Kahden XML-kyselykielen vertaileva käyttäjätutkimus

Matti Lassila

Tampereen yliopisto
Informaatiotieteiden yksikkö
Informaatiotutkimus ja
interaktiivinen media
Pro gradu -tutkielma
Maaliskuu 2013

Tiivistelmä

Tampereen yliopisto
Informaatiotieteiden yksikkö
Informaatiotutkimus ja interaktiivinen media
Lassila Matti: Kahden XML-kyselykielen vertaileva käyttäjätutkimus
Pro gradu -tutkielma, 107 s., 28 liites.
Informaatiotutkimus ja interaktiivinen media
Maaliskuu 2013

XML-merkatun tiedon tehokasta käsittelyä ja hakua varten on kehitetty relaatiotietokantojen SQL-kieltä vastaavia kyselykieliä. Tutkielmassa tarkastellaan kahden, lähtökohdiltaan ja ilmaisuvoimaltaan erilaisen XML-kyselykielen soveltuvuutta vuorovaikutteiseen ad hoc -käyttöön. Vertailuparina käytetään XQuery ja XIL-kieliä, joiden kummankin suunnittelussa on otettu vaikutteita relaatiotietokantojen SQL-kyselykielestä. Kieliä tarkastellaan erityisesti dokumenttiorientoituneen XML-tiedonhaun näkökulmasta.

Tutkielmassa selvitetään ad hoc -käyttötilannetta jäljittelevien käyttäjäkokeiden avulla, tarjoaako ad hoc -käyttöön suunniteltu XIL käyttäjälleen hyötyä verrattaessa sitä yleiskäyttöiseen XQuery-kieleen. Kieliä verrataan suhteessa koetilanteissa annettujen oikeiden vastauksien määrään sekä vastauksissa tehtyjen virheiden määrään ja laatuun. Käyttäjäkokeiden tulosten sekä tutkimuskirjallisuuden pohjalta esitetään XIL-kieltä koskevia kehitysehdotuksia.

Saadut tulokset kuvaavat testatuilla kielillä saavutettavaa suoriutumistasoa käyttötilanteessa, jossa tiedonhakija joutuu muotoilemaan kyselyn muistinvaraisesti ilman käyttöliittymän tai dokumentaation tukea. Mikäli XIL-kieltä halutaan kehittää tässä tutkielmassa esitettyjen kehitysehdotusten pohjalta, käyttäjäkokeiden tulokset tulisi validoida uusintatestillä mahdollisimman todenmukaisessa käyttöliittymässä.

Avainsanat: tiedonhakujärjestelmät, käyttäjäkokeet, kyselykielet, XML, loppukäyttäjäohjelmointi

Sisällysluettelo

1 Johdanto

Vapaasti saatavilla olevien aineistojen – avoimen datan – on ennustettu olevan yksi lähivuosikymmenten keskeisimmistä muutosvoimista. Avoimen datan uskotaan luovan uusia liiketoimintamahdollisuuksia, edistävän tieteiden kehitystä ja auttavan yhteiskunnallisten haasteiden ratkaisemisessa (Euroopan Komissio 2011, 3). Rahassa mitattuna näiden hyötyjen on arvioitu olevan Euroopan Unionin alueella jopa 40 miljardia euroa vuosittain. Koska potentiaaliset hyödyt ovat suuret, datan uudelleenkäyttöä on edistetty sekä kansallisella että EU-tasolla lainsäädännöllä ja erilaisilla politiikkaohjelmilla (Poikola et al 2010, Euroopan Komissio 2011).

Pelkkä poliittinen tahto ei kuitenkaan yksin riitä. Arvioidut hyödyt jäävät toteutumatta, mikäli vapaasti saataville asetettua dataa ei saada hyödynnettyä tehokkaasti. Datan hallinnointiin, käsittelyyn ja analysointiin tarvitaan tekninen infrastruktuuri, jonka kehittämistä EU-komissio pitää ensiarvoisen tärkeänä (Euroopan Komissio 2011).

Tieteellisen tutkimustyön katsotaan olevan astumassa uuteen aikakauteen käytettävissä olevien tutkimusaineistojen valtavan kasvun myötä (Hey et al 2009). Ongelmaksi on kuitenkin osoittautunut se, että organisaatioiden kyky kerätä ja tallentaa dataa on kehittynyt paljon nopeammin kuin datan hallinnoinnin ja analyysin keinot (Gray et al 2005). Toisin kuin koskaan aiemmin, ongelma ei ole enää datan niukkuus vaan ylenmääräisyys, tulva (Hey & Trefethen 2003). Käytettävissä olevien aineistojen valtava kasvu on luonut ongelmia aloilla, joissa tarve organisaatioiden tarjoamalle datankäsittelytuelle on ollut perinteisesti pieni. Näihin lukeutuvat esimerkiksi mikrobiologia, kemia, sosiaalitieteet sekä humanistiset alat (Howe & Halperin 2012). Datatulvaa edeltävänä aikana näillä aloilla on saatettu selvitä datanhallinta- ja analyysiongelmista taulukkolaskentataulukoiden avulla (emt.). Sama tilanne vallitsee myös yritysmaailmassa, sillä maailmanlaajuisesti suurta osaa kaikesta liiketoimintadatasta käsitellään ja hallinnoidaan taulukkolaskennassa (Panko & Port 2012).

Ennen datatulvaa taulukkolaskentaan nojautuvat työnkulut ovat olleet riittävä ratkaisu useimpiin tutkimus- ja yritysorganisaatioissa kohdattuihin datankäsittelypulmiin. Joitain satoja rivejä sisältävän datan käsittely kourallisena taulukkolaskentatauluja on jouhevaa. Ongelmat alkavat, kun rivien määrää mitataan kymmenissä tuhansissa, datan jakautuessa satoihin taulukkolaskentatiedostoihin. Tällöin datan hallinnointi voi viedä suuren osan analyytikon tai tutkijan työajasta (Howe & Cole 2010). Taulukkomuotoisen käyttöliittymän skaalautuvuuden rajat ovat tulleet vastaan.

