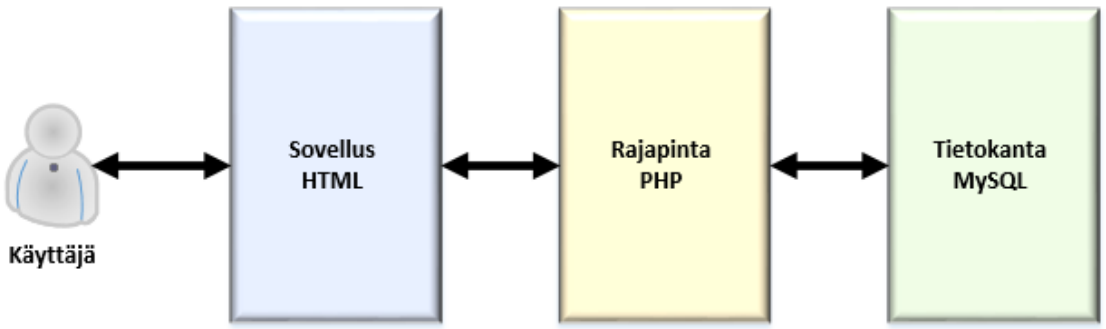
Yllä olevalla lausekkeella luotiin näkymä nimeltään suomalaiset. Näkymällä haetaan EMPL-taulusta kaikki ne tietueet, joissa maa-kentän arvo on *Suomi*. Näkymää pystytään jatkossa kutsumaan yksinkertaisemmin seuraavalla komennolla:

```
select * from suomalaiset;
```

Näkymän tulostejoukkoa pystytään rajoittamaan normaalin lausekkeen tavoin tarkentavilla parametreilla, joilla saadaan halutut tietueet näkyviin.

2.7 Ohjelmointirajapinta

Rajapinta toimii tulkkina tietokannan ja käytettävän sovelluksen välillä. Tässä opinnäytetyössä sovelluksena toimi verkkopohjainen HTML-sivusto, johon tietoja haettiin tietokannasta PHP-kielen avulla, joka puolestaan toimii ohjelmointirajapintana tietokantaan. Käyttäjä käyttää sovellusta eikä ole tietoinen taustalla tapahtuvista muista prosesseista. Kuviossa 2 on esiteltynä prosessien eri osapuolet ja nuolilla on kuvattu keskustelusuhteet eri osapuolien välillä.



Kuvio 2 Rajapinnat ja niiden väliset keskustelusuhteet

3 Työn suunnittelu

3.1 Palvelin

Kaikki työssä tarvittavat palvelut voidaan toteuttaa käyttämällä yhtä palvelinta. Opinnäytetyötä tehtäessä ei nähty tarpeelliseksi eikä järkeväksi erottaa MySQL-tietokantaa ja verkkosivuja erillisille palvelimille. Verkkopohjainen vuorolista ei vaadi jatkuvia resursseja palvelimelta, vaan resursseja vaaditaan ainoastaan käyttäjien tehdessä hakuja tietokantaan. Tästä syystä virtuaalinen palvelin on kaikkein paras vaihtoehto, koska palvelin pystyy varaamaan tarvittavat resurssit silloin, kun on tarve, ja luovuttamaan niitä muiden käyttöön silloin, kun resursseille ei ole tarvetta.

3.2 Tietokanta

Tietokantaan luodaan jokaiselle kolmesta ryhmästä oma taulu työvuorotietojen säilyttämiselle. Hakujen optimoimiseksi jokaiselle ryhmälle luodaan oma taulu. Jokaista työntekijää kohden vuodessa tulee 365 riviä tietoja, ja työntekijöiden määrän kasvaessa puhutaan useista tuhansista riveistä tietoja ryhmää kohden vuodessa. Myös ryhmien jako omiin listoihin selkeyttää hallittavuutta ja vuorojen osoittamista tietyn ryhmän vuoroiksi. Jokaiselle ryhmälle luodaan myös oma taulu vuoropohjakiertoa varten, koska vuorokierrot ja halutut roolit eroavat toisistaan. Näin ollen ryhmät pystyvät toimimaan eriytettynä muista ryhmistä. Pääavaimena näissä tauluissa toimii juokseva indeksinumero.

Kaikki työntekijät pidetään yhdessä tietokannan tauluista, jossa säilytetään tarpeellinen tieto työntekijästä. Työntekijöillä on käytössään käyttäjätunnus, joka on uniikki, ja siksi sitä käytetään pääavaimena taulussa. Työntekijän tiedoissa säilytetään järjestelmän toiminnan kannalta oleellisia tietoja, kuten työsuhteen muoto, henkilön vuorolistan määrittävä ryhmä sekä onko käyttäjällä oikeus hallinnoida vuoroja. Työntekijätaulu on kytköksissä vuorolistatauluun yksi-monessa-suhteella.

Tämän lisäksi työ tulee vaatimaan hallinnollisen datan säilyttämistä tietokannassa eri tauluissa. Hallinnollinen data ei ole yhteydessä muihin tietokannan tauluihin, vaan

sisältää työtä määrittäviä parametreja, kuten päiväkohtaiset työntekijöiden vahvuudet, pyhäpäivien tiedot ja jaksotyön jakson alkamisen vuoden alussa.

3.3 Verkkosivut

Verkkosivut toteutettiin perinteisellä HTML-kielellä, jossa käytettiin PHP:ta rajapintana tietokannan ja verkkosivujen välillä. Sivustosta oli tarkoitus luoda mahdollisimman yksinkertainen ja peruselementeillä rakennettu kevyt versio. Sivusto rakennettiin käyttämällä itse luotua koodia ja vain tarvittavissa tilanteissa turvauduttiin käyttämään valmiiksi luotuja toteutuksia ja koodinpätkiä kuten skriptejä.

Sivustolle luotiin kirjautumispalvelut, josta päädytään vuorolistaan oman ryhmän kohdalle. Sivuston optimoimiseksi näytetään vain muutama viikko kerrallaan vuorolistaa, koska silloin saadaan käyttäjälle näkymään kaikkein oleellisin tieto ja tarvittaessa työviikkoja pystytään selaamaan eteenpäin ja taaksepäin. Vuorojen hallinnointia varten luotiin erillinen tila, jonka pystyvät aktivoimaan vain siihen oikeutetut henkilöt.

Vuorotietojen muokkaamista varten luotiin erillinen sivu, jonka kautta toteutetaan kaikki muutokset vuoroihin. Kaikki mahdollinen on toteutettu ennalta määriteltyjen arvojen avulla käyttäen apuna alasvetovalikoita, radiopainikkeita ja valintaruutuja. Näin pyritään minimoimaan käyttäjän tekemiä virheitä tietoja syötettäessä. Myös MySQL-injektion toteuttaminen on pyritty estämään mahdollisimman tehokkaasti. On selvää, että kaikkia kohtia ei voida ennalta määrittää käyttäjän tarpeiden mukaan, kuten työvuoroa koskevaa kommenttia. Tähän toteutettiin suojaus MySQL-injektion varalta.

4 Työympäristön kuvaus

4.1 Palvelut

Työn toteutus vaati verkkosivuinfrastruktuurin luomisen, tietokannan, jossa tietoja säilytetään, sekä rajapinnan verkkosivujen ja tietokannan väliin. Työ toteutettiin avoimen lähdekoodin sovelluksilla, ja siitä syystä palvelimen käyttöjärjestelmäksi valikoitui Linux. Sekä kehitysalustalle että tuotantoverkkoon asennettiin LAMP-ympäristö eli Linux-palvelin, jossa pyörivät Apache-, MySQL- ja PHP-palvelut.

Apache

Apache on avoimen lähdekoodin projekti, jonka tarkoituksena on mahdollistaa HTTP-palvelimen ylläpito paikallisessa verkossa tai julkisessa Internetissä. Julkisessa Internetissä näkyminen ei vaadi Apachelta mitään, vaan verkkolaitteiden pitää osata ohjata pyynnöt palvelimelle, jossa Apache pyörii. Tässä työssä palvelin jätettiin käytettäväksi pelkästään yrityksen sisäverkosta.

MySQL

MySQL on avoimeen lähdekoodiin perustuva tietokantasovellus. Se mahdollistaa tietokannan hallinnoimisen suoraan palvelimelta tai toiselta koneelta. Hallinnointi tapahtuu komentorivin (CLI) kautta tai erillisellä sovelluksella graafisen käyttöliittymän avulla.

PHP

PHP on kehitetty toimimaan ohjelmointirajapintana verkkosivujen ja tietokannan välillä. PHP on palvelinpään ohjelmointikieli, ja tämän vuoksi koodi suoritetaan palvelimen laskentateholla ja saatu tulos toimitetaan ainoastaan käyttäjälle. Näin ollen käyttäjällä ei ole tietoa siitä, mitä PHP suorittaa taustalla ja miten, vaan hän näkee ainoastaan tuloksen.

4.2 Kehitysympäristö

Opinnäytetyötä varten pystytettiin erillinen ympäristö, jossa työ suunniteltiin ja kehitettiin. Ympäristö pidettiin hyvin yksinkertaisena, ja siihen kuuluivat ainoastaan palvelin, työasema ja välissä oleva kytkinlaite.

Palvelin

Palvelimena toimi Raspberry Pi model B+, joka oli varustettu Class 10-nopeuden muistikortilla. Palvelimelle asennettiin Raspbian käyttöjärjestelmä, joka pohjautuu Debian Linuxin käyttöjärjestelmäversioon Wheezy, ja se on tehty yhteensopivaksi Raspberry Pi -alustalle. Palvelimelle asennettiin Apache2, MySQL 5.5 ja PHP 5.4 versiot.

Työasema

Työasemaa käytettiin myös todentamaan verkkosivujen toimintaa, ja se vastasi tuotantoverkossa olevia työasemia. Tämän lisäksi työasemalla suunniteltiin ja ylläpidettiin palvelimella olevaa tietokantaa MySQL Workbench -ohjelmalla.

4.3 Tuotantoverkko

Tuotantoverkkoon opinnäytetyötä varten luotiin erillinen virtuaalipalvelin, joka pyörii VMWare-ohjelmiston päällä. Virtuaalipalvelimelle asennettiin Red Hat Enterprise Linux 6, johon asennettiin vastaavat palvelut, joita kehitysympäristössäkin käytettiin. Kehitysympäristöstä poiketen Red Hat Linux ei pohjaudu Debian-käyttöjärjestelmään, ja palveluiden välillä ilmenikin versioeroja. Tuotantoon saatiin muutoin asennettua vastaavat versiot, mutta PHP:sta asennettiin uudempi 5.6-versio.

Tuotantoverkon palvelin on sijoitettu yrityksen sisäverkkoon siten, että sinne ei ole pääsyä ulkoverkosta ilman VPN-yhteyttä. Tämä estää ulkopuoliset palvelunestohyökkäykset ja tietomurrot. Mutta vaikka sivut ovat yrityksen sisäverkossa, on tietoturvaan kiinnitetty huomiota.

5 Ympäristön käyttöönotto

5.1 Laitteiston asennus

Kehitysympäristö

Raspbian-käyttöjärjestelmä asennettiin muistikortille Raspberry Pi -sivuston ohjeiden mukaisesti. Opinnäytetyön teossa käytettiin versiota, joka on julkaistu 9.9.2014. Ensimmäisen käynnistyskerran aikana käyttäjälle näytetään konfigurointivalikko, jossa määritetään tarkemmat asetukset käyttöjärjestelmälle. Mikäli tuo valikko pitää myöhemmin saada auki, niin sen saa kirjoittamalla komennon

```
sudo raspi-config
```

Konfigurointivalikosta tarkistetaan ensimmäisenä, että koko levytila on käytössä muistikortilta. Tämä tapahtuu valikon 1 kautta, josta määritetään koko muistikortti järjestelmän käyttöön.

Seuraavaksi tarkistetaan järjestelmän aika, paikka ja näppäimistökieli. Näppäimistökieli suositellaan vaihdettavaksi ennen salasanan määrittämistä, koska oletuksena näppäimistökieli on englanti, jossa erikoismerkit ovat eri näppäimien takana kuin suomalaisessa näppäimistökielessä. Asetukset löytyvät valikon 4 alta.

Oletuksena järjestelmän käyttäjätunnuksena on *pi* ja salasana *raspberry*. On suositeltavaa vaihtaa tämä salasana, koska muutoin kuka tahansa pääsee kirjautumaan palvelimelle eikä se ole tietoturvallista. Salasana vaihdetaan valikon kohdasta 2.

Lopuksi määritellään vielä kohdan 8 lisävalinnat alta järjestelmän nimi, muistinkäyttö ja SSH-asetukset. Järjestelmän nimellä tarkoitetaan nimeä, jolla laite näkyy verkossa muille koneille. Myös komentorivillä näkyy laitteen nimi käyttäjän perässä. Muistinkäytössä määritellään laitteen oman muistin ja GPU-muistin välinen suhde. Palvelinkäytössä ei tarvita graafista käyttöliittymää, joten GPU-arvo asetetaan pienimmäksi mahdolliseksi arvoksi, jolloin muille toiminnoille jää enemmän muistia käytettäväksi. Mikäli järjestelmään halutaan muodostaa etäyhteys SSH-protokollan avulla, täytyy lisäasetuksista aktivoida SSH-optio. Mikäli SSH aktivoidaan, on erittäin tärkeää, että

oletuskäyttäjä ja salasana muutetaan järjestelmälle, koska järjestelmän turvallisuus on muutoin vaarannettu.

