Nicola Nykopp

Libstat – Helsingin yliopiston kirjaston tilastopalvelin

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tietotekniikka

Insinöörityö

14.12.2015

Metropolia

Tekijä(t) Otsikko Sivumäärä Aika	Nicola Nykopp Libstat – Helsingin yliopiston kirjaston tilastopalvelin 43 sivua 14.12.2015
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Tietotekniikka
Suuntautumisvaihtoehto	Tietoverkot
Ohjaaja(t)	Yliopettaja Janne Salonen Kehittämispäällikkö Eeva Laurila

Helsingin yliopiston kirjaston toiminnasta kertyy tietoa moneen eri järjestelmään. Tämän työn tarkoitus on luoda tietokanta, johon tietoa voi tallentaa sekä käyttöliittymä, jolla tietoa voi tarkastella ja jota kautta tietoa voi syöttää. Lisäksi on tarkoitus luoda ohjelmat, jotka automaattisesti hakevat ja tallentavat (tai pelkästään tallentavat) tietoa järjestelmään.

Alustana toimii yliopiston tietotekniikkakeskuksen tarjoama RedHat-virtuaalipalvelin. Tietokantaohjelmana käytetään MariaDB:tä, ja käyttöliittymä on toteutettu PHP:llä (tietokannan abstrahointiin käytetään Doctrinea, ja esim. lomakkeen validoinnissa JavaScriptia). Tietoja hakevat ja tallentavat ohjelmat on toteutettu Javalla.

Työssä esitellään kolme tapaa viedä tietoja järjestelmään: manuaalinen, automaattinen ja puoliautomaattinen.

Manuaalisessa tavassa käyttöliittymässä on lomake, jonka avulla tiedot viedään tietokantaan. Esimerkkeinä manuaalisesta syöttötavasta esiteltiin käyttäjäkoulutuksen tilastointilomake sekä CTT (Count the Traffic)-lomake, jolla mitataan kirjaston eri tilojen käyttöasteita otantapäivinä.

Automaattisessa tavassa käynnistetään ajastetusti ohjelma, joka hakee tietoa ja tallentaa sen omaan kantaan. Esimerkkeinä tästä esiteltiin kirjastojärjestelmä, josta haetaan mm. painetun aineiston lainoihin, palautuksiin ja varauksiin liittyvät tiedot, MikroVäylän tietokanta, josta haetaan kävijäluvut sekä kirjaston oma aukiolotietokanta, josta haetaan aukiolopäivät ja aukiolotunnit.

Puoliautomaattisessa tavassa käsitellään tietoja, joita ei voi saada automaattisesti, mutta ne saadaan kuitenkin csv-muotoon, josta ohjelma lukee ne ja tallentaa tietokantaan. Esimerkkinä tästä esitettiin E-lehtipakettien käyttötilastot.

Järjestelmään jää vielä paljon laajentamisen varaa, ja monia mahdollisia tietolähteitä on vielä käyttämättä, erityisesti elektronisen aineiston käyttöön liittyviä.

Avainsanat	tilastot, kirjastot, MariaDB, PHP, Doctrine, Java

Author(s) Title Number of Pages Date	Nicola Nykopp Libstat – Statistics Server for the Helsinki University Library 43 pages 14 December 2015
Degree	Bachelor of Engineering
Degree Programme	Information and Communication Technology
Specialisation option	Network Engineering
Instructor(s)	Janne Salonen, Principal Lecturer Eeva Laurila, Development Director

The purpose of this thesis was to create a system, called Libstat, into which statistical data about the Helsinki University Library can be entered. The main objective was to create a simple user interface to provide access to the information stored in the system. The system was built for the Helsinki University Library.

The system was built on a Red Hat-server provided by the IT department at the University of Helsinki. MariaDB is used as a database, and the user interface uses PHP. For database abstraction Doctrine is used, and there is some JavaScript, for example for form validation. In addition, there are programs written in Java that retrieve information and store it in the database.

The thesis presents three ways of entering data into the system, namely manual, automatic and semi-automatic.

Manual data entry involves using an online form, from which data is entered and then stored in the database. As examples of this two forms are presented, one for entering information about user training and one for CTT, which is a way of measuring utilization levels of various parts of the library during sample days.

Automatic data entry involves programs that retrieve information from another system and store it in Libstat. There are three examples of automatic data entry: the library system, providing information about circulation, the database of MikroVäylä, providing visitor counters and the library's own opening hour database, providing the amount of opening days and hours.

Semi-automatic data entry involves information that cannot be obtained automatically, but it can be obtained in csv-form. A programs reads the csv-file and enters the data into the database. As an example of this there is usage statistics of electronic journal packages.

The system leaves much space for enlargement, and there are many potential sources of information that are still unused, especially concerning the utilization of electronic resources.

Keywords	libraries, statistics, MariaDB, PHP, Doctrine, Java

Sisällys

Lyhenteet

ORM	Object-relational mapping. Oliomallin mukaisen esityksen kuvaus relaatiomallin mukaiseksi esitykseksi.
CTT	Count the Traffic – kirjastoissa käytetty kävijälaskentamenetelmä
CSC	CSC – Tieteen tietotekniikan keskus Oy, valtion voittoa tavoittelematon osakeyhtiö

1 Johdanto

Helsingin yliopiston kirjasto tarjoaa kirjasto- ja tietopalveluja ensisijaisesti Helsingin yliopiston opiskelijoille ja henkilökunnalle, mutta kirjasto on avoin myös kaikille muille tiedontarvitsijoille. Vuonna 2014 kirjastossa asioi n. 2 miljoonaa kävijää, sieltä annettiin 475 000 lainaa, ladattiin 2,8 elektronista lehtiartikkelia ja 1,95 miljoonaa e-kirjaa (1). Yliopiston kirjastolla on toimipaikat neljällä kampuksella:

- Keskusta: humanistis-yhteiskuntatieteelliset alat
- Kumpula: luonnontieteet
- Meilahti: terveystieteet
- Viikki: biotieteet.

Viimeisen viidentoista vuoden ajan kirjastoalalla on yleistynyt ajattelutapa, jonka mukaan palveluiden kehittämistä koskevien päätösten pitäisi niin pitkälle kuin mahdollista perustua tutkittuun tietoon ja selvityksiin. Suuntaus tunnetaan nimellä *evidence-based librarianship* (3). Helsingin yliopiston kirjastossa tämä on näkynyt esim. laajoissa asiakaskyselyissä, joista ensimmäinen toteutettiin vuonna 2004 (2, s. 2).

Helsingin yliopiston kirjaston toiminnasta kertyy tietoa moneen eri järjestelmään. Toiminnanohjauksen kannalta olisi hyödyllistä, jos tiedot saisi koottua yhteen paikkaan, jossa ne olisivat suunnittelijoiden ja johtajien käytettävissä aina, kun niitä tarvitaan. Kirjaston pitää myös vuosittain ilmoittaa tietyt tiedot Tieteellisten kirjastojen tilastotietokantaan (https://yhteistilasto.lib.helsinki.fi/), ja myös näiden tietojen kerääminen helpottuisi, jos tiedot saisi kootusti yhdestä paikasta.

Tämän työn tavoitteena on luoda järjestelmä, johon voidaan tallentaa koostetietoa kirjaston toiminnasta eri tavoilla ja jossa tiedot ovat tarkasteltavissa selaimella. Tietojen tuominen voi olla

- manuaalista: tiedot syötetään selaimen kautta suoraan omaan tietokantaan. Esimerkiksi käyttäjäkoulutusten tiedot voidaan tallentaa näin.

- automaattista: ohjelma hakee toisesta järjestelmästä tietoa ja tallentaa sen omaan tietokantaan ajastetusti. Esimerkiksi tiedot painetun aineiston lainoista voi saada näin kirjastojärjestelmästä.

- puoliautomaattista: muista järjestelmistä otetaan raportteja, jotka tallennetaan csv-muodossa omalle palvelimelle. Sen jälkeen suoritetaan ohjelma, joka siirtää tiedot omaan tietokantaan. Esimerkiksi e-aineistojen käyttötilastot ovat saatavissa ainoastaan eri kustantajien omista portaaleista, joihin ei ole ohjelmallisia rajapintoja, mutta tiedot voidaan kerätä csv-tiedostoon ja siirtää ohjelmalla tietokantaan.

Tavoitteena on, että työn aikana toteutetaan 1-2 tapausta kustakin tallennustavasta sekä luodaan käyttöliittymä, jossa tietoja voi tarkastella. Järjestelmää on tarkoitus tämän työn valmistuttua edelleen laajentaa.

Ohjelmointityössä olen jatkuvasti käyttänyt apuna erityisesti seuraavia dokumentaatioita, vaikka en niihin suoraan viittaakaan:

- PHP dokumentaatio. http://php.net/manual/en/index.php (luettu 10.4.2015).
- Java 7 API. http://docs.oracle.com/javase/7/docs/api/ (luettu 10.4.2015).
- Java 6 API. http://docs.oracle.com/javase/6/docs/api/ (luettu 10.4.2015).
- Joda-Time dokumentaatio. http://www.joda.org/joda-time/ (luettu 10.4.2015).

2 Palvelin ja ohjelmistot

2.1 Palvelinalusta

Helsingin yliopiston tietotekniikkakeskuksen yliopiston yksiköille tarjoamilla Linux-virtuaalipalvelimilla yksiköt voivat toteuttaa omia palvelujaan. Tätä työtä varten sain käyttööni virtuaalipalvelimen, jossa on käyttöjärjestelmä Red Hat Enterprise Linux Server 7.1 (Maipo). Palvelimelle oli valmiiksi asennettu Apache (2.4.6) ja PHP (5.4.16). Asensin näiden lisäksi seuraavat ohjelmat:

- MariaDB (5.5.41)
- phpMyAdmin (4.4.9)

- Composer (1.0)
- Java (1.7.0, eli Java 7).

Palvelimen hostname on libstat-0.hulib.helsinki.fi, alias libstat.hulib.helsinki.fi.

Tietotekniikkaosasto vastaa käyttäjätunnuksista ja verkkoyhteyksistä, mutta käyttäjä voi asentaa ja konfiguroida ohjelmia vapaasti. Lähtökohtaisesti tilaajalla (tässä tapauksessa minulla) on täydet oikeudet palvelimelle, mutta ssh-kirjautuminen onnistuu suoraan vain yliopiston sisältä. Yliopiston ulkopuolelta voi kirjautua niin, että ottaa ensin yhteyttä johonkin yliopiston eteisverkossa olevaan palvelimeen (esim. cedi.it.helsinki.fi) ja sieltä edelleen Libstatille. Käytännössä tiedostojen siirto on siis tehtävä kahdessa vaiheessa, jos ottaa yhteyttä yliopiston ulkopuolelta: ensin sftp:llä edustapalvelimelle ja sieltä edelleen scp:llä.

Pyysin tietotekniikkaosastolta http(s)-porttien (80, 443) avaamista ensin yliopiston sisällä (tammikuussa 2015) ja myöhemmin (huhtikuussa 2015) myös ulkomaailmaan. Samoin huhtikuussa tietotekniikkaosasto teki reiän palomuuriin kirjastojärjestelmän palvelimelta tuleville yhteyksille.

2.2 Tietokannan hallintajärjestelmä

MariaDB:n oletusasetukset ovat enimmäkseen hyvin toimivia. Tietokantamoottorina on InnoDB, joka osaa huolehtia eheysrajoitteiden noudattamisesta. Ainoa perusasia, jota oli muutettava, oli merkistöasetukset. Vaikka UTF-8 on nopeasti yleistynyt, MariaDB:ssä oletusmerkistö on edelleen Latin1. MariaDB:ssä ei ole myöskään yhtä asetusta merkistölle, vaan sen voi määritellä vaikka kenttäkohtaisesti ja monella muullakin tasolla. Stephen Balukoff on blogikirjoituksessa (4) esitellyt hyvin millaisia solmuja ristiriitaisilla asetuksilla saa aikaan. Muutin kaikilla tasoilla oletusmerkistöksi UTF-8:n.

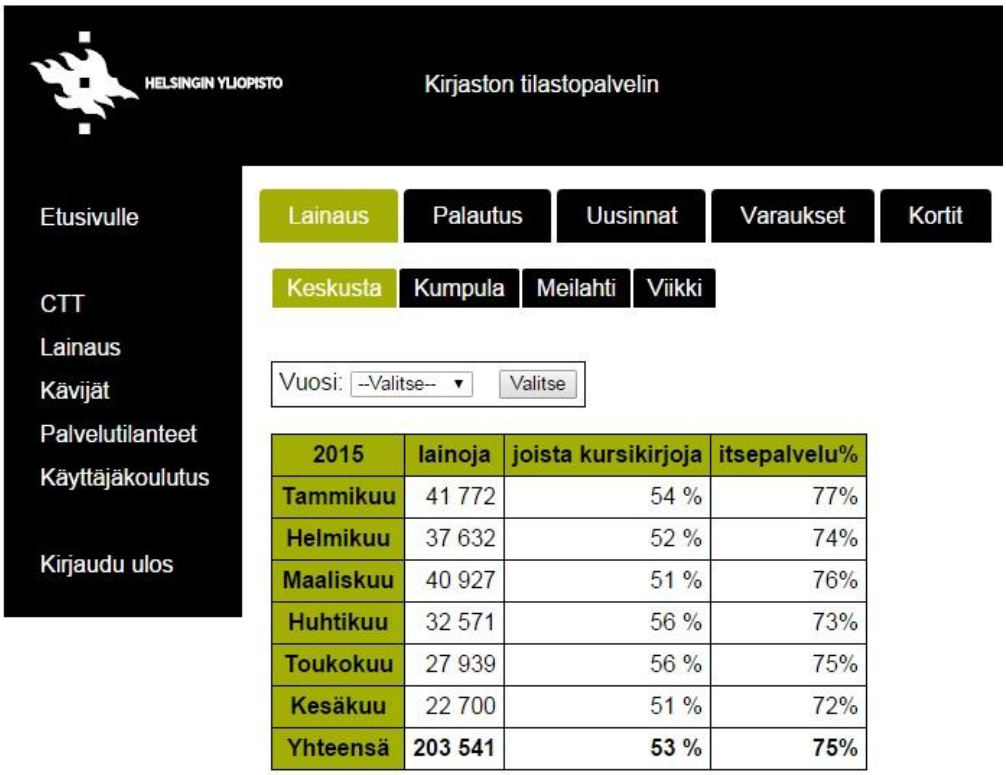
2.3 Käyttöliittymä

Käyttöliittymä on toteutettu PHP:lla ja JavaScriptilla. JavaScriptia tarvitaan lähinnä lomakesyötteiden validointiin, kun taas koostetietojen esitteleminen onnistuu hyvin pelkällä PHP:llä.

Käyttöliittymässä on käytetty yliopiston logoa sekä yliopiston visuaalisen ilmeen mukaista kirjaston tehosteväriä (yliopiston tiedekunnille ja erillislaitoksille on määrätty omat tehostevärit, joita voi käyttää esittelymateriaaleissa ja verkkosivuilla).