Vielä 1980-luvulla tavanomaisessa toimistotyössä käytettiin komentorivi- tai tekstivalikkopohjaisia työvälineitä. Liikuttaessa vielä yksi vuosikymmen taaksepäin, 1970-luvulle, tietokoneen käyttö ei olennaisesti eronnut ohjelmoinnista (Rushkoff 2010, 133). Sittemmin tekstipohjaiset ihmisen ja koneen vuorovaikutustavat ovat väistyneet graafisten käyttöliittymien tieltä. Nykyään komentorivityökalujen katsotaan palvelevan pääasiassa teknisesti orientoituneiden henkilöiden tarpeita.

Datatulva on kuitenkin tuonut esille osoita-klikkaa-vuorovaikutuksen heikkouden: graafinen käyttöliittymä toimii hyvin tilanteissa, joissa käsiteltävien kohteiden määrä on pieni, kun taas kohteiden määrän kasvaessa vuorovaikutus muuttuu tehottomaksi. On nähtävissä merkkejä siitä, että komentokielikäyttöliittymät olisivat tekemässä paluuta uudistuneessa muodossa, alkuperäisiä toteutuksia virhesietoisempina ja syntaksiltaan lähempänä luonnollista kieltä. (Peffers et al 2007.)

Myös tietokantojen alueella on tapahtumassa kehitystä, joka edustaa 1970-luvun tietokanta-ajattelun uudelleentulemista. Vaikka relaatiotietokannat ovat koko kolmekymmenvuotisen käyttöhistoriansa ajan olleet pääasiassa tietokanta-ammattilaisten sekä ohjelmoijien työvälineitä, niiden alkuperäinen tarkoitus oli palvella toimistotyön osana tapahtuvaa ad hoc -tiedonhakua (Chamberlin & Boyce 1974). Relaatiotietokantojen kanssa samaan aikaan markkinoille tulleet taulukkolaskentaohjelmat täyttivät kuitenkin oletetun käyttäjäkunnan tarpeet paremmin, ja tietokannat ottivat pääosin taustajärjestelmän roolin seuraavien vuosikymmenten ajaksi (McJones 1997). Datatulvan myötä tietokannat saattavat kuitenkin olla ottamassa niille 1970-luvulla hahmotellun paikan: tutkijoille ja muille datatulvan vaivaamille tietotyöläisille on kehitetty tietokantojen kyselykäyttöliittymiä, jotka opastus- ja tukitoiminnallisuuksineen tukevat tietokantojen vuorovaikutteista käyttöä kyselykielen avulla (Howe & Cole 2010, Howe & Halperin 2012). Näistä järjestelmistä saadut kokemukset ovat osoittaneet kyselykielten palvelevan hyvin myös tutkijoiden tarpeita, kunhan kyselyn muotoiluun on tarjolla riittävästi järjestelmän tukea (Howe & Halperin 2012).

Relaatiotietokantojen lisäksi tutkimus- ja liiketoimintadataa tallennetaan myös muita tiedontallennusparadigmoja edustaviin järjestelmiin. XML (eXtensible Markup Language) -merkkaus on eräs yleisimmistä relationaaliselle muodolle vaihtoehtoisista tiedontallennustavoista. Hierarkkiseen tietomalliin perustuva XML on metakieli, jota käytetään toteutettaessa sovellusaluekohtaisia merkkauskieliä (Bray et al 1998). XML-merkatun tiedon tehokasta käsittelyä ja hakua varten on kehitetty relaatiotietokantojen SQL-kieltä vastaavia kyselykieliä. Nämä kyselykielet antavat tälle tutkielmalle aiheen.

Tutkielmassa tarkastellaan kahden, lähtökohdiltaan ja ilmaisuvoimaltaan erilaisen XML-kyselykielen soveltuvuutta vuorovaikutteiseen ad hoc -käyttöön. Vertailuparina käytetään XQuery ja XIL-kieliä, joi-

den kummankin suunnittelussa on otettu vaikutteita relaatiotietokantojen SQL-kyselykielestä. Kieliä tarkastellaan erityisesti dokumenttiorientoituneen XML-tiedonhaun näkökulmasta. Vertailussa tarvittava tutkimusaineisto kerätään käyttäjäkokeilla.

Tutkielmassa sovelletaan suunnittelutieteellistä tutkimusotetta. Suunnittelutieteellisellä tutkimuksella tarkoitetaan tässä tutkimustapaa, jossa yhdistyvät jokin inhimillistä tarkoitusta palveleva artefaktin tuottaminen sekä sen arviointi, kuinka hyvin luotu artefakti lopulta onnistuu palvelemaan aiottua käyttötarkoitusta. Tutkimusprosessissa noudatetaan Peffersin ja muiden (Peffers et al 2007) kehittämää *design science research* -metodia.

XQuery on W3C-organisaation standardoima XML-kyselykieli, joka on ensisijaisesti tarkoitettu ohjelmointitaitoisten käyttäjien ei-vuorovaikutteiseen käyttöön (Buxton et al 2011). Satunnainen ad hoc -käyttö on kuitenkin otettu huomioon kielen suunnittelussa (Chamberlin 2003), ja saataville on tullut vuorovaikutteiseen käyttöön tarkoitettuja kielen laajennoksia ja työkaluja (kts. esim. Grün et al 2007). Tästä syystä kieltä on mielekästä tutkia ad hoc -käyttötilannetta jäljittelevässä kokeessa, vaikka kieli onkin ilmaisuvoimaltaan verrattavissa yleisohjelmointikieliin.

XIL on kokeellinen, vuorovaikutteisen dokumenttiorientoituneen XML-tiedonhaun tutkimuskieli (Junkkari et al 2006). Kieli noudattaa SQL-kielen alkuperäisiä suunnitteluperiaatteita: se on tarkoitettu ensisijaisesti vuorovaikutteiseen ad hoc -käyttöön. Koska kielen käyttöalue ja -tapa on ennalta tunnettu, sen ilmaisuvoimaa on voitu rajata tietoisesti, tarkoituksena helpottaa kielen oppimista ja käyttöä. Kielen syntaksi on lähellä alkuperäistä SQL-kieltä, jonka alkeet hallitsevan henkilön voidaan olettaa oppivan XIL-kieli verrattain helposti niin että kielen omaksuminen tapahtuu osin siirtovaikutuksen kautta.