Nyt kaikki tarvittavat asetukset on määritelty järjestelmän perustoimintojen osalta. Konfigurointivalikko voidaan sulkea etusivun *Finish*-painikkeen avulla.

Tuotantoverkko

Tuotantoverkon palvelin tuli esiasennettuna opinnäytetyötä varten. Järjestelmään ei ollut asennettu mitään ylimääräisiä sovelluksia tai palveluita. Järjestelmään ei ole aktivoitu graafista käyttöliittymää ja palvelinta käytetään SSH-yhteyden yli komentoriviltä. Tuotantoverkon palvelimella järjestelmänä on Red Hat 6.6 Santiago.

5.2 Palveluiden asennus

Palvelimelle tuli asentaa LAMP-ympäristö. Kehitysympäristö on tehty Debian-pohjaisessa käyttöjärjestelmässä, joten pakettihallintaohjelmana toimii APT. Tuotantoverkon palvelin on Red Hat, joka käyttää YUM-pakettienhallintaa.

Kehitysympäristö

Ennen uusien pakettien asentamista on tärkeää päivittää pakettitiedot järjestelmässä. Käyttöjärjestelmän mukana tulevat pakettitiedot voivat olla vanhentuneita ja ajantasaisimmat versiot saadaan asennettua vasta tietojen päivityksen jälkeen. Päivitys tapahtuu seuraavalla komennolla

```
sudo apt-get update
```

Järjestelmän mukana asennetut paketit kannattaa päivittää uusimpiin paketteihin ennen järjestelmän lopullista käyttöönottoa. Näin järjestelmä saadaan luotua uusimpien versioiden päälle. Uusimmat paketit saa asennettua komennolla

```
sudo apt-get upgrade
```

Komento listaa paketit, joista on saatavilla uudempi versio. Ja vastaamalla *Y*-järjestelmän esittämään kysymykseen saadaan paketit ladattua Internetistä, ja ohjelma asentaa tarvittavat paketit automaattisesti.

Varsinainen LAMP-ympäristö asennetaan tämän jälkeen komennolla

```
sudo apt-get install apache2 mysql-client mysql-server php5
```

Komennolla asennetaan uusimmat saatavilla olevat paketit Apachesta, MySQL:stä ja PHP:sta. MySQL:n asennuksen yhteydessä täytyy vastata ohjelman asetuksia määritteleviin kysymyksiin. Kysymyksissä määritetään pääkäyttäjän salasana sekä muita asetuksia, jotka näkyvät alla.

```
Set root password? [Y/n] y
Remove anonymous users? [Y/n] y
Disallow root login remotely? [Y/n] y
Reload privilege tables now? [Y/n] y
```

Nyt palvelimelle on asennettu tarvittavat palvelut, mutta PHP pitää vielä konfiguroida käyttämään oikeaa merkistöä. PHP:n hallintatiedosto avataan käyttäjän haluamalla tekstieditorilla.

```
sudo nano /etc/php5/apache2/php.ini
```

Tiedostosta käydään pakottamassa PHP:n merkistö UTF-8:aan. Tarvittava tieto löytyy riviltä 683 ja määrityksen tulee olla seuraava:

```
default_charset = "UTF-8"
```

Tämän jälkeen muutokset tallennetaan ja ne astuvat voimaan palveluiden uudelleenkäynnistämisen yhteydessä. Palveluiden uudelleenkäynnistys onnistuu komennolla

```
sudo service apache2 restart
```

Nyt ympäristön palvelut on asennettu ja esikonfiguraatiot tehty. Apachen ja PHP:n toiminta palvelimella voidaan todentaa tekemällä Apachen oletushakemistoon PHP-tiedosto, jossa kutsutaan PHP:n info-funktiota. Tiedoston voi luoda haluamallaan tekstieditorilla, esimerkissä käytämme Nanoa.

```
sudo nano /var/www/info.php
```

Tiedostoon tallennetaan seuraavat rivit:

```
<?php
phpinfo();
?>
```

Tiedosto tallennetaan painamalla *CTRL + O* ja vastataan kysymykseen painamalla *Y*. Tämän jälkeen ohjelmasta voidaan poistua painamalla *CTRL + X*. Tämän jälkeen Internet-selaimella yhdistetään palvelimella sijaitsevaan tiedostoon, esimerkiksi http://192.168.1.2/info.php. Mikäli sivusto latautuu, on Apache ja PHP asennettu

palvelimelle onnistuneesti. Käyttäjälle auennut sivu sisältää PHP:n version ja PHP:ta määrittävät parametrit sekä niiden arvot.

Tuotantoverkko

Tuotantoverkon palvelimelle asennetaan vastaavat palvelut, kuin kehitysympäristössä käytettiin. Alustojen käyttöjärjestelmät poikkeavat toisistaan, joten versioiden välisiä eroja ilmenee palvelinten välillä. Apache asennetaan komennolla

```
sudo yum install httpd httpd-devel
```

Red Hat -käyttöjärjestelmän 6.6-versio tarjoaa oletuksena asennettavaksi PHP:n 5.3-version ja MySQL:n 5.1-version. Työtä varten tarvitaan vähintään MySQL-versio 5.5, jotta skandinaaviset merkit toimivat oikein tietokannassa. Myös PHP:n versio päivitetään, jotta voidaan kirjautumisvaiheessa käyttää edistyneempää salasanan kryptaamista.

Jakelun mukana tulevia paketteja uudempien versioiden käyttö vaatii, että järjestelmälle näytetään, mistä paketit löytyvät. Tämä onnistuu määrittämällä uusi repositorio järjestelmään. Repositorio sisältää tiedon paketeista ja niiden sijainnista Internetissä. Uusi repositorio asennetaan seuraavasti:

```
sudo rpm -Uvh https://mirror.webtatic.com/yum/el6/latest.rpm
```

Kun tiedoston haku Internetistä onnistuu ja paketti saadaan asennettua osaksi järjestelmää, ovat paketin mukana tulleet tiedot pakettienhallinnan käytettävissä. Tämän jälkeen voidaan asentaa MySQL:n uudempi versio uudesta lähteestä, joka onnistuu seuraavalla komennolla

```
sudo yum install mysql55w mysql55w-server
```

Asennus vaiheessa käydään läpi vastaavat kysymykset, joita kehitysympäristössä tuli vastaan. Mikäli palvelu ei suoraan käynnisty asennuksen jälkeen, voidaan se pakottaa käyntiin suorittamalla komento

```
sudo service mysqld start
```

Tämän jälkeen asennetaan vielä PHP ja siihen MySQLi-lisäosa, joka mahdollistaa tietokantaan yhdistämisen PHP:n avulla. Uusi PHP-versio löytyy samasta repositoriosta, jonka aiemmin sisällytimme järjestelmään, joten voimme asentaa PHP:n komennolla

```
sudo yum install php56w php56w-opcache php56w-mysql
```

Komento asentaa samalla muut tarvittavat lisäosat. Kuten kehitysympäristössä, myös tuotantoverkossa on tarve käydä muokkaamassa PHP:n asetuksista merkistöksi UTF-8. Tarvittava kohta löytyy riviltä 683 tiedostosta

```
sudo nano /etc/php.ini
```

Tämän jälkeen palvelu täytyy taas käynnistää uudestaan, jotta tehdyt muutokset tulevat voimaan. Palvelun saa käynnistettyä uudestaan komennolla:

```
sudo service httpd restart
```

5.3 Tietokannan luonti

Esivalmistelut

Tietokanta luotiin kehitysympäristössä MySQL Workbench-ohjelmalla. Valmiista tietokannasta luotiin lopuksi varmuuskopio, joka vietiin tuotantoverkon MySQL-tietokantaan. Tuotantoverkossa ei ole tehty tietokantaan rakenteellisia muutoksia.

Oletuksena tietokantaan pääsee kirjautumaan vain paikallisesti pääkäyttäjänä. Tämän vuoksi ensimmäinen kirjautuminen MySQL-tietokantaan tehdään palvelimelta paikallisesti ja käydään luomassa uusi käyttäjä, joka saa kirjautua tietokantaan etänä. Tätä käyttäjää käytetään MySQL Workbench -ohjelmassa tietokannan luontiin ja hallinnointiin. Tietokantaan päästään kirjautumaan palvelimelta komennolla

```
mysql -u root -p
```

Tämän jälkeen järjestelmä kysyy pääkäyttäjän salasanan, minkä jälkeen käyttäjä on kirjautuneena MySQL-sovellukseen pääkäyttäjänä. Kirjautumisen onnistumisen tietää siitä, että komentorivillä näkyy *mysql>*. Tässä tilassa kirjoitetaan kaikki tietokantaa koskevat komennot.

Tietokannan luominen

Kun ollaan kirjautuneena MySQL-tietokantaan, voidaan luoda uusi tietokanta, jota opinnäytetyössä käytetään. Uusi tietokanta luodaan ja otetaan käyttöön komennoilla

```
CREATE DATABASE tietokanta
USE tietokanta
```

Käyttäjän luominen

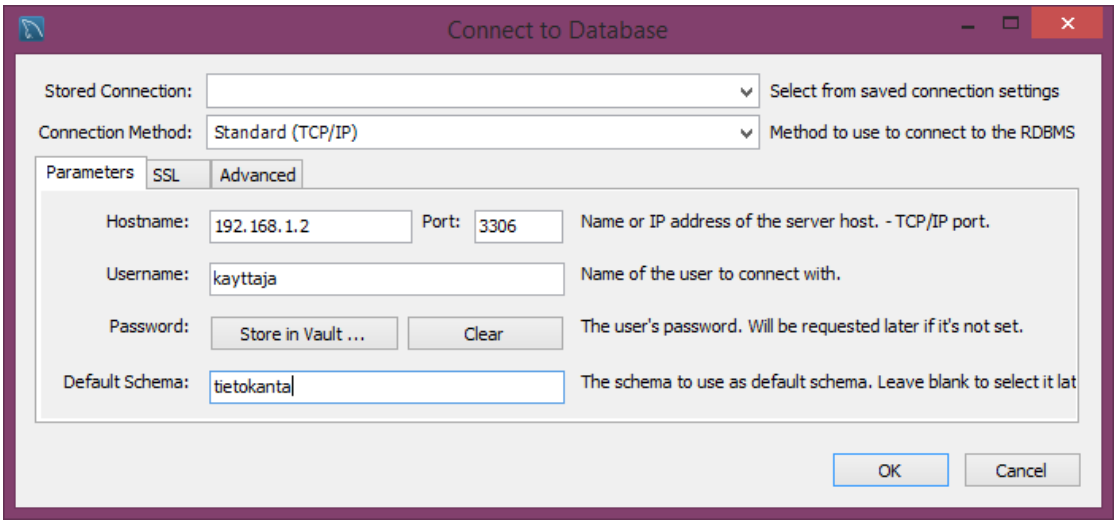
Edellä kuvattujen toimintojen jälkeen luodaan uusi käyttäjä ja määritetään mihin tietokantaan käyttäjällä on oikeudet. Oikeuksien määrittämisessä voidaan käyttää tarkkaa määrettä kirjoittamalla se niin kuin se on tai käyttää jokerimerkkiä, joka on *-merkki.

```
CREATE USER 'kayttaja'@'localhost' IDENTIFIED BY 'salasana';
GRANT ALL PRIVILEGES ON tietokanta.* TO 'kayttaja'@'localhost';
FLUSH PRIVILEGES;
```

On huomioitavaa, että yllä olevassa esimerkissä annettiin oikeus käyttäjälle kirjautua paikallisesti palvelimelta. Tämä on riittävä määritys tuotantoverkossa, kun PHP ja MySQL sijaitsevat samalla palvelimella. Mutta kehitysympäristössä tietokannan hallinnointi suoritetaan erilliseltä tietokoneelta, joten sille on myös luotava oikeudet. Oikeudet ovat muuten vastaavat, mutta *localhost*-arvo muutetaan vastaamaan yhdistettävän koneen IP-osoitetta tai korvataan *-merkillä, jolloin käyttäjä voi kirjautua mistä osoitteesta tahansa tietokantaan.

MySQL Workbench-ohjelman yhdistäminen tietokantaan

Sovelluksen ylävalikosta valitaan *Database* ja *Connect to Database*. Aukeava ikkuna näyttää kuvion 3 mukaiselta ja siihen täytetään tietokantaa koskevat tiedot. Esimerkissä tietokanta sijaitsee palvelimella, jonka osoite on 192.168.1.2. Käyttäjä ja tietokanta ovat aiemmassa kohdassa luodut tiedot.

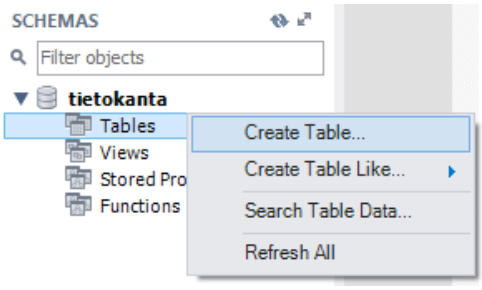


Kuvio 3 Asetukset tietokantaan yhdistämistä varten

Painamalla *OK* ohjelma yrittää yhdistää tietokantaan ja kysyy käyttäjän salasanaa. Syötetään käyttäjälle määritetty salasana kenttään, jonka jälkeen yhteys muodostetaan kantaan.