Vasemmassa reunassa on pystysuora päänavigaatio mustalla pohjalla, ja sen lisäksi on kulloinkin valitun osion mukainen vaakasuora navigaatio, jossa valittu kohta on korostettu kirjaston tehosteväriällä. Vaakasuoraa navigaatiota voi olla useita tasoja (Kuva 1).

Tyylit määritellään erillisessä css-tiedostossa.



2015	lainoja	joista kursikirjoja	itsepalvelu%
Tammikuu	41 772	54 %	77%
Helmikuu	37 632	52 %	74%
Maaliskuu	40 927	51 %	76%
Huhtikuu	32 571	56 %	73%
Toukokuu	27 939	56 %	75%
Kesäkuu	22 700	51 %	72%
Yhteensä	**203 541**	**53 %**	**75%**

Kuva 1. Näkymä käyttliittymän kohdasta lainaus, valittuna Keskustan lainaus vuonna 2015 (kuvaa).

Jos näytettävien tietojen hakeminen tietokannasta tehdään samassa tiedostossa, joka myös näyttää tiedot liittymässä, tiedostosta tulee pitkä ja vaikealukuinen. Yksi tapa siirtää varsinaiset SQL-kyselyt muualle on ottaa käyttöön PHP:n kanssa toimiva Doctrine, mikä tarjoaa olio-relaatiomuunnoksen sekä käteviä funktioita tiedon tallentamiseen ja hakemiseen tietokannasta.

Aikaisemmin asennetun Composer-ohjelman avulla ilmoitetaan composer.json-tiedostossa, että halutaan käyttää Doctrine ORM:in versiota 2.4 ja että annotoidut luokat, joilla mapataan tietokannan tauluja olioiksi, sijaitsee kansiossa *entities*.

```
{
    "require": {
        "doctrine/orm": "2.4.*"
    },
    "autoload": {
        "psr-0": {"": "entities/"}
    }
}
```

Tiedostossa *bootstrap.php* luodaan tietokantayhteys ja instanssi EntityManagerista, joka huolehtii tietojen hakemisesta ja tallentamisesta. Tietokantayhteyden ja EntityManagerin tarjoamat palvelut saa käyttöön *require_once()*-funktiolla. Parametriksi annetaan '*bootstrap.php'*.

Jokaista tietokantataulua kohden, jota haluaa käsitellä Doctrine ORM:illa, luodaan PHP-luokka, jossa annotaatiot kertovat, mitkä jäsenet vastaavat mitä kenttiä. Kaikille jäsenille määritellään getterit ja setterit. Puhtaille välitauluille, joissa ei ole muita tietoja, ei tarvitse määritellä omaa luokkaa. Jokaisella näin määritellyllä luokalla on oma EntityRepository, jonka saa käyttöön pyytämällä EntityManagerilta. Oletuksena tarjotun EntityRepositoryn kautta voi noutaa dataa tietokannasta oliomuotoon, mutta jos haluaa käyttää kyselyssä esim. funktioita, on määriteltävä oma EntityRepository-luokasta periytyvä luokka, joka nimetään kuvailuluokan annotaatiossa. Omaan luokkaan voi sitten määritellä funktioita, joilla haetaan tietoja tietokannasta.

Doctrine ORM mahdollistaa kyselyiden tekemisen monilla eri tavoilla:

- EntityRepository-luokan *find(), findBy(), findAll()* ja *findOneBy()*-funktioilla voi hakea oliomuotoon useita rivejä tai yhden rivin tietokannasta. Hakuehdoissa ei voi käyttää <, > tai funktioita kuten COUNT() tai SUM(), mutta esim. hakutuloksen lajittelu haluttujen kenttien mukaan onnistuu. On myös mahdollista määritellä useita hakuehtoja (sekä AND että OR-tyyppiset yhdistämiset mahdollisia, vaikka näitä avainsanoja ei käytetä), ja hakutulosten maksimimäärä voidaan määritellä. Tulos on aina olio tai oliotaulukko.

- Itse määritellyssä EntityRepository-luokassa voi rakentaa kyselyitä QueryBuilder-oliolla. Kyselyä ei kirjoiteta yhdeksi merkkijonoksi, vaan QueryBuilder-olio kutsuu funktioita kuten *where()* ja *from()*, jossa määritellään haluttu kysely. Kyselykieli ei ole SQL, vaan DQL (olioita käsittelevä kyselykieli, joka muistuttaa Hibernaten HQL-kyselykieltä). Käytössä ovat funktiot MIN(), MAX(), COUNT() ja SUM(), mutta valitettavasti ei esim. MariaDB:n tarjoamat YEAR()- ja GROUP_CONCAT() -funktioita. Tulos voidaan saada joko oliomuodossa tai taulukkona (jossa kentän nimi on avain ja arvo kyseisen kentän arvo).

- Itse määritellyssä EntityRepository-luokassa voi rakentaa DQL-kyselyn myös merkkijonona, mutta tätä ei suositella. QueryBuilder-oliota käyttämällä kyselyn rakentaminen on rakenteista, ja varsinkin pitkät ja monimutkaiset kyselyt jäsentyvät selkeämmin QueryBuilderin avulla.

- Itse määritellyssä EntityRepository-luokassa voi tehdä kyselyitä myös käytössä olevan tietokantaohjelman (tässä tapauksessa MariaDB) omalla SQL-murteella. Silloin pitää ensin määritellä ResultSet-olio, jossa kerrotaan mitä sarakkeita tuloksessa on ja millä nimellä niitä haluaa tuloksessa käsitellä. Kyselyssä voi luonnollisesti käyttää kaikkia kyseisen SQL-murteen funktioita (ml. edellä mainitut YEAR() ja GROUP_CONCAT()).

Uusien tietojen vieminen tietokantaan (tai olemassa olevien tietueiden päivitys) tapahtuu EntityManager-olion funktioilla *persist()* (parametrina annetaan tallennettava olio, ja EntityManager laittaa lisäyksen/päivityksen työlistalle) ja *flush()* (vasta nyt EntityManager suorittaa kaikki työlistalla olleet tehtävät ja tietokanta päivittyy).

Käytössä olen todennut, että Doctrine on erinomainen apu, kun haluaa tallentaa tietueita kantaan tai kun haluaa päivittää olemassa olevia tietueita. Myös transaktioiden hallinta on Doctrinessa helppoa.

Suuri osa tästä työstä on kuitenkin koostetietojen esittämistä, ja kyselyissä käytetään yleensä COUNT()- tai SUM()-funktioita. Tämän tyyppisissä kyselyissä Doctrinesta on vähemmän apua, sillä tulosten käsitteleminen olioina ei ole järkevää – käytännössä halutaan usein yksi tai useampi kokonaisluku tulokseksi. Joidenkin MariaDB:ssä tarjolla olevien funktioiden puuttuminen johti siihen, että usein käytin suoraan SQL:ää.

Kokonaisuutena katsottuna on kyseenalaista, saiko Doctrinesta niin paljon hyötyä, että sitä kannatti käyttää, mutta toisaalta, vaikka olisi käyttänyt suoraan PHP:n PDO-luokkaa tietokantayhteyden luomiseen ja kyselyiden rakentamiseen, toiminta olisi joka tapauksessa pitänyt siirtää pois sivuilta, joilla tiedot esitetään. Doctrine ORM tarjosi tähän valmiin mallin.

Doctrine ORMin toiminnasta kerrotaan projektin dokumentaatiossa (5) ja API:ssa (6).

2.4 Tietoturva

Helsingin yliopiston tietotekniikkakeskus vastaa palvelimen tietoliikenneyhteyksistä ja niiden turvallisuudesta. Ssh-yhteydet sallitaan vain yliopiston verkon sisältä, ellei erikseen tehdä reikää palomuuriin tietylle IP-osoitteelle.

Tietokantaan ei tallenneta mitään tietoja, millä voisi yksilöidä kirjaston asiakkaita, eikä muutakaan salassa pidettävää tietoa. Koska tietokanta on kuitenkin tarkoitettu ensisijaisesti kirjaston sisäiseen käyttöön, käyttöliittymään kirjaudutaan yhteissalasanalla (määritelty suoraan PHP-koodissa). Koska salasana kulkee selväkielisenä, sitä ei olisi vaikea saada haltuunsa, mutta tietojen ”vuotaminen” ei aiheuttaisi vahinkoa. Tietokannan tiedoista otetaan viikottain kopio *mysqldump*-ohjelmalla. Käyttöliittymän kautta tulevat syötteet validoidaan ennen kuin tietokantaa päivitetään, ja kaikissa päivityksissä käytetään esivalmisteltuja kyselyitä tai Doctrinea.

Käytännössä siis tietokanta ja palvelin on varsin hyvin suojattu (ssh-kirjautuminen sallittu vain yliopiston verkosta ja muutamalle käyttäjätunnukselle), kun taas verkkosivuille on melko helppo päästä. Mahdollinen vahinko kuitenkin rajoittuu mahdollisuuteen syöttää ilkivaltaisesti tietoja tietokantaan siltä osin kuin tietoja kerätään suoraan käyttöliittymässä. Esim. SQL-injektio ei onnistu mm. siksi, että kaikki syötteet validoidaan ja siivotaan mahdollisesta koodista.

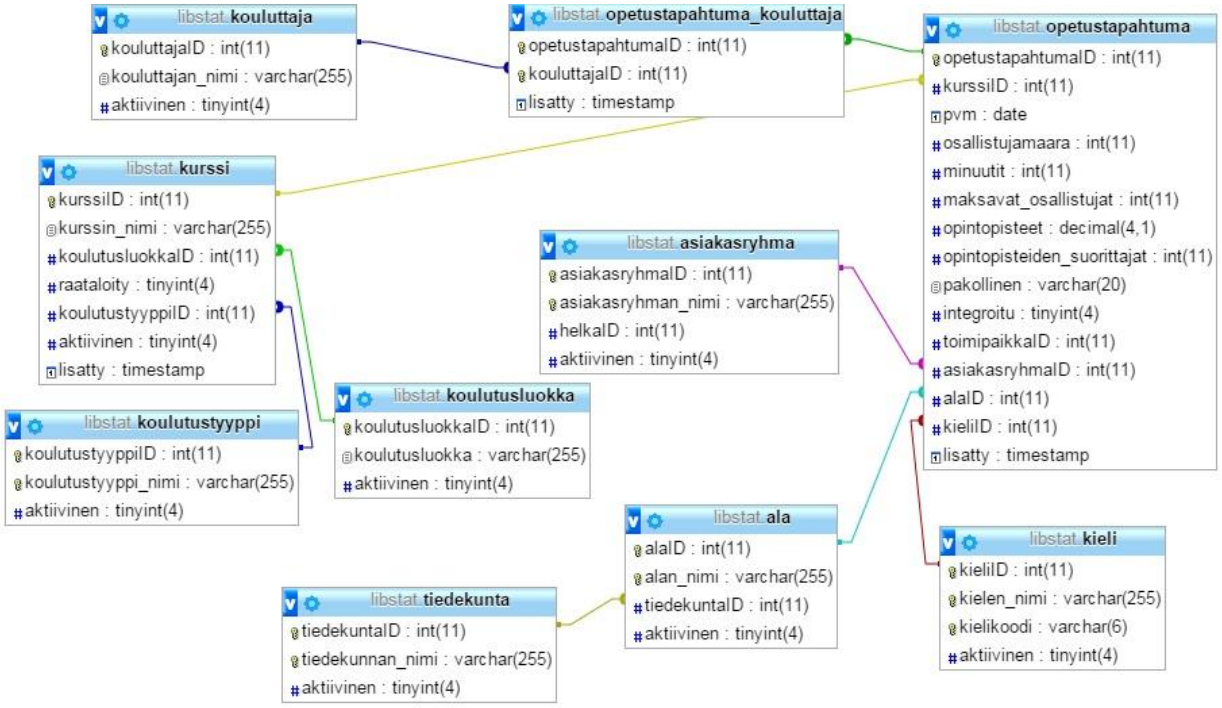
3 Manuaalinen tiedon syöttäminen

3.1 Käyttäjäkoulutuksen tilastoiminen

Helsingin yliopiston kirjasto tarjoaa käyttäjäkoulutusta neljällä kampuksella ja verkossa. Tiedot pidetyistä koulutuksista on aikaisemmin kerätty eri tavoilla eri kampuksilla. Käytössä on ollut ainakin Excel ja yliopiston tarjoama e-lomakepalvelu (https://elomake.helsinki.fi). Selvitin ensin, mitä tietoja on kerätty ja keskustelin käyttäjäkoulutuksesta vastaavien kanssa. Tämän perusteella tein alustavan taulurakenteen ja lomakkeen, jota kokeiltiin pari kuukautta. Saadun palautteen perusteella tein vielä joitain korjauksia lomakkeeseen ja taulurakenteeseen.

3.1.1 Tietokannan taulut

Tietokannan taulurakenteessa (kuva 2) on taulu opetustapahtumalle, joka sisältää vierasavaimet kurssille, asiakasryhmälle, (asiakasryhmän edustamalle) alalle, kielelle ja toimipaikalle. Kurssi-taulussa määritellään koulutustyyppi (esim. lähiopetus tietokoneella, luento) ja luokitus sen mukaan, mihin opintojen vaiheeseen kurssi liittyy. Kenttä ”raataloity” vastaa lomakkeen kohtaa kertaluonteinen, ja kenttä aktiivinen mahdollistaa vanhojen kurssien pudottamisen lomakkeen pudotusvalikosta (lomakkeella näytetään vain aktiiviset kurssit). Vastaavat aktiivinen-kentät löytyvät myös asiakasryhmä-, ala- ja toimipaikka-tauluista siltä varalta, että näitä halutaan myöhemmin kirjata eri tavalla. Kouluttajan ja opetustapahtuman välillä on monen suhde moneen, joten niitä yhdistää välitaulu. Kieli, toimipaikka- ja tiedekunta-tauluja käytetään myös muiden kuin koulutustietojen yhteydessä.

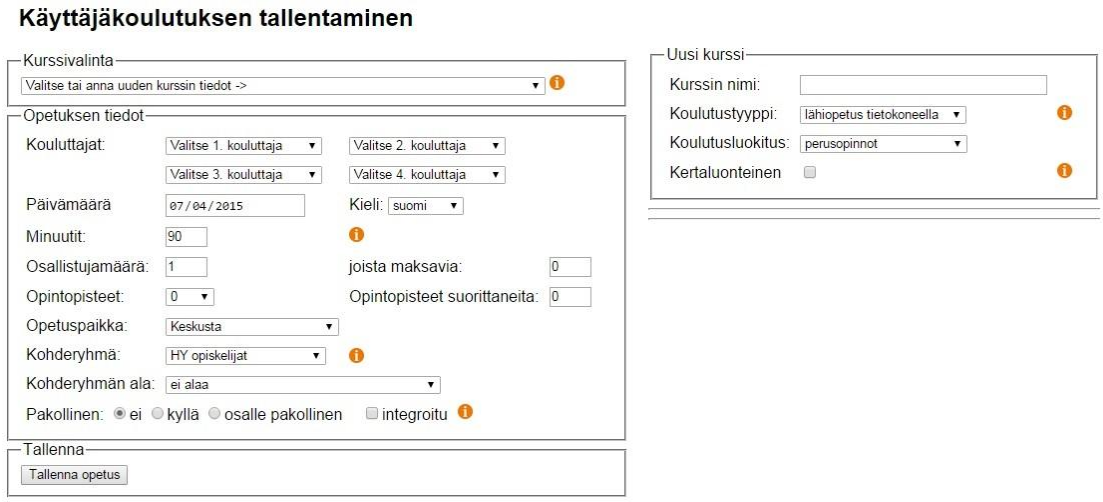


Kuva 2. Käyttäjäkoulutuksiin liittyvät taulut.