Tässä tutkielmassa selvitetään ad hoc -käyttötilannetta jäljittelevien käyttäjäkokeiden avulla, tarjoaako ad hoc -käyttöön suunniteltu XIL käyttäjälleen hyötyä verrattaessa sitä yleiskäyttöiseen XQuery-kieleen. Kieliä verrataan suhteessa koetilanteissa annettujen oikeiden vastauksien lukumäärään sekä vastauksissa tehtyjen virheiden määrään ja laatuun. Käyttäjäkokeiden tulokset analysoidaan, ja analyysitulosten pohjalta esitetään XIL-kieltä koskevia kehitysehdotuksia. Lisäksi nostetaan esiin joitain tutkimuskirjallisuudessa mainittuja ad hoc -kyselykielten kehitysehdotuksia.

Tutkielmassa on seitsemän päälukua. Ensimmäinen pääluku esittelee tutkielmassa sovelletun suunnittelutieteellisen tutkimusmetodologian ja asemoi tutkielman myöhemmät osat luvussa kuvatun tutkimusprosessin vaiheisiin. Tätä seuraa johdatus tutkielman laajempaan kontekstiin, XML-tiedonhakuun.

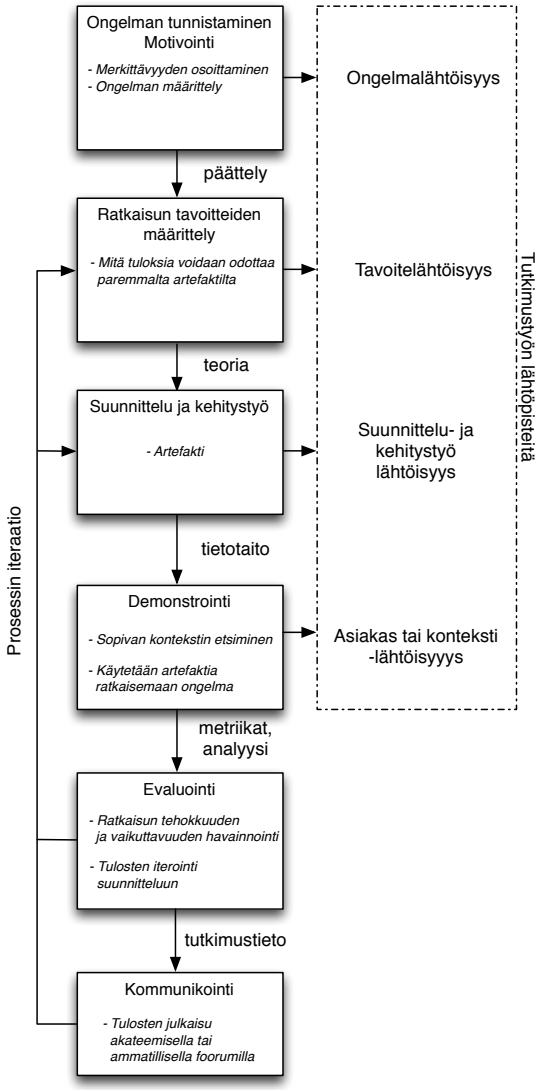
Lisäksi käsitellään yleisemmin dokumenttien konelukuiseen esittämiseen liittyviä kysymyksiä. Kolmannessa pääluvussa problematisoidaan ja määritellään tutkielman kannalta olennaiset käsitteet. Tämän jälkeen esitellään käyttäjäkokein vertailtavat kyselykielet XIL ja XQuery sekä näiden yhteinen esivanhempi SQL. Viidennessä pääluvussa perehdytään kyselykielten kokeellisen tutkimuksen historiaan sekä viimeaikaisiin kyselykielitutkimuksiin, tarkoituksena tutustua tutkimusalalla vallitseviin tutkimuksen teon käytäntöihin ja käyttäjäkokeiden tuloksiin. Aiempien tutkimustulosten pohjalta muodostetaan suuntaa antava käsitys tasosta, jolle nyt tutkittavien kyselykielten – erityisesti XIL-kielen – tulisi päästä, jotta kielen suunnittelua voidaan pitää onnistuneena.

Kuudennen pääluvun aloittaa yhteenveto kyselykielten tutkimisen menetelmistä sekä koeasetelmin kerätyn aineiston analyysitavoista. Tätä seuraa tehtyjen käyttäjäkokeiden kuvaukset sekä koetulosten analyysi. Koska tutkielmalla on XIL-kielen kehittämiseen liittyvä suunnittelutieteellinen tavoite, luvussa paneudutaan erityisesti XIL-kielellä laadittujen vastausten virheiden erittelyyn. Koetulosten analyysiä jatketaan luokittelemalla XIL- ja XQuery-vastaukset yhteismitallisesti, jonka jälkeen luokittelua käytetään kielten tilastollisen vertailun pohjana. Luvun päättää regressionanalyysin avulla tehty tarkastelu testitehtävien malliratkaisujen monimutkaisuuden ja tehtäväsuoriutumisen yhteydestä. Malliratkaisujen monimutkaisuutta mitataan Halsteadin ohjelmistometriikoilla.

Seitsemännessä pääluvussa kootaan yhteen koetulosten analyysin yhteydessä esitetyt XIL-kielen kehitysehdotukset, joita täydennetään tutkimuskirjallisuuteen perustuvalla pohdinnalla kyselykielten ja hakujärjestelmien ominaisuuksista, jotka parantaisivat vuorovaikutteisessa käyttötilanteessa tehtyjen hakujen tuloksellisuutta. Tutkielman päättää loppuyhteenveto, jossa tutkimustulosten valossa arvioidaan XIL-kielen toimivuutta dokumenttiorientoituneen XML-tiedonhaun kyselykielenä.

2 Suunnittelutieteellinen tutkimus

Peffers ja muut (2007) ovat kehittäneet kirjallisuuteen perustuvan suunnittelutieteellisen tutkimuksen prosessimallin, jonka tarkoituksena on tukea suunnittelutavoitteita sisältävän tutkimuksen tekoa, raportointia sekä tutkimustulosten arviointia. He korostavat, ettei malli ole normatiivinen, vaan se esittää erään mahdollisen, kirjallisuudessa toistuvan tavan toteuttaa suunnittelutieteellinen tutkimus.



Kuva 1: Suunnittelutieteellisen tutkimuksen prosessi. Lähde: Peffers ja muut 2007, mukaillen.