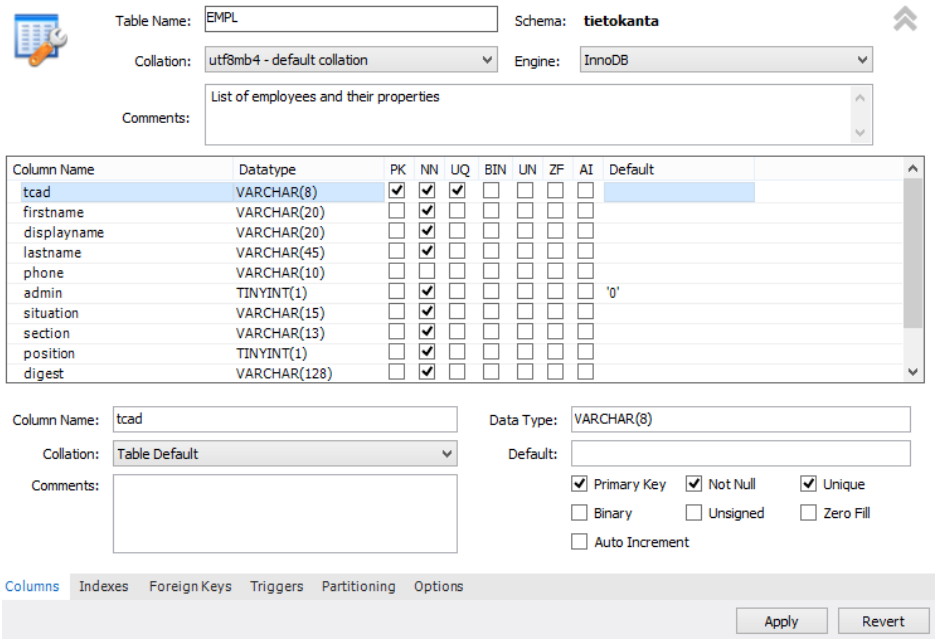
Taulun luominen tietokantaan

Taulu voidaan luoda ohjelmalla graafisesti valitsemalla ohjelman vasemmasta laidasta kuvion 4 mukainen kohta. Valikko saadaan auki painamalla hiiren oikeata painiketta kohdan *Tables* päällä.



Kuvio 4 Aloitetaan uuden taulun luominen tietokantaan

Ohjelman keskiruutuun avautuu välilehti, jossa tietokannan taulu voidaan luoda. Täytetään taulun nimi ja halutut kentät tietokantaan ominaisuuksillaan kuvion 5 mukaisesti. Lopuksi painetaan *Apply*, mikä muuttaa kenttiin syötetyt arvot MySQL-komennoksi, joka ajetaan tietokantaan. Komento näkyy avautuvassa ikkunassa, ja se pitää vielä hyväksyä ennen varsinaista ajoa.



Kuvio 5 Uuden taulun luominen graafisesti

Vastaavasti tietokannan voi luoda itse komentomuodossa ohjelman *Query*-välilehdellä, joka aukesi, kun tietokantaan yhdistettiin. Siihen kirjoitetaan MySQL-komento siinä muodossa, että se voidaan ajaa tietokantaan. Aiemmin kuviossa 5 graafisesti luotu EMPL taulu näyttää ajettavana MySQL-komentona seuraavalta:

```
CREATE TABLE `EMPL` (
  `tcad` varchar(8) NOT NULL,
  `firstname` varchar(20) NOT NULL,
  `displayname` varchar(20) NOT NULL,
  `lastname` varchar(45) NOT NULL,
  `phone` varchar(10) DEFAULT NULL,
  `admin` tinyint(1) NOT NULL DEFAULT '0',
  `situation` varchar(15) NOT NULL,
  `section` varchar(13) NOT NULL,
  `position` tinyint(1) NOT NULL,
  `digest` varchar(128) NOT NULL,
  PRIMARY KEY (`tcad`),
  UNIQUE KEY `tcad_UNIQUE` (`tcad`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='List of employees and
their properties';
```

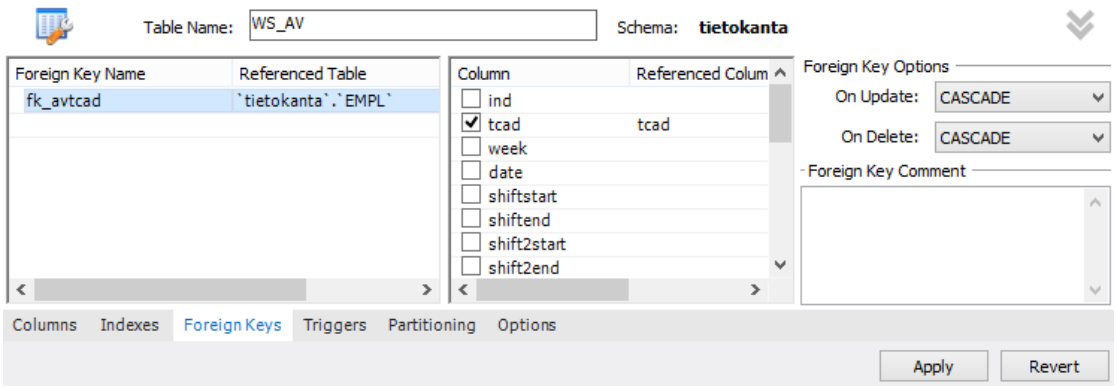
Komento suoritetaan valitsemalla yläpalkista *Query* ja sen alta *Execute current statement*, tai saman voi tehdä myös näppäinyhdistelmällä *CTRL + Enter*. Mikäli tietokannan hallinta tehdään puhtaasti palvelimen komentoriviltä suoraan tietokantaan, niin komentomuodossa kirjoitettu koodi pystytään suorittamaan siellä.

On huomioitavaa, että merkistön arvon tulee olla utf8mb4, joka mahdollistaa å-, ä- ja ö-merkkien käyttämisen tietueiden arvoissa. Muut työssä käytetyt taulut ja niiden kentät sekä arvot löytyvät liitteestä 1 ja kaaviokuva koko tietokannasta liitteestä 2.

Viiteavaimen luominen taulujen välille

Taulujen liitosehdot määritetään taulun luonnin yhteydessä *Foreign Keys*-välilehdeltä. Mikäli liitosehto tehdään taulun luonnin jälkeen, valitaan taulu hiiren oikealla painikkeella ja tämän jälkeen valitaan valikosta *Alter table*. Liitosehdolla tulee olla yksilöllinen ja kuvaava nimi. Viiteavain liitetään aina viitattavan taulun pääavainta vasten. Kuviossa 6 on määritelty WS_AV-taulun liitosehto aiemmin luodun EMPL-taulun kanssa, joka määritetään kohdassa *Referenced Table*. Muokattavan taulun kentät näkyvät *Column*-sarakkeessa, josta valitaan mikä taulun kentistä halutaan liittää osaksi toisen taulun tietoja. Tässä tapauksessa valitaan arvo *tcad*, joka on liitetty osaksi EMPL-taulun *tcad*-kenttää. Liitettävien taulujen kenttien arvot pitää vastata toisiaan, jotta liitosehto toimii. *Foreign Key Options* -kohdassa voidaan määrittää tie-

tojen päivittämisen ja poistamisen ehdot. Vaihtoehtoina ovat muutoksen estäminen, päivittäminen, puuttuva arvo (null) tai no action -valinta, jolla ei tehdä mitään muutoksia.



Kuvio 6 Viiteavaimen määrittäminen taulujen välille

Työssä käytetyt viiteavaimet on kuvattuna liitteessä 2, jossa näkyy tietokannan taulut kaaviokuvana. Viiteavaimet voidaan myös määrittää MySQL-komentojen avulla ilman graafista käyttöliittymää seuraavasti:

```
ALTER TABLE `tietokanta`.`WS_AV`
ADD CONSTRAINT `fk_avtcad`
  FOREIGN KEY (`tcad`)
  REFERENCES `tietokanta`.`EMPL` (`tcad`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;
```

5.4 Graafinen käyttöliittymä

HTML

Sivuston pohjalla toimii HTML-koodilla toteutettu kehysrakenne, jonka päälle sisältöä tuotetaan tietokannasta PHP:n avulla. Sivustolta selkeimpänä HTML-koodilla toteutettu elementti on vuorolistan taulukkonäkymä, johon on tuotu työntekijöiden työvuorot. Taulukoita on käytetty myös muualla työssä apuna rakenteen säilyttämiseksi. Tästä hyvänä esimerkkinä on kuvio liitteessä 3, jossa yläpuolella on työssä oleva näkymä ja alapuolella havainnollistava kuva, jossa on tuotu näkyviin taulukon pohjarakenne.

Työssä liikutaan paljon sivustolta toiselle, ja siirryttäessä täytyy saada siirrettyä tietoa. Sitä siirretään sekä osoiterivin että lomakkeiden avulla. Lomakkeet on luotu HTML-koodia hyödyntäen ja niihin on haettu tietokannasta pohjatiedot, joita käyttäjä

voi muokata. Näistä helpoin dokumentoida ja lähes kokonaan HTML-koodilla toteutettu on kirjautumissivu. Kirjautumissivun koodi on nähtävillä liitteestä 4.

CSS

Sivuston yleisilmettä määrittävät ominaisuudet on kirjattu palvelimella sijaitsevaan CSS-tiedostoon, johon jokaisen sivun alussa viitataan. CSS-tiedoston tarkoituksena on yhtenäistää ja helpottaa sivuston ulkoasun muuttamista. Muuttamalla tiedostossa olevaa määritystä muutos ohjautuu kaikille sivuille ja niihin kohtiin, joita muutos koskee. Tämä yksinkertaistaa sivuston ylläpitoa, ja esimerkiksi sivuston värimaailmaa on helppo muuntaa tyylitiedostojen avulla. Työssä käytetyn tyylitiedoston sisältö löytyy liitteestä 5. Tyylitiedosto sisältää myös navigointipalkkia määrittäviä ominaisuuksia.

JQuery

JQuery on erillinen kirjasto, joka sisältää valmiiksi määriteltyjä funktioita ja ominaisuuksia, joita pystytään hyödyntämään verkkosivuilla. JQuery täytyy erikseen ottaa käyttöön lataamalla Internetistä palvelimelle Javascript-tiedosto tai viittaamalla sellaiseen Internetissä. Opinnäytetyössä käytettiin JQueryn versiota 2.1.1. Tiedosto ladattiin JQueryn verkkosivuilta paikalliseksi tiedostoksi. Näin ollen pystytään varmistumaan siitä, että kirjasto on aina sivuston käytettävissä. Paikallisen kopion tallentaminen onnistuu komentorivillä W*get*-sovelluksen avulla.

```
sudo wget http://code.jquery.com/jquery-2.1.1.min.js
```

Tiedosto tallentuu siihen kansioon, jossa komento on kirjoitettu. Tiedosto on siirretty palvelimella verkkosivujen kanssa samaan kansioon, jolloin siihen voidaan viitata HTML-koodissa seuraavasti:

```
<script type="text/javascript" src="jquery-2.1.1.min.js"></script>
```

Opinnäytetyötä tehtäessä jouduttiin hyödyntämään JQueryä yhden osan toteuttamisessa. Kyseessä on työntekijöiden paikan muuttaminen vuorolistalla. Tämän toteutus haluttiin tehdä kaksiulotteisella matriisilla, joka on toteutettu radiopainikkeiden avulla. Ainoastaan HTML ja PHP-kieltä hyödyntämällä ei voida luoda kaksiulotteista matriisia, koska radiopainike pystytään liittämään vain yhteen ryhmään. JQuery mahdollistaa *Class*-määrityksen, jonka avulla pystytään liittämään yksi radiopainike osaksi kahta eri ryhmää.

Liitteestä 6 löytyy radiopainikematriisin koodi, josta näemme, kuinka radiopainikkeeseen on määritetty kaksi ominaisuutta *name* ja *data-col*. Näistä *name* on HTML-kielessä määritetty ominaisuus ja *data-col* on JQueryn tuoma ominaisuus. Koodissa vaakasuorat rivit on sidottu JQueryllä ja pystysarakkeet HTML-kielellä.

5.5 Ohjelmointirajapinta

Ohjelmointirajapinnan koodi, tässä tapauksessa PHP, kirjoitetaan osaksi sovelluksen koodia. Koodi suoritetaan siinä vaiheessa, kun se tulee sovelluksen koodissa vastaan. PHP koodin aloittaa merkintä *<?php* ja lopettaa *?>*-merkintä. Ohjelmointirajapinnan toimintaa varten täytyy PHP:hen asentaa MySQLi-laajennus. Se mahdollistaa tietokantaan yhdistämisen ja tietokantakomentojen suorittamisen PHP:n kautta.