3.1.2 Syöttölomake

Lomakkeella (ks. kuva 2) kouluttaja voi joko täyttää uuden kurssin tiedot oikealla tai valita olemassa olevan kurssin pudotusvalikosta. Osuus ”Opetuksen tiedot” on aina täytettävä. Jos uuden kurssin kohdalla ruksaa ”Kertaluonteinen” kurssi ei tule näkyviin vasemman puolen pudotusvalikkoon. Alkuperäisessä määrittelyssä kieli, minuutit ja opintopisteet olivat kurssin ominaisuuksia, mutta ne haluttiin siirtää opetustapahtuman puolelle, jotta pudotusvalikon kurssilista ei paisuisi kovin pitkäksi.

Lomakkeen validointi (JavaScript) tarkistaa, että lomakkeella on joko valittu olemassa oleva kurssi tai syötetty uusi, että on valittu vähintään yksi kouluttaja, että on annettu päivämäärä muodossa VVVV-KK-PP ja että on vähintään yksi osallistuja. PHP-koodissa tarkistetaan vielä uudestaan, että on valittu kurssi tai syötetty uusi ja että päivämäärä on oikein. Lisäksi uuden kurssin nimestä siivotaan mahdollinen koodi *filter_input()*-funktiolla.



Kuva 3. Lomake, jolla syötetään käyttäjäkoulutus.

Tietojen tallennus tapahtuu seuraavasti:

- Jos on syötetty uusi kurssi, se tallennetaan ja haetaan uusimman tallennuksen ID yhdessä transaktiossa.

- Tallennetaan opetustapahtuma (käyttäen edellä saatua ID:tä jos kyse on uudesta kurssista) ja haetaan uusimman tallennuksen ID yhdessä transaktiossa.

- Käytetään edellä saatua ID:tä ja lisätään jokaisen valitun kouluttajan kohdalle yhteys opetuskerran ja kouluttajan välille.

Tallennukset tehdään oliopohjaisesti Doctrinen kautta, eli luodaan esim. uusi Kurssi-luokan olio, asetetaan settereiden avulla kenttien arvot ja tallennetaan (*persist(), flush()*).

Käyttöliittymässä on myös lomake, jolla voi ilmoittaa uuden kouluttajan tai deaktivoida kouluttaja Työtehtävät saattavat muuttua tai työsuhde päättyä. Deaktivoitu kouluttaja ei näy lomakkeen pudotusvalikoissa.

3.1.3 Tietojen esittäminen

Tallennetuista opetuksista on aina näkyvissä ajantasainen kooste (ks. kuva 4).

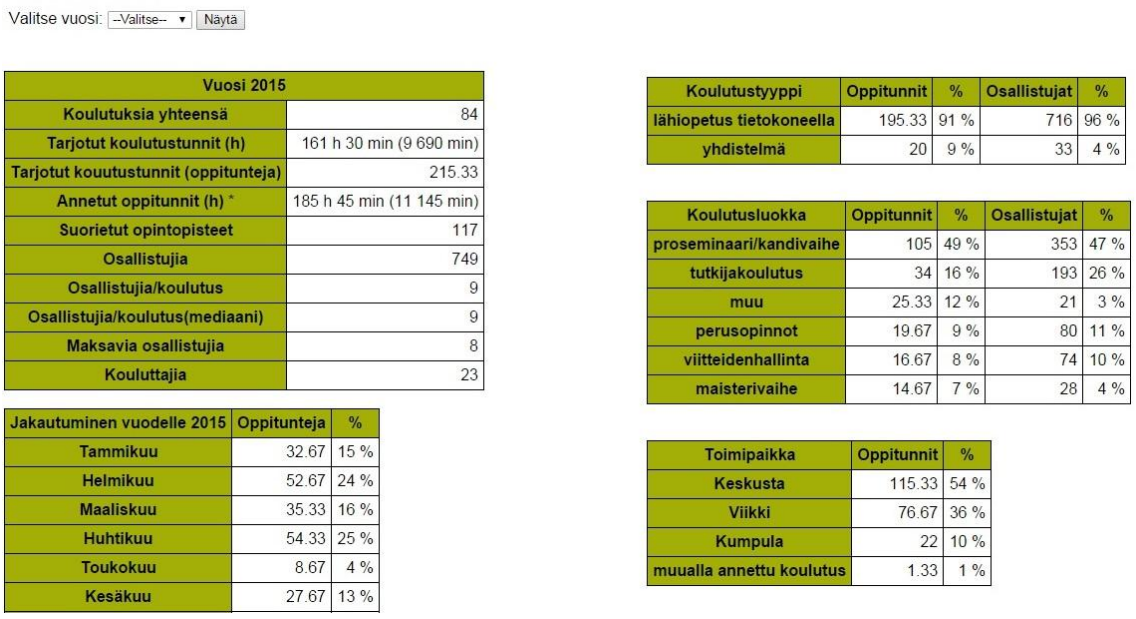
Tämän lisäksi on mahdollista ladata suoraan csv-muotoon kaikki valitun toimipaikan koulutustiedot. Yliopiston Windows-työasemilla csv-tiedostot avataan oletusarvoisesti Excelillä, joka ei osaa automaattisesti tunnistaa csv-tiedoston merkistöksi UTF-8. Excelille voi kuitenkin eksplisiittisesti kertoa, että kysymys on UTF-8-koodatusta tekstistä kirjoittamalla tiedostoon alkuun UTF-8:aa edustava BOM-koodi (Byte Order Mark, https://en.wikipedia.org/wiki/Byte_order_mark#Representations_of_byte_order_marks_by_encoding), joka on heksadesimaaleissa EF BB BF. PHP:ssa:

```
fwrite($output, chr(0xEF) . chr(0xBB) . chr(0xBF));,
```

jossa $output on tiedostokahva (file pointer). Näin ääkköset ja erikoismerkit näkyvät Excelissäkin oikein.

Tiedot hakevassa kyselyssä käytetään GROUP_CONCAT-funktiota niin, että saa samalle riville yhteen koulutustapahtumaan liittyvät useat kouluttajat:

```
SELECT …, GROUP_CONCAT(DISTINCT kouluttajan_nimi SEPARATOR ' & '
) AS kouluttajat, …
```



Valitse vuosi: --Valitse-- Näytä

Vuosi 2015	
Koulutuksia yhteensä	84
Tarjotut koulutustunnit (h)	161 h 30 min (9 690 min)
Tarjotut kouutustunnit (oppitunteja)	215.33
Annetut oppitunnit (h) *	185 h 45 min (11 145 min)
Suorietut opintopisteet	117
Osallistujia	749
Osallistujia/koulutus	9
Osallistujia/koulutus(mediaani)	9
Maksavia osallistujia	8
Kouluttajia	23

Jakautuminen vuodelle 2015	Oppitunteja	%
Tammikuu	32.67	15 %
Helmikuu	52.67	24 %
Maaliskuu	35.33	16 %
Huhtikuu	54.33	25 %
Toukokuu	8.67	4 %
Kesäkuu	27.67	13 %

Koulutustyyppi	Oppitunnit	%	Osallistujat	%
lähiopetus tietokoneella	195.33	91 %	716	96 %
yhdistelmä	20	9 %	33	4 %

Koulutusluokka	Oppitunnit	%	Osallistujat	%
proseminaari/kandivaihe	105	49 %	353	47 %
tutkijakoulutus	34	16 %	193	26 %
muu	25.33	12 %	21	3 %
perusopinnot	19.67	9 %	80	11 %
viitteidenhallinta	16.67	8 %	74	10 %
maisterivaihe	14.67	7 %	28	4 %

Toimipaikka	Oppitunnit	%
Keskusta	115.33	54 %
Viikki	76.67	36 %
Kumpula	22	10 %
muualla annettu koulutus	1.33	1 %

Kuva 4. Osa sivusta, jolla tilastot esitetään.

3.2 CTT

CTT (Count the Traffic) on Tanskassa kehitetty menetelmä kirjastojen kävijälaskentaan, jossa kirjastotila jaetaan eri vyöhykkeisiin ja lasketaan käyttäjät vyöhykkeillä niin, että huomioidaan myös, mitä käyttäjät alueella tekevät. Tarkoitus on kerätä systemaattista dataa siitä, mitä asiakkaat kirjastossa tekevät ja eri tilojen käyttöasteista (7). Helsingin yliopiston kirjasto on ottanut käyttöön tanskalaisesta menetelmästä yksinkertaistetun version jossa lasketaan vain työskentelypaikoilla olevat asiakkaat (mutta ei esim. hyllyjen väleissä tai liikkeellä olevat), ja kirjataan vain, työskenteleekö asiakas laitteella vai ilman. Kaksi kertaa vuodessa viikon pituisen jakson aikana lasketaan kävijät kolme kertaa päivässä (viikonloppuisin kerran). Näin saadaan tiedot alueiden käyttöasteista ja karkealla tasolla käyttötavasta (laite / ilman). Aikaisemmin tiedot on ensin syötetty paperille ja sen jälkeen edelleen yliopiston tarjoamalla e-lomakepalvelulla luodulle lomakkeelle, josta sen on voinut saada esim. Exceliin.

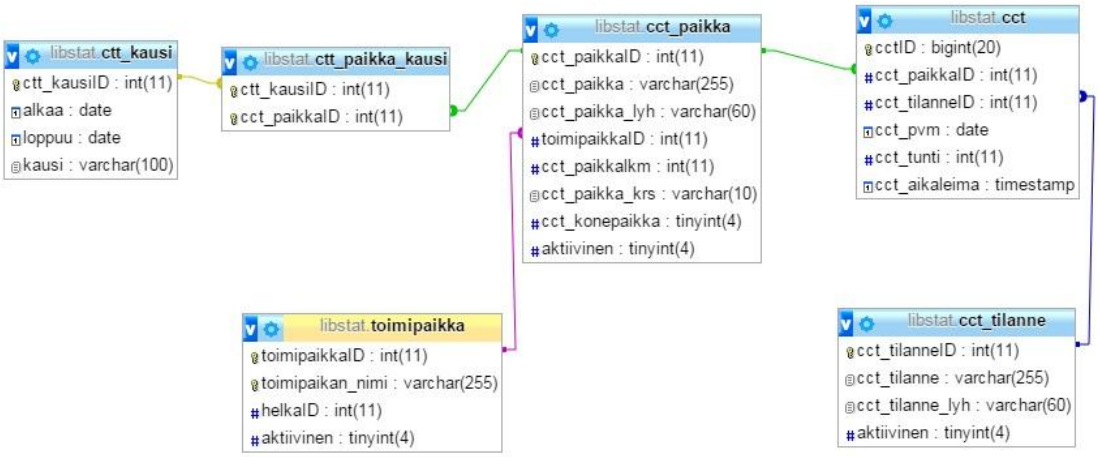
3.2.1 Tietokannan taulut

Tietokannassa suurin osa CTT-laskentaan liittyvistä tauluista ja kentistä on nimetty virheellisesti cct-alkuiseksi. Kun huomasin tämän, olin jo käyttänyt nimiä niin monessa

paikassa, että sitä ei kannattanut enää muuttaa. Taulujen nimet eivät näy käyttäjälle, ja esim. kaikissa verkkosivujen osoitteissa näkyy oikea muoto ctt.

Taulussa cct yksi rivi edustaa yhtä havaintoa. Taulussa on vierasavain, joka viittaa tauluun cct_tilanne, johon voi määritellä tilastoitavia tilanteita – nykyisellään lasketaan vain työskentely koneella ja työskentely ilman, mutta jos halutaan tulevissa laskennoissa huomioida muitakin tilanteita, se on mahdollista. Cct-taulu sisältää myös vierasavaimen tauluun cct_paikka, joka edustaa vyöhykettä, jossa kävijöitä lasketaan. Vyöhykkeestä kirjataan paikkojen lukumäärä (jotta saadaan laskettua käyttöaste), kerros (jotta voidaan laskea kerroskohtaiset käyttöasteet) ja tieto siitä, onko kyseessä työasemalla varustetuita paikoista vai ei (jos paikoilla on työasemat, tallennuslomakkeella ei tarjota vaihtoehtoa ”työskentelee ilman laitetta”). Cct-paikka-taulussa on vierasavaimena viittaus toimipaikka-tauluun.

Taulu ctt_kausi sisältää tiedon millä aikavälillä mittauksia on tehty, ja tällä taululla on monen suhde moneen cct_paikka-tauluun. Täytyy tietää, mitkä alueet ovat olleet mukana missäkin mittauksessa – se, että joltain alueelta ei ole yhtään kirjausta voi tarkoittaa joko sitä, että alue ei ollut mukana mittauksessa, tai sitä, että siellä ei havaittu yhtään asiakasta. Koska kirjastojen tiloissa tapahtuu jatkuvasti suuria ja pieniä muutoksia, on tärkeää, että tietokannan rakenne on riittävän joustava heijastamaan tätä.



Kuva 5. CTT-laskentaan liittyvät taulut.

3.2.2 Syöttölomakkeet

CTT-syöttölomakkeilla (jokaisella toimipaikalla on oma) on joukko painikkeita, joista jokainen edustaa yhtä paikkaa ja tilannetta (ks. kuva 6, jossa on esimerkkinä Viikissä käytetty lomake). Tilastoiminen tapahtuu painikkeita painamalla, ja valittu havainto tallentuu saman tien tietokantaan. Napin vieressä näkyvä laskuri näyttää tallennusten määrän, niin että tallentaja näkee, että tallennukset onnistuvat. Laskurin arvo tallennetaan sessiomuuttujaan. Esim. ”1. krs kiinteät” viittaa paikkoihin, joissa on työasemat, ja siksi tästä alueesta ei tarjota vaihtoehtoa ”ilman”. Muuten paikoista on aina kaksi tapaa tilastoida, ”laitteella” tai ”ilman”. Keskustassa, jossa kirjastotiloja on yhdeksässä kerroksessa, kaikki painikkeet eivät mahdu yhdelle sivulle.

Lomaketta on tarkoitus käyttää kirjaston 10” tablettilaitteilla (iPad ja Samsung), jotka on yhdistetty yliopiston langattomaan verkkoon. Koska tilastoitava tieto tallentuu heti tietokantaan, verkkoyhteyden pitää toimia koko ajan. Vaikka kirjaston tiloissa pitäisi olla kaikkialla yliopiston langaton verkko, erityisesti kellarikerroksissa esiintyi ongelmia tilastoinnin yhteydessä. Luotettavamman verkkoyhteyden sai käyttämällä matkapuhelimesta jaettua yhteyttä. Lomaketta voisi kehittää niin, että se ei enää tallenna jokaista havaintoa heti, vaan tallentaa ensin tiedot paikallisesti ja vasta erillisen tallenna-painikkeen kautta vie tiedot palvelimelle ja tietokantaan.