Malli kuvaa suunnittelutieteellisen tutkimuksen prosessin kuutena palautekytkentöjä sisältävänä vaiheena (Kuva 1). Suunnittelutieteellisen menetelmän käyttö ei kuitenkaan edellytä vaiheiden läpikäyntiä prosessin ensimmäisestä vaiheesta alkaen. Tutkimusideasta riippuen työn lähtöpisteenä voi toimia mikä tahansa prosessin vaihe. Suunnittelutieteellisen tutkimuksen vaiheita ovat:

1. *Ongelman tunnistaminen ja motivointi*. Suunnittelutieteellisen tutkimuksen tuottama artefakti pyrkii ratkaisemaan jonkin maailmassa havaitun ongelman. Vaiheessa kuvaillaan ja käsitteellistetään havaittu ongelma ja perustellaan, miksi ongelma tulisi ottaa ratkaistavaksi. Vaiheen läpiviemiseen tarvitaan tietoa ongelman tilasta sekä sen ratkaisemisen tuomista eduista. (Peffers et al 2007.)
2. *Ratkaisun tavoitteiden määrittely*. Ratkaisun tavoitteet johdetaan ongelman määrittelystä. Tavoitteet voivat olla määrällisesti mitattavissa, kuten ratkaisun tuoma parannus suhteessa nykyisiin saman ongelman ratkaiseviin toteutuksiin tai laadullisia kuvauksia siitä, kuinka suunniteltava artefakti ratkaisee ongelman, jonka ratkaisua ei aiemmin ole yritetty. Tavoitteiden määrittelyssä tarvitaan yksityiskohtaista tietoa ongelmasta ja mikäli aiempia ratkaisuja on olemassa, tietoa niiden tehokkuudesta. (Peffers et al 2007.)
3. *Suunnittelu- ja kehitystyö*. Vaiheen lopputuloksena syntyvä artefakti voi olla olemukseltaan materiaalinen tai immateriaalinen objekti, esimerkiksi laite, ohjelmisto tai käsitteellinen malli. Suunnittelu- ja kehitystyön apuna voidaan hyödyntää ongelman aihepiiriä käsittelevää teoreettista tietoa. (Peffers et al 2007.)
4. *Demonstrointi*. Tässä vaiheessa suunnittelu ja kehitystyön tuottamalla artefaktilla ratkaistaan yksi tai useampi tunnistetun ongelman ilmentymä. Käytettävää menetelmää ei ole rajattu, vaan valinta voidaan tehdä tarkoituksen mukaan. Peffers ja muut (2007) mainitsevat mahdollisiksi menetelmiksi esimerkiksi koeasetelman, simulaation, tapaustutkimuksen tai matemaattisen todistuksen. Vaiheessa tarvitaan ymmärrystä siitä, kuinka artefaktia voidaan käyttää sen suunnittelun motiivina olleen ongelman ratkaisemisen apuna. (Peffers et al 2007.)
5. *Evaluointi*. Evaluoinnissa ratkaisulle asetettuja tavoitteita verrataan demonstroinnissa saavuttetuihin tuloksiin. Evaluointimenetelmän valinta tehdään ongelman alan ja artefaktin luonteen ohjaamana. Tarvittava tietämys liittyy ongelman alan kannalta relevantteihin metriikoihin ja analyysimenetelmiin. Evaluoinnin jälkeen päätetään, palataanko prosessissa takaisin suunnittelu- ja kehitystyöhön vai siirrytäänkö prosessissa eteenpäin, raportoimaan tutkimuksessa saavutetut tulokset. (Peffers et al 2007.)
6. *Kommunikointi*. Tutkimuksen tulokset raportoidaan. Tutkimusraportin muoto voi noudattaa prosessimallin vaiheita, alkaen ongelmakuvaksesta ja päättyen evaluointitulosten raportointiin. (Peffers et al 2007.)

Tässä tutkielmassa työ aloitetaan demonstroinnista. Tähän mennessä XIL-kielen parissa tehty tutkimus (Junkkari et al 2006, Tuomisto 2011, Vainio 2012) liittyy pääosin demonstrointia edeltävään, suun-

nittelu ja kehitystyön vaiheeseen. Pefferin ja muiden (2007) mukaan suunnittelu- ja kehitystyölähtöinen tutkimustapa soveltuu käytettäväksi esimerkiksi silloin, kun halutaan kokeilla jotain toista kontekstia varten kehitetyn ratkaisun toimivuutta uudessa käyttöyhteydessä. XIL-kielen tapauksessa tämä kokeilu on ollut relaatiotietokantojen SQL-kielen käyttö XML-kyselykielen kehittämisen lähtökohtana.

Peffersin ja muiden (2007) malliin suhteuttaen tämän tutkielman johdanto sekä toinen ja kolmas pääluku käsittelevät tutkimusaihetta *ongelman tunnistamisen ja motivoinnin* näkökulmasta. Viidennessä pääluvussa tapahtuva muuhun tutkimukseen perehtyminen liittyy *ratkaisun tavoitteiden määrittelyyn*. Tutkimuskirjallisuuden kautta luodaan käsitys siitä, mitä suorituskykytasoa XIL-kieleltä voidaan odottaa. Koska kyseessä on kyselykieliä vertaileva tutkimus, tavoiteltava suorituskykytaso määrittyy myös suhteessa XQuery-kielellä saataviin koetuloksiin. Vasta koetulokset paljastavat, tarjoaako ad hoc -käyttöön suunniteltu XIL käyttäjälleen hyötyä verrattaessa sitä yleiskäyttöiseen XQuery-kieleen.

Ratkaisun tavoitteiden määrittelyn ohella viides pääluku käsittelee myös *evaluointia*, sillä luvussa aloitetaan tutustuminen kyselykielten tutkimisen menetelmiin. Menetelmien käsittelyä syventävä sekä tutkimusasetelmat ja tulokset kuvaava kuudes pääluku kuuluu kokonaisuudessaan evaluoinnin vaiheeseen. Tutkielma tekstinä edustaa prosessimallissa *kommunikointia*.

3 XML ja tiedon rakenteisuuden jatkumo

XML-perustaisia merkkauskieliä käytetään laajasti eri sovellusalueilla. Tässä pääluvussa tutustutaan XML-merkkauksen yleispiirteisiin, pohditaan eri sovellusalueilla tavattujen dokumenttien olemusta ja kuvaillaan tutkielman aiheen kannalta olennaisen, dokumenttiorientoituneen XML-merkkauksen, ominaispiirteitä.