Tietokantaan yhdistäminen ja yhteyden päättäminen

PHP:n avulla tehdyt MySQL-kyselyt vaativat, että tietokantaan on luotu istunto. Istunto luodaan aina ennen varsinaisten hakujen suorittamista tietokantaan. Yhdistäminen tapahtuu seuraavilla parametreilla:

```
<?php
$servername = "localhost”;
$username = "kayttaja";
$password = "salasana";
$schema = "tietokanta";

// Yhteyden muodostaminen tallennetaan muuttujaan $conn
$conn = new mysqli($servername, $username, $password, $schema);
?>
```

Mikäli MySQL hyväksyy kirjautumisen yhteydessä annetun käyttäjätunnus ja salasana -parin, on yhteys muodostettu ja PHP:n avulla voidaan tehdä hakuja tietokantaan. Kun tarvittavat haut on tehty, tulee yhteys tietokantaan sulkea. Vaihtoehtoisesti yhteys voidaan sulkea vasta PHP-sivun lopussa. Sulkeminen tapahtuu yksinkertaisella komennolla

```
$conn->close();
```

Käyttäjän kirjautuminen palveluun

Työvuorolistan käyttäminen edellyttää aktiivista istuntoa. Istunto luodaan käyttäjän kirjautuessa sivustolle. Jokaisen sivun alussa tarkistetaan, onko istunto aktiivinen, ja

mikäli havaitaan, että istunto on vanhentunut tai käyttäjä yrittää sivustolle ilman kirjautumista, niin käyttäjä uudelleenohjataan kirjautumissivulle. Käyttäjän kirjautumisessa suoritettu salasanan tarkistus sekä istunnon luonti on selitetty liitteessä 7.

Kirjautumisvaiheessa suoritettu MySQL-kysely käyttäjätietojen hakemista varten on suojattu MySQL-injektiolta. Suojaus tapahtuu tekemällä *prepare*d *statement* -kysely, jossa MySQL-kysely lähetetään palvelimelle ensin ja arvot syötetään myöhemmin. Näin ollen MySQL-injektion suorittaminen kyselyssä on mahdotonta, koska kyselyyn lisättävät arvot eivät ole osa MySQL-kyselyä.

Käyttäjän salasanan vaihtaminen

Käyttäjän on mahdollista vaihtaa salasana omalle tunnukselleen kirjautumissivulta. Käyttäjältä kysytään tunnus, vanha salasana ja uusi salasana kaksi kertaa, jotta vältytään kirjoitusvirheiltä uudessa salasanassa. Käyttäjän oikeus vaihtaa salasana varmistetaan samalla tavalla kuin kirjautumisvaiheessa, mutta kirjautumisen sijasta käyttäjän uusi syötetty salasana kryptataan ja tallennetaan tietokantaan. Salasanan vaihtamisessa käytetty PHP-koodi löytyy liitteestä 8.

Koodissa suodatetaan käyttäjän syöttämät arvot MySQL-injektion varalta käyttämällä *real_escape_string*-funktiota. Funktio poistaa syötteestä pakenemismerkit, joita ovat NUL (ASCII 0), \n, \r, \, ’, ” ja CTRL+Z. (Mysqli::real_escape_string n.d.)

Arkipyhän tai koulutuksen merkitseminen tietokantaan

Vuorolistassa on mahdollista merkitä koulutuksia ja arkipyhäpäiviä päivän tietoihin. Tietojen tallennuksen yhteydessä selvitetään PHP:n IF-lausekkeilla, tarvitseeko aiemmat tiedot poistaa tai päivittää vai luodaanko kokonaan uusi tieto tietokantaan. Päiväkohtaiset merkinnät säilytetään tietokannassa omassa DAYS-nimisessä taulussa, jossa päivämäärä toimii taulun yksilöllisenä pääavaimena.

Tietojen kerääminen, käsittely ja tietokantaan kyselyn suorittaminen toteutetaan PHP-koodilla. Käytetty koodi löytyy liitteestä 9. Koodissa kerätään aluksi tarvittavat tiedot lomakkeen tiedoista ja osoiteriviltä, minkä jälkeen selvitetään tehtävä toimenpide ja luodaan sitä vastaava MySQL-lauseke. Lopuksi luotu lauseke ajetaan tietokantaan komentona.

6 Tietoturva

6.1 Salasanan kryptaus

Kryptaus

Käyttäjän syöttämä salasana kryptataan password_hash()-funktion avulla käyttäen Blowfish-algoritmia ja satunnaista suolausta. Kryptauksen ansiosta käyttäjän salasanaa ei tallenneta selkokielisenä tietokantaan, joten mahdollinen tietomurto tietokantaan ei anna käyttäjien salasanoja selville. Tämän lisäksi satunnainen suolaus mahdollistaa sen, että vaikka kahdella käyttäjällä olisi sama salasana, ei tietokantaan tallennetuista salasanatiedoista pystytä yhdistämään, että käyttäjillä on sama salasana.

Käyttäjän salasanan kryptaaminen tapahtuu kuvion 7 mukaisesti, jossa käyttäjän salasana ajetaan *Password_hash()*-funktion läpi käyttäen satunnaista suolausta. Lopputuloksena on kryptattu salasanatieto, joka tallennetaan tietokantaan.

Kuvio 7 Salasanan kryptaaminen password_hash()-funktiolla

Tietokantaan tallennettu salasana sisältää tiedon käytetystä salausalgoritmista, salauksen suolan sekä salatun salasanan. Pelkän salasanan tallentamisen lisäksi on tärkeää tallentaa käytetty salausalgoritmi sekä suola, jotta salasanaa voidaan verrata myöhemmin syötettävään kirjautumistietoon. Salasanan varmentaminen kirjautumisen yhteydessä tapahtuu kuvion 8 mukaisesti, jossa käyttäjän syöttämä salasana suolataan tietokantaan tallennetulla salasanatiedolla, joka sisälsi käytetyn suolan. Funktion antama arvo joko täsmää tietokannan tietoon, jolloin syötetty salasana on sama kuin tietokantaan tallennettu, tai sitten ne eroavat, jolloin syötetty salasana on väärä.

Kuvio 8 Salasanan varmentaminen kirjauduttaessa

Blowfish

Blowfish-algoritmin käyttäminen kryptauksessa rajoittaa käyttäjän syöttämän salasanan pituuden 72 merkkiin. Jos käyttäjä syöttää pidemmän salasanan, jätetään ylimenevät merkit huomioimatta. Salasanan kryptauksen vahvuus riippuu käytettävän algoritmin lisäksi myös kryptauksen vahvuudesta, joka voidaan määrittää *cost* -parametrillä. Oletuksena vahvuudeksi on asetettu *10*, mutta sitä voidaan muuttaa tarvittaessa. Mitä suurempi vahvuus on, sitä enemmän resursseja vaaditaan koneelta, ja sitä kauemmin koneella menee suorittaa salasanan kryptaaminen.

Kryptaamisen monimutkaisuuden määrittäminen laitteistolle sopivaksi

Seuraavalla koodilla määritetään, missä ajassa algoritmi tulee olla suoritettu, ja saadaan suurin käytettävä *cost*-parametri, joka kykenee aikamääreeseen:

```
<?php
$timeTarget = 0.05; // 50 millisekuntia
$cost = 8;
do {
   $cost++;
   $start = microtime(true);
   password_hash("test", PASSWORD_BCRYPT, ["cost" => $cost]);
   $end = microtime(true);
} while (($end - $start) < $timeTarget);
echo "Appropriate Cost Found: " . $cost . "\n";
?>
```

On tärkeää, että käytettävä arvo ei ole liian pieni, jolloin kryptaus on helpompi purkaa, mutta on myös pidettävä huolta siitä, että käyttäjäkokemus ei heikkene liian pitkän vasteen seurauksena. Tästä syystä on sopivaa valita 50-100 ms kryptaamisen enimmäiskestoksi palvelimen tehoja selvittäessä.

6.2 MySQL-injektio

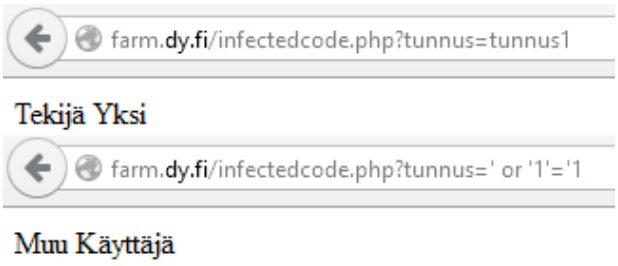
MySQL-injektiolla tarkoitetaan tilannetta, jossa tietokantaan päästään syöttämään haitallisia kyselyitä, jotka voivat paljastaa arkaluontoisia tietoja tietokannasta tai mahdollistaa pääsyn tietokantaan. MySQL-injektio on mahdollista toteuttaa paikoissa, joissa sivuilta luetaan muuttuvaa tietoa ja se syötetään tietokantaan. Yleensä tietoja luetaan esimerkiksi verkkosivujen osoiteriviltä tai tekstikentistä, kuten kirjautumistietoja syötettäessä. On tärkeää suojata tällaiset kentät injektiota vastaan, sillä taitavat hakkerit osaavat hyödyntää pienimmänkin haavoittuvuuden ja saada merkittäviä vahinkoja aikaan.

Tässä työssä käytetty palvelin on kokonaan sisäverkossa. Tästä huolimatta MySQL-injektio on otettu huomioon sellaisissa kohdissa, jossa sen toteuttaminen on mahdollista.

Seuraavassa esimerkkitapauksessa poimitaan sivun osoiteriviltä käyttäjän tunnus, jonka perusteella tulostetaan käyttäjän etu- ja sukunimi tietokannasta. Ensimmäisessä tapauksessa koodia ei ole suojattu MySQL-injektiolta ja se on altis murroille. Sivun koodi on pidetty esimerkin osalta yksinkertaisena.

```
<?php
include "connect.php";

// Poimitaan osoiterivin tunnus haavoittuvaisena
$tunnus = $_GET['tunnus'];
$sqlhaku = $conn->query("select * from EMPL where tcad='$tunnus'");
$haku = $sqlhaku->fetch_assoc();
echo $kayttaja = $haku['firstname'] . " " . $haku['lastname'];
?>
```

Kuviossa 9 on esiteltynä, kuinka aluksi on haettu normaalisti käyttäjän tunnuksella tiedot ja jälkimmäisessä haussa kuviota on hyödynnetty MySQL-injektion peruselementtiä, eli todetaan haun olevan aina totta merkkaamalla hakuun *or 1 = 1*.

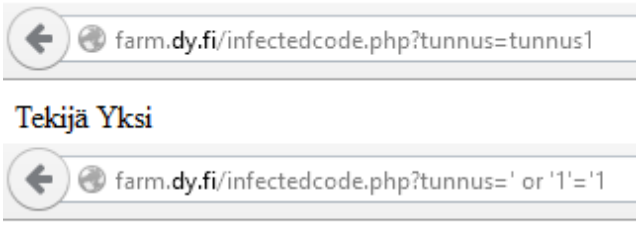


Kuvio 9 Haavoittuva MySQL-kysely tietokantaan

Seuraavaksi suojataan koodi yksinkertaisella *real_escape_string*-funktiolla, joka estää injektion toteuttamisen ja avataan vastaavat sivut kuin haavoittuvaisessa esimerkissä. MySQL-injektiosuojauksen kanssa koodi näyttää seuraavanlaiselta:

```
<?php
include "connect.php";

// Poimitaan osoiterivin tunnus suojattuna injektiolta
$tunnus = $conn->real_escape_string($_GET['tunnus']);
$sqlhaku = $conn->query("select * from EMPL where tcad='$tunnus'");
$haku = $sqlhaku->fetch_assoc();
echo $kayttaja = $haku['firstname'] . " " . $haku['lastname'];
?>
```

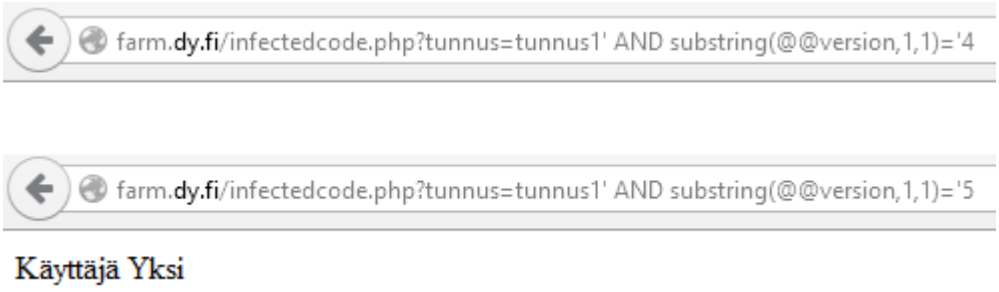
Nyt suojattuna sivusto toimii oikein eikä luovuta muita käyttäjätietoja kuin halutun. Tästä esimerkkinä kuviossa 10 nähty alempi haku, jossa tosilauseke *or 1 = 1* ei luovuta tietokannan ensimmäisen tietueen arvoja.



Kuvio 10 Suojattu MySQL-kysely tietokantaan

Ylemmät esimerkit ovat hyvin yksinkertaisia eivätkä vuoda itsessään arkaluontoista tietoa, mutta asiansa osaavalle henkilölle yksinkertaisella testillä löydetty haavoittuva kohta avaa ovet, ja hän pystyy ottamaan tietokannan haltuunsa ja saamaan sieltä tarvittavat tiedot ulos.

Seuraava yksinkertainen koe on kuvattu kuviossa 11, ja se on tehty haavoittuvaan järjestelmään. Siinä käyttäjä saa selville MySQL-tietokannan version.



Kuvio 11 Haavoittuvuus paljastaa käytetyn version MySQL-tietokannasta

7 Käyttöliittymä

7.1 Kirjautuminen

Autentikaatio

Työn edetessä selvitettiin mahdollisuutta Active Directory (AD) -ympäristöön integroimiselle, mutta koska oli aikataulullisesti epävarmaa, milloin tarvittavat oikeudet AD-autentikoinnin toteuttamiselle saadaan, päädyttiin tekemään paikallinen autentikaatio. Paikallisessa autentikaatiossa käyttäjän tunnistetiedot säilytetään tietokannassa osana käyttäjän muita tietoja.

