LAUREA
AMMATTIKORKEAKOULU
Yhdessä enemmän

QlikView-raportointisovelluksen luominen

Mella, Marko

2016 Laurea

Laurea-ammattikorkeakoulu
Leppävaara

QlikView-raportointisovelluksen luominen

Marko Mella
Tietojenkäsittely
Opinnäytetyö
Huhtikuu, 2016

Laurea-ammattikorkeakoulu **Tiivistelmä**
Laurea Leppävaara
Tietojenkäsittelyn koulutusohjelma

Marko Mella

QlikView-raportointisovelluksen luominen

Vuosi 2016 Sivumäärä 26

Tämän toiminnallisen opinnäytetyön tavoitteena on rakentaa raportointisovellus kohdeorganisaatiolle. Tuloksena syntynyt sovellus tulee olemaan organisaation asiantuntijoiden jatkuvassa käytössä. Sovellus nopeuttaa huomattavasti asiantuntijoiden tiedonkeruuta tarvitsemistaan asioista.

Opinnäytetyöni toimeksiantaja käyttää QlikView-raportointisovellusta Business Intelligence raportointityökalunaan. Raportointisovellusta lähdettiin tekemään eri tavalla kuin olisi tarkoitus. Riskejä oli paljon, mutta niistä oli rakennettu riskianalyysi, joten ne olivat tiedossa. Sovelluksen rakentaminen lähti aivan alusta, joten raportointitarpeet piti selvittää rakentamisen yhteydessä. Sovelluksen rakentamisen aikana tietolähteenä oli saatavilla ainoastaan testiaineistoa, mutta tulevaisuudessa tietolähteeksi vaihdetaan rakenteilla oleva tietovarasto.

Työn teoriaosuudessa esitellään Business Intelligencen hyötyjä sekä esitellään raportointisovelluksen luomisen lähtökohta. Teknillisessä osassa toteutetaan itse sovellus. Toteutuksessa paneudutaan sovelluksen pohjan rakentamiseen sekä skriptiin, eikä keskitytä lopulliselle käyttäjälle näkyvien raportointitaulujen tekemiseen.

Opinnäytetyön tuotoksena organisaatiolle on QlikView Deployment Framework ympäristö sekä kaksi QlikView-sovellusta. Toisen sovelluksista on tarkoitettu tuottamaan tiedostoja, joita toinen sovellus käyttää tietolähteenään.

Tekijällä ei ollut aikaisemmin lainkaan kokemuksia QlikView ohjelmasta. Tämä opeteltiin täysin alusta. Projektissa mukana ollut QlikViewin konsultti sekä Qlikin oman yhteisön sivustot toimivat erittäin hyvänä apuna. Työn tuotosta pidetään organisaatiossa suurena apuna ja oltiin positiivisesti yllättyneitä aikaansaannoksista.

Asiasanat: QlikView, Business Intelligence, Raportointi

Laurea University of Applied Sciences **Abstract**
Laurea Leppävaara
Bachelor's Degree Programme in Business Information Technology

Marko Mella

Creating QlikView reporting tool

Year 2016 Pages 26

The purpose of this functional thesis was to create a reporting application to the commissioner. The application will be in daily use of the professionals and it will speed up data collection significantly.

The commissioner of this thesis uses QlikView reporting application as its Business Intelligence reporting tool. We started creating the application differently as it should be done. There were several risks, but as we made a risk analysis, they were considered. We started this process from the beginning so we had to figure out all the reporting requirements during the creation of this application. The database of this application was filled with test data, but in the future the database will be a data warehouse.

The theoretical part of this thesis covers the benefits of the Business Intelligence and QlikView Deployment Framework. I will also introduce the starting point of this project. In the technical part I will go into the groundwork of the application and to the script. I will not focus on the reports that will be shown to the final user of this application.

The result of this thesis to the organization is QlikView Deployment Framework architecture and two QlikView applications. The purpose of the first application is to create data files that the second application will use as a data source.

The author who created this thesis had no earlier experience with QlikView application. QlikView's own consultant and QlikView community website were very good help during the project. These applications were created with the consultant. The final application is regarded as a very useful tool and the users were very positively surprised about the results.

Keywords: QlikView, Business Intelligence, Reporting

Sisällys

1 Johdanto

Yrityksillä on hallussaan suuria määriä dataa, mutta kukaan ei tiedä, että mitä kaikkea. Tiedostoja on monissa eri lähteissä ja eri kansioissa, eikä kukaan oikein tiedä, mistä mitäkin löytyy. Miten saataisiin kaikki tieto irti ja hyötykäyttöön? Business Intelligence työkaluilla saadaan varsin hyvin data esille. Varsinkin yrityksen johdolle on tärkeää saada vaivattomasti ja nopeasti tieto yrityksen kehityksen suunnasta.

Opinnäytetyössäni keskityn yhteen BI-ratkaisuun, QlikView-raportointisovellukseen, joka toteutetaan kohdeorganisaatiolle. Työssä keskitytään siihen, miten tieto ladataan sovelluksen käyttöön ja mitä toimenpiteitä sovellukseen tarvitsee tehdä, jotta eri raportteja ja analyysejä saadaan yrityksen käyttöön.

Opinnäytetyöni jää kohdeorganisaation käyttöön, sillä se toimii myös jatkoa ajatellen organisaatiossa oppaana koskien QlikView ympäristöä sekä sovellusta.

1.1 Sanasto

Opinnäytetyössäni käytetään paljon lyhenteitä. Varsinkin QlikView-sovelluksen tiedostoista käytettäviä lyhenteitä on hyvä tarkentaa.

QlikTech	Ruotsalainen ohjelmistoyritys
QlikView	QlikTechin luoma raportointiohjelma
QWV	Kehittäjän tekemä QlikView-sovellus.
QVD	QlikView tiedosto, jotka tehdään ennen lopullista QWV sovelluksen tekoa. QVD:hin ladataan tietokannasta data, joita QWV käyttää lähteenään.
BI	Business Intelligence

1.2 Kohdeorganisaatio

Kohdeorganisaatiossa työskentelee noin 150 henkilöä. Organisaatiolla on kaksi eri toimipaikkaa Suomessa. Organisaatiolla on käytössään paljon ehdottomasti seurattavaa tietoa. Organisaatio on ottanut käyttöönsä QlikView-raportointisovelluksen Business intelligence sovellukseen.

1.3 QlikTech

QlikTech on Ruotsissa vuonna 1993 perustettu tietokoneohjelmistoyritys. Sillä on noin 36 000 asiakasta yli 100:ssa maassa sekä maailmanlaajuisesti yli 2000 työntekijää.

QlikView on QlikTechin luoma Business Intelligence-sovellus. Sovellus mahdollistaa käyttäjälle datan keräämisen eri tietolähteistä, kuten esimerkiksi SQL- palvelimesta, Oraclesta ja myös esimerkiksi Excel-tiedostoista.

Sovelluksen avulla käyttäjä voi tarkastella käytössä olevia tietoja monilla eri tavoilla. Raportteja voi tehdä erilaisina graafeina, tauluina, diagrammeina ja kaavioina. Sovelluksella voi myös tehdä trendianalyyseja, mitkä ovat tärkeitä tietoja varsinkin organisaation johdolle. Tiedot sekä graafit ja kuviot ovat myös ladattavissa sovelluksesta esimerkiksi Excel- tai PDF-tiedostoina.

QlikViewista on ilmainen ”Professional edition” versio. Siinä on täydet toiminnallisuudet, mutta käyttäjä voi avata vain omalla koneellaan tehtyjä sovelluksia. Muiden tekemiä sovelluksia ei siis ole mahdollista saada auki. Muuten QlikView toimii lisensseillä, joita on kahdenlaisia. Toinen on Document CAL-lisenssi, mikä on sovelluskohtainen lisenssi. Yhdellä Document-CAL lisenssilä käyttäjä saa käyttöönsä yhden raporttisovelluksen. Named user CAL-lisenssi on laajempi ja lisenssin haltija saa käyttöönsä rajattoman määrän sovelluksia.