Tietojen hakeminen on toteutettu Doctrinen avulla. Tässä tapauksessa tallennuksen yhteydessä ei tarvita transaktioita. Jokainen painallus vain lisää rivin cct-tauluun.

Kuva 6. Viikin toimipaikassa käytetty lomake.

Saatujen kokemusten perusteella lomaketta kannattaisi ehkä kehittää niin, että painallus ei suoraan tallentaisi tietoja tietokantaan, vaan ensin paikallisesti evästeisiin, ja vasta erillisen ”Tallenna”-napin kautta palvelimelle. Silloin verkko-ongelmat eivät haittaisi niin paljon, ja olisi myös mahdollisuus korjata virheelliset painallukset ennen kuin havainto tallennetaan tietokantaan.

3.2.3 Tietojen esittäminen

Jokaisella toimipaikalla on oma sivu CTT-tulosten tarkastelua varten. Tuloksia voi tarkastella joko valitun päivän mukaan (päivä valitaan pudotusvalikosta), jolloin näkee tulokset ja täyttöasteet alueiden mukaan kaikilta mittausajankohdilta (ks. kuva 7).

2015-04-28 tiistai	11					14					17				
	laitteella	ilman	yhteensä	paikkoja	täyttöaste	laitteella	ilman	yhteensä	paikkoja	täyttöaste	laitteella	ilman	yhteensä	paikkoja	täyttöaste
1. krs -kiinteät koneet	2	0	2	2	100 %	1	0	1	2	50 %	1	0	1	2	50 %
1. krs kirjastokioskit	0	0	0	1	0 %	0	0	0	1	0 %	0	0	0	1	0 %
1. krs lukupaikat	1	0	1	7	14 %	4	0	4	7	57 %	3	1	4	7	57 %
1. krs Ryhmätyöhuoneet	9	0	9	22	41 %	2	2	4	22	18 %	1	0	1	22	5 %
2. krs -kiinteät koneet	14	0	14	16	88 %	13	0	13	16	81 %	11	0	11	16	69 %
2. krs kirjastokioskit	0	0	0	3	0 %	1	0	1	3	33 %	1	0	1	3	33 %
2. krs lukupaikat	0	5	5	14	36 %	1	0	1	14	7 %	1	3	4	14	29 %
2. krs Niilin puutarha	0	0	0	9	0 %	2	2	4	9	44 %	0	0	0	9	0 %
3. krs -kiinteät koneet	16	0	16	19	84 %	19	0	19	19	100 %	10	0	10	19	53 %
3. krs kirjastokioskit	0	0	0	2	0 %	0	0	0	2	0 %	0	0	0	2	0 %
3. krs lukupaikat	16	15	31	53	58 %	20	17	37	53	70 %	12	6	18	53	34 %
3. krs Ryhmätyöhuoneet	4	0	4	20	20 %	4	1	5	20	25 %	3	2	5	20	25 %
4. krs -kiinteät koneet	8	0	8	8	100 %	8	0	8	8	100 %	3	0	3	8	38 %
4. krs kirjastokioskit	0	0	0	2	0 %	0	0	0	2	0 %	0	0	0	2	0 %
4. krs lukupaikat	8	23	31	73	42 %	8	21	29	73	40 %	4	15	19	73	26 %
4. krs Ryhmätyöhuoneet	7	0	7	28	25 %	8	0	8	28	29 %	4	0	4	28	14 %
Yhteensä	**85**	**43**	**128**	**279**	**46 %**	**91**	**43**	**134**	**279**	**48 %**	**54**	**27**	**81**	**279**	**29 %**

Kuva 7. Yhden päivän tietojen tarkastelu, esimerkkinä Viikin toimipaikka.

Toinen tapa tarkastella tuloksia on valita pudotusvalikosta jokin paikka, jolloin saa kyseisen paikan kaikki kirjatut käyttöasteet kaudesta riippumatta (ks. kuva 8).

Hiljainen lukusali (2. krs), 40 paikkaa	
2015-04-22 keskiviikko klo 11	100 %
2015-04-22 keskiviikko klo 14	108 %
2015-04-22 keskiviikko klo 18	53 %
2015-04-23 torstai klo 11	78 %
2015-04-23 torstai klo 14	95 %
2015-04-23 torstai klo 18	50 %
2015-04-24 perjantai klo 11	85 %
2015-04-24 perjantai klo 14	88 %
2015-04-24 perjantai klo 18	43 %
2015-04-25 lauantai klo 14	45 %
2015-04-26 sunnuntai klo 14	73 %
2015-04-27 maanantai klo 11	80 %
2015-04-27 maanantai klo 14	75 %
2015-04-27 maanantai klo 18	65 %
2015-04-28 tiistai klo 11	75 %
2015-04-28 tiistai klo 14	80 %
2015-04-28 tiistai klo 18	60 %
2015-05-11 maanantai klo 11	100 %
2015-05-11 maanantai klo 14	85 %

Kuva 8. Keskustan hiljaisen lukusalin mitatut käyttöasteet.

Kolmas tapa tarkastella tuloksia on valita pudotusvalikosta kausi, jolloin saa koko toimipaikan käyttöasteen kyseisenä kautena (ks. kuva 9).

	11	13	17
2015-04-22 keskiviikko	40 %	41 %	36 %
	11	13	17
2015-04-23 torstai	41 %	50 %	30 %
	11	13	17
2015-04-24 perjantai	34 %	39 %	27 %
	11	13	17
2015-04-27 maanantai	38 %	52 %	28 %
	11	13	17
2015-04-28 tiistai	39 %	64 %	24 %

Kuva 9. Kumpulan mitatut käyttöasteet keväällä 2015.

Kun järjestelmässä on enemmän tallennettuja kausia, tähän voisi vielä lisätä aikajanan, josta näkee käyttöasteiden kehityksen.

4 Tietojen hakeminen automaattisesti

4.1 Kirjastojärjestelmän lainaustiedot

4.1.1 Kirjastojärjestelmä ja tarvittavat ohjelmat

Helsingin yliopiston kirjastolla on – kuten kaikilla Suomen korkeakoulukirjastoilla – käytössään Voyager-kirjastojärjestelmä jonka taustalla on Oracle-tietokanta. Tietokanta sijaitsee CSC:n ylläpitämällä palvelimella, jossa ovat myös muiden korkeakoulukirjastojen kannat. Palvelin tunnetaan yleisesti nimellä Sanni, ja Helsingin yliopiston kirjaston kanta nimellä Helka. Helkassa on Helsingin yliopiston kirjaston lisäksi mukana myös Kansalliskirjasto ja joitain yiopiston ulkopuolisia tieteellisiä kirjastoja, kuten Suomalaisen kirjallisuuden seuran ja Museoviraston kirjastot. Tässä työssä käsitellään vain Helsingin yliopiston kirjastoa koskevia tietoja ja muutamassa tapauksessa koko Helkaa koskevia tietoja.

Helkaan on tallennettu tiedot kirjaston kokoelmista, asiakkaista ja fyysisen aineiston käyttöön liittyvistä tapahtumista kuten varauksista, lainoista, palautuksista, uusinnoista ja myöhästymismaksuista. Koska Helkassa on asiakkaiden yhteystiedot, henkilötunnukset ja lainaustiedot, tietokantaan ei ole tietoturvasyistä mahdollista tehdä kyselyitä toiselta palvelimelta käsin. Halutut kyselyt on tehtävä Sannilla, ja tulokset tallennetaan yhteen tai useampaan tiedostoon, jotka voi sitten scp:llä siirtää muualle.

Tiedostojen siirto Sannilta Libstatille järjestettiin Helka-ylläpidon käytössä olevalla käyttäjätunnuksella, jolle pyydettiin tietotekniikkaosastolta oikeudet kirjautua Libstatille Sannilta avainautentikoinnilla. Helka-palvelut loi sitten kirjautumiseen tarvittavat avaimet ja varmisti yhteyden toimivuuden. Libstatilla toinen ohjelma siirtää tiedot omaan tietokantaan.

Prosessi kokonaisuudessaan:

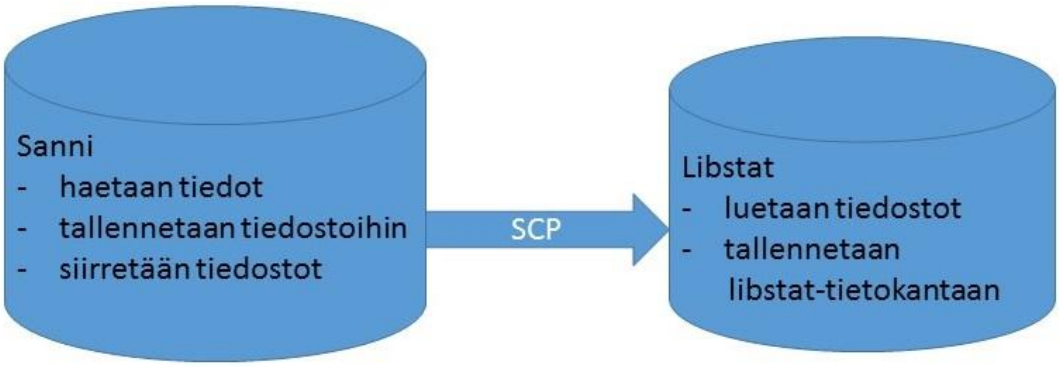
1. Sannilla käynnistetään ajastetusti (crontab) shell wrapper-skripti

2. Wrapper-skripti

    - käynnistää ohjelman, joka suorittaa halutut kyselyt ja tallentaa tulokset tiedostoon / tiedostoihin

    - siirtää scp-ohjelmalla tiedoston / tiedostot Sannilta Libstatille

3. Libstatilla on toinen wrapper-skripti, joka lukee tallennetut tiedostot ja vie tiedot omaan tietokantaan. Wrapper-skripti käynnistetään ajastetusti (crontab).



Kuva 10. Tietojen hakeminen kirjastojärjestelmästä.

Sannilla on mahdollista käyttää useita ohjelmointikieliä, mm. Javaa (versio 6), Pythonia (versio 2) ja Perliä (versio 5.8.4). Sannilla käytetyin skriptauskieli on Perl, ja Helkan osalta on käytetty myös Pythonia (josta tosin on käytössä vain Python2, joka eroaa aika paljon uudemmasta Python3:sesta). Päätin käyttää ohjelmiin, jotka hakevat ja tallentavat tietoja helppokäyttöistä Javaa, josta minulla on myös eniten kokemusta.

Javan mukana tulevien luokkien lisäksi tarvitsin seuraavat lisäosat:

- Joda-Time helpottaa suuresti päivämäärien ja aikojen käsittelyä (http://www.joda.org/joda-time/index.html). Joda-Time-toiminnallisuudet ovat suoraan käytettävissä uusimmassa Java-versiossa (Java 8) paketissa java.time, mutta koska Sannilla on käytössä Java 6 ja Libstatilla Java 7, tarvitsen edelleen erillisiä Joda-Time jar -tiedostoja.

- ajuri mahdollistaa kyselyiden tekemisen Oracle-tietokantoihin (oracle6.jar, ladattavissa Oraclen verkkosivuilta). Tarvitaan vain Sannilla.

- ajuri mahdollistaa kyselyiden tekemisen MySQL- ja MariaDB-tietokantoihin (MySQL JDBC Driver, tulee NetBeans-kehitysympäristön mukana ja otetaan käyttöön kohdassa Libraries).

4.1.2 Päivittäin tapahtuva tietojen haku

Vaikka useimmat tiedot ovat haettavissa kirjastojärjestelmästä jälkikäteen, jotkut tiedot ovat luonteeltaan sellaisia, että ne on haettava useammin kuin kuukaudessa. Näitä ovat erityisesti varauksiin liittyvät tiedot, koska noutamattomista varauksista (varaukset, joita ei peruta eikä noudeta) ei jää järjestelmään mitään tietoa. Niin kauan kun kirja on vielä palauttamatta ja viemättä takaisin hyllyyn varaus näkyy järjestelmässä, ja samoin näkyy päivämäärä, jolloin se on umpeutunut. Noutamattomat varaukset käsitellään aamulla, joten noutamattomien varausten määrän saa järjestelmästä vain kirjastojen sulkeuduttua ja ennen seuraavan aamun käsittelyä. Tieto on siinä mielessä tärkeä, että noutamattomat varaukset aiheuttavat paljon työtä, ja Keskustassa niistä – samoin kuin jo saapuneista perutuista varauksista - onkin aikaisemmin pidetty kirjaa niin, että käsittelijä merkitsee määrän palvelualueen paperikalenteriin.

Samoin esim. tieto siitä, kuinka paljon kirjoja kulloinkin odottaa noutoa varaushyllyssä on haettava kirjaston ollessa kiinni yöllä. Selkeyden vuoksi kaikki muutkin varauksia koskevat tiedot on syytä hakea päivittäin, myös ne, jotka olisi myöhemminkin haettavissa. Varaustietojen lisäksi otin päivittäin haettaviin tietoihin mukaan sen, paljonko on käsitelty muiden toimipaikkojen palautuksia; tieto löytyy aamuyöstä tyhjennettävästä logitaulusta. Kyseinen tieto tulee tarkemmin kuukausitasolla palautusten yhteydessä, jolloin näkee myös sen, minkä kirjaston palautuksista oli kyse.

Haettavat tiedot:

- noutoa odottavat varaukset kampuksittain ja aineistopyynnöt Keskustassa (aineistopyyntöjä ei ole muilla kampuksilla): 5 kokonaislukua

- noutamattomat varaukset kampuksittain (edellisenä päivänä vanhentuneet): 4 kokonaislukua

- noutoa odottaneet, perutut varaukset kampuksittain: 4 kokonaislukua
- käsitellyt muiden kirjastojen palautukset kampuksittain: 4 kokonaislukua
- tehdyt aineistopyynnöt kokoelmittain (on neljä suljetussa varastossa olevaa kokoelmaa, joihin voi tehdä aineistopyyntöjä): 4 kokonaislukua
- verkossa tehdyt varaukset kampuksittain: 4 kokonaislukua
- virkailijoiden tekemät varaukset kampuksittain: 4 kokonaislukua
- noudetut varaukset kampuksittain: 4 kokonaislukua.

Nuo 33 lukua tallennetaan tiedostoon, joka nimetään xdaily.txt, jossa x on kuukauden päivä, jolloin ohjelma suoritetaan. Tiedostoja on siis palvelimilla 31, ja joka yö yhden päälle kirjoitetaan uusi tiedosto. Tiedoston ensimmäisellä rivillä on kuluvan päivän päivämäärä ja sen jälkeen em. luvut.