Salasanan vaihtaminen

Kirjautumissivulta käyttäjälle on luotu mahdollisuus vaihtaa itse käytössä oleva salasana. Salasanan vaihto vaatii perinteisesti käyttäjätunnuksen, vanhan salasanan ja kaksi kertaa uuden salasanan kirjoitusvirheiden välttämiseksi. Järjestelmä tarkistaa ensin vanhan salasanan oikeellisuuden järjestelmässä olevaan tietoon kryptaamalla sen ja vertaamalla sitä tietokantaan tallennettuun tietoon. Tämän jälkeen tarkistetaan, että uusi salasana on syötetty kaksi kertaa ja ne täsmäävät keskenään. Mikäli kaikki on kunnossa, salasana kryptataan ja tallennetaan tietokantaan, jonka jälkeen uudella salasanalla voidaan kirjautua.

7.2 Navigointi

Jokaisen sivun ylälaitaan on rakennettu navigointipalkki, joka sisältää olennaisimmat toiminnot kyseiselle sivulle. Työvuorolista, joka avautuu kirjauduttaessa, on sivuston oletussivu ja toimii puurakenteen juurena. Eri osioiden välillä pystytään liikkumaan navigointipalkin kautta, ja hierarkian syvemmiltä tasoilta päästään navigointipalkin avulla ylöspäin. Tietyille sivuille ei pääse navigointipalkin kautta vaan ainoastaan sivuilla olevista muista linkeistä.

7.3 Katselutila

Sivuston perusnäkymänä toimii kaikille käytössä oleva katselutila, jossa näkyvät kerrallaan enintään kolmen (3) viikon työvuorot yhdeltä vuorolistalta. Vuorolistojen välillä pystytään liikkumaan navigointipalkissa olevasta valikosta. Jokaisen työntekijän vuoron kohdalla on olennainen tieto nähtävillä eli työvuoron aloitus- ja lopetusaika sekä mahdollinen rooli ja vuorovastaavan tehtävä. Tämän lisäksi viikoittainen varallaolo merkitään vihreällä taustavärillä, ja lisätietoja saa viemällä hiiren työvuoron päälle. Myös työajan lyhennykset, arkipyhän poistot ja kommentit muuttavat työvuoron pohjaväriä, ja tarkemmat tiedot saadaan viemällä osoitin työvuoron päälle.

Jokaiselle päivälle tietokantaan on tallennettu tieto tarvittavien työntekijöiden määrästä. Arkipyhinä käytetään sunnuntain vahvuutta. Jokaisen päivän kohdalla tarkistetaan aamu-, ilta- ja yövuorolaisten lukumäärä ja verrataan sitä tietokantaan tallennettuun tietoon. Mikäli työntekijöiden lukumäärä poikkeaa normaalista, vuorolista ilmoittaa siitä korostamalla vahvuuden punaisella tai vihreällä värillä riippuen siitä, ylitetäänkö vai alitetaanko vahvuus. Vastaavaa toimenpidettä hyödynnetään myös roolien kanssa, eli mikäli roolitettuja henkilöitä on normaalista poikkeava määrä, siitä nousee vuorolistaa hallinnoiville henkilöille huomio. Aiemmin mainitut asiakohdat on esitelty liitteessä 10.

7.4 Hallinnointitila

Vuorolistaa muokkaavilla henkilöillä on mahdollisuus aktivoida hallinnointitila navigointipalkista. Tämä mahdollistaa yksittäisten vuorojen muokkaamisen, uusien vuorojen lisäämisen vuorolistaan, käyttäjien hallinnan sekä lomien ja muiden vuorolistaa koskevien muutoksien kirjaamisen. Jokaisen työntekijän vuoroon on lisätty radiopainike, joka tulee näkyviin, kun hallinnointitila on käytössä. Klikkaamalla painiketta siirrytään erilliselle sivulle muokkaamaan kyseistä työvuoroa. Muutokset tallennetaan ja ne viedään tietokantaan, josta ne ovat heti kaikkien saatavilla. Kuviossa 12 näkyvät hallinnointitilan tuomat painikkeet.

			Työntekijä1	Työntekijä2	Työntekijä3	Työntekijä4	Työntekijä5	Työntekijä6
2015 / viikko 6			○ Siirrä ei julkaistu rivi tähän					
SU	1. helmi ○	2 / 2 / 1	7-17 S ○ SL	7-14:30 AV ○	--- ○	13-23 S ○	-7:30 ○	21:30- ○ SL2
MA	2. helmi ○	4 / 2 / 1	12-22 S ○	7-17 ○	13-23 AV ○ SL2	--- ○	--- ○	-7:30 ○
TI	3. helmi ○	3 / 2 / 1	21:30- ○ SL2	12-22 AV ○	13-23 S ○ SL	--- ○	--- ○	--- ○
KE	4. helmi ○	3 / 2 / 1	-7:30 ○	21:30- ○ SL	--- ○	7-14:30 S ○	7-17 AV ○	--- ○
TO	5. helmi ○	3 / 2 / 1	--- ○	-7:30 ○	--- ○	15:30-23 S ○	7-17 S ○	12-22 AV ○ SL2
PE	6. helmi ○	3 / 2 / 1	--- ○	--- ○	12-22 AV ○	21:30- ○ SL2	7-17 AV ○	15:30-23 S ○ SL2
LA	7. helmi ○	2 / 2 / 1	7-14:30 AV ○	--- ○	14:30-22 S ○ SL2	-7:30 ○ 21:30-	--- ○	13-23 AV ○
vko. 2			37.5h	37.5h	37.5h	37.5h	37.5h	37.5h

Kuvio 12 Hallinnointitilan tuomat radiopainikkeet

7.5 Vuoropohja

Jokaiselle vuorolistalle on luotu oma vuoropohjakierto, jota pystytään tarvittaessa muokkaamaan. Vuoropohjaan tehtävät muutokset eivät muuta jo vuorolistoilla olevia vuoroja, mutta vaikuttavat listalle lisättäviin vuoroihin. Vuoropohja huomioi normaalin vuorolistan tavoin vahvuuden sekä roolien määrän. Vuorolistan hallinnoijilla on normaalin näkymän tavoin mahdollisuus vuorojen muokkaamiseen.

7.6 Oma sivu

Jokaisella käyttäjällä on oma sivu, jossa näkyvät henkilön omat työvuorot koko vuoden ajalta. Jatkossa opinnäytetyön ulkopuolelta tänne osioon on suunniteltu toteutettavaksi vuorolisien laskenta, käyttäjän kielivalinta sekä vuorotietojen tulostaminen paperiseen versioon ja työvuorojen vienti kalenteriin. Työvuorojen kalenteriin vienti toteutetaan tallentamalla tietokannan tiedot Comma-Separated Values (.CSV) -muotoisessa tiedostossa tietokoneelle, minkä jälkeen se voidaan viedä haluttuun sähköpostijärjestelmään. Tuettuja sähköpostijärjestelmiä tulevat olemaan Googlen ja Microsoft Outlookin kalenterit.

7.7 Käyttäjien hallinta

Tietokanta sisältää käyttäjän kirjautumistietojen lisäksi myös vuorolistan kannalta merkittäviä tietoja kuten työsuhteen tyypin, mihin vuorolistaan henkilö kuuluu ja onko henkilöllä hallintaoikeutta järjestelmässä. Tämän lisäksi käyttäjältä on kerätty nimitiedot ja työpuhelinnumero, joka näytetään henkilön varallaolon yhteydessä muille työntekijöille. Käyttäjän hallintasivu on nähtävillä liitteessä 11.

Uusia käyttäjiä voidaan lisätä järjestelmään, minkä jälkeen he ovat käytettävissä vuorolistoilla. Käyttäjä voidaan myös siirtää vuorolistalta toiselle tai poistaa kokonaan. Vakituisia työntekijöitä pystytään siirtämään vuorolistalla paikalta toiselle liitteessä 12 näkyvällä radiopainikematriisilla.

8 Järjestelmän pilotointi

8.1 Toteutus

Kun opinnäytetyö oli saatu siirrettyä kehitysympäristöstä tuotantoverkon palvelimelle ja tarkistettua, että kaikki tarvittavat ominaisuudet toimivat niin kuin niiden kuuluikin, aloitettiin käyttäjille avoin pilottijakso. Pilottijakson aikana vanhaa ja uutta vuorolistaa ajetaan rinnakkain, ja sen tarkoituksena on löytää mahdolliset ongelmat järjestelmän käytössä ja kerätä kehitysideoita jatkoa varten. Jakson on suunniteltu kestävän kaksi (2) viikkoa, mutta riippuen palautteen määrästä ja tarvittavista korjaustoimenpiteistä voidaan sitä laajentaa tarvittaessa 1-2 viikolla.

8.2 Käyttäjäpalaute

Käyttäjiä on pyydetty antamaan järjestelmästä palautetta, jolla järjestelmää pystytään kehittämään ja ennen kaikkea korjaamaan kriittiset järjestelmässä olevat virheet. Työtä tehtäessä järjestelmästä on pyritty karsimaan pois virheelliset toiminnallisuudet, mutta näinkin isossa työssä niitä voi edelleen esiintyä. Käyttäjäpalautteen kerääminen on erittäin tärkeää työn kannalta, koska eri käyttäjät käyttävät palvelua eri tavalla ja osaavat havainnoida epäkohtia, joita työn tekijä ei osaa ottaa huomioon.

8.3 Analysointi

Annettu käyttäjäpalaute analysoitiin ja jaettiin kahteen eri luokkaan. Oli kehittäviä kommentteja, jotka kirjataan ylös ja toteutetaan mahdollisuuksien mukaan opinnäytetyön ulkopuolella. Tämän lisäksi oli järjestelmän kannalta kriittisiä palautteita, eli käyttäjien havaitsemia epäkohtia tai virheitä toiminnoissa, jotka vaikuttavat koko järjestelmän toimintaan. Näiden löytäminen ja korjaaminen on erittäin tärkeää, jotta järjestelmä voidaan ottaa käyttöön tuotannossa.

9 Yhteenveto

9.1 Työn lopputulos

Opinnäytetyö sai alkunsa tarpeesta luoda uusi toimivampi järjestelmä työvuorojen ylläpitämiseen. Työ päätettiin tehdä tietokantojen ja verkkosivujen avulla ja työn lopputuloksena saatiin toteutettua toimiva järjestelmä, joka pystytään ottamaan käyttöön tuotannossa. Näin ollen opinnäytetyö saavutti sille määritetyt vaatimukset ja työn lopputulos oli onnistunut. Työn suorittaminen oli oletettua hankalampi ja työläämpi projekti. Työn laajuudesta antaa viitteitä PHP-tiedostoissa olevat noin 4400 riviä koodia.

Työn teoriaosuus päätettiin kirjoittaa tietokantojen perusteista, ja tämän lisäksi käytiin läpi järjestelmän kannalta oleelliset palvelut ja niiden toimintaa. Dokumentoinnin osalta ei ollut järkevää käydä koko työtä läpi, koska kyseessä on massiivinen työ. Sen sijaan keskityttiin yleisiin toiminnallisuuksiin, joita voidaan hyödyntää yksittäisinä palasina missä tahansa työssä ilman, että ne ovat tähän työhön sidonnaisia.

Testausvaihe ja korjaukset

Työn testausjaksoa jouduttiin pidentämään, koska testaajilla oli muita työkiireitä eikä järjestelmää keritty näin ollen täysin testaamaan ensimmäisen kahden viikon aikana. Isompia ongelmia ei havaittu työssä testausjakson aikana, mutta muutamia pieniä kehitysideoita testaajilta löytyi sivustolle. Nämä muutokset oli helppo toteuttaa järjestelmään, ja koska kriittisiä ongelmia ei havaittu, nämä korjaukset tehtiin järjestelmään heti.

9.2 Kehitysideat

Opinnäytetyötä tehtäessä tuli vuorolistalle kehitysideoita sekä itseltä että työtovereilta. Näitä ideoita on tarkoitus alkaa työstämään opinnäytetyön jälkeen. Alla esitetään lista ominaisuuksista, joiden toteuttamista tullaan pohtimaan ja selvitellään mahdollisuuksia, kuinka ominaisuudet voitaisiin toteuttaa.

- AD-autentikaation totetutus
- Vuorolisien listaus henkilön omilla sivuilla
- Taustahenkilöiden vuorovastaavan kierron lisääminen
- Ohjeistukset työntekijän tehtävistä ja rooleista
- Kaksikielinen toteutus vuorolistalle
- Sivujen hallinnan helpottaminen ja optimointi
- Työvuorojen vienti kalenteriin
- Vuorolistan käytön laajentaminen yrityksen muissa yksiköissä?

9.3 Pohdinta

Opinnäytetyön toiminnallisen osion työstäminen oli mielekästä, ja se opetti minulle paljon erilaisia toteutustapoja verkkosivujen puolelta. Lisäksi sain syvennettyä omaa osaamistani tietokantojen kanssa. Oli hienoa nähdä työn kehittyminen vaihe vaiheelta, vaikka aluksi näin isossa työssä oli ongelmana päästä vauhtiin, koska palaset ovat niin paljon sidoksissa toisiinsa, ettei yhtä asiaa voi nut saada kokonaan valmiiksi ilman toisten palasien tukea. Työ myös eli jonkin verran tekemisen mukana, koska suunnitteluvaiheessa ei ole osattu ottaa kaikkia asioita huomioon ja niitä on jälkikäteen lisätty työhön. Työ oli vähällä lähteä myös laajenemaan vielä mittavampiin suhteisiin, joten oli erittäin hyvä tehdä tarkat rajaukset opinnäytetyölle. Rajausten ansiosta pystyttiin siirtämään valtaosa uusista ideoista opinnäytetyön ulkopuolella suoritettaviksi jatkotoimenpiteiksi.