2 Business intelligence

Business intelligence ratkaisujen avulla yritysten ja julkisten organisaatioiden henkilöstö pääsee käsiksi toimintaa kuvaavaan informaatioon (Hovi, s.74) Kerätty ja analysoitu informaatio auttaa organisaatiota kehittämään toimintaansa sekä tekemään valistuneempia päätöksiä. Käyttäjän ei tarvitse tuntea lainkaan lähdejärjestelmää eikä tietokantojen rakenteita, vaan sovelluksen avulla tieto on helposti saatavilla sekä käytettävissä. BI-ratkaisujen avulla tietoa voidaan tietojärjestelmistä kerätä, muokata ja siirtää organisaation käyttöön. Nykyään tietoa kerätään enemmän kuin sen hyödyntämiseen olisi tarvetta, sillä tallennuskapasiteetit ovat nykypäivänä niin suuria. Kohdeorganisaatiossani tietoa lisätään päivittäin ja tämän tiedon on oltava mahdollisimman reaaliaikaista.

3 Raportointisovellus

Opinnäytetyössäni rakennan raportointisovellusta kohdeorganisaatiolle. Tein sovelluksen osana työharjoittelua. QlikView-sovelluksen tekemisessä avusti heidän oma konsulttinsa. Projekti eteni työpajoina, joissa kävimme läpi hankalia asioita ja teimme toimeksiantoja seuraavaan työpajaan. Konsultin kanssa työskentely oli erittäin sujuvaa. Suurena apuna raporttien tekemiseen toimi myös QlikViewin oma community-sivusto, joka on yli 100 000:n käyttäjän yhteisö. Yhteisössä käyttäjät kommunikoivat keskenään ja ratkovat toistensa ongelmia.

Organisaatioon oli jo aikaisemmin tehty QlikView-sovelluksia, mutta sovelluksien tekijät olivat vaihtaneet jo työpaikkaa. Näiden sovelluksien ympärille ei ollut rakennettu Deployment Fra-

mework-ympäristöä, joten tämä tuli koko organisaatiolle uutena asiana. Tässä opinnäytetyössäni selvitän myös tämän ympäristön etuja ja käytettävyyttä.

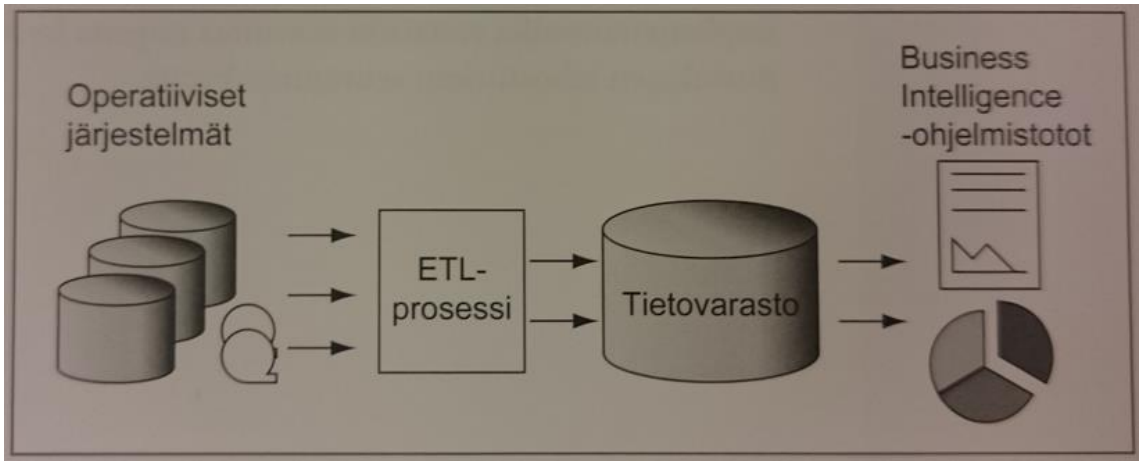
3.1 Raportointitarpeet

Sovelluksen raportointitarpeet tulivat raportin tulevilta käyttäjiltä. Heille oli hieman epäselvää, mitä raportoitavaa koko sovellukseen oli tarkoitus rakentaa, joten tarpeita tuli lisää pitkin sovelluksen tekemistä. QlikView-ohjelmakaan ei ollut kovin tuttu tuleville käyttäjille, joten he eivät tienneet, millaisia raportteja sovellukseen olisi mahdollista tehdä. Tämä hankaloitti hieman projektia. Raportointitarpeita tuli ilmi myös demosovelluksen esittelyissä, kun jotain tarpeellista huomattiin puuttuvan. Raportointitarpeiden lisäksi tehtyjä ratkaisuja ja raportteja todettiin käytännöllisiksi ja näitä haluttiin jättää käyttöön.

Raportin päätehtävänä on tarkastella historiatietoja sekä nykyhetkeä, eikä tulevaisuuden analysointi ollut tarpeellista. Tärkeää oli myös saada vuosivertailua nykyhetkestä edellisiin vuosiin. Oleellista oli myös saada sovelluksesta valmiita listoja käyttäjän valitsemilla ehdoilla.

3.2 Käytössä oleva data

Sovelluksen rakentamisen ajan käytössäni oli ainoastaan testidata, mikä ladattiin PostreSQL-relaatiotietokanasta. Organisaatiossa oli tulossa myöhemmin käyttöön tietovarasto, joka tulee jatkossa olemaan raportin tietolähteenä. Kuvassa 1 nähdään tiedon kulku tietovaraston ollessa käytössä. Tietovaraston tullessa käyttöön data tulee operatiivisista tietolähteistä tietovarastoon, joista sovellus tulee hakemaan tiedon. Operatiivisiin järjestelmiin käyttäjät lisäävät dataa päivittäin.



Kuva 1 Tietovarasto ja BI-ratkaisun jalostusketju

Testidata hankaloitti hieman prosessia, sillä osa datasta oli hyvin epämääräisessä muodossa. Raporttia tehdessä ei ollut varmaa, ovatko sovelluksessa tehdyt mäppäykset turhia. Taulujen

kentistä osa oli myös tyhjänä, eli ei sisältänyt lainkaan testidataa, joten oli mahdoton tietää, missä muodossa tieto tulisi raporttiin. Suurin ongelma testidatan käyttämisessä oli tietokannan muuttuminen. Tietokanta ei ollut aloittaessa täysin valmis, sillä raportoitava aineisto on alun perin tarkoitettu vain tietojen keräämiseen. Raportointi ei ole ollut alun perin tarkoituksena. Tästä johtuen tietokanta muuttui ja päivittyi kesken projektin. Tämä vaikutti QVD-sovelluksen luomisessa siten, että QVD-tiedostoja piti päivitellä kesken projektin ja se vaikutti myös lopulliseen sovellukseen. Kannan olisi hyvä olla valmis ennen sovelluksen aloittamista, muuten raportin tekeminen on työläämpää, koska joitakin asioita joutuu tekemään useampaan kertaan. Tämä oli riskinä tiedossa jo ennen projektia.

4 Qlikview Deployment Framework

Organisaatiolle tuli uutena asiana QlikView Deployment Framework. Sen tarpeellisuus aiheutti hieman kummastusta organisaatiossa, sillä ympäristön etuja ei täysin hahmotettu. Kappaleessa selvitän ympäristön idean ja sen edut sovelluksien kehittäjälle.