3.1 XML-merkkaus

XML-merkkaus koostuu merkkijonojen joukosta, joka muuntaa rakenteettoman merkkivirran joukoksi elementtejä. Elementit koostuvat nimetyistä tunnisteista ja sisällöstä, jota tunnisteet ympäröivät. Kulmasulut merkkaavat tunnisteen ja tekstisisällön rajan. Nimetyt tunnisteet esiintyvät pareittain, joissa parin ensimmäistä tunnistetta kutsutaan avaus- tai alkutunnisteeksi ja jälkimmäistä sulku- tai lopputunnisteeksi. Avaustunnisteeseen voidaan liittää yksi tai useampi lisäominaisuus – attribuutti – joiden järjestys ei ole dokumenttia käsiteltäessä merkityksellinen.

Avaustunnisteessa vasemmalle päin osoittavaa kulmasulkua seuraa elementin nimi, sekä elementtiin liitetyt attribuutit. Tunnisteen päättää oikealle osoittava kulmasulku. Attribuutti-arvo-pareissa attribuutin arvo tulee ympäröidä lainausmerkeillä. Avaustunnisteen jälkeen elementti voi sisältää tavanomaista tekstisisältöä tai toisia elementtejä, joiden esiintymisjärjestys on merkityksellinen (Glushko & McGrath 2005, 45).

Tarkastellaan seuraavaksi listauksen 1 dokumenttikatkelman elementtiä *kappale*. Tässä elementissä `<kappale>` on avaustunniste, `tyyli` on elementin attribuutti ja `puheenvuoro` attribuutin arvo. Sisältö koostuu elementeistä `puhuja` ja `repliikki`, jotka ovat lapsi-vanhempi-suhteessa ympäröivään kappale-elementtiin.

Listaus 1: Esimerkkidokumentti

```
<kappale tyyli=”puheenvuoro”>
    <puhuja>JUHANI<puhuja>
    <repliikki>Ken aukaisi ovea?</repliikki>
</kappale>
```

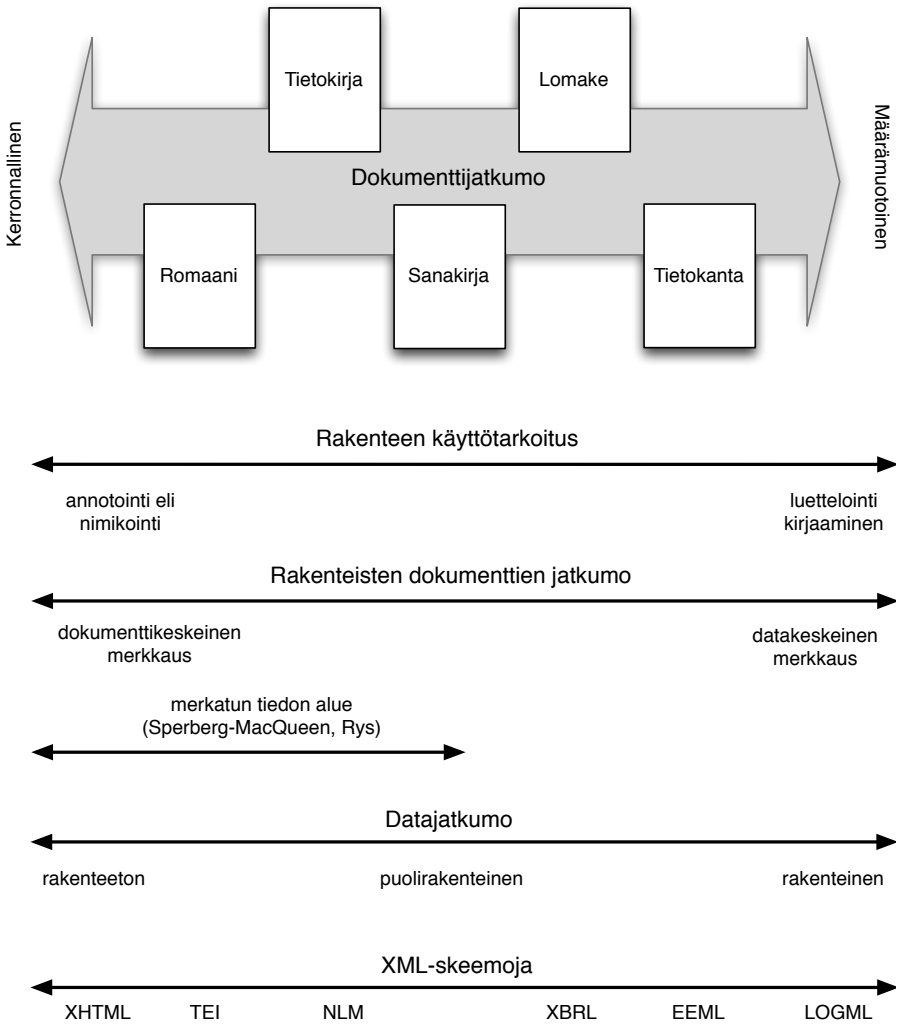
Vasemmalle päin osoittava kulmasulku merkkaa sulkutunnisteen alun. Tätä seuraa vinoviiva, elementin nimi ja oikealle päin osoittava kulmasulku, joka päättää sulkutunnisteen. Elementti, jolla ei ole tekstisisältöä tai lapsia on tyhjä, ja sitä voidaan merkitä avaustunnisteella jonka päättävää kulmasulkua edeltää vinoviiva.

Dokumentin lapsi-vanhempi-suhteessa olevat elementit voivat muodostaa rajoittamattoman syviä hierarkioita. Kaikkein ylimmällä tasolla sijaitsee dokumenttielementti, jota kutsutaan myös dokumentin juureksi. Dokumentilla voi olla vain yksi juuri.

3.2 Dokumentit ja tiedon rakenteisuus

Glushkon ja McGrathin (2005, 11) määritelmän mukaan dokumentti on jotain tarkoitusta varten oleva, itsenäinen kokoelma informaatiota. Määritelmä on teknologianeutraali, joten se kattaa fyysiseltä olomuodoltaan monimuotoisen joukon informaatiotallenteita.