Laajuudesta johtuen myös kirjallisen työn kirjoittaminen oli ajoittain hankalaa. Oli haasteita etsiä työstä sellaisia komponentteja, jotka voidaan sisällyttää raportointiin. Komponentit eivät saa myöskään olla liian syvällisiä toiminteita työn kannalta, vaan tarkoituksena oli, että niitä voidaan käyttää missä tahansa tilanteessa ilman kytköstä juuri tähän työhön.

Lähteet

Häkkinen, A. 2014a. Tietokantarajapinnat johdanto. Jyväskylän ammattikorkeakoulun Optima-opiskeluympäristö, työtila IIZO3021 Tietokantarajapinnat. Viitattu 23.4.2015. https://optima.jamk.fi.

Häkkinen, A. 2014b. Tietokantarajapinnat SQL-perusteet osa 3. Jyväskylän ammattikorkeakoulun Optima-opiskeluympäristö, työtila IIZO3021 Tietokantarajapinnat. Viitattu 23.4.2015. https://optima.jamk.fi.

Häkkinen, A. 2014c. Tietokantarajapinnat SQL-perusteet osa 4. Jyväskylän ammattikorkeakoulun Optima-opiskeluympäristö, työtila IIZO3021 Tietokantarajapinnat. Viitattu 23.4.2015. https://optima.jamk.fi.

Häkkinen, A. 2014d. Ohjelmointirajapinta. Jyväskylän ammattikorkeakoulun Optima-opiskeluympäristö, työtila IIZO3021 Tietokantarajapinnat. Viitattu 27.4.2015. https://optima.jamk.fi.

Mysqli::real_escape_string. N.d. Dokumentaatio PHP:n sivuilla. Viitattu 8.5.2015. http://php.net/manual/en/mysqli.real-escape-string.php.

Privileges provided by MySQL. N.d. Manuaali ohjelmiston sivuilla. Viitattu 23.4.2015. https://dev.mysql.com/doc/refman/5.5/en/privileges-provided.html.

Relaatiotietokannat luento 3. 2011. Avoimen yliopiston sivusto. Viitattu 19.5.2015. http://appro.mit.jyu.fi/tiedonhallinta/luennot/luento3/.

Relaatiotietokantojen peruskäsitteet. 2004. Avoimen yliopiston sivusto. Viitattu 23.4.2015. http://appro.mit.jyu.fi/doc/tiedonhallinta/tietokannat/index2.html.

The MySQL Access Privilege System. N.d. Manuaali ohjelmiston sivuilla. Viitattu 23.4.2015. https://dev.mysql.com/doc/refman/5.5/en/privilege-system.html.

TeliaSonera in brief. 29.1.2015. Yrityksen intrassa oleva esittelymateriaali. Viitattu 14.5.2015. http://ts.teliasonera.net/GeneralInformation/about/Documents/Corporate%20presentation%20Q4%202014.pptx.

Liitteet

Liite 1 MySQL-tietokannan taulut

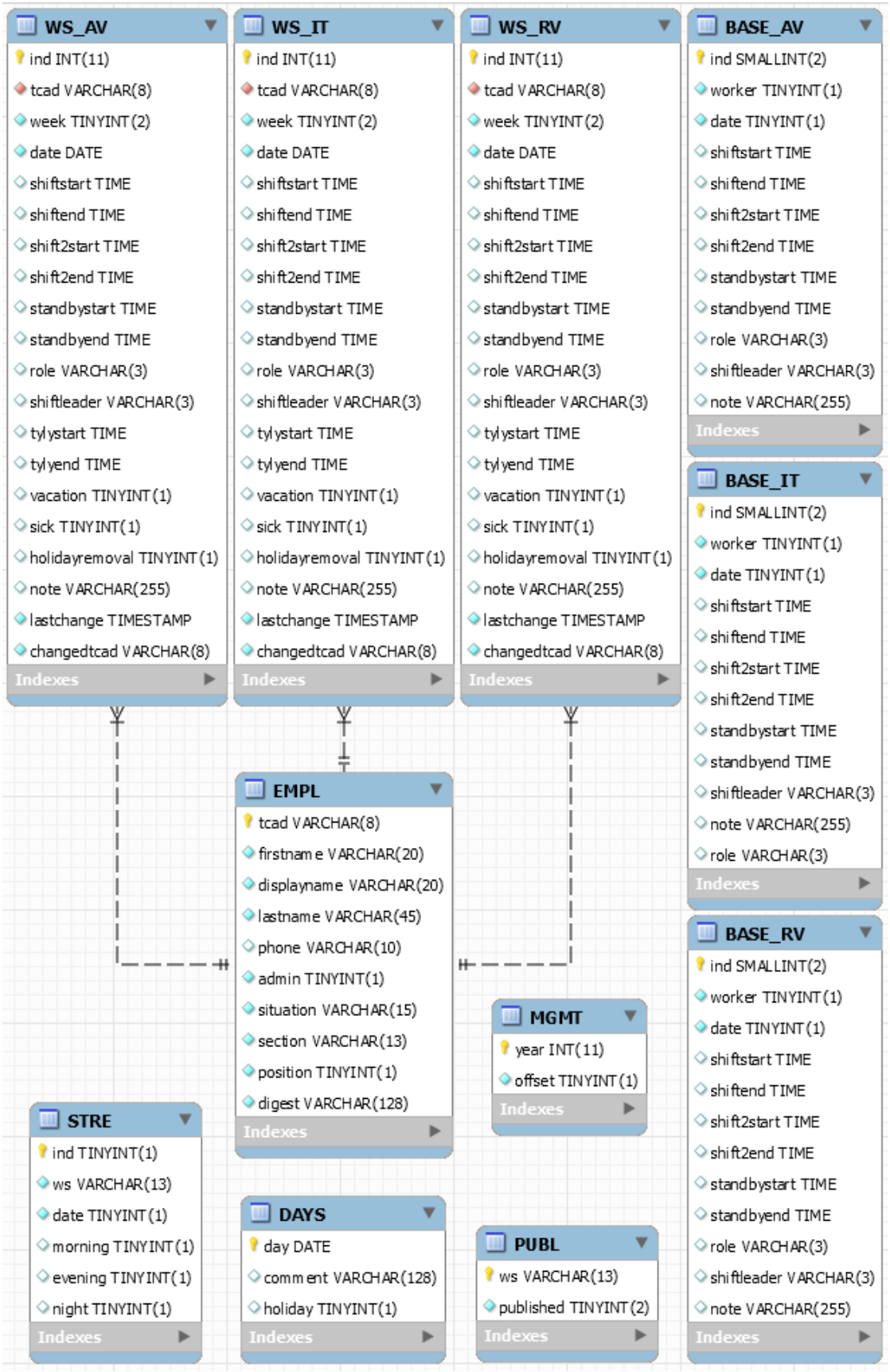
```
CREATE TABLE `BASE_AV` (
  `ind` smallint(2) NOT NULL AUTO_INCREMENT,
  `worker` tinyint(1) NOT NULL,
  `date` tinyint(1) NOT NULL,
  `shiftstart` time DEFAULT NULL,
  `shiftend` time DEFAULT NULL,
  `shift2start` time DEFAULT NULL,
  `shift2end` time DEFAULT NULL,
  `standbystart` time DEFAULT NULL,
  `standbyend` time DEFAULT NULL,
  `role` varchar(3) DEFAULT NULL,
  `shiftleader` varchar(3) DEFAULT NULL,
  `note` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`ind`),
  UNIQUE KEY `index_UNIQUE` (`ind`)
) ENGINE=InnoDB AUTO_INCREMENT=71 DEFAULT CHARSET=utf8mb4 COM-
MENT='Shift rotation for AV employees';

CREATE TABLE `BASE_IT` (
  `ind` smallint(2) NOT NULL AUTO_INCREMENT,
  `worker` tinyint(1) NOT NULL,
  `date` tinyint(1) NOT NULL,
  `shiftstart` time DEFAULT NULL,
  `shiftend` time DEFAULT NULL,
  `shift2start` time DEFAULT NULL,
  `shift2end` time DEFAULT NULL,
  `standbystart` time DEFAULT NULL,
  `standbyend` time DEFAULT NULL,
  `shiftleader` varchar(3) DEFAULT NULL,
  `note` varchar(255) DEFAULT NULL,
  `role` varchar(3) DEFAULT NULL,
  PRIMARY KEY (`ind`),
  UNIQUE KEY `index_UNIQUE` (`ind`)
) ENGINE=InnoDB AUTO_INCREMENT=64 DEFAULT CHARSET=utf8mb4 COM-
MENT='Shift rotation for IT employees';

CREATE TABLE `BASE_RV` (
  `ind` smallint(2) NOT NULL AUTO_INCREMENT,
  `worker` tinyint(1) NOT NULL,
  `date` tinyint(1) NOT NULL,
  `shiftstart` time DEFAULT NULL,
  `shiftend` time DEFAULT NULL,
  `shift2start` time DEFAULT NULL,
  `shift2end` time DEFAULT NULL,
  `standbystart` time DEFAULT NULL,
  `standbyend` time DEFAULT NULL,
  `role` varchar(3) DEFAULT NULL,
  `shiftleader` varchar(3) DEFAULT NULL,
  `note` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`ind`),
  UNIQUE KEY `index_UNIQUE` (`ind`)
) ENGINE=InnoDB AUTO_INCREMENT=71 DEFAULT CHARSET=utf8mb4 COM-
MENT='Shift rotation for RV employees';
```

```
CREATE TABLE `DAYS` (
  `day` date NOT NULL,
  `comment` varchar(128) DEFAULT NULL,
  `holiday` tinyint(1) DEFAULT NULL,
  PRIMARY KEY (`day`),
  UNIQUE KEY `day_UNIQUE` (`day`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='Information regarding
specific date and also holiday marking';

CREATE TABLE `MGMT` (
  `year` int(11) NOT NULL,
  `offset` tinyint(1) NOT NULL,
  PRIMARY KEY (`year`),
  UNIQUE KEY `year_UNIQUE` (`year`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

CREATE TABLE `PUBL` (
  `ws` varchar(13) NOT NULL,
  `published` tinyint(2) NOT NULL,
  PRIMARY KEY (`ws`),
  UNIQUE KEY `WS_UNIQUE` (`ws`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='Week indicator of pub-
lished shifts';

CREATE TABLE `STRE` (
  `ind` tinyint(1) NOT NULL AUTO_INCREMENT,
  `ws` varchar(13) NOT NULL,
  `date` tinyint(1) NOT NULL,
  `morning` tinyint(1) DEFAULT NULL,
  `evening` tinyint(1) DEFAULT NULL,
  `night` tinyint(1) DEFAULT NULL,
  PRIMARY KEY (`ind`),
  UNIQUE KEY `ind_UNIQUE` (`ind`)
) ENGINE=InnoDB AUTO_INCREMENT=22 DEFAULT CHARSET=utf8mb4 COM-
MENT='Strengths of shifts for each list';
```