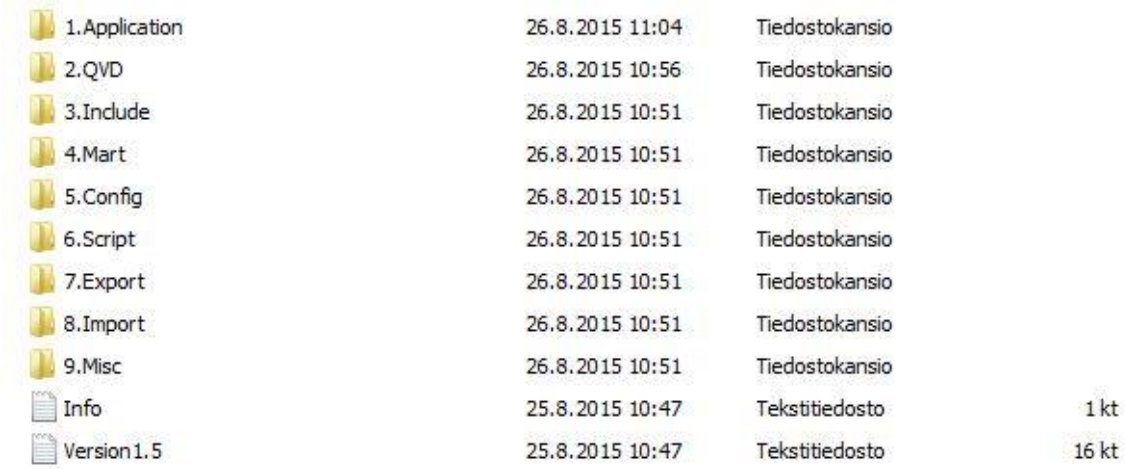
QlikView Deployment Framework on ympäristö, johon QlikView-tiedostot on mahdollista sijoittaa. Ympäristö on tehty mahdollisimman yksinkertaiseksi ja tekee sovelluksien tekemisestä sekä sovellusympäristön käytöstä huomattavasti käyttäjäystävällisempää. Se ei käytännössä ole erillinen sovellus, vaan oma ympäristö sekä valmiita skriptimalleja ympäristön toimimiseen. QlikView Deployment Framework on saatavilla QlikViewin yhteisöstä ja saatavuuden ehtona on Qlik Communityn jäsenyys. QlikView Deployment Framework ei ole QlikTechin tuote, vaan se on QDF ympäristön jäsenien tukema. Yhteisöön kuuluu QlickTechin työntekijöitä, yhteistyökumppaneita, asiakkaita ja harrastajia.

4.1 QDF ympäristö

Kun Deployment Framework ympäristö on asennettu tietokoneelle, syntyy käyttäjälle kuvan 2 mukainen ympäristö.

Kuva 2QDF ympäristö

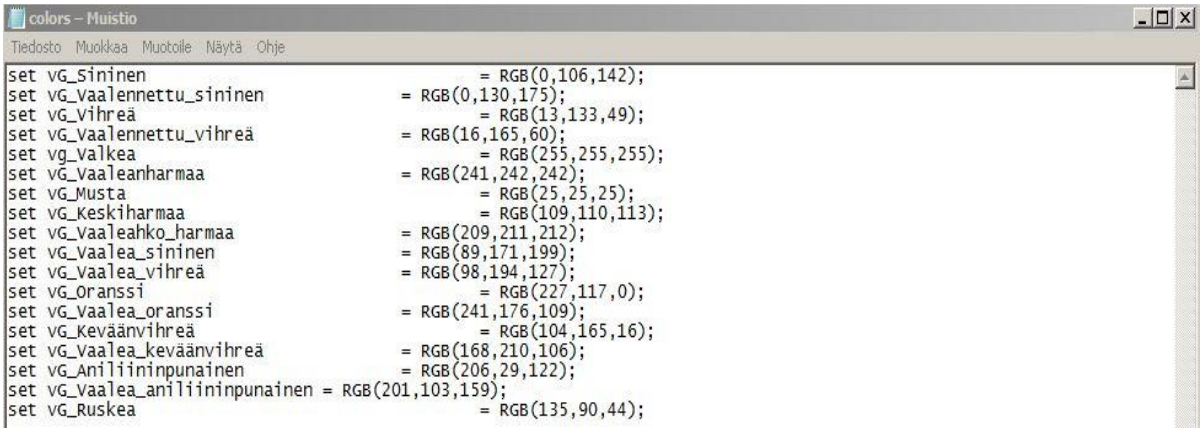
Vihreällä kuviolla olevat kansiot ovat sijainteja, mihin sovelluksia sijoitetaan. Tässä tapauksessa asetimme sijainnit 01.Tuotanto ja 01.Testi nimisiksi. Testi ympäristössä on tarkoituksena tehdä muokkauksia, joiden jälkeen valmis sovellus siirretään tuotantokansioon. Tuotanto kansiossa on tarkoitus säilyttää pelkästään valmiita sovelluksia ja tiedostoja. Näin sovellukset on helppo asentaa tuotantoympäristöön polkuja muuttamatta. Sijainteja voi myös käyttää niin, että erittelee jokaisen projektin tai sovelluksen eri kansioon, jolloin jokaisessa kansiossa on vain tietyn projektin tiedostot. Example kansiossa on esimerkkisovelluksia, skriptejä sovelluksiin sekä tiedostoja selventääkseen käyttäjälle, miten ympäristö toimii.
Valittaessa kansion, mihin sovellusta rakennetaan, ilmestyy kuvan 3 mukainen ympäristö.



Kuva 3 QDF ympäristö

Tämä ympäristö on samanlainen, riippumatta valitseeko minkä tahansa vihreän kansion. Kuvan mukaisesti ympäristössä on yhdeksän eri kansiota ja jokaisella kansiolla on oma tarkoituksensa. Tässä työssä käytettiin seuraavia kansioita:

- 1.Application
  - Kansioon on tarkoitus sijoittaa QWV sovellukset, eli sovellukset, mitä luodaan.
- 2.QVD
  - Kansioon sijoitetaan QVD-luonti-sovelluksella tehdyt QVD-tiedostot.
- 3.Include
  - Kansioon sijoitetaan muun muassa variaabeleja, kieliasetuksia ja värikoodeja. Esimerkiksi värikoodit voidaan tehdä valmiiksi kuvan 4 mukaisesti muistioon ja sijoittaa se oikeaan kansioon.
- 8.Import
  - Kansioon sijoitetaan esimerkiksi mahdolliset Excel tiedostot, mitä sovellus käyttäisi lähteenään.



Kuva 4 Värivariaabelin muodostaminen muistioon

Jokaisella kansiolla on oma sijaintilyhenteensä, joita käytetään sovelluksessa. Tämä tarkoittaa, että QlikView-sovellus hakee lyhennettä käyttäen datan kyseisen ympäristön sisältä. Tämä säästää tiedostojen sijaintien kirjoittamisen sovellukseen. Tämä mahdollistaa esimerkiksi eri sovelluksien muokkaamisen vaivattomasti, kun kaikki ovat saman tiedostopolun alla. Sovellukset voidaan myös siirtää esimerkiksi tietokoneelta toiselle, kunhan ympäristö on toisessakin koneessa käytössä. Tässä projektissa käytetyn Testi-kansion sovellukset hakevat datan testikansiosta näillä lyhenteillä, mutta jos testi-kansiosta kopioi sovelluksen tuotanto-kansioon, hakee sovellus datan tuotanto-kansiosta automaattisesti, kunhan lähdetiedostot ovat myös sijoitettu sinne.

4.2 QlikView Deployment Frameworkin hyödyt

Raportointisovelluksia ja QVD-tiedostoja tuottaessa ympäristö ja ympäristön hallinta monimutkaistuu, mitä enemmän sovelluksia tehdään. Tälläisia ympäristöjä on vaikea ylläpitää varsinkin, jos yrityksessä työntekijät vaihtuu. QlikView Deployment Framework on käytännössä vain struktuuri tiedostojen säilyttämiseen ja hallitsemiseen. Se tuo jokaiselle tiedostolle ja toiminnolle oman paikkansa. Sovelluksia ja tiedostoja siirtäessä testikansiosta tuotantokansioon ei tarvitse tehdä mitään skriptimuutoksia, sillä ympäristön rakenteet ovat identtisiä. Ympäristö ei tuo itse sovelluksiin käytännössä mitään uutta, mutta se helpottaa sovelluksen hallintaa. Raporttien lopullinen käyttäjä ei siis huomaa mitään eroa, onko ympäristö käytössä vai ei.

5 Toteutus

Tässä kappaleessa käydään läpi sovelluksien toteutus sekä siinä käytettyjä menetelmiä. Raporttien tekemistä ei käydä läpi, vain sovelluksen sisällä, eli skriptissä tehtyjä asioita.

Ennen sovelluksien tekemistä tulisi olla pohjatyö mahdollisimman hyvin tehty. Sovelluksen kehittäjällä tulisi olla tiedossa, mitä käyttäjät haluavat sovellukseen ja mitä tietoa sovellukseen ei tarvitse. Myös tietokannan olisi hyvä olla valmiina, jotta jälkeenpäin ei tarvitse tehdä muutoksia sovelluksiin, eikä turhaa työtä tulisi.