Kyselyt suorittavassa ja tiedoston tuottavassa ohjelmassa on kolme luokkaa:

- Oracle hoitaa tietojen hakemisen tietokannasta. Konstruktorissa luodaan yhteys tietokantaan, ja metodit, jotka hakevat tiedot tietokannasta palauttavat kokonaislukutaulukoita (yhdessä tapauksessa kokonaisluvun).
- Tiedosto hoitaa tiedoston luomisen ja siihen kirjoittamisen. Konstruktorissa luodaan tiedosto ja kirjoitetaan kuluvan päivän päivämäärä ensimmäiselle riville, metodit lisäävät tiedostoon lukutaulukkoja tai lukuja.
- Paivittain sisältää main-metodin, jossa kutsutaan Oraclen ja Tiedoston metodeja.

Tietojen hakeminen kannasta noudattaa yleensä sellaista kaavaa, että määritellään kokoelmatunnukset (myös toimipaikat määritellään kokoelmatunnuksilla), joista halutaan hakea, valmistellaan kysely ja suoritetaan se silmukassa halutuilla tunnuksilla. Usein olisi mahdollista tehdä kysely myös yhdellä kertaa ja määritellä tunnukset suoraan kyselyssä, mutta mahdolliset muutokset on myöhemmin helpompi tehdä jos kokoelmatunnukset ovat itse kyselystä erillään. Jos kyselyjä haluaan muuttaa, todennäköisimmin muutokset

liittyvät juuri käytettäviin kokoelmatunnuksiin. Esimerkkinä on metodi, jolla haetaan noutoa odottavat varaukset:

```
public int[] haeNoudettavat(String tanaan) {
// 443=Kaisa-talo, 176=Minerva, 179=Kumpula, 180=Terkko, 202=Hammas, 177=Viikki
// lokaatiot sisältää listan niistä lokaatioista, joista haetaan noudettavana olevat
// varaukset
        int[] lokaatiot = {443, 176, 179, 180, 202, 177};
        // maarat-taulukkoon tallennetaan kyselyiden tulokset
        int[] maarat = new int[6];
        //taulukkoon kampuskohtaiset tallennetaan kampuskohtaiset tiedot järjestyksessä
// Keskusta, Kumpula, Terkko, Viikki
        int[] kampuskohtaiset = new int[4];
        try {
            //Valmistellaan kysely pending-tilassa olevista varauksista jotka eivät ole
// vanhentuneet eivätkä ole aineistopyyntöjä
PreparedStatement ps = conn.prepareStatement("SELECT COUNT(HOLD_RECALL_ITEMS.ITEM_ID) "
                  + "FROM HOLD_RECALL_ITEMS, HOLD_RECALL "
                  + "WHERE HOLD_RECALL_ITEMS.HOLD_RECALL_ID=HOLD_RECALL.HOLD_RECALL_ID  "
                    + "AND PICKUP_LOCATION=? AND HOLD_RECALL_STATUS=2 AND CALL_SLIP_ID=0
AND EXPIRE_DATE >= TO_DATE('" + tanaan + "', 'yyyy-mm-dd') ");
//Suoritetaan kysely jokaista lokaatiota kohti ja tallennetaan tulokset maarat-taulukkoon.
            for (int x = 0; x < lokaatiot.length; x++) {
                ps.setInt(1, lokaatiot[x]);
                ResultSet tulos = ps.executeQuery();

                while (tulos.next()) {
                    maarat[x] = tulos.getInt(1);

                }
            }
            //Kaisa + Minerva
            kampuskohtaiset[0] = maarat[0] + maarat[1];
            //Kumpula
            kampuskohtaiset[1] = maarat[2];
            //Terkko + Hammas
            kampuskohtaiset[2] = maarat[3] + maarat[4];
            //Viikki
            kampuskohtaiset[3] = maarat[5];


            return kampuskohtaiset;

        } catch (Exception e) {
            System.out.println(e.getMessage());
            return null;
        }
    }
```

Kuten kyselystä näkee, Keskustan luku muodostuu Kaisa-talon ja Minervan yhteenlasketuista luvuista ja Meilahden luku Terkon ja Hammastieteellisen kirjaston yhteenlasketuista luvuista. Minervan toimipaikka on yhdistynyt Kaisa-taloon keväällä 2015 ja Hammaslääketieteellinen yhdistetään Terkkoon kesällä 2016. Kun nämä muutot on saatu päätökseen, voi hieman yksinkertaistaa hakuja tekeviä metodeja.

Paivittain-luokan main-metodi luo instanssit Oracle- ja Tiedosto-luokista ja kutsuu niiden metodeja antaen tarvittavat päivämäärät parametreiksi:

```
   public static void main(String[] args) {
        String tanaan = new LocalDate().toString();
        String eilen = new LocalDate().minusDays(1).toString();
```

```
        Oracle o = new Oracle();
        Tiedosto t = new Tiedosto(new LocalDate().getDayOfMonth(), tanaan);


        //hae noudettavat varaukset (ei call slippejä)
       int noudettavat[]= o.haeNoudettavat(tanaan);
       t.kirjoitaLukuTaulukko(noudettavat);

        // hae noudettavat call slipit
        int noudettavatCS= o.haeNoudettavatCallSlipit();
        t.kirjoitaLuku(noudettavatCS);

        int [] noutamattomat = o.haeNoutamattomat(eilen, tanaan);
        t.kirjoitaLukuTaulukko(noutamattomat);


        int [] perutut = o.haePerutut(eilen, tanaan);
        t.kirjoitaLukuTaulukko(perutut);

        int[] vieraatPalautukset = o.haeVieraatPalautukset();
        t.kirjoitaLukuTaulukko(vieraatPalautukset);

        int [] tehdytCallSlipit = o.haeTehdytCallSlipit(eilen, tanaan);
        t.kirjoitaLukuTaulukko(tehdytCallSlipit);

        int [] tehdytHoldit = o.haeTehdytHoldit(eilen, tanaan);
        t.kirjoitaLukuTaulukko(tehdytHoldit);

        int[] haetutVaraukset = o.haeNoudetutVaraukset(eilen, tanaan);
        t.kirjoitaLukuTaulukko(haetutVaraukset);

        t.lopeta();
        o.suljeYhteys();
    }
```

Koska Sannilla on Java 6, NetBeansissa valitaan kohdassa Properties – Sources Source/Binary Format: JDK6. Tämän jälkeen projekti rakennetaan ja dist-kansiosta siirretään Sannille jar-tiedosto ja lib-kansio, joka sisältää tarvittavat lisäkirjastot. Samaan kansioon Sannille laitetaan run.sh-niminen skripti, joka käynnistää ohjelman:

```
#! /usr/bin/bash
# Ajetaan Nicolan crontabista paivittain 00:11:
cd /export/home/OMATUNNUS/libstat/paivittain
java –jar libstatPaivittain.jar
cd /export/home/OMATUNNUS/libstat/reports/
/usr/local/bin/scp *daily.txt YHTEISTUNNUS@libstat-0.hulib.helsinki.fi:paivittain/
```

Tämä skripti käynnistyy siis joka yö klo 00:11. Kun käyttää scp:tä, kaikki tiedostot siirretään joka yö riippumatta siitä, onko niitä päivitetty tai ei. Parempi olisi käyttää rsync-ohjelmaa, mutta sitä ei saatu toimimaan Sannin ja Libstatin välillä. Käytännössä tiedostot ovat niin pieniä, ettei sillä tässä tapauksessa ole juuri väliä, siirretäänkö yksi vai 31 tiedostoa.

Libstatilla on toinen Java-ohjelma, jossa on kolme luokkaa:

- MariaDB tallentaa tiedostosta luetut luvut tietokantaan.

- TiedostoLuku lukee päivän tiedoston ja tallentaa luvut HashMap-rakenteeseen, jossa avaimena on merkkijono ja arvona lukutaulukko.

- SanniPaivittain sisältää em. luokkien metodeja kutsuvan main-metodin.

TiedostoLuku-luokan metodi tarkistaa ensin, että kuukauden päivää vastaavassa tiedostossa on kuluvain päivän päivämäärä, niin että tietokantaan ei tule vietyä vanhoja tietoja, jos tiedostonsiirto on jostain syystä epäonnistunut. Jos tiedostonsiirto on onnistunut, tallennetut luvut luetaan HashMap-rakenteeseen, jossa avaimena oleva merkkijono kuvailee luettavia lukuja (esim. ”noutoaOdottavat”) ja itse luvut luetaan lukutaulukkoon. Näin saadaan niputettua eri tauluihin tallennettavat tiedot omiin taulukoihin. Avaimina toimivat merkkijonot ja kunkin lukutaulukon pituus määritellään ensin omissa taulukoissa, jonka jälkeen itse lukeminen tehdään silmukassa:

```
    public Map<String, Integer[]> luePaivanTiedosto(String pvm, int paiva) {

        Map<String, Integer[]> listat = new HashMap<String, Integer[]>();

        try {

            // in = new BufferedReader(new FileReader("/home/YHTEISTUNNUS/paivittain/" +
paiva + "daily.txt"));
            if (in.readLine().equals(pvm)) {

               String[] listaNimet = {"noutoaOdottavat", "noudettavatAineistopyynnot",
              "noutamattomat", "perutut", "vieraatPalautukset",
              "aineistopyynnotKok",
                     "holdOPAC", "holdVirk", "noudetut"};

               int[] listaLkm = {4, 1, 4, 4, 4, 4, 4, 4, 8};

               for (int y = 0; y < listaNimet.length; y++) {
                   listat.put(listaNimet[y], new Integer[listaLkm[y]]);
                   Integer[] lista = listat.get(listaNimet[y]);
                   for (int x = 0; x < lista.length; x++) {
                       lista[x] = Integer.valueOf(in.readLine());
                   }
               }

               return listat;

            } else {
                return null;
            }

        } catch (Exception e) {
            System.out.println(e.getMessage());;
            return null;
        }

    }
```

Kun tiedoston lukutaulukko on näin pätkitty, on koko ajan helppo nähdä, mitä ollaan kulloinkin käsittelemässä, ja jos ohjelmaa muutetaan niin, että tietokannasta haetaan

lisää lukuja, tarvitsee vain lisätä uudelle lukusetille nimi ja ilmoittaa koko listaNimet- ja listaLkm-taulukoihin.

MariaDB-luokan konstruktorissa luodaan yhteys libstat-tietokantaan ja metodeissa tallennetaan tiedot tietokantaan. Metodissa yleensä valmistellaan insert-lause ja suoritetaan se sitten silmukassa niin, että kaikki luvut saadaan kantaan. Esimerkkinä on metodi, joka tallentaa jokaisen kampuksen kohdalle noutamattomat varaukset:

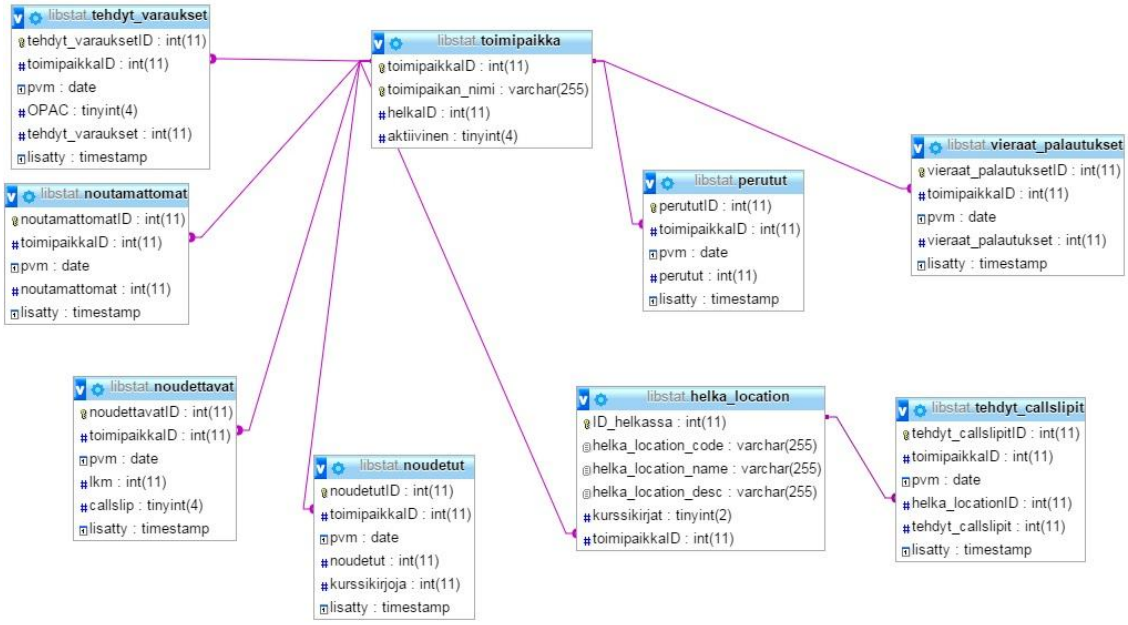
```
    public void tallennaNoutamattomat(Integer[] lkmt, String pvm) {
        try {
            PreparedStatement ps = yhteys.prepareStatement("INSERT INTO noutamatto-
mat(pvm, toimipaikkaID, noutamattomat) VALUES('" + pvm + "',?,?)");

            for (int x = 1; x < 5; x++) {

                ps.setInt(1, x);
                ps.setInt(2, lkmt[x - 1]);
                ps.executeUpdate();

            }

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
```

SanniPaivittain-luokan main-metodi luo instanssit MariaDB- ja TiedostoLuku-luokista ja kutsuu niiden metodeja antaen tarvittavat parametrit:

```
public static void main(String[] args) {
        TiedostoLuku tl = new TiedostoLuku();
        MariaDB mdb = new MariaDB();

        String tanaan = new LocalDate().toString();
        String eilen = new LocalDate().minusDays(1).toString();

       Map<String, Integer[]> listat = tl.luePaivanTiedosto(tanaan, new LocalDate().get-
DayOfMonth());

       mdb.tallennaNoudettavat(listat.get("noutoaOdottavat"), tanaan);

       mdb.tallennaNoudettavatCallSlipit((listat.get("noudettavatAineistopyynnot"))[0],
tanaan);

       mdb.tallennaNoutamattomat(listat.get("noutamattomat"), tanaan);

       mdb.tallennaPerutut(listat.get("perutut"), eilen);

       mdb.tallennaVieraatPalautukset(listat.get("vieraatPalautukset"), eilen);

       mdb.tallennaAineistopKok(listat.get("aineistopyynnotKok"), eilen);

       mdb.tallennaHolditOPAC(listat.get("holdOPAC"), eilen);

       mdb.tallennaHolditVirk(listat.get("holdVirk"), eilen);

       mdb.tallennaNoudetut(listat.get("noudetut"), eilen);

        tl.sulje();
        mdb.suljeYhteys();

    }
```

Koska Libstatilla käytössä on Java 7, binääritiedoston formaatiksi valitaan JDK7. Kuten Sannilla olevan ohjelman kohdalla jar-tiedosto siirretään palvelimelle ja ohjelma ajastetaan käynnistymään crontabilla.

Päivittäin haettavat tiedot tallennetaan erillisiin tauluihin, joilla yleensä on yhteys vain toimipaikka-tauluun tai kokoelmatunnukset sisältävään helka_location-tauluun:



Kuva 11. Taulut, joihin tallennetaan päivittäin haettavat tiedot.