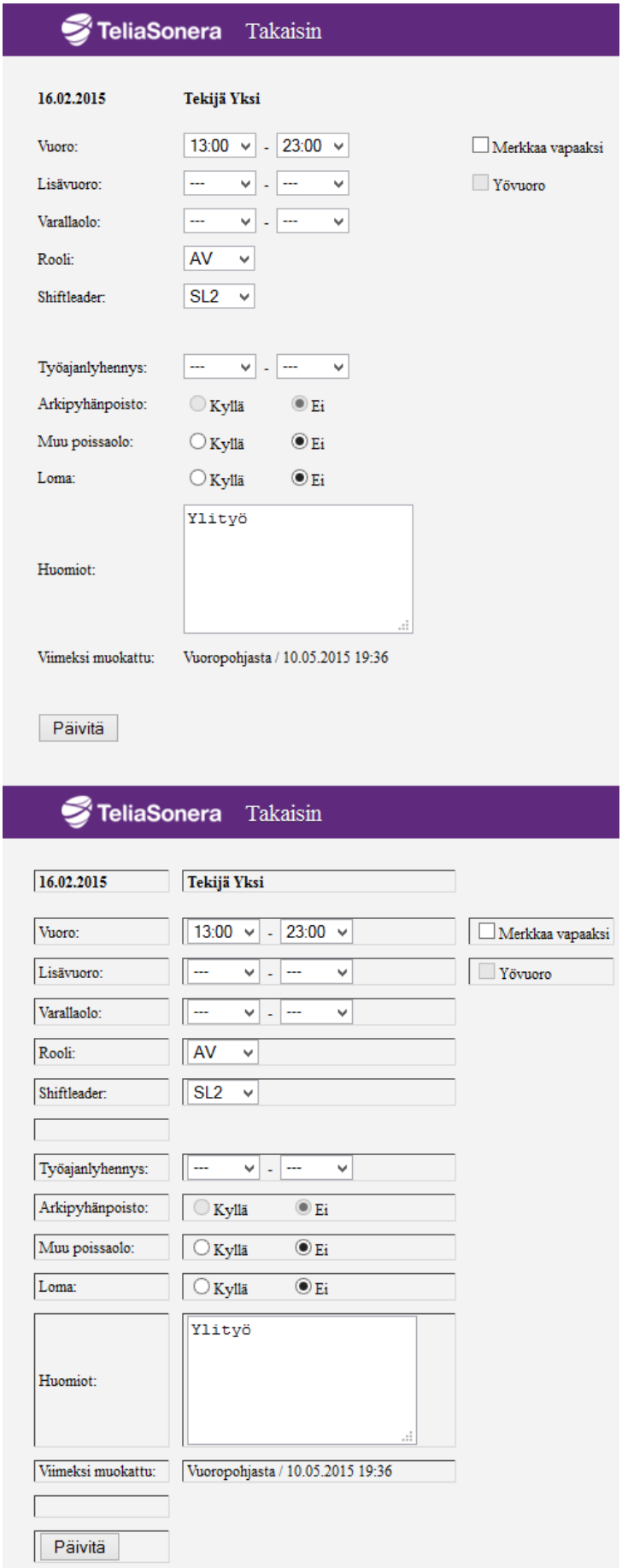
```
CREATE TABLE `WS_AV` (
  `ind` int(11) NOT NULL AUTO_INCREMENT,
  `tcad` varchar(8) NOT NULL,
  `week` tinyint(2) NOT NULL,
  `date` date NOT NULL,
  `shiftstart` time DEFAULT NULL,
  `shiftend` time DEFAULT NULL,
  `shift2start` time DEFAULT NULL,
  `shift2end` time DEFAULT NULL,
  `standbystart` time DEFAULT NULL,
  `standbyend` time DEFAULT NULL,
  `role` varchar(3) DEFAULT NULL,
  `shiftleader` varchar(3) DEFAULT NULL,
  `tylystart` time DEFAULT NULL,
  `tylyend` time DEFAULT NULL,
  `vacation` tinyint(1) DEFAULT '0',
  `sick` tinyint(1) DEFAULT '0',
  `holidayremoval` tinyint(1) DEFAULT '0',
  `note` varchar(255) DEFAULT NULL,
  `lastchange` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `changedtcad` varchar(8) NOT NULL DEFAULT 'system',
  PRIMARY KEY (`ind`),
  UNIQUE KEY `index_UNIQUE` (`ind`),
  KEY `fk_avtcad_idx` (`tcad`),
  CONSTRAINT `fk_avtcad` FOREIGN KEY (`tcad`) REFERENCES `EMPL` (`tcad`) ON DE-
LETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=99 DEFAULT CHARSET=utf8mb4 COM-
MENT='Worksheet of AV';

CREATE TABLE `WS_IT` (
  `ind` int(11) NOT NULL AUTO_INCREMENT,
  `tcad` varchar(8) NOT NULL,
  `week` tinyint(2) NOT NULL,
  `date` date NOT NULL,
  `shiftstart` time DEFAULT NULL,
  `shiftend` time DEFAULT NULL,
  `shift2start` time DEFAULT NULL,
  `shift2end` time DEFAULT NULL,
  `standbystart` time DEFAULT NULL,
  `standbyend` time DEFAULT NULL,
  `role` varchar(3) DEFAULT NULL,
  `shiftleader` varchar(3) DEFAULT NULL,
  `tylystart` time DEFAULT NULL,
  `tylyend` time DEFAULT NULL,
  `vacation` tinyint(1) DEFAULT '0',
  `sick` tinyint(1) DEFAULT '0',
  `holidayremoval` tinyint(1) DEFAULT '0',
  `note` varchar(255) DEFAULT NULL,
  `lastchange` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `changedtcad` varchar(8) NOT NULL DEFAULT 'system',
  PRIMARY KEY (`ind`),
  UNIQUE KEY `index_UNIQUE` (`ind`),
  KEY `fk_ittcad_idx` (`tcad`),
  CONSTRAINT `fk_ittcad` FOREIGN KEY (`tcad`) REFERENCES `EMPL` (`tcad`) ON DE-
LETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=85 DEFAULT CHARSET=utf8mb4 COM-
MENT='Worksheet of IT';
```

```
CREATE TABLE `WS_RV` (
  `ind` int(11) NOT NULL AUTO_INCREMENT,
  `tcad` varchar(8) NOT NULL,
  `week` tinyint(2) NOT NULL,
  `date` date NOT NULL,
  `shiftstart` time DEFAULT NULL,
  `shiftend` time DEFAULT NULL,
  `shift2start` time DEFAULT NULL,
  `shift2end` time DEFAULT NULL,
  `standbystart` time DEFAULT NULL,
  `standbyend` time DEFAULT NULL,
  `role` varchar(3) DEFAULT NULL,
  `shiftleader` varchar(3) DEFAULT NULL,
  `tylystart` time DEFAULT NULL,
  `tylyend` time DEFAULT NULL,
  `vacation` tinyint(1) DEFAULT '0',
  `sick` tinyint(1) DEFAULT '0',
  `holidayremoval` tinyint(1) DEFAULT '0',
  `note` varchar(255) DEFAULT NULL,
  `lastchange` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `changedtcad` varchar(8) NOT NULL DEFAULT 'system',
  PRIMARY KEY (`ind`),
  UNIQUE KEY `index_UNIQUE` (`ind`),
  KEY `fk_rvtcad_idx` (`tcad`),
  CONSTRAINT `fk_rvtcad` FOREIGN KEY (`tcad`) REFERENCES `EMPL` (`tcad`) ON DE-
LETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=99 DEFAULT CHARSET=utf8mb4 COM-
MENT='Worksheet of RV';
```

Liite 2 Tietokannan kaaviokuva



Liite 3 Taulukkorakenteen hyödyntäminen työssä



Liite 4 Lomakkeen käyttö kirjautumisessa

```
<form method="post" action="login.php">
 <table border="0px" cellspacing="10px" cellpadding="5px">
 <th width="225px" colspan="2">
  <h1>Command Center<br>-vuorolista</h1>
 </th>
 <tr>
 <?php if($error == '1') {?>
 <td colspan="2"><font color="red"><b>Käyttäjätunnus ja/tai salasana virheelli-
nen</b></font></td>
 <tr>
 <?php } ?>
 <td>Käyttäjätunnus:</td>
 <td><input type="text" name="tcad" required></td>
 <tr>
 <td>Salasana:</td>
 <td><input type="password" name="passwd" width="200px" required></td>
 <tr>
 <td></td>
 <td><a href="changepw.php">Vaihda salasana</a></td>
 <tr>
 <tr>
 <td></td>
 <td><input type="submit" value="Kirjaudu sisään"></td>
 <tr>
 <td width="90px"></td>
 <td width="150px"></td>
 <td width="230px"></td>
 <tr>
 <td height="220px" colspan="3">
  <div id="changelog">
   <?php include "changelog.txt";?>
  </div>
 </td>
 <tr>
 <td>Version: 0.9 alpha</td>
 <td colspan="2">
  <right>Järjestelmään liittyvissä ongelmissa ota yhteys <a
href="mailto:f1217@student.jamk.fi?Subject=Vuorolista" tar-
get="_top">ylläpitäjään.</a></right>
 </td>
 </table>
</form>
```

Liite 5 Tyylitiedoston sisältö

```
html,body {
	margin: 0px;
	padding: 0px;
	width: 100%;
	background-color: #5F2A7E;
	position: absolute;
	top: -7px;
}

#header {
	position: fixed;
	background-color: #5F2A7E;
	color: white;
	height: 42px;
	width: 100%;
	top: 0px;
	left: 0px;
}

#content {
	position: fixed;
	top: 42px;
	bottom: 0px;
	width: 100%;
	left: 0px;
	right: 0px;
	overflow-y: auto;
	overflow-x: auto;
	background-color: #F3F3F3;
	color: black;
}

#ownshifts {
	border: 1px solid;
	position:absolute;
	overflow-x: hidden;
	overflow-y: auto;
	background-color: #F3F3F3;
	width: 500px;
	height: 600px;
}

#shiftexport {
	position: fixed;
	top: 700px;
	left: 15px;
	background-color: #F3F3F3;
	width: 500px;
	height: 220px;
}
```

```
#changelog {
	position: static;
	width: 500px;
	height: 200px;
	overflow-y: auto;
	overflow-x: hidden;
	border: 1px solid;
	padding: 5px;
}

table {
	padding-left: 15px;
	padding-top: 15px;
	border: 1px;
	font-size: 12px;
}

td {
	padding: 0px 3px 0px 3px;
}

top {
	vertical-align: top;
}

bottom {
	vertical-align: bottom;
}

left {
	text-align: left;
}

right {
	text-align: right;
	float: right;
	right: 2px;
}

center {
	text-align: center;
	vertical-align: center;
}

rb {
	height: 10px;
	width: 10px;
}

a {
	color: #5F2A7E;
	text-decoration:none;
}
a:hover {
	color: #5F2A7E;
	text-decoration:underline;

}
```

```
/*Navigointi palkki on otettu sivustolta ja muokattu omiin tarpeisiin soveltuvaksi.
http://inspirationalpixels.com/tutorials/creating-a-dropdown-menu-with-html-css */

.clearfix:after {
	display:block;
	clear:both;
}

/*----- Menu Outline -----*/
.menu-wrap {
	width:100%;
	box-shadow:0px 1px 3px rgba(0,0,0,0.2);
	background:#5F2A7E;
}

.menu {
	width:100%;
	left: 0px;
	margin:0px;
	background:#5F2A7E;
}

.menu li {
	margin:0px;
	list-style:none;
}

.menu a {
	transition:all linear 0.1s;
	color:#FFFFFF;
}

.menu li:hover > a, .menu .current-item > a {
	text-decoration:none;
	color:#D4D4D4;
}

.menu .arrow {
	font-size:12px;
	line-height:0%;
}

/*----- Top Level -----*/
.menu > ul > li {
	float:left;
	display:inline-block;
	position:relative;
	font-size:19px;
}

.menu > ul > li > a {
	padding:9px 10px;
	display:inline-block;
	text-shadow:0px 1px 0px rgba(0,0,0,0.4);
}

.menu > ul > li:hover > a, .menu > ul > .current-item > a {
	background:#5F2A7E;
}
```

```
/*----- Bottom Level -----*/
.menu li:hover .sub-menu {
	z-index:1;
	opacity:1;
}

.sub-menu {
	width:160px;
	padding:0px 0px;
	position:absolute;
	top:100%;
	left:0px;
	z-index:-1;
	opacity:0;
	transition:opacity linear 0.15s;
	box-shadow:0px 2px 3px rgba(0,0,0,0.2);
	background:#5F2A7E;
}

.sub-menu li {
	display:block;
	font-size:16px;
}

.sub-menu li a {
	padding:10px 20px;
	display:block;
}

.sub-menu li a:hover, .sub-menu .current-item a {
	background:#5F2A7E;
}
```

Liite 6 Paikanhallintamatriisi koodina

```
<table border="0" cellspacing="10px" cellpadding="5px">
 <form action="users_posdb.php" method="POST">
  <?php
  for($r=0;$r<=$userrow;$r++) {
   if($r == '0') { ?>
   <th></th>
   <?php
   $tcadarray = array("empty");
    $namearray = array("empty");
    for($c=1;$c<=$userrow;$c++) {
     $user=$userhaku->fetch_assoc();
     array_push($tcadarray, $user['tcad']);
     array_push($namearray, $user['displayname']);?>
     <th><?php echo $c?></th>
    <?php } ?>
    <tr>
   <?php } else {
    for($c=0;$c<=$userrow;$c++){
     if($c == '0') { ?>
      <th width="75px"><?php echo $namearray[$r]?></th>
     <?php } else { ?>
      <td>
       <input type="radio" name="<?php echo $tcadarray[$r]?>" data-col="<?php
echo $c?>" value="<?php echo $c?>"
        <?php if($tcadarray[$r] == $tcadarray[$c]){
         echo checked;
        }?>>
      </td>
     <?php }
    } ?>
    <tr>
   <?php }
  }?>
 </table>
 <table border="0" cellspacing="10px" cellpadding="5px">
  <td width="85px"></td>
  <td>
   <input type="submit" name="paivita" value="Päivitä järjestys"></input>
   <input type="text" name="curlista" hidden value="<?php echo $current-
list?>"></input>
  </td>
 </form>
</table>
```

Liite 7 MySQL-injektiolta suojattu kirjautuminen

```
$inputtcad = $_POST['tcad'];
$inputpw = $_POST['passwd'];

if($stmt = $conn->prepare("select tcad, section, digest from EMPL where tcad=?")) {
  // bind  variable as string
  $stmt->bind_param("s", $inputtcad);

  // execute statement
  $stmt->execute();

  // save variables from query
  $stmt->bind_result($tcad, $currentlist, $digest);

  // fetch the data
  $stmt->fetch();

  // close connection
  $stmt->close();
}

// check password, if ok then make global variable user with user account
if(crypt($inputpw, $digest) == $digest) {
  $_SESSION["user"] = $tcad;
  header("refresh:0;url=shifts.php?list=$currentlist");
} else {
  header("refresh:0;url=index.php?error=1");
}
```