Tässä projektissa lähtökohdat poikkesivat ideaalitilanteesta melko paljon. Aloimme rakentaa sovellusta ennen kuin tiesimme täysin, mitä sovellukseen tulisi tehdä. Raportointitarpeet eivät olleet täysin tiedossa, sillä käyttäjätkään eivät tienneet, minkälaisia raportteja sovellukseen voisi tehdä. Teimme sovellukseen paljon omia ideoita ja arvelimme, mikä voisi olla hyvä tulevan käyttäjän kannalta. Testidata oli hieman puutteellista ja sen päivittyminen aiheutti lisätyötä. Sovelluksen tekeminen päätettiin kuitenkin aloittaa näistä puutteista huolimatta, sillä oppimiseni kannalta tärkeää aloittaa mahdollisimman nopeasti sovelluksen tekeminen. Testidata muuttui kuukausia aloittamisen jälkeenkin, joten muuten aloittaminen olisi viivästynyt huomattavasti. Lisäksi oli helpompi reagoida käyttäjien antamiin raportointitarpeisiin, kun tunsi enemmän QlikView-sovellusta ja sen tuomia mahdollisuuksia.

5.1 Sovelluksen luominen

Organisaation tarvitseman raportointisovelluksen luomisessa käytetään kahta eri sovellusta. Ensimmäiseksi tehdään QVD-generaattori-sovellus, jolla luodaan tietokannasta QVD-tiedostot. Tämän jälkeen tehdään varsinainen raportointisovellus käyttäjälle.

Kuva 5 Arkkitehtuuri ilman tietovarastoa

Kuvan 5 mukaisesti tarvitaan tietokantayhteys. Tässä tapauksessa käytössä oli testitietokanta. Kun tietokantaan on saatu yhteys, voidaan aloittaa QVD-tiedostojen tekeminen QVD-generaattorilla, mikä tulee olemaan oma tekemä QlikView-sovellus. Tämä sovellus ei näy lopulliselle käyttäjälle. QVD-tiedostojen valmistuttua rakennetaan itse raportointisovellus.

5.1.1 QVD-generaattori-sovelluksen luominen

QVD-generaattori-sovelluksen luomisessa rakennetaan pohja koko raportointisovellukselle. QVD-generaattorissa luodaan alkuperäisestä tietolähteestä saatavilla olevista tiedoista QVD-tiedostoja, joita lopullinen raportointisovellus käyttää tietolähteenään. QVD-tiedostot ovat

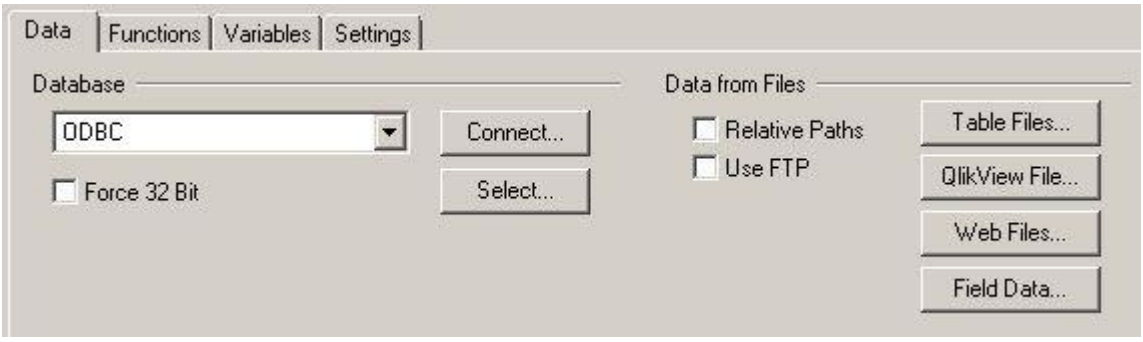
yksittäisiä tiedostoja ja niitä voidaan avata myös yksittäin QlikView-sovelluksella. QVD-tiedostot päivittyvät käyttäjän haluamin väliajoin ja tämän jälkeen ladataan tiedostoihin pohjautuva raportointisovellus. (qlikview scripting, s.78)

QVD-tiedostojen suurimpana etuna on niiden nopeus verrattuna lataukseen suoraan tietolähteestä. Kun käyttäjä haluaa muokata sovellusta ja skripti täytyy ladata uudestaan, on lataus huomattavasti nopeampaa sovelluksen ladatessaan tiedot QVD-tiedostoista, kuin että se lataisi ne suoraan tietokannasta. QVD-tiedostojen etuna on myös niiden uudelleenkäytettävyys, sillä tiedostoja voi käyttää toisissakin sovelluksissa.

QVD-generaattori-sovelluksen latausskriptiin asetetaan ensimmäiseksi QlikView Deployment Framework asetukset, jotta koko arkkitehtuuri toimisi. Nämä löytyvät QDF asennuskansioista. Asetukset määrittelevät esimerkiksi lyhenteiden toimivuuden sovelluksessa sekä koko arkkitehtuurin toimivuuden.

5.1.2 Tietokantayhteys

Tietokantayhteys muodostetaan kohdasta Database valitsemalla haluttu yhteys, jonka jälkeen painetaan Connect. Opinnäytetyöni tapauksessa muodostetaan ODB-yhteys. ODBC-yhteys on avoin rajapinta tietokannoille. Se käyttää ajuria tietokannan ja ohjelmiston välissä. Seuraavaksi valittavaksi tulee mahdolliset ODBC-tietokantayhteydet ja niistä valitaan haluttu tietokanta.



Kuva 6 Tietokantayhteyden muodostaminen

Tietokantaan otetaan yhteys ennen varsinaisen skriptin aloittamista ja se ilmestyy kuvan 7 rivin 39 mukaisesti, kun yhteys on valittu.

```
38 // Tietokantayhteys
39 ODBC CONNECT32 TO (tietokannan nimi);
```

Kuva 7 Tietokantayhteys muodostettu

5.1.3 Sovelluksen skripti

Sovelluksen skripti tehdään QlikViewin omassa latausskriptissä. Tietokannasta haettava tieto haetaan valitsemalla kohdasta Database haluttu yhteys, eli nyt ODBC yhteys ja sitten painetaan Select. Tämän jälkeen valitaan haluttu tietokanta ja sieltä halutut taulut. Halutessaan käyttää Excel tiedostoja, valitaan Data from files- kohdasta Table files ja hakemistosta etsitään haluttu lähdetiedosto. Excel-tiedostot pysyvät ominaan, eikä niistä synny QVD-tiedostoja.

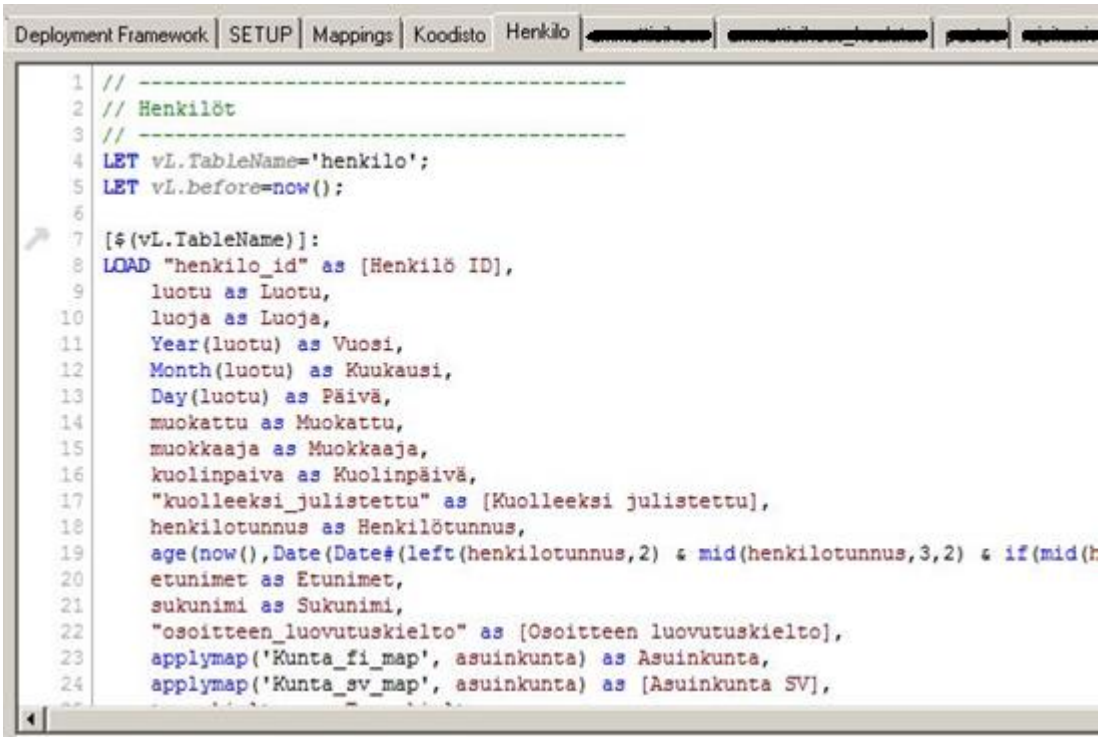
Jokainen tietokannasta valittu taulu kannattaa rakentaa eri välilehdille. Näin jokainen luotu QVD-tiedosto on sovelluksessa erikseen välilehdellä ja selkeyttää huomattavasti käytettävyyttä.