Dokumentteja voidaan tarkastella jatkumona, jonka toisessa ääripäässä sijaitsevat kerronnalliset (narrative) dokumentit ja toisessa määrämuotoiset (transactional) dokumentit, kuten lomakkeet ja lokitiedot (Kuva 2). Jatkumon ääripäiden dokumenteilla on selkeä ominaisluonne ja ihmislukija voi vaivattomasti erottaa nämä dokumenttityypit toisistaan. Monet arkipäiväiset dokumentit, kuten sanakirjat, luettelot tai pöytäkirjat eivät kuitenkaan selkeästi kuulu kumpaakaan ryhmään, sillä niissä on sekä kerronnallisia että määrämuotoisia osia. On esitetty arvioita, että suurin osa maailmassa käsitellystä tiedosta kuuluisi tähän dokumenttijatkumon keskivaiheille sijoittuvien sekamuotoisten dokumenttien joukkoon. (Glushko & McGrath 2005, 10.)



Kuva 2: Dokumenttijatkumo. Lähde: Glushko 2005, mukaillen.

De Rose (1997) luonnehtii jatkumon ääripäiden dokumenttien piirteitä seuraavasti. Määrämuotoisille dokumenteille on luonteenomaista:

– Rakenteeltaan identtisten, mutta sisällöltään poikkeavien dokumenttien suuri määrä: esimerkiksi veroilmoituslomake, joka on jokaisen Suomen kansalaisen kohdalla samanlainen, mutta henkilöiden välillä uniikki.

– Järjestyksen merkityksettömyys sekä dokumentin tietosisältöjen että dokumenttien keskinäisen järjestyksen suhteen

– Kontekstin merkityksettömyys: jokainen kokonaisuudesta poimittu dokumentti on ymmärrettävissä erillään muista.

– Poikkeamat ennalta suunnitellusta tietosisältöjen rakenteesta ovat tulkittavissa virheiksi rakenteen suunnittelussa: esimerkiksi vapaamuotoiset kommentit määrämuotoisessa lomakkeessa vastausvaihtojen oltua riittämättömiä.

– Tietosisällön litteä rakenne tai matala hierarkia, jonka tietoalkioiden määrä on tunnettu ennalta.

Kerronnallisissa dokumenteissa puolestaan:

– Rakenne vaihtelee voimakkaasti dokumenttityypin sisällä, esimerkiksi on hyvin epätodennäköistä, että kaksi sattumanvaraisesti valittua kirjaa sisältävät saman määrän kappaleita ja lukuja.

– Poikkeamat dokumentin päärakenteesta ovat tavanomaisia, esimerkiksi pääosin kappaleista koostuva kirja voi sisältää alaviitteitä, sivuhuomautuksia ja lainauksia.

– Tietosisältöjen sarjallinen järjestys ja konteksti kantavat merkitystä, ja järjestyksen tai kontekstin muuttaminen voi vaikuttaa sisällön tulkintaan.

– Dokumentin tietosisällöt ovat toistuessaankin ainutkertaisia: esimerkiksi ei voida väittää, että samansisältöiden tekstikappale kirjan alussa ja lopussa olisivat identiteetiltään sama tekstikappale.

– Tietosisältö muodostaa syvän hierarkian.

De Rosen (1997) mukaan tiedon rakenteistaminen merkitsee tiedon tulkintaa – sen jakamista yhä pienempiin osiin ja näiden osien nimeämistä. Tällöin sisältöön kiinnitetään rakenne ja merkitys, joka muutoin saattaa olla olemassa kulttuurikonventiona. Esimerkiksi tekstin osio- ja kappalejako, listat ja alaviitteet ovat rakenteita, jotka ohjaavat tekstin tulkintaa ja joiden merkitys on selvä kulttuurikonventioiden kautta. Tiedon koneluettavuus tulee kuitenkin mahdolliseksi vasta rakenteen eksplisiittisen merkitsemisen myötä. (DeRose 1997.)

Tietojenkäsittelytieteissä dokumenttijatkumoa tarkastellaan tavallisesti käsitteiden rakenteeton tieto, puolirakenteinen tieto ja rakenteinen tieto kautta. Elmasri & Navathen (2011, 416) sekä Sperberg-McQueenin (2005) mukaan käsitteellä rakenteinen tieto viitataan tiedon lajeihin, jotka noudattavat jotain ennalta määrättyä tietomallia ja tyyppijärjestelmää. Silloin kun nämä ehdot eivät täyty, kyse on puolirakenteisesta tai rakenteettomasta tiedosta.

Puolirakenteinen tieto esiintyy moninaisissa muodoissa. Elmasri ja Navathen mukaan olennaisin ero puolirakenteisen ja rakenteisen tiedon välillä on, että puolirakenteisessa tiedossa tiedon kuvaus ja tietosisällöt esiintyvät limittäin, eikä tietosisältöjen arvoja ole kiinnitetty tyyppijärjestelmään (Elmasri & Navathe 2011, 416). Sperberg-McQueenin (2005) määritelmä on miltei sama: puolirakenteiset tietosisällöt koostuvat rakenteisista osista, mutta tietosisällöstä riippuen rakenneosien kokoonpano voi vaihdella eikä niiden tyyppiä tunneta. Hän antaa esimerkkinä osoitetiedon: jonkin tietosisällön yhteydessä osoite voi olla ilmaistuna merkkijonona, jossain toisessa objektina, jolla on oma sisäinen rakenteensa. Rysin (2003) mukaan puolirakenteiselle tiedolle tyypillisiä piirteitä ovat rakenteen suuri vaihtelu ja hierarkkiset, toistuvat rakenteet. Rakenteettomasta tiedosta nämä piirteet puuttuvat kokonaan. (Rys 2003.)

Sperberg-MacQueen (2005) sekä Rys (2003) täydentävät edellä kuvattua kolmijakoista kategorisointia neljännellä, merkatun tiedon kategorialla. Kategoria on tulkittavissa puolirakenteisen tiedon erityistapaukseksi. Sperberg-MacQueen määrittelee merkatun tiedon olevan tekstiä, jossa on joitain rakenteeseen liittyviä merkintöjä. Rysin (2003) mukaan puolirakenteisen ja merkatun tiedon välillä on olennainen ero. Merkatun tiedon tapauksessa tekstiin perustuva informaatiosisältö on ihmislukijan tulkittavissa, vaikka merkkaus poistettaisiin. Rakenteisen ja puolirakenteisen tiedon tapauksessa merkkaus kantaa tulkinnan kannalta olennaista tietoa, jolloin sen poistaminen hävittää tekstin merkityssisällön. Sisältö voi olla edelleen ihmisen tulkittavissa, mutta vain suurta vaivaa nähden. (Rys 2003.)