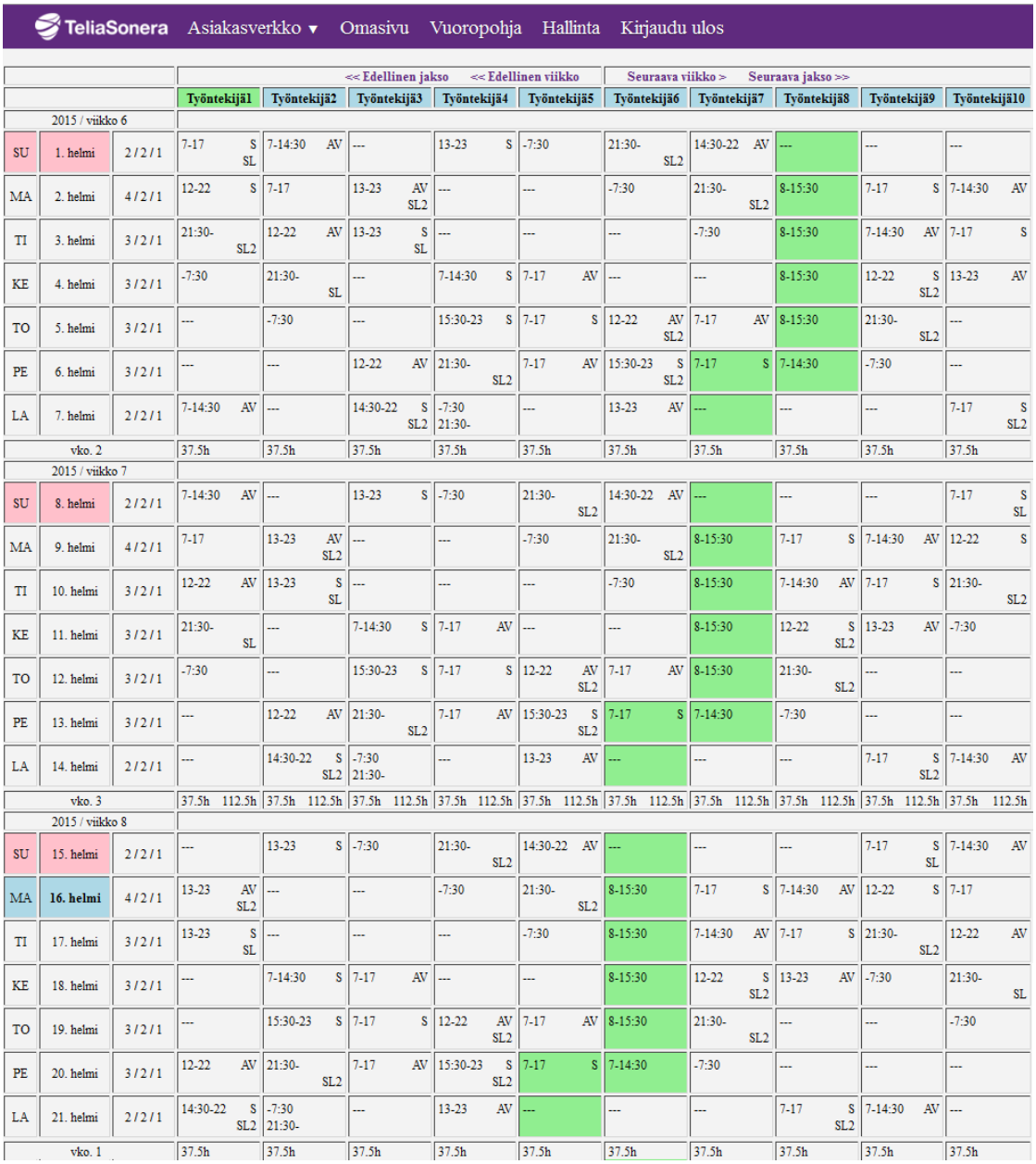
Liite 8 Käyttäjän salasanan vaihtaminen

```
$tcad = $conn->real_escape_string($_POST['tcad']);
$oldpw = $conn->real_escape_string($_POST['passwd']);
$newpw = $conn->real_escape_string($_POST['newpw']);
$newpw2 = $conn->real_escape_string($_POST['newpw2']);

$kayttajahaku = $conn->query("select tcad, digest from EMPL where tcad='$tcad'");
$kayttaja = $kayttajahaku->fetch_assoc();

if($kayttaja['tcad'] != NULL) {
 if(crypt($oldpw, $kayttaja['digest']) == $kayttaja['digest']) {
  if($newpw == $newpw2) {
   $hashpw =  password_hash($newpw, PASSWORD_BCRYPT);
   $sql = "UPDATE EMPL SET digest='$hashpw' where tcad='$tcad'";
   $conn->query($sql);
   echo "Salasana vaihdettu, sivu uudelleen ohjautuu 3 sek kuluttua...";
   header("refresh:3;url=index.php");
  } else {
   header("refresh:0;url=changepw.php?error=3");
  }
 } else {
  header("refresh:0;url=changepw.php?error=2");
 }
} else {
 header("refresh:0;url=changepw.php?error=1");
}
```

Liite 9 Arkipyhän merkitseminen tietokantaan

```
$ind = $conn->real_escape_string($_GET['ind']);
$t_holiday = $conn->real_escape_string($_POST['arkipyha']);
$t_comment = $conn->real_escape_string($_POST['comment']);
$t_poista = $conn->real_escape_string($_POST['poisto']);

$tietohaku = $conn->query("select * from DAYS where day = '$ind'");
$tietorow = $tietohaku->num_rows;

if($t_poista == 'on') {
  $sql = "delete from DAYS where day='$ind'";
} else {
  if($tietorow > '0') {
    $sql = "update DAYS SET day='$ind', comment='$t_comment', holiday='$_holiday'
where day='$ind'";
  } else {
    $sql = "insert into DAYS (day, comment, holiday) values ('$ind', '$t_comment',
'$t_holiday')";
  }
}

$conn->query($sql);
```

Liite 10 Vuorolistan esikatselutila

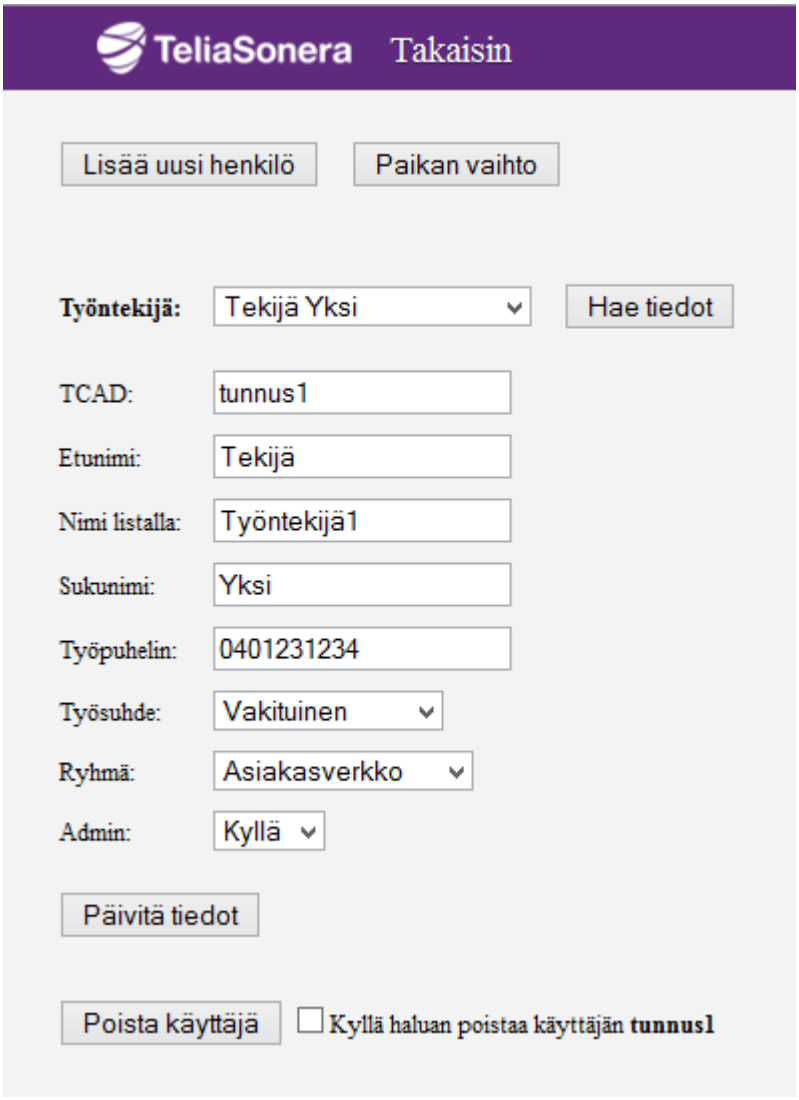


TeliaSonera Asiakasverkko ▾ Omasivu Vuoropohja Hallinta Kirjaudu ulos

<< Edellinen jakso << Edellinen viikko Seuraava viikko > Seuraava jakso >>

			Työntekijä1	Työntekijä2	Työntekijä3	Työntekijä4	Työntekijä5	Työntekijä6	Työntekijä7	Työntekijä8	Työntekijä9	Työntekijä10
2015 / viikko 6												
SU	1. helmi	2 / 2 / 1	7-17 S SL	7-14:30 AV	---	13-23 S	-7:30	21:30- SL2	14:30-22 AV	---	---	---
MA	2. helmi	4 / 2 / 1	12-22 S	7-17	13-23 AV SL2	---	---	-7:30	21:30- SL2	8-15:30	7-17 S	7-14:30 AV
TI	3. helmi	3 / 2 / 1	21:30- SL2	12-22 AV	13-23 S SL	---	---	---	-7:30	8-15:30	7-14:30 AV	7-17 S
KE	4. helmi	3 / 2 / 1	-7:30	21:30- SL	---	7-14:30 S	7-17 AV	---	---	8-15:30	12-22 S SL2	13-23 AV
TO	5. helmi	3 / 2 / 1	---	-7:30	---	15:30-23 S	7-17 S	12-22 AV SL2	7-17 AV	8-15:30	21:30- SL2	---
PE	6. helmi	3 / 2 / 1	---	---	12-22 AV	21:30- SL2	7-17 AV	15:30-23 S SL2	7-17 S	7-14:30	-7:30	---
LA	7. helmi	2 / 2 / 1	7-14:30 AV	---	14:30-22 S SL2	-7:30 21:30-	---	13-23 AV	---	---	---	7-17 S SL2
vko. 2			37.5h	37.5h	37.5h	37.5h	37.5h	37.5h	37.5h	37.5h	37.5h	37.5h
2015 / viikko 7												
SU	8. helmi	2 / 2 / 1	7-14:30 AV	---	13-23 S	-7:30	21:30- SL2	14:30-22 AV	---	---	---	7-17 S SL
MA	9. helmi	4 / 2 / 1	7-17	13-23 AV SL2	---	---	-7:30	21:30- SL2	8-15:30	7-17 S	7-14:30 AV	12-22 S
TI	10. helmi	3 / 2 / 1	12-22 AV	13-23 S SL	---	---	---	-7:30	8-15:30	7-14:30 AV	7-17 S	21:30- SL2
KE	11. helmi	3 / 2 / 1	21:30- SL	---	7-14:30 S	7-17 AV	---	---	8-15:30	12-22 S SL2	13-23 AV	-7:30
TO	12. helmi	3 / 2 / 1	-7:30	---	15:30-23 S	7-17 S	12-22 AV SL2	7-17 AV	8-15:30	21:30- SL2	---	---
PE	13. helmi	3 / 2 / 1	---	12-22 AV	21:30- SL2	7-17 AV	15:30-23 S SL2	7-17 S	7-14:30	-7:30	---	---
LA	14. helmi	2 / 2 / 1	---	14:30-22 S SL2	-7:30 21:30-	---	13-23 AV	---	---	---	7-17 S SL2	7-14:30 AV
vko. 3			37.5h 112.5h	37.5h 112.5h	37.5h 112.5h	37.5h 112.5h	37.5h 112.5h	37.5h 112.5h	37.5h 112.5h	37.5h 112.5h	37.5h 112.5h	37.5h 112.5h
2015 / viikko 8												
SU	15. helmi	2 / 2 / 1	---	13-23 S	-7:30	21:30- SL2	14:30-22 AV	---	---	---	7-17 S SL	7-14:30 AV
MA	**16. helmi**	4 / 2 / 1	13-23 AV SL2	---	---	-7:30	21:30- SL2	8-15:30	7-17 S	7-14:30 AV	12-22 S	7-17
TI	17. helmi	3 / 2 / 1	13-23 S SL	---	---	---	-7:30	8-15:30	7-14:30 AV	7-17 S	21:30- SL2	12-22 AV
KE	18. helmi	3 / 2 / 1	---	7-14:30 S	7-17 AV	---	---	8-15:30	12-22 S SL2	13-23 AV	-7:30	21:30- SL
TO	19. helmi	3 / 2 / 1	---	15:30-23 S	7-17 S	12-22 AV SL2	7-17 AV	8-15:30	21:30- SL2	---	---	-7:30
PE	20. helmi	3 / 2 / 1	12-22 AV	21:30- SL2	7-17 AV	15:30-23 S SL2	7-17 S	7-14:30	-7:30	---	---	---
LA	21. helmi	2 / 2 / 1	14:30-22 S SL2	-7:30 21:30-	---	13-23 AV	---	---	---	7-17 S SL2	7-14:30 AV	---
vko. 1			37.5h	37.5h	37.5h	37.5h	37.5h	37.5h	37.5h	37.5h	37.5h	37.5h

Liite 11 Työntekijän tietojen hallinta



Liite 12 Käyttäjien paikanhallintamatriisi