Päivittäin haettavat tiedot ovat siis valmiita koostelukuja, jotka eivät sisällä tietoa esim. yksittäisistä varauksista.

Käyttöliittymässä tiedot esitetään kuukausitasolla ja kampuksittain vuositasolla:

Vuosi 2015 Valitse

2015	Tehdyt (Hold)	joista itsepalveluna	itsepalvelu%	Noudetut	joista kurssikirjoja	kurssikirja %	Noutamattomat	Perutut	Noutoa odottavat min (Hold)	Noutoa odottavat max (Hold)	Tehdyt aineistopyynnöt	joista graduja
Tammikuu	0	0		0	0		768	501	0	0	0	0
Helmikuu	0	0		0	0		904	452	0	0	0	0
Maaliskuu	0	0		0	0		970	418	0	0	0	0
Huhtikuu	5407	4293	79%	4116	0	0%	918	416	761	1280	679	519
Toukokuu	4467	3580	80%	3116	1174	38%	797	380	547	896	448	376
Kesäkuu	3502	2666	76%	2400	1152	48%	713	288	437	610	364	36
Heinäkuu	1474	1125	76%	863	315	37%	372	105	288	438	166	0
Elokuu	0	0		0	0		0	0	0	0	0	0
Syyskuu	0	0		0	0		0	0	0	0	0	0
Lokakuu	0	0		0	0		0	0	0	0	0	0
Marraskuu	0	0		0	0		0	0	0	0	0	0
Joulukuu	0	0		0	0		0	0	0	0	0	0
Yhteensä	**14850**	**11664**	**79%**	**10495**	**2641**		**5442**	**2560**			**1657**	**931**

Kampuksittain

2015	Tehdyt (Hold)	joista itsepalveluna	itsepalvelu%	Noudetut	joista kurssikirjoja	Noutamattomat	Perutut	Noutoa odottavat min (Hold)	Noutoa odottavat max (Hold)
Keskusta	14236	11189	79%	10051		5221	2501	283	1186
Kumpula	126	106	84%	125		39	8	0	26
Meilahti	163	112	69%	87		85	19	1	23
Viikki	325	257	79%	232		97	32	3	45

Kuva 12. Varaustietojen esitteleminen.

Tiedot noutamattomista ja perutuista varauksista on Keskustan osalta olemassa vuoden 2015 alusta, koska alkuvuoden tiedot oli helppo siirtää tietokantaan paperikalenterista – muilta osin tiedot saadaan alkaen 7.4.2015.

Koosteen lisäksi on lähinnä asiakaspalveluissa työskentelevien iloksi ”Top 10” -listat, jotka varsin hyvin kuvastavat sitä, mitkä päivät ovat olleet asiakaspalvelussa kaikkein kiireisempiä:

Top10 Noudetut	
pvm	lkm
2015-04-13 maanantai	324
2015-04-10 perjantai	302
2015-04-14 tiistai	294
2015-04-09 torstai	283
2015-04-15 keskiviikko	271
2015-05-25 maanantai	262
2015-04-08 keskiviikko	244
2015-04-16 torstai	231
2015-04-27 maanantai	229
2015-05-04 maanantai	228

Top10 Noutamattomat	
pvm	lkm
2015-01-16 perjantai	257
2015-05-26 tiistai	126
2015-04-16 torstai	117
2015-04-21 tiistai	97
2015-02-10 tiistai	90
2015-04-11 lauantai	79
2015-05-12 tiistai	78
2015-03-31 tiistai	77
2015-06-30 tiistai	68
2015-03-10 tiistai	67

Top10 Perutut	
pvm	lkm
2015-01-15 torstai	83
2015-05-15 perjantai	76
2015-06-05 perjantai	73
2015-03-09 maanantai	60
2015-01-09 perjantai	54
2015-01-27 tiistai	54
2015-02-02 maanantai	50
2015-02-24 tiistai	46
2015-04-13 maanantai	44
2015-03-10 tiistai	42

Top10 Aineistopyynnöt	
pvm	lkm
2015-04-15 keskiviikko	95
2015-04-14 tiistai	46
2015-04-16 torstai	42
2015-04-10 perjantai	40
2015-04-13 maanantai	39
2015-04-22 keskiviikko	35
2015-04-23 torstai	35
2015-05-04 maanantai	35
2015-05-15 perjantai	34
2015-04-28 tiistai	31

Kuva 13. Varauksiin liittyvät Top 10- listat.

4.1.3 Kuukausittain tapahtuva tietojen haku

Kuukausittain kirjastojärjestelmästä halutaan hakea aineiston lainaukseen liittyvät luvut kuten lainaukset, palautukset, uusinnat, tehdyt kortit ja suoritetut maksut.

Teknisesti toteutus on muuten samanlainen kuin päivittäin haettavissa tiedoissa, mutta suuri osa tiedoista tallennetaan olioina, jokainen oliotyyppi omaan tiedostoonsa, ja sen lisäksi tallennetaan yksi kokonaislukuja sisältävä tiedosto samaan tyyliin kuin päivittäin. Kun halutaan tallentaa valmiiden koostelukujen sijasta esim. yksittäisiä lainaustapahtumia, on selkeää niputtaa yhteenkuuluvat arvot olioksi. Ohjelmassa on seuraavat luokat olioiden tallennusta varten:

- Laina
- Palautus
- Kortti
- LainaavatAsiakkaatTiski
- PalauttavatAsiakkaatTiski
- UusivatAsiakkaatTiski
- PoistettuMaksu.

Laina-luokalla on kentät seuraaville tiedoille:

- lainauspäivämäärä
- lainahetki (päivämäärä ja aika)
- viikonpäivä
- lainauspaikka
- lainatun niteen kokoelmatunnus
- lainatun niteen toimipaikka

- automaattilainaus (kyllä / ei)
- niteen signum (”hyllyosoite”)
- kurssikirja (kyllä/ei)
- lainaustyyppi (normal / override)
- asiakasryhmä
- asiakkaan tilastoryhmä.

Lainauspäivämäärä on pelkkä päivämäärä ja lainaushetki on päivämäärä ja aika – jälkimmäinen tietenkin riittäisi, mutta päivämäärän tallentaminen erikseen nopeuttaa käyttöliittymässä tehtäviä kyselyitä. Myös lainatun niteen toimipaikka on kyselyitä nopeuttava tieto, sillä se olisi haettavissa myös niteen kokoelmatunnuksen perusteella. Kokoelmatunnuksen perusteella olisi pääteltävissä myös se, onko kirja kurssikirja tai ei.

Niteen signumin ottaminen mukaan tietokantaan mahdollistaa lainauslukujen tarkastelun kokoelmakohtaisesti halutulla tarkkuudella. Tämä tieto ei säily luotettavasti kirjastojärjestelmässä, sillä niteitä siirretään kokoelmien välillä ja myös poistetaan.

Palautus-luokalla on kentät seuraaville tiedoille:

- palautuspäivämäärä
- palautushetki (päivämäärä ja aika)
- viikonpäivä
- eräpäivä palautushetkellä
- palautuspaikka
- palautetun niteen kokoelmatunnus
- palautetun niteen toimipaikka
- automaattipalautus (kyllä / ei)

- kurssikirja (kyllä / ei)
- myöhässä (kyllä / ei)
- myöhässä yli kaksi viikkoa (kyllä / ei)
- myöhässä yli vuoden (kyllä / ei).

Kuten lainojen kohdalla myös palautuksissa siis tallennetaan jonkin verran tietoa, joka olisi laskettavissa toisesta tiedosta. Suurin osa palautuksen tiedoista on samantapaisia kuin lainojen kohdalla, mutta palautuksille ominaisia tietoja ovat tietenkin eräpäivä ja mahdollinen myöhästyminen. Yli kahden viikon myöhästyminen on siinä mielessä merkittävä, että asiakkaalta pitää poistaa käsin lainakielto, jolloin kyseinen palautus siis aiheuttaa työtä asiakaspalvelussa.

Kortti-luokalla on kentät seuraaville tiedoille:

- vuosi, jolloin kortti on tehty
- kuukausi, jolloin kortti on tehty
- asiakasryhmä
- asiakkaan tilastoryhmä
- luontipaikan ”kokoelmatunnus”
- toimipaikka, jossa luotu.

Toimipaikka olisi pääteltävissä kokoelmatunnuksen perusteella. Asiakasryhmän (Patron Group) mukaan on kirjastojärjestelmässä mahdollista määritellä lainaukseen liittyviä oikeuksia, kuten laina-aikoja, myöhästymismaksujen määriä jne. Helkassa valtaosa henkilöasiakkaista kuuluu yhteen kolmesta asiakasryhmästä: HY opiskelija, HY henkilökunta tai Ulkopuolinen. Asiakasryhmä on pakollinen tieto, ja jokaiseen lainaan liittyy yksi asiakasryhmä. Tilastoryhmät sen sijaan ovat pelkästään tilastoimista varten luotuja, huomattavasti tarkempia ryhmiä. Tilastoryhmissä huomioidaan esim. tiedekunta,

niin että jokaista tiedekuntaa kohti on ryhmät opiskelijoita ja henkilökuntaa varten. Tilastoryhmä ei ole pakollinen tieto, ja yhteen lainaan voi liittyä myös useita tilastoryhmiä. Jotta samaa lainaa ei laskettaisi kahteen kertaan, lasketaan vain ensimmäinen ryhmä (eli se, jonka id on pienin), jos on asetettu useita tilastoryhmiä.

Vaikka sekä asiakasryhmä että tilastoryhmä ovat kirjaston kannalta hyvin mielenkiintoisia tietoja, ne eivät valitettavasti ole kovin luotettavia: asiakasryhmä jää päivittämättä esim. kun opiskelija valmistuu (suuri osa tietenkin poistuu samalla kokonaan asiakaskunnasta, mutta monet jäävät asiakkaiksi joko yliopiston henkilökuntana, tai sitten ulkopuolisena käyttäjänä). Tilastoryhmä unohdetaan valitettavasti usein kokonaan, ja myös päivittämättä jättäminen on ongelma. Näistä ongelmista huolimatta käyttäjäryhmistä saatavia tietoja käytetään kirjastossa.

Luokkiin LainaavatAsiakkaatTiski, PalauttavatAsiakkaatTiski ja UusivatAsiakkaatTiski tallennetaan tiedot siitä, paljonko eri kirjastoissa hoidetaan lainausta, palautusta ja uusimista tiskeillä sekä mitä asiakasryhmiä ko. asiakkaat edustavat. On kiinnostavaa, paljonko edelleen tehdään tiskillä sellaista, johon kirjasto tarjoaa itsepalvelua.

Luokassa PoistetutMaksut tarkastellaan sitä, paljonko (euroissa) on poistettu syntyneitä maksuja kuittauksella *Forgive* tai *Error*. Nämä ovat siis asiakkaiden maksuja, jotka on poistettu joko virheellisinä tai suurien maksujen kohdalla tehtävän kohtuullistamisen yhteydessä.

Näiden lisäksi haetaan seuraavat lukuina käsiteltävät tiedot:

- Paljonko on tehty uusia kortteja kirjaston toimipaikoissa sekä koko Helkassa (5 kokonaislukua)?

- Paljonko on suoritettu maksuja kirjaston toimipaikoissa ja verkossa, riippumatta siitä, minkä kirjastoja maksuista on kyse; kultakin kampukselta (Keskusta, Kumpula, Meilahti, Viikki) lasketaan: myöhästymismaksut, noutamattomat varaukset, muut maksut (15 kokonaislukua)?

- Paljonko on syntynyt maksuja kirjaston kirjoista kultakin kampukselta (Keskusta, Kumpula, Meilahti, Viikki): myöhästymismaksut, noutamattomat varaukset, muut maksut (12 kokonaislukua)?

- Paljonko on suoritettu maksuja kirjaston kirjoista maksupaikasta riippumatta, kultakin kampukselta (Keskusta, Kumpula, Meilahti, Viikki): maksut (minkä tahansa kirjaston) tiskille, maksut verkossa (8 kokonaislukua)?

- Paljonko on uusittu eri toimipaikkojen niteitä verkossa ja tiskeillä (uusintapaikkoja ei eritelty) (8 kokonaislukua)?

- Paljonko HY kirjastojen tiskeillä on uusittu lainoja toimipaikoittain sekä kaikilla muilla tiskeillä tehdyt uusinnat (5 kokonaislukua)?

- Paljonko on kuukaudessa lainavia asiakkaita toimipaikoittain sekä koko Helkassa (5 kokonaislukua)?

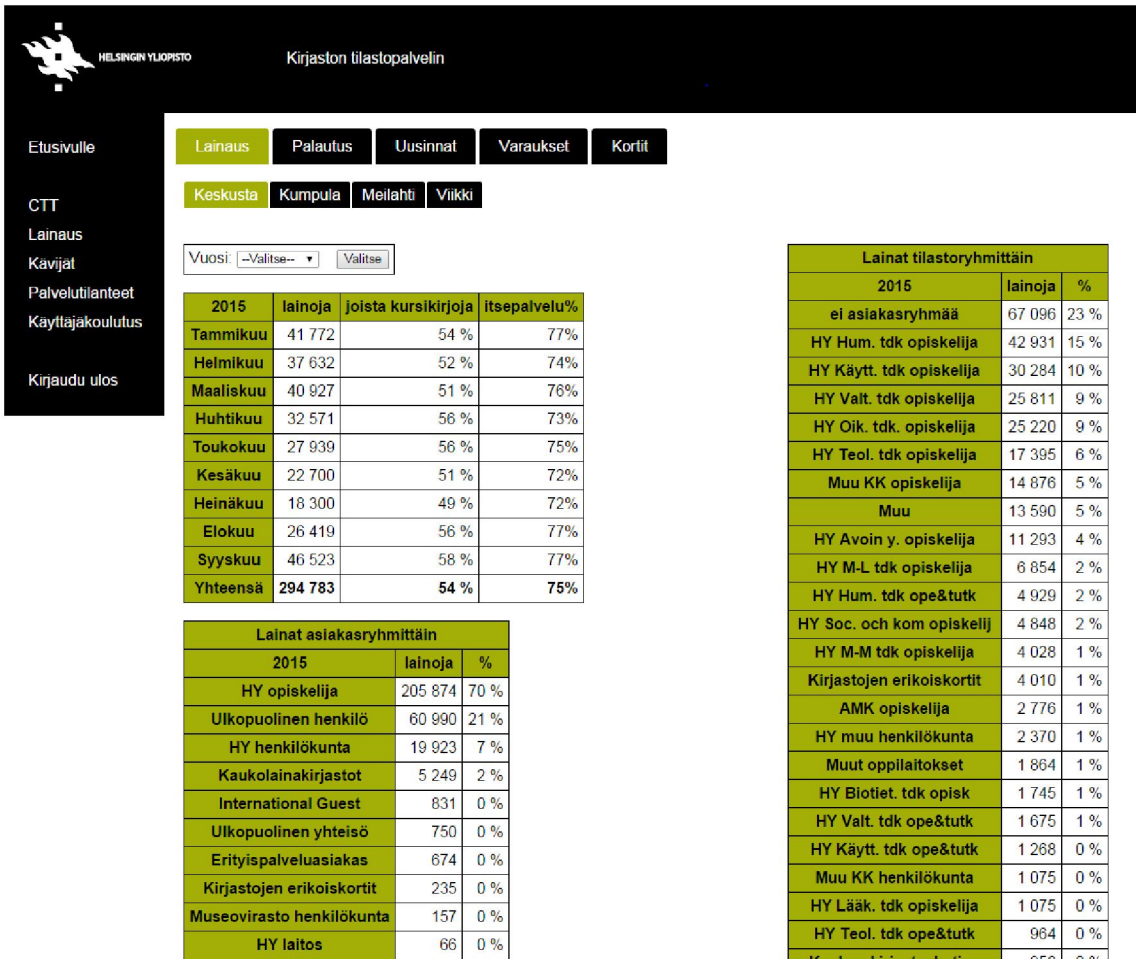
Kuten edellä päivittäin haettavien tietojen kohdalla käynnistetään ajastetusti Sannilla ohjelma, joka hakee tiedot Helkasta ja tallentaa ne tiedostoihin. Jokaista oliotyyppiä kohden kirjoitetaan oma tiedosto (käytetään ObjectOutputStream- ja FileOutputStream-luokkia), koska tietojen lukeminen on helpompaa, jos jokaisessa tiedostossa on vain yhdentyyppisiä olioita.