QlikView luo haetun taulun skriptiin kuvan 8 mukaisesti riviltä 8 alaspäin. Rivit neljä ja viisi ovat variaabeleita. Nämä variaabelit ovat edellytys Deployment Framework ympäristössä. Taulut ladataan valmiiseen skriptipohjaan, joka on saatavilla Deployment Frameworkin Example-kansiosta.

5.1.4 Datan manipulointi

QVD-generaattori-sovelluksessa manipuloidaan dataa mahdollisimman paljon. Datan manipuloinnilla tarkoitetaan latausskriptissä tapahtuvaa datan muokkausta. Manipulointi tapahtuu tässä vaiheessa, jotta lopullisen sovelluksen skripti olisi mahdollisimman selkeä. Manipuloinnin määrä kuitenkin riippuu siitä, tullaanko luotuja QVD-tiedostoja käyttämään jatkossa eri raportointisovellukseen. Jos käytetään, jätetään manipulaatiota runsaasti myös raportointisovellukseen.

Yksi manipulointi on nimien muuttaminen yksityiskohtaisempaan muotoon. Nimien kanssa täytyy olla tarkkaavainen, sillä ne eivät saa olla identtisiä toisissa tauluissa olevien kenttien kanssa tarkoittaen eri asiaa. QlikView ei tunnista näitä eri asioiksi ja lopullisen sovelluksen tietomalli yhdistää nämä nimet, mikä tuottaa niiden välille virheellisen yhteyden.

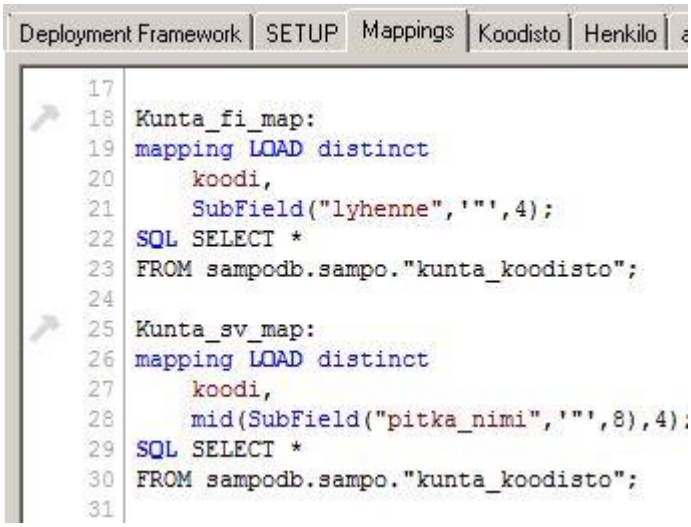


Kuva 8 Sovelluksen skriptiä

Toinen manipulointi, mitä QVD-generaattori-sovelluksessa tulee tehdä, on koodien muuntaminen luettavaan muotoon. Opinnäytetyössäni esimerkiksi kunnat oli merkitty kuntakoodeilla, ja ne täytyi saada luettavaan muotoon käyttäjälle. Yksi tapa tehdä tämä muutos on mäppääminen. Tehdään kuvan 9 mukainen Mappings-välilehti, mihin mäppäykset tehdään. Tässä tapauksessa haluttu kunnan nimi on keskellä tekstiä, joten kunnan nimi on eroteltu tekstistä käyttäen SubField -komentoa. Tämä välilehti ei luo QVD-tiedostoa, sillä tähän merkataan vain mäppäykset, eikä taulua tallenneta mihinkään. Tämän jälkeen mäppäys viedään loppuun välilehdellä, missä kuntakoodi on. Tämä tapahtuu kuvan 8 mukaisesti, rivillä 23 ja 23 olevalla applymap-koodilla.

Kolmas manipulaatio on päivämäärien erittely. Tietokannassa päivämäärät ovat muodossa pp.kk.vvvv ja raportointisovellukseen siitä eritellään päivä, kuukausi ja vuosi. Tämä helpottaa raporttien luomista sovelluksessa, varsinkin jos halutaan eritellä dataa esimerkiksi tietyn vuoden mukaan.

QVD-generaattoriin kannattaa myös tehdä muut tietokannassa olevat yleiset datan manipuloinnit. Niitä voi olla yllä mainittujen lisäksi esimerkiksi isot alkukirjaimet ja henkilön iän muodostaminen henkilötunnuksesta. Raporttikohtaiset datamanipulaatiot tehdään raportointisovelluksen skriptissä. Ikäryhmien muodostaminen on yksi tämmöinen manipulaatio.

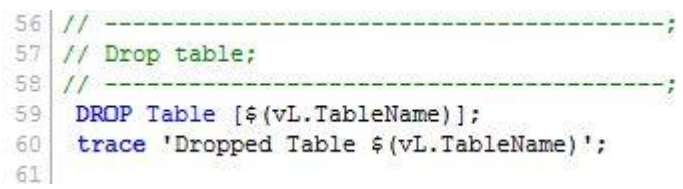


Kuva 9 Mappings välilehti

5.1.5 Tyhjä QVD-sovellus

QVD-generaattori-sovellus tulee pitää tyhjänä, sillä tämä sovellus ladataan päivittäin. Näin pidetään muistin käyttö mahdollisimman vähäisenä. Muistin käyttö voi olla hyvin suuri lukuoperaatioiden kautta tulleen datan tallentamisessa QVD-sovellukseen. Tämän takia QVD-taulut tulee tuhota sen jälkeen, kun taulun sisältö on kirjoitettu QVD-tiedostoon. Tämä tapahtuu kuvan 10 mukaisesti Drop table-komennolla. Lopputuloksena jää siis tyhjä QVW-tiedosto ja joukko QVD-tiedostoja; QVD-tiedostojen lukumäärä vastaa suunnilleen ladattujen taulujen määrää. Jos halutaan tarkistaa QVD-tiedostojen dataa, kannattaa avata taulu kerrallaan. Liian monen taulun avaaminen aiheuttaa RAM-muistin loppumisen, minkä seurauksena kone jumiutuu täysin. Halutun taulun voi avata yksinään QlikView-sovelluksella tai sitten poistaa drop table-komento skriptistä, jolloin kyseisen taulun data tulee käyttäjälle esille.

Kuvasta 10 huomaa myös, miten Deployment Framework käyttää vakituisia komentoja. Käyttäjän ei siis tarvitse itse kirjoittaa tiedostopolkuja skriptiin.