XML-esitysmuotoa voidaan soveltaa koko dokumenttijatkumon yli. Relevanssilajittelevasta XML-tiedonhausta on kuitenkin suurin hyöty dokumenttijatkumon keskivaiheilla, jolle sijottuvat dokumentit voidaan katsoa kuuluvan edella kuvattun *merkatun tiedon* kategoriaan (Kuva 2). Merkatun tiedon kategoriaan kuuluvien XML-dokumenttien elementit voivat sisältää toisten elementtien lisäksi tekstisisältöä (Listaus 2).

Listaus 2: TEI-merkattu katkelma näytelmästä Edward II[1]

```
<sp who=”prin”>
<speaker>Prince</speaker>
<l>I thinke king <name>Edward</name> will out-run us all.</l>
</sp>
```

Kuten edellä todettiin, XML-esitysmuotoa voidaan käyttää koko dokumenttijatkomon yli. Myös dokumenttijatkumon kerronnalliset dokumentit voivat noudattaa jotain ennalta määrättyä kaaviota, jos näi-

[1] http://teibyexample.org/examples/TBED07v00.htm?target=marlowe

den rakenne on merkitty jollain yleisesti tunnetulla tavalla koneluettavassa muodossa. Tällaisen skeeman sisältöä ei kuitenkaan ole määritelty tiukasti; esimerkiksi XHTML-skeema ei määritä, millä arvoalueilla elementtien sisällön tulee olla. Dokumenttijatkumon määrämuotoisten dokumenttien sisältö voidaan taas määritellä tiukasti, arvoalueet kiinnittäen.

Sitä, kuuluuko dokumentin tietosisältö rakenteettoman, puolirakenteisen tai rakenteisen tiedon kategoriaan, ei voida päätellä pelkästään sen perusteella, että dokumentti noudattaa XML-muotoa. Tästä syystä tässä työssä vältetään käyttämästä käsitettä *rakenteiset dokumentit*, joka on jollain merkkauskielellä kuvattuihin dokumentteihin viittaava vakiintunut, mutta epätäsmällinen käsite. Käsitteen ongelmana on siinä luotu yhteys merkkauskielellä tapahtuvan tiedon kuvaamisen sekä kuvatun tiedon lajin välille. Tässä työssä, tarkasteltaessa XML-esitysmuotoon tallennettujen dokumenttien tiedonhakua, haun kohteena olevien dokumenttien oletetaan kuuluvan merkatun tiedon kategoriaan.

4 Keskeiset käsitteet

Arvioitaessa järjestelmän käytettävyyttä usein viitataan karkeasti käyttäjiin, ohjelmointiin ja ohjelmointitaitoihin. Nämä ovat kuitenkin monisyisiä käsitteitä, koska käyttäjiä on eri tasoisia ja ohjelmointi voi olla eri tasoista. Tässä luvussa paneudutaan näihin ja tutkielman kannalta muihin olennaisiin käsitteisiin.

4.1 Loppukäyttäjät

Tietojärjestelmäkirjallisuudessa on lukuisia loppukäyttäjän määritelmiä. Jokainen niistä tunnistaa joukon ominaisuuksia, jotka luonnehtivat käyttäjien tehtäviä ja tehtävien suorittamisessa vaadittavia taitoja. Suoraviivaisimmillaan loppukäyttäjät jaetaan kahteen pääjoukkoon: asiantuntijakäyttäjiin (expert users), joilla on tietoteknistä kokemusta ja kokemattomiin käyttäjiin (novice users; naive users), joilla ei ole erityiskoulutusta tietotekniikan näkökulmasta.

Kaksiluokkainen jaottelu hukkaa kuitenkin paljon yksityiskohtia, jotka voivat auttaa ymmärtämään, mitä loppukäyttäjän käsitteellä tarkkaan ottaen tarkoitetaan. Tässä luvussa esiteltäviä loppukäyttäjäluokitteluja yhdistää näkemys siitä, että loppukäyttäjyys on pikemminkin työtehtävärooli, kuin tietotekniseen osaamistasoon liittyvä ominaisuus.

Käsiteltävät käyttäjäluokittelut kuvaavat alkuperäisyhteydessään yleensä *tietokantojen* loppukäyttäjiä. Käyttäjäryhmäkuvaukset on tulkittu tässä yhteydessä laajemmin niin että niiden ymmärretään kuvaavan minkä tahansa *tietojärjestelmän* loppukäyttäjiä.

Rockart ja Flanneryn (1983) käyttäjäluokittelu perustuu loppukäyttäjien ($n = 200$) ja IT-ammattilaisen ($n = 50$) haastatteluihin, joita tehtiin yhteensä seitsemässä eri organisaatiossa. Haastatteluaineiston analyysin pohjalta syntyi kuusikohtainen käyttäjäluokittelu, jonka luokkia ovat *ei-ohjelmoivat loppukäyttäjät*, *komentokielikäyttäjät*, *loppukäyttäjäohjelmoijat*, *edistyneet loppukäyttäjäohjelmoijat*, *loppukäyttäjätukihenkilöt* sekä *sovelluskehittäjät*.

- *Ei-ohjelmoivat loppukäyttäjät* (non-programming end users) tarvitsevat työssään ainoastaan valmisohjelmia, joita he käyttävät valikkojen tai yksinkertaisen, rajoitetun komentokielen kautta (Rockart & Flannery 1983, 6). Nykyisten toimistosovellusten kontekstissa ei-ohjelmoiva loppukäyttäjä voisi olla esimerkiksi taulukkolaskentaa tai tilasto-ohjelmistoa käyttävä henkilö, jonka tarvitsemat toiminnot ovat suoraan esillä ohjelmiston käyttöliittymässä.