Oracle-niminen luokka huolehtii taas tietojen hakemisesta, ja Kirjoittaja-luokka tiedostojen kirjoittamisesta. LibstatKuukausittain-luokan main-metodi luo instanssit Oracle- ja Kirjoittaja-luokista sekä kutsuu niiden metodeja.

Kun luodut tiedostot on siirretty Libstatille, toinen Java-ohjelma käy lukemassa tiedot ja tallentaa ne omaan kantaan.

Käyttöliittymässä on toistaiseksi mahdollista tarkastella lainauksia, palautuksia ja tehtyjä kirjastokortteja, muilta osin kerääntyy jo dataa, mutta sitä ei voi vielä tarkastella liittymässä.

Kuvassa 13 on esimerkkinäkymä Keskustan lainaustiedoista käyttöliittymässä.



2015	lainoja	joista kursikirjoja	itsepalvelu%
Tammikuu	41 772	54 %	77%
Helmikuu	37 632	52 %	74%
Maaliskuu	40 927	51 %	76%
Huhtikuu	32 571	56 %	73%
Toukokuu	27 939	56 %	75%
Kesäkuu	22 700	51 %	72%
Heinäkuu	18 300	49 %	72%
Elokuu	26 419	56 %	77%
Syyskuu	46 523	58 %	77%
Yhteensä	**294 783**	**54 %**	**75%**

Lainat asiakasryhmittäin		
2015	lainoja	%
HY opiskelija	205 874	70 %
Ulkopuolinen henkilö	60 990	21 %
HY henkilökunta	19 923	7 %
Kaukolainakirjastot	5 249	2 %
International Guest	831	0 %
Ulkopuolinen yhteisö	750	0 %
Erityispalveluasiakas	674	0 %
Kirjastojen erikoiskortit	235	0 %
Museovirasto henkilökunta	157	0 %
HY laitos	66	0 %

Lainat tilastoryhmittäin		
2015	lainoja	%
ei asiakasryhmää	67 096	23 %
HY Hum. tdk opiskelija	42 931	15 %
HY Käytt. tdk opiskelija	30 284	10 %
HY Valt. tdk opiskelija	25 811	9 %
HY Oik. tdk. opiskelija	25 220	9 %
HY Teol. tdk opiskelija	17 395	6 %
Muu KK opiskelija	14 876	5 %
Muu	13 590	5 %
HY Avoin y. opiskelija	11 293	4 %
HY M-L tdk opiskelija	6 854	2 %
HY Hum. tdk ope&tutk	4 929	2 %
HY Soc. och kom opiskelij	4 848	2 %
HY M-M tdk opiskelija	4 028	1 %
Kirjastojen erikoiskortit	4 010	1 %
AMK opiskelija	2 776	1 %
HY muu henkilökunta	2 370	1 %
Muut oppilaitokset	1 864	1 %
HY Biotiet. tdk opisk	1 745	1 %
HY Valt. tdk ope&tutk	1 675	1 %
HY Käytt. tdk ope&tutk	1 268	0 %
Muu KK henkilökunta	1 075	0 %
HY Lääk. tdk opiskelija	1 075	0 %
HY Teol. tdk ope&tutk	964	0 %

Kuva 14. Osa näkymässä, jossa on Keskustan lainaustiedot vuodelta 2015.

4.2 MikroVäylän tuottamat kävijäluvut

Kirjasto on hankkinut lainauksessa ja palautuksessa käytettävät automaattinsa ja hävikinestiporttinsa MikroVäylä-nimisestä yrityksestä (http://www.mikrovayla.fi/). Keskustassa uusista automaateista ja porteista kertyy tietoa MikroVäylän omaan tietokantaan, jonka tietoja kirjastot voivat tarkkailla MikroVäylän Servmanager-palvelussa. Keskustan osalta Servmanagerissa on tietoja syksystä 2012 alkaen. MikroVäylä on ystävällisesti sallinut minulle pääsyn omaan tietokantansa näkymiin (View), joissa on Helsingin yliopiston kirjastoa koskevat tiedot. Koska tietokannassa ei ole henkilötietoja tai muita arkaluontoisia tietoja, kyselyt voi tehdä suoraan Libstatilta.

Verkossa olevissa hävikinestoporteissa on kävijälaskurit, joista MikroVäylän ohjelma käy joka tunti hakemassa lukemat. Portit arvioivat myös suuntaa, eli laskuri ilmoittaa yhden luvun sisään tulleille ja toisen ulosmenneille.

Servmanagerissa tietoja on mahdollista tarkastella kuukausitasolla (kun valitsee vuoden) tai päivätasolla (kun valitsee kuukauden). Tuntikohtaiset tilastot saa porttikohtaisesti csv-muotoon niin, että on itse laskettava lukujen erotus edelliseltä tunnilta sekä siltä tunnilta, jolta haluaa tietää sisään- ja ulosmenijät. Kun talossa on 7 kävijöitä laskevaa porttia ja tiedostoon tulee aina koko vuoden luvut, tuntikohtaisten lukujen saaminen siihen muotoon, että voisi oikeasti tarkastella kävijälukuja tunneittain, on aika työlästä. Kirjastossa onkin toivottu kätevämpää tapaa tarkastella tuntikohtaisia tilastoja.

Periaatteessa olisi mahdollista esittää tiedot suoraan MikroVäylän kannasta omassa käyttöliittymässä, mutta tietojen saaminen haluttuun muotoon edellyttää paljon laskentaa (jolloin sivun latautuminen olisi hyvin hidasta), joten päädyin siihen, että tallennan tiedot vähän helpommassa muodossa omaan kantaan.

Kerran kuukaudessa haetaan ajastetusti edellisen kuukauden tiedot niin, että lasketaan jokaiselle tunnille sisään ja ulos kulkeneiden määrät. Näitä ei lukuja ei valitettavasti voi käyttää suoraan, sillä portit rekisteröivät säännöllisesti enemmän ulosmenoja kuin sisääntuloja. Toimittajan mukaan tämä johtuu siitä, että portit eivät aina osaa arvioida kulkijoiden suuntaa oikein. Jos tarkastellaan kokonaista päivää (tai pidempää jaksoa), voi tehdä niin, että laskee yhteen sisääntulijat ja ulosmenijät ja jakaa luvun kahdella. Mutta jos haluaa tietää, paljonko talossa on kulloinkin väkeä, tämä ei riitä. Koska ulosmenijöitä on liikaa, luvut olisivat aina liian pieniä, kun lasketaan yhteen tiettyyn ajankohtaan asti sisään tulleet ja vähennetään siitä ulosmenneet. Vuoden 2014 toteuman perusteella laskin sisään tulleille kertoimeksi 1,02971 ja ulos menneille 0,97196. Oman tietokannan tauluun tallennetaan sekä alkuperäiset luvut että kertoimella saadut luvut.

Jokaisen tunnin kohdalle tallennetaan myös talossa olevien lukumäärä sekä tieto siitä, onko kirjasto auki vai ei. Aukiolo päätellään niin, että jos kävijöitä on alle 15 tai kyse on aamutunneista (7, 8, 9, viikonloppuisin myös 10 ja 11) kirjasto lasketaan suljetuksi, muuten avoimeksi. Kirjasto on avoinna arkisin 9-20 ja lauantaisin ja sunnuntaisin 11-17 (sunnuntait vain kiireisimpinä aikoina). Erityisesti arkiaamuisin kirjastoon tulee suuri joukko henkilökuntaa töihin aamuisin, joten ne olen yrittänyt näin rajata pois tuloksista.

Myös aukioloaikojen ulkopuolella talossa liikkuu jonkin verran kirjaston henkilökuntaa ja vartioita, mutta ne jäävät yleensä alle 15 kävijän rajan. Kirjastossa on satunnaisesti isoja iltatilaisuuksia, ja niiden kävijät halutaan mukaan lukuihin – siksi illalla ei ole aikarajaa.

Joskus käy niin, että lukuja ei ole saatu porteista. Porttien laskurit toimivat silloinkin koko ajan, ainoastaan tiedonsiirrossa on välillä ongelmia, eli kun luvut taas saadaan, yhdelle tunnilla kirjautuu monen tunnin (pahimmassa tapauksessa jopa usean päivän) kävijät. Usein katkot ovat lyhyitä ja sattuvat öisin, jolloin niistä ei ole haittaa. Ohjelma, joka hakee tiedot MikroVäylän kannasta kirjoittaa logitiedostoon ne tunnit, joilta ei ole kirjauksia.

Käyttöliittymässä tuntikohtaisia tilastoja voi tarkastella kolmella tavalla:

- yksittäisen päivän tasolla
- valitun vuoden osalta keskimääräiset kävijämäärät viikonpäivittäin ja tunneittain
- valitun kuukauden osalta keskimääräiset kävijämäärät viikonpäivittäin ja tunneittain.

Päiväkohtaisen tilaston kohdalla valitaan ensin vuosi (tarjotaan niitä vuosia, joilta on kirjauksia), sen jälkeen pääsee valitsemaan kuukauden (tarjotaan niitä kuukausia, joilta on kirjauksia) ja lopuksi päivän. Valinta on toteutettu Ajaxilla (8), ja ohjelma tarjoaa valittaviksi vain sellaisia päiviä, joilta on merkittyjä aukiolotunteja. Jos kello 16 ei ole merkitty aukiolotunniksi, koko päivä jätetään pois. Jos haluaisi olla varma, että mukaan ei tule lainkaan päiviä, joissa on aukioloaikoina ollut katkoksia lukujen saamisessa, pitäisi katsoa logia ja merkitä kaikki ongelmallisen päivän tunnit sulkupäiviksi. Käytännössä tämä kevyempi tarkistus on toistaiseksi riittänyt.

Kuvassa 14 näkyy syyskuulta 2015 tarjottavat päivät. Kuukauden alussa oli pitkiä katkoksia tietojen saannissa porteista, niin että esim. 1.-4.9. puuttuu listasta.

Kuva 15. Syyskuun 2015 päivät, joista on tuntikohtaiset tiedot kävijämääristä.

Kuvassa 16 näkyvät tuntikohtaiset kävijämäärät maanantaina 7.9.2015.

Kävijöitä kirjastossa tunneittain (valittu päivä) Vuosi: --Valitse--

Tunti	10	11	12	13	14	15	16	17	18	19	20
2015-09-07 maanantai	335	490	622	619	793	837	636	609	319	312	312

Kuva 16. Kävijämäärät kirjastossa yhden päivän aikana. Kello 10 oli siis 335 henkilö kirjastossa jne.

Kuvassa 16 näkyy maaliskuun 2015 keskimääräiset kävijämäärät viikonpäivittäin ja tunneittain. Näitä lukuja voi hyödyntää mm. kun suunnittelee asiakaspalveluvuoroissa kulloinkin tarvittavia henkilömääriä.

Kävijöitä keskimäärin viikonpäivittäin ja tunneittain (kuukausi) | Vuosi: --Valitse-- | Kuukausi: --Valitse-- | Näytä

2015 3	10	11	12	13	14	15	16	17	18	19	20
maanantai, kävijöitä keskimäärin	402	570	619	681	787	826	787	643	529	351	43
tiistai, kävijöitä keskimäärin	464	636	706	754	859	854	784	661	545	344	40
keskiviikko, kävijöitä keskimäärin	458	651	666	721	865	842	786	649	515	350	30
torstai, kävijöitä keskimäärin	415	601	652	685	817	845	788	632	495	324	54
perjantai, kävijöitä keskimäärin	363	534	560	596	710	722	666	525	391	241	45
lauantai, kävijöitä keskimäärin	0	255	365	426	485	472	92	14	0	0	0
sunnuntai, kävijöitä keskimäärin	0	0	0	0	0	0	0	0	0	0	0

Kuva 17. Keskimääräiset kävijämäärät maaliskuussa 2015.

Tuntikohtaisten tilastojen lisäksi kirjastossa kaivattiin myös sellaista näkymää, jossa voisi verrata usean vuoden lukuja kuukausitasolla, puolivuosittain ja neljännesvuosittain. Tämä olisi periaatteessa laskettavissa tuntikohtaisista tilastoista, mutta koska niistä puuttuu tietoja, päätin hakea erikseen vielä kuukausittaiset luvut ja tallentaa ne omaan tauluunsa. Kunkin kävijöitä laskevan portin kohdalla vähennetään kuukauden viimeisestä lukemasta kuukauden ensimmäinen lukema, sisään ja ulos erikseen. Sen jälkeen sisään ja ulos lasketaan yhteen ja jaetaan kahdella. Näistä tiedoista saa nopeasti laskettua luvut koostesivulle, joka näkyy kuvassa 17.