Kuva 10 Drop table-komento

5.2 Raportointisovellus

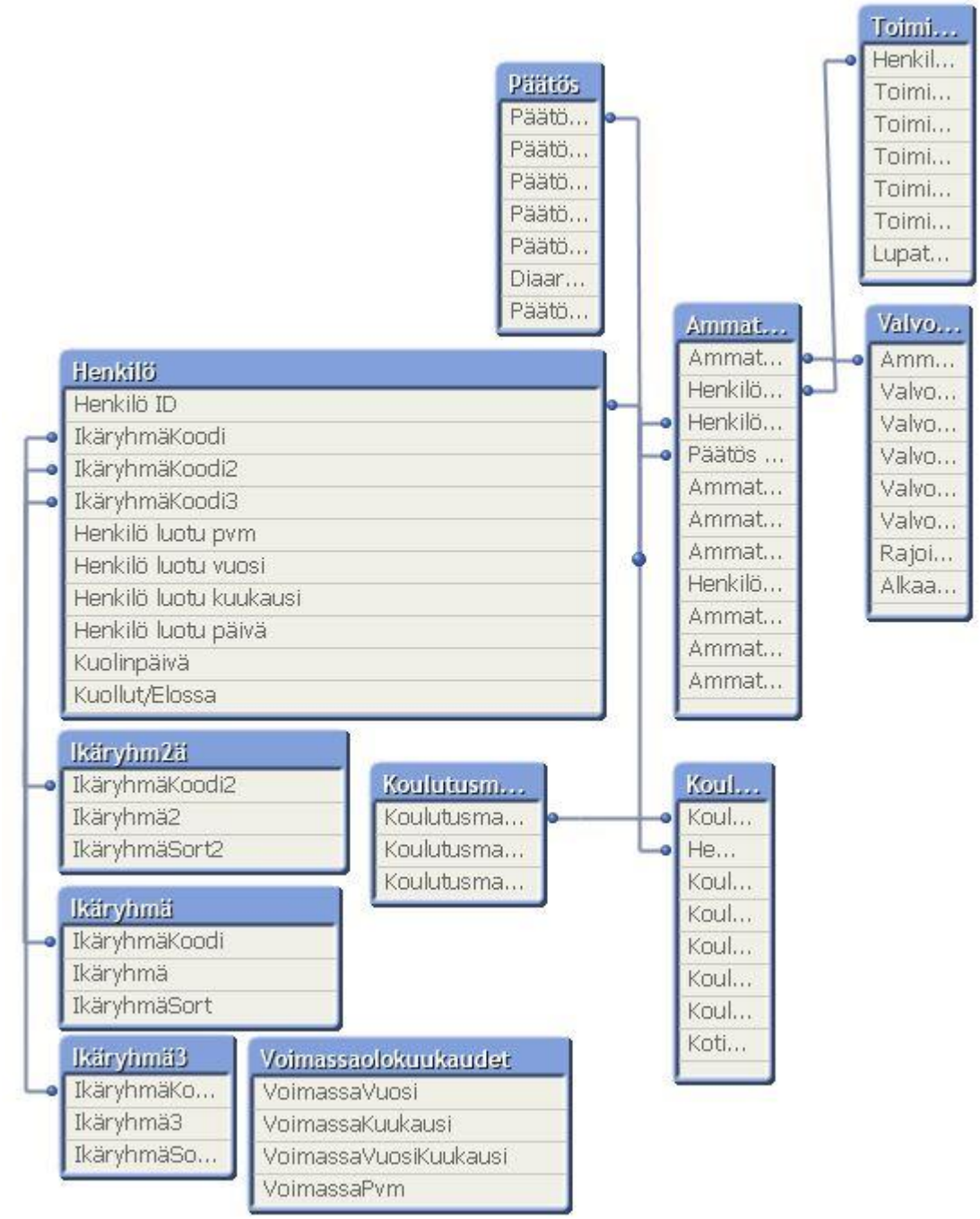
QVD-tiedostojen valmistuessa on aika tehdä raportointisovellus käyttäjälle. Ensimmäinen vaihe on QVD-generaattorilla rakennettujen QVD-tiedostojen lataaminen sovelluksen skriptiin. Tämän jälkeen tehdään tietomalli, mikä sekin muodostetaan latausskriptissä. Kun tämä pohja on kunnossa, on mahdollista aloittaa raporttien muodostaminen käyttäjille.

5.2.1 Aloitus

Sovelluksen alkuun täytyy tehdä Deployment Framework asetukset, kuten tehtiin QVD-generaattori-sovellukseenkin. Sovelluksessa ei enää luoda yhteyttä tietokantaan tai tietovarastoon. Ainoat tietolähteet ovat jo aikaisemmin luodussa QVD-sovelluksessa tehdyt QVD-tiedostot, sekä mahdolliset erilliset lähdetiedostot, kuten Excel-tiedostot. Tietolähteet ladataan skriptiin samaa polkua pitkin kuin avattaisiin esimerkiksi Excel tiedostot.

5.2.2 Tietomalli

Sovelluksen tietomalli on tärkeä tehdä huolella. On hyvin tärkeää, että tietolähteestä ladattu data on nimetty tarkasti, jotta tietomallissa ei tule sekaannuksia. On eri tapoja tehdä tietomalli.



Kuva 11 Sovelluksen tietomalli

Kuvassa 11 on tämän sovelluksen tietomalli. Arkaluontoisen tiedon takia en voi esittää kaikkia tietomallin kenttiä. QlikView muodostaa oman tietomallinsa jo ladatuista lähteistä. Nämä ovat luultavimmin virheellisiä, sillä joko tauluille ei löydy yhteyksiä tai sitten yhteydet ovat vääriä looppien takia. Tämä tietomalli on tehty niin, että tauluja on kutakuinkin saman verran kuin oli luotuja QVD-tiedostoja. Jokaisesta luodusta QVD-tiedostosta syntyy oma taulunsa. Tietomalli on rakenteeltaan tähtimallinen. Taulut on yhdistetty faktatauluun suoraan, jotta

niiden välille saataisiin yhteys. Poikkeuksena ovat skriptissä tehdyt datamanipulaatiosta tulleet taulut. Faktatauluna tässä sovelluksessa on ”Ammat…” taulu.

Tietomallissa ei ole lainkaan virheellisiä linkkejä, eli looppeja taulujen välillä. Looppeja, eli synteettisiä avaimia syntyy, jos kaksi tai useampi taulu sisältää kahden tai useamman samannimisen kentän. Loopit poistetaan huolellisella kenttien nimeämisellä. Tämä nimeäminen tapahtuu pääosin QVD-luonti-sovelluksessa ja nimeämiset, joita ei siellä tehdä, tehdään lopullisessa sovelluksessa. Kenttien yhdistämisellä saadaan tehtyä avainkenttiä, jotka myös estävät looppien syntymisen. Tämä tarkoittaa kahden kentän yhdistämistä yhdeksi kentäksi ja se tapahtuu sovelluksen skriptissä. Yksi tapa looppien välttämiseen on myös poistaa tarpeettomat kentät sovelluksesta, mutta tämä ei ole järkevin ratkaisu, sillä tulevaisuudessa kentät saattavatkin olla tarpeellisia. Esimerkiksi jokaisella taululla oli luotu-kenttä, mutta koska se tarkoittaa jokaisessa taulussa eri asiaa, on se nimetty jokaiseen tauluun uudelleen yksilöivästi. Esimerkiksi henkilö tauluun nimeksi on vaihdettu ’Henkilö luotu pvm’. Ikäryhmät ovat sovelluksen latausskriptiin tehtyjä ryhmittelyitä ja niitä käydään läpi seuraavassa kappaleessa.

5.2.3 Datamanipulaatiot

Latausskriptin on hyvä olla tässä vaiheessa mahdollisimman yksinkertaista. Tämä kuitenkin riippuu siitä, käytetäänkö jo luotuja QVD-tiedostoja jatkossa johonkin muuhun sovellukseen. Tässä projektissa ajateltiin tulevaisuutta ja QVD-generaattoriin on tehty vain yleiset manipulaatiot, jotta QVD-tiedostot olisivat helposti käytettävissä jatkossa. Tästä johtuen sovelluksen skriptiin on tehty runsaasti manipulaatioita ja ne saattavat olla hyvinkin monimutkaisia. Tietomallissa esiintyneet ikäryhmät ovat yksi esimerkki sovellukseen tehdyistä datamanipulaatioista. Kuvassa 12 on määritelty yksi kolmesta ikäjakaumasta, joita sovelluksessa käytetään.