- *Komentokielikäyttäjien* (command level users) tiedontarpeet vaihtelevat ja tästä syystä valmisohjelmien oletustoiminnot eivät ole riittäviä käyttäjäryhmän työtehtävien näkökulmasta. Komentokielikäyttäjät ovat motivoituneita perehtymään työssään tarvitsemiinsa ohjelmiston osakokonaisuuksiin syvällisesti. He eivät ole kuitenkaan kiinnostuneet ohjelmistosta itsessään, vaan työtehtävät ovat ohjelmiston opiskelun ensisijainen motiivi. (Rockart & Flannery 1983, 6.) Toimistosovellusten kontekstissa tähän käyttäjäryhmään voidaan lukea kuuluvaksi esimerksi makroja tai säännöllisiä lausekkeita työtehtävissään hyödyntävät henkilöt.

- *Loppukäyttäjäohjelmoijat* (end user programmers) käyttävät ohjelmointikieliä työkaluna. Tähän käyttäjäryhmään kuuluvien henkilöiden työtä ovat analyyttisiset tehtävät, joiden hoitaminen vaatii ohjelmointitaitoa. Näissä tehtävissä syntyvät ohjelmat ovat kuitenkin vain työn oheistuote ja tyypillisesti ainoastaan henkilökohtaisessa käytössä. Sovelluksilla voi kuitenkin olla myös muita käyttäjiä. (Rockart & Flannery 1983, 6.) Toimistosovellusten ohjelmointiominaisuuksien tehokäyttäjiä voidaan pitää tämän käyttäjäryhmän tavallisimpana edustajina.

- *Edistyneet loppukäyttäjäohjelmoijat* (functional support personnel) hallitsevat työssään tarvitsemansa ohjelmointityökalut suvereenisti ja ovat tästä syystä työyhteisönsä epävirallisen loppukäyttäjäohjelmoinnin tukihenkilön asemassa. Tähän ryhmään kuuluvat henkilöt eivät miellä itseään kuitenkaan ohjelmoijiksi, vaikka käyttäisivät suuren osan työajastaan ohjelmoinin kaltaisten tehtävien parissa ja työn sivutuloksena syntyneillä ohjelmilla olisi laaja käyttäjäkunta. (Rockart & Flannery 1983, 6.)

- *Loppukäyttäjätukihenkilöt* (end user computing support personnel) työskentelevät tyypillisesti organisaation IT-tuessa. Järjestelmien ylläpitoon liittyvien tehtävien lisäksi tähän ryhmään kuuluvat henkilöt kehittävät apuohjelmistoja ja työkaluja muita käyttäjiä varten. (Rockart & Flannery 1983, 6.)

- *Sovelluskehittäjät* (database programmers) ovat organisaation palveluksessa olevia ohjelmoijia, joiden tehtävänä on kehittää organisaation sisäiseen käyttöön tarkoitettuja sovelluksia, jotka vähentävät johdon tarvetta hankkia ohjelmisto-osaamista ulkopuolisilta asiantuntijoilta. (Rockart & Flannery 1983, 6.)

Rockart ja Flannery kuitenkin korostavat, että loppukäyttäjät ovat heterogeeninen joukko ja stereotyyppistä loppukäyttäjää ei ole olemassa. Eri käyttäjäryhmiin kuuluvien henkilöiden käyttötarpeet vaihtelevat, kuten myöskin tarvittavan käyttäjäkoulutuksen määrä. (Rockart & Flannery 1983, 8.)

Catarci (1995) jakaa loppukäyttäjät kolmeen ryhmään – kokemattomiin (naive), kokeneisiin (intermediate) ja eksperttikäyttäjiin. Kokemattomat käyttäjät eivät ole halukkaita opiskelemaan järjestelmän käyttöä ja olettavat käyttöliittymän opastavan heitä. Kokeneilla käyttäjillä on teknistä tietojenkäsittelyn ymmärrystä, mutta tietämys on luonteeltaan yleistä eikä liity käytössä olevaan järjestelmään. Eksperttikäyttäjän tietämys on laajaa: hän tuntee useita eri ohjelmointi- ja kyselykieliä ja kokee tärkeäksi hallita käytössä olevan järjestelmän läpikotaisin. (Catarci & Santucci 1995.)

Jarken ja Vassiloun (1985) käyttäjäluokittelun kriteerit perustuvat kirjallisuuteen. Kriteereitä ovat *ohjelmointitietämys*, *kyselykielen käyttötiheys*, *sovellusaluetietämys*, ja *työtehtävissä tarvittavien toimintojen lukumäärä*. Ohjelmointitietämys ja käyttötiheys määrittävät yhdessä luokittelun *vuorovaikutustaito*-ulottuvuuden kun taas sovellusaluetietämys ja työtehtävissä tarvittavien toimintojen lukumäärä määrittävät *tehtävärakenne*-ulottuvuuden. (Jarke & Vassiliou 1985.)

Ohjelmointitietämys on Jarken ja Vassiloun mukaan yleisemmällä tasolla oleva käsite kuin tyypillisesti tehty jako ohjelmoijiin ja ei-ohjelmoijiin. Tämä voi johtaa epäyhteneviin tulkintoihin, koska ohjelmoijuus on voitu määritellä eri yhteyksissä eri tavoin. Ohjelmointitietämys on yhteydessä henkilön loogis-matemaattiseen ymmärrykseen sekä myönteiseen käsitykseen tietotekniikasta. Käyttötiheys määrittää mielekkäänä pidettävän järjestelmäkoulutuksen määrän – mitä enemmän järjestelmää tullaan käyttämään, sitä suurempi alkuinvestointi opiskeluun on perusteltua. Koulutuspanos määrittää järjestelmän käyttäjän oletetun lähtötason. Sovellustietämyksellä viitataan käyttäjän tietokannan rakennetta ja sisältöä koskevan mentaalisen mallin kattavuuteen. (Jarke & Vassiliou 1985.)

Vuorovaikutustaito ja tehtävärakenne -ulottuvuudet määrittävät neljä käyttäjätyyppiä: *satunnais-*, *rutiini-*, *johtoporras-* ja *spesialistikäyttäjät* - ja näille *kokematon*, *kokenut* ja *ekspertti* -taitotasot.

Jarken ja Vassiloun määrittelemät käyttäjäluokat ovat:

- *Satunnaiskäyttäjät* (casual users) eivät ole kiinnostuneet järjestelmän yksityiskohdista. Heille tarkoitetun käyttöliittymän tulisi perustua valikoihin, mutta antaa kuitenkin mahdollisuus taitotasoltaan edistyneelle satunnaiskäyttäjälle