Takaisin hakusivulle

Kaisa-talo 2015	Kävijöitä
Tammikuu	138 667
Helmikuu	154 527
Maaliskuu	167 530
Huhtikuu	163 279
Toukokuu	126 471
Kesäkuu	74 985
Heinäkuu	57 965
Elokuu	92 735
Syyskuu	178 895
Lokakuu	0
Marraskuu	0
Joulukuu	0
Yhteensä	1 155 054

Kaisa-talo 2014	Kävijöitä
Tammikuu	138 644
Helmikuu	148 040
Maaliskuu	154 816
Huhtikuu	145 208
Toukokuu	120 646
Kesäkuu	62 047
Heinäkuu	50 511
Elokuu	84 114
Syyskuu	157 968
Lokakuu	169 918
Marraskuu	153 808
Joulukuu	103 768
Yhteensä	1 489 488

Kaisa-talo 2013	Kävijöitä
Tammikuu	123 546
Helmikuu	137 196
Maaliskuu	134 922
Huhtikuu	140 800
Toukokuu	118 918
Kesäkuu	56 145
Heinäkuu	45 080
Elokuu	77 566
Syyskuu	147 078
Lokakuu	161 290
Marraskuu	153 694
Joulukuu	88 380
Yhteensä	1 384 615

Kaisa-talo 2015	Kävijöitä
Tammi-kesäkuu	825 459
Heinä-joulukuu	329 595
Yhteensä	1 155 054

Kaisa-talo 2014	Kävijöitä
Tammi-kesäkuu	769 401
Heinä-joulukuu	720 087
Yhteensä	1 489 488

Kaisa-talo 2013	Kävijöitä
Tammi-kesäkuu	711 527
Heinä-joulukuu	673 088
Yhteensä	1 384 615

Kaisa-talo 2015	Kävijöitä
Tammi-maaliskuu	460 724
Huhti-kesäkuu	364 735
Heinä-syyskuu	329 595
Loka-joulukuu	0
Yhteensä	1 155 054

Kaisa-talo 2014	Kävijöitä
Tammi-maaliskuu	441 500
Huhti-kesäkuu	327 901
Heinä-syyskuu	292 593
Loka-joulukuu	427 494
Yhteensä	1 489 488

Kaisa-talo 2013	Kävijöitä
Tammi-maaliskuu	395 664
Huhti-kesäkuu	315 863
Heinä-syyskuu	269 724
Loka-joulukuu	403 364
Yhteensä	1 384 615

Kuva 18. Kooste kävijäluvuista eri vuosina.

Toistaiseksi vain keskustakampuksella on verkossa olevat portit. Muualla portit laskevat kyllä kävijöitä (tosin erottelematta suuntaa), mutta luvut pitää käydä katsomassa porteista ja kirjata käsin – näin ollen tuntikohtaisten lukujen ottaminen ei tule kysymykseen. Tarkoitus on tehdä muita toimipaikkoja varten lomake, jolla kävijäluvut syötetään kantaan.

4.3 Aukiolotiedot kirjaston SQLite aukiolokannasta

Kirjaston pitää vuosittain ilmoittaa tieteellisten kirjastojen yhteistilastoon kirjaston toimipaikkojen aukiolopäivät ja aukiolotunnit. Nämä tiedot on saatavissa kirjaston SQLite-aukiolotietokannasta, mutta dataa ei voi näyttää suoraan sieltä, koska jos normaaleja aukioloaikoja muuttaa, se vaikuttaa myös takautuvasti, jolloin aikaisempien kuukausien ja vuosien tiedot vääristyvät. Aukiolotietokantaa käytetään esittämään ajankohtaiset ja tulevat aukioloajat kirjaston verkkosivuilla. Siksi tiedot haetaan Libstatille aukiolotietokannasta kuukausittain ja tallennetaan omaan tietokantaan.

Aukiolotietokantaan on avoin rajapinta, joka tarjoaa HTTP-protokollan GET-metodilla toimipaikkojen aukioloajat. Tietoja voi pyytää enintään 31 päivältä kerralla, ja vastaus toimitetaan JSON-formaatissa. Esim. pyynnöllä

```
http://hulib.hulib.helsinki.fi/openhours/list?start_day=2015-01-01&end_day=2015-01-31&office=1
```

saa Keskustan toimipaikan (office=1) aukiolotiedot tammikuulta 2015. Jokaista päivää kohti palautetaan seuraavan muotoinen JSON-objekti:

```
{
  "office": 1
  "type": "exception",
  "day": "2015-01-02",
  "open_time": "09.00",
  "close_time": "18.00"
}
```

Kirjastossa on vuoden 2015 syksyllä kokeiltu sunnuntaisin uudenlaista aukiolotapaa, jossa kirjaston tilat ovat asiakkaiden käytettävissä, henkilökunta ei päivystä tiskillä. Kirjastossa on kuitenkin pieni määrä henkilökuntaa, joka voi auttaa ongelmatilanteissa ja tarvittaessa tyhjentää talon. Aukiolotietokannassa tällaista aukioloa ei eroteta

tavallisesta, täyden palvelun aukiolosta. Siksi ohjelma, joka hakee tiedot aukiolotietokannasta lukee ensin tiedoston, jossa kerrotaan rajatun aukiolon ajat. Omaan tietokantaan tallennetaan sitten aukiolopäivien ja -minuuttien lisäksi minuutit, jolloin on ollut rajattu aukioloaika.

Java-ohjelmassa, joka hakee ja tallentaa tiedot, käytetään org.json-paketin luokkia JSONin käsittelyyn ja Apachen tarjoamia commons- sekä http-paketteja http-pyyntöjen käsittelyyn.

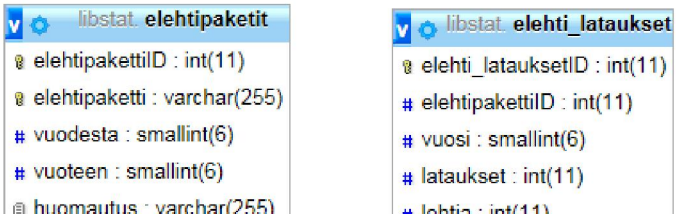
5 Tietojen hakeminen manuaalisesti ja syöttäminen ohjelman avulla

On myös tapauksia, joissa täysin automaattinen tietojen keräily ei ole mahdollista, mutta ohjelmalla voidaan kuitenkin helpottaa tietojen syöttämistä tietokantaan.

Kirjaston elektronisten aineistojen käytöstä saadaan tietoa aineistoja tarjoavien kustantajien omista portaaleista, joihin ei ole tarjolla sellaisia rajapintoja, että tietoja voisi hakea automaattisesti. Joskus tilastoja lasketaan myös eri tavoilla, niin että eri palveluista saadut luvut eivät ole yhteismitallisia, esim. e-kirjojen kohdalla jotkut laskevat avatut kirjat ja toiset avatut luvut. Pahimmillaan kirjasto ei edes tiedä tarkalleen millä tavalla käyttötilastot lasketaan. Kirjastolle on kuitenkin erittäin tärkeää saada tietoa hankittujen aineistojen käytöstä. Tässä työssä esitetään ainoastaan suurten e-lehtipakettien vuositilastoa.

Kirjaston saatavuuspalvelut tuottavat vuosittain Excelin, jossa on lehtipaketin nimi, artikkelilatausten määrä, lehtien määrä sekä mahdolliset huomautukset. Eri paketeissa voi olla päällekkäisyyttä, eli sama lehti voi esiintyä useammassa lehtipaketissa. Tiedostosta poistetaan otsikot ja rivi, jolla on yhteenlasketut tiedot. Tiedosto tallennetaan UTF8-muotoon ja siirretään palvelimelle samaan kansioon kuin ohjelma, joka sen lukee. Ohjelma käynnistetään komentokehotteesta ja parametriksi annetaan vuosi, jolta tiedostossa olevat luvut ovat.

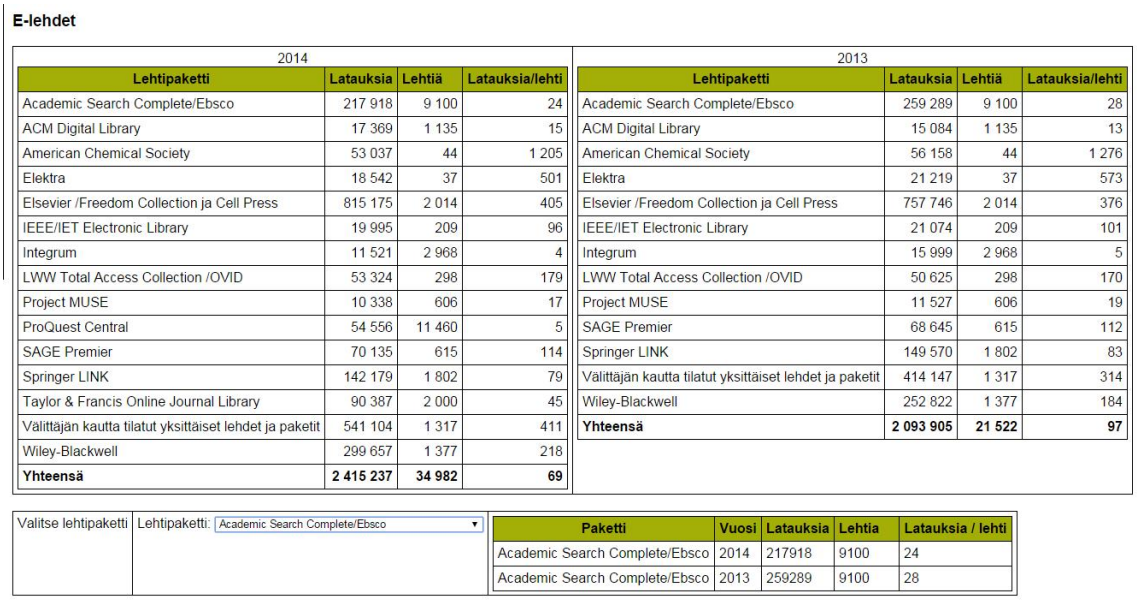
Tietokannassa on kaksi taulua, yksi e-lehtipaketteja varten ja toinen latauksia varten (ks. kuva 18).



Kuva 19. E-lehtipaketteihin liittyvät taulut.

Ohjelma, joka lukee csv-tiedoston, tarkistaa ensin paketin nimen perusteella, onko paketti jo elehtipaketit-taulussa. Jos ei ole, niin paketti lisätään, ja lopuksi lisätään rivi tauluun elehti_lataukset. Tiedostossa olevasta e-lehtipaketin nimestä poistetaan mahdolliset välilyönnit alusta ja lopusta, mutta muuten paketin nimen pitää olla täsmälleen oikein kirjoitettu, jotta ei tulisi tuplia. Koska koodia tuli alle 100 riviä, en katsonut aiheelliseksi erottaa tietokantaoperaatioita ja tiedoston lukemista omiin luokkiinsa.

Sivulla, jossa näytetään e-lehtipakettien tilastot, näytetään ensin kultakin vuodelta paketit, lataukset, lehtien määrän sekä latauksia / lehti. Tämän alla on valikko, josta voi valita paketin, jolloin saa kyseisen paketin tiedot kaikilta vuosilta. Yksittäisen paketin valinta on toteutettu Ajaxin avulla niin, että ei tarvitse päivittää koko sivua, sillä pienehköllä näytöllä voi olla, että tieto, jonka käyttäjä on juuri valinnut, jäisi vuositaulukoiden alle.



E-lehdet

2014

Lehtipaketti	Latauksia	Lehtiä	Latauksia/lehti
Academic Search Complete/Ebsco	217 918	9 100	24
ACM Digital Library	17 369	1 135	15
American Chemical Society	53 037	44	1 205
Elektra	18 542	37	501
Elsevier /Freedom Collection ja Cell Press	815 175	2 014	405
IEEE/IET Electronic Library	19 995	209	96
Integrum	11 521	2 968	4
LWW Total Access Collection /OVID	53 324	298	179
Project MUSE	10 338	606	17
ProQuest Central	54 556	11 460	5
SAGE Premier	70 135	615	114
Springer LINK	142 179	1 802	79
Taylor & Francis Online Journal Library	90 387	2 000	45
Välittäjän kautta tilatut yksittäiset lehdet ja paketit	541 104	1 317	411
Wiley-Blackwell	299 657	1 377	218
Yhteensä	**2 415 237**	**34 982**	**69**

2013

Lehtipaketti	Latauksia	Lehtiä	Latauksia/lehti
Academic Search Complete/Ebsco	259 289	9 100	28
ACM Digital Library	15 084	1 135	13
American Chemical Society	56 158	44	1 276
Elektra	21 219	37	573
Elsevier /Freedom Collection ja Cell Press	757 746	2 014	376
IEEE/IET Electronic Library	21 074	209	101
Integrum	15 999	2 968	5
LWW Total Access Collection /OVID	50 625	298	170
Project MUSE	11 527	606	19
SAGE Premier	68 645	615	112
Springer LINK	149 570	1 802	83
Välittäjän kautta tilatut yksittäiset lehdet ja paketit	414 147	1 317	314
Wiley-Blackwell	252 822	1 377	184
Yhteensä	**2 093 905**	**21 522**	**97**

Valitse lehtipaketti | Lehtipaketti: Academic Search Complete/Ebsco

Paketti	Vuosi	Latauksia	Lehtia	Latauksia / lehti
Academic Search Complete/Ebsco	2014	217918	9100	24
Academic Search Complete/Ebsco	2013	259289	9100	28

Kuva 20. E-lehtipakettien tilastot.

Jatkossa on tarkoitus tutkia, miten voisi koostaa järkeviä tietoja e-kirjojen käytöstä ja mahdollisesti myös yksittäisten lehtien käytöstä.

Puoliautomaattisesta tallennustavasta ehti tähän työhön ainoastaan yksi esimerkki, mutta seuraava kohde on jo tiedossa: joulukuussa 2015 käyttöönotetusta vuoronumerojärjestelmästä saa csv-muodossa tietoa mm. siitä, paljonko tiskillä on käynyt asiakkaita ja kauanko he ovat joutuneet jonottamaan. Tätä dataa on tarkoitus tuoda omaan kantaan ja esittää käyttöliittymässä.

6 Loppusanat

Kuten johdannossa todettiin, nyt luotu järjestelmä ei ole valmis, vaan sitä on tarkoitus edelleen kehittää. Tärkein alue, jolta tarvitaan vielä lisää tietoja on varmasti elektronisten aineistojen käyttö. Osaa tiedosta, joka menee kantaan, ei vielä ole mahdollista tarkastella käyttöliittymässä (esim. maksuihin liittyvät tiedot). Käyttöliittymään voisi myös lisätä aikasarjoja ja graafeja sitä mukaa. kun tietoa kertyy enemmän.

Lähteet

1 Helsingin yliopiston kirjaston verkkosivut
http://www.helsinki.fi/kirjasto/fi/yhteystiedot/kirjaston-toiminta/ (luettu 2.7.2015).

2 Kirjastojen kansallinen käyttäjäkysely 2010 – Helsingin yliopiston kirjaston tulokset, http://wiki.helsinki.fi/display/opiskelijatoimikunta/Opiskelijatoimikunta, Liite 3 (luettu 2.7.2015).

3 Koufogiannakis, Denise & Crumley, Ellen, ”Evidence-Based Librarianship”, Feliciter 2002, vol 48, p. 112.

4 Balukoff, Stephen, ”Getting out of MySQL Character Set Hell”, blogimerkintä 2009, https://www.blueboxcloud.com/insight/blog-article/getting-out-of-mysql-character-set-hell (luettu 2.7.2015).

5 Doctrine 2 ORM documention, http://doctrine-orm.readthedocs.org/en/latest/index.html (luettu 3.7.2015).

6 Doctrine 2 ORM API, http://www.doctrine-project.org/api/orm/2.4/namespace-Doctrine.ORM.html (luettu 3.7.2015).

7 Høivik, Tord, ”Count the traffic”, IFLA 2008, http://es.akronlibrary.org/wp-content/blogs.dir/19/files/2011/07/Count-the-traffic.pdf (luettu 5.7.2015).

8 Wikipedia-artikkeli “Ajax”, https://en.wikipedia.org/wiki/Ajax_(programming) (luettu 12.10.2015).