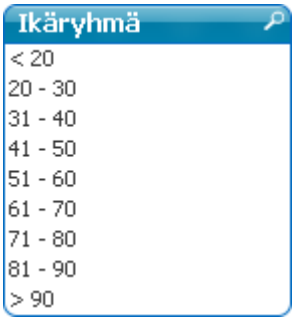
```
24      Ikä,
25      if(Ikä < 20,1,
26         if( Ikä <= 30,2,
27             if( Ikä <= 40,3,
28                 if( Ikä <= 50,4,
29                     if( Ikä <= 60,5,
30                         if( Ikä <= 70,6,
31                             if( Ikä <= 80,7,
32                                 if( Ikä <= 90,8,
```

Kuva 12 Ikäjakauman muodostaminen skriptiin

Ryhmittely pitää suorittaa loppuun INLINE -komennolla. Kuvan 13 mukaisesti tehdään lajittelujärjestys. Lopputuloksena ovat kuvan 14 mukaiset listaukset, jossa haluttua ikähaarukkaa painamalla saadaan valittu rajaus käyttöön.

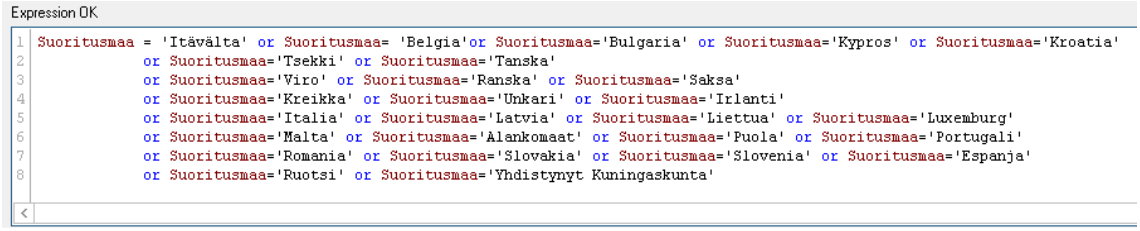
```
Ikäryhmä:
LOAD * INLINE
[ IkäryhmäKoodi, Ikäryhmä, IkäryhmäSort
1,'< 20',1
2,'20 - 30',2
3,'31 - 40',3
4,'41 - 50',4
5,'51 - 60',5
6,'61 - 70',6
7,'71 - 80',7
8,'81 - 90',8
9,'> 90',9
];

```

Kuva 13 Ikäryhmien lajittelujärjestys



Kuva 14 Ikäryhmän näkyminen käyttäjälle

Sovelluksessa on myös käytetty toista datamanipulaatiota, joka rikastaa tietomallia. Tässä datamanipulaatiossa on käytetty apuna variaabelia, mikä on muodostettu. Variaabeli tarkoittaa, että on määritelty jo valmiiksi esimerkiksi ryhmä tai jaottelu, mitä voi raporttia tehdessä käyttää. Tämä tekee myös skriptistä visuaalisesti yksinkertaisempaa. Variaabelilla on helppo tehdä esimerkiksi haluttu joukko tietoa, mikä halutaan vetää yhteen. Tässä tapauksessa haluttiin listata EU maat yhteen kuvan 15 tavalla ja variaabeli nimettiin v_KoulutusEU.



Kuva 15 Variaabelin muodostaminen

Nyt on muodostettu variaabeli, jossa on määritelty EU:hun kuuluvat maat. Seuraavaksi lataus-skriptiin lisätään osio, jossa jaotellaan ryhmät kolmeen eri numeroon kuvan 16 mukaisesti. Tämän jälkeen INLINE -komennolla nimetään kyseiset numerot halutuiksi kuvan 17 tapaan.

```
164         if(Suoritusmaa = 'Suomi',1,
165           if( $(v_KoulutusEU),2,
166               if( Suoritusmaa-$(v_KoulutusEU) or Suoritusmaa <> 'Suomi',3,
167       ))) as Koulutusmaakoodi,
```

Kuva 16 Koulutusmaiden numerointi

```
46  //-----------------------------------------------------------
47  // Tehdään Koulutusmaa järjestys
48  //-----------------------------------------------------------
49  Koulutusmaaryhmä:
50  LOAD * INLINE
51  [ Koulutusmaakoodi, Koulutusmaaryhmä, Koulutusmaaryhmäsort
52  1, 'Suomi',1
53  2, 'EU',2
54  3, 'Muut',3
55  ];
```

Kuva 17 Numeroinnin nimeäminen

Lopullinen jaottelu käyttäjälle tulee näyttämään kuvan 18 mukaisena, ja hän voi helposti rajata hakutuloksia koulutusmaan mukaan.

Kuva 18 Koulutusmaan näkyminen käyttäjälle

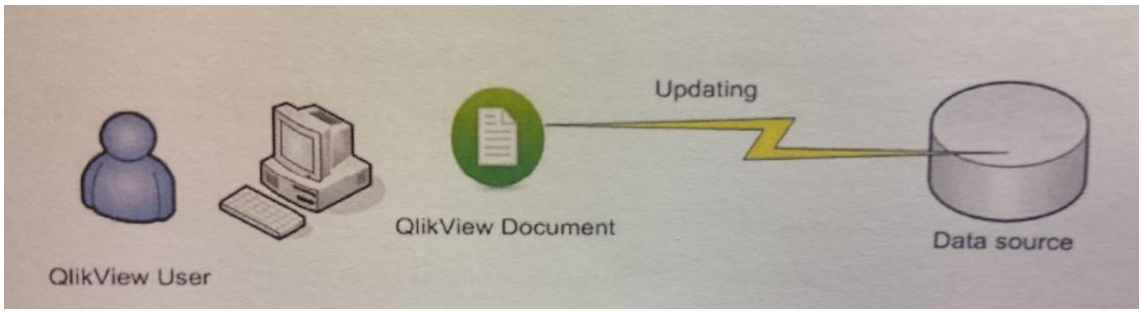
6 QlikView-ympäristö

QlikViewiä on mahdollista käyttää kolmen eri ympäristön mukaan.

- QlikView Standalone
- QlikView Server
- QlikView Server Publisher

6.1 QlikView Standalone

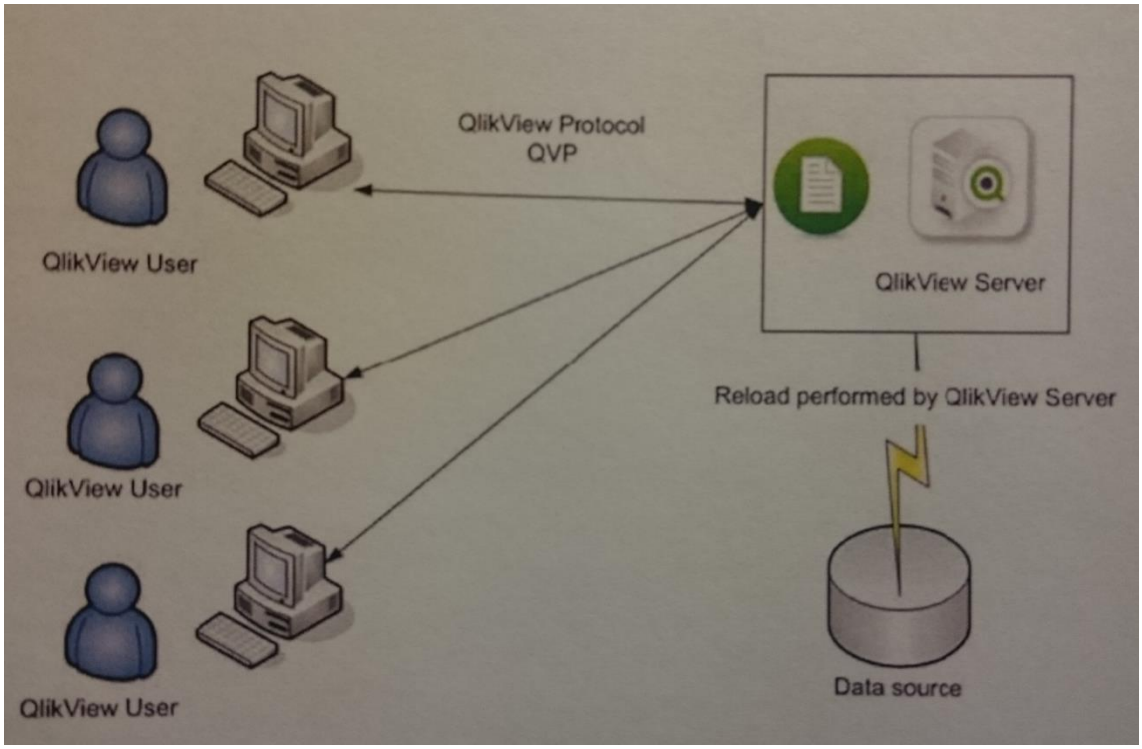
QlikView Standalone on tarkoitettu yksittäiselle käyttäjälle. Se tarkoittaa kuvan 19 mukaisesti, että sovelluksesta otetaan suora yhteys tietolähteeseen ilman välikäsiä. Tämän ympäristö on helppokäyttöinen, sillä kaikki asetukset tehdään suoraan sovelluksesta. Ympäristöä voi käyttää QlikView Personal Editionilla, mikä on ilmainen versio käyttäjälle. Tämän haittapuolena on, että käyttäjä ei saa kuin itse tekemiä sovelluksiaan auki sillä tietokoneella, jolla sovellukset on tehty.



Kuva 19 QlikView Standalone

6.2 QlikView Server

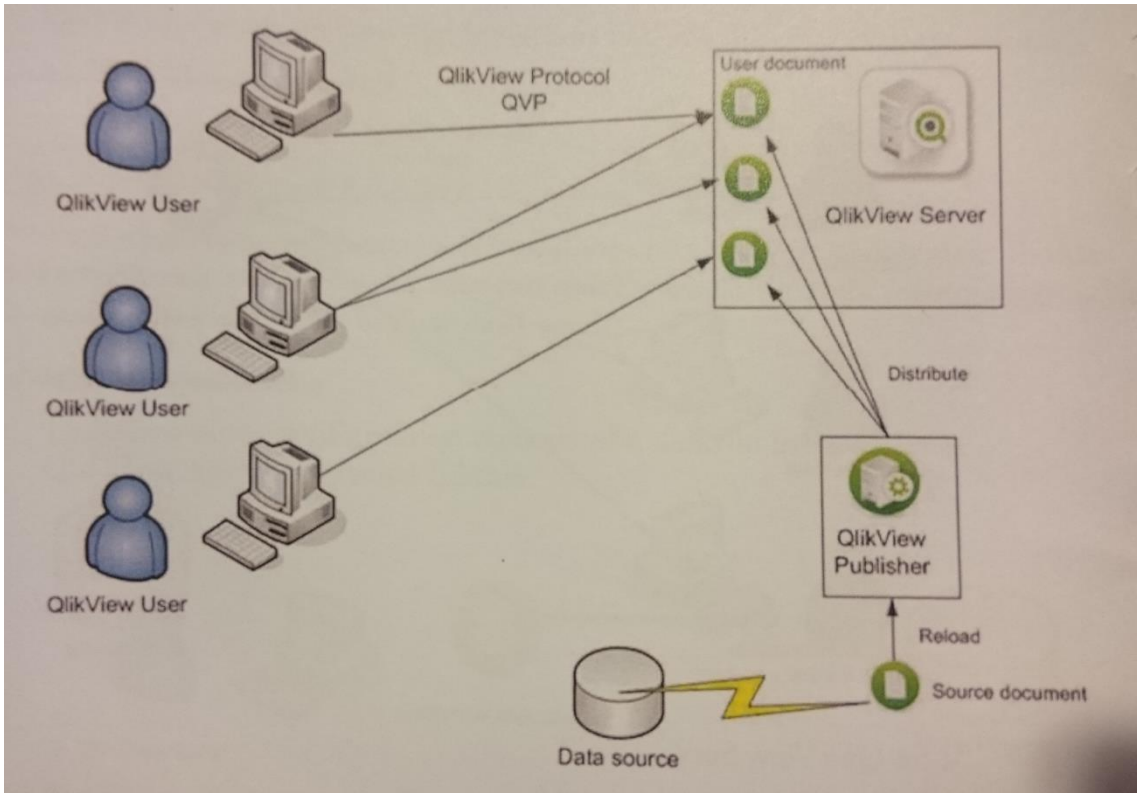
QlikView Serverin avulla käyttäjät voivat käyttää sovelluksia selaimen kautta AccessPointilla. AccessPoint on QlikViewin oma web portaali. Käyttäjä pääsee sinne tunnistautumalla ja saa käyttöönsä dokumentit. AccessPointissa käyttäjä ei pysty muokkaamaan sovelluksien skriptiä, mutta hän voi tehdä omia raportointitauluja käytössä olevista datan tauluista. QlikView Serverin ylläpitäjä käyttää Management Consolea hallitakseen ympäristöä ja sovelluksia.



Kuva 20 QlikView Server

6.3 QlikView Publisher

Kohdeorganisaatiollani on käytössään QlikView Publisher. Tässä organisaatiolla on käytössään kaksi eri palvelinta, Publisher- ja Server-palvelin. Publisher ympäristössä käyttäjä luo ja muokkaa sovelluksia, jakaa käyttöoikeuksia Management Consolen kautta sekä määrittelee raporttien latausajat. Lataukset on asetettu lataamaan öisin työajan ulkopuolella, jotta seuraavana päivänä käyttäjillä olisi käytössään reaaliaikaiset tiedot. Käyttöoikeudet tarkoittavat oikeuksia selaimen kautta pääsyä QlikView serverille, josta raportit ovat saatavilla. Management Console sijaitsee Publisher-palvelimella. Sovellukset julkaistaan Serverillä, johon raporttien käyttäjät pääsevät AccessPointin avulla. Tämä vaihtoehto on hyvä yrityskäyttöön.



Kuva 21 QVD Publisher

7 Yhteenveto

Tämän toiminnallisen opinnäytetyön tarkoituksena oli rakentaa QlikView-raportointisovellus kohdeorganisaatiolleni. Sovellusta tulevat käyttämään oman alansa ammattilaiset. He käyttävät sovellusta tiedon keräämiseen, lukumäärien tarkasteluun sekä yleiseen tarkkailuun, sillä heidän työnkuvansa kohdistuu valvontaan. Organisaatiolle uutena asiana tullut QlikView Deployment Framework on otettu käyttöön tuotannossa ja ympäristö on rakennettu muihinkin QlikView-sovelluksiin.

Sovelluksesta pidettiin demo tilaisuuksia, joissa tulevat käyttäjät pääsivät tutustumaan sovellukseen. Tilaisuuksissa anntettiin kehitysideoita sekä tarpeita. Teimme sovelluksiin paljon omia ratkaisuja sekä graafeja, joita tulevat käyttäjät pitivät hyvinä ratkaisuina ja niitä haluttiin pitää sovelluksessa. Käyttäjät yllättyivät, mitä kaikkea tietoa sovelluksesta oli mahdollista saada ulos.

Vaikka sovelluksen tekeminen alkoi aivan alusta, eikä minulla ollut lainkaan aiempaa kokemusta QlikViewistä, onnistui sovelluksen tekeminen mielestäni hyvin. QlikViewin konsultti oli aktiivisesti mukana ja teimme hyvää yhteistyötä. Testidatan oleminen tietolähteenä oli oppi-

miseni kannalta hyvä asia. Testikantaa täytyi jatkuvasti tarkkailla ja sen seurauksena sen oppi tuntemaan varsin hyvin. Opin sovelluksen skriptin lisäksi myös paljon QlikViewistä, kuten Management Consolesta sekä QlikView-palvelinympäristöstä. Raportointitarpeiden puutteellisuus opetti työskentelemään luovemmin. Tämän projektin lähtökohta ei ole poikkeuksellista, vaan samankaltaisia tilanteita saattaa olla hyvin yleisestikin, varsinkin jos QlikView tulee koko organisaatiolle uutena asiana ja sovelluksen tekeminen halutaan pitää mahdollisimman paljon organisaation sisällä.

Lähteet

Floyd M. 2013. QlikView Scripting. Birmingham: Packt Publishing Ltd

Hovi, A, Hervonen, H & Koistinen, H. 2009. Tietovarastot ja Business Intelligence. Porvoo: WSOY

QlikTech International AB. 2012 Qlik Developer, QlikView version 11.2 English. (Nidottu koulutusmateriaali.

QlikTech International AB. 2012 Qlik Designer, QlikView version 11.2 English. (Nidottu koulutusmateriaali

Qlik community. 2013. Introducing the QlikView Deployment Framework. Viitattu 20.2.2016 https://community.qlik.com/blogs/theqlikviewblog/2013/11/15/introducing-the-qlikview-deployment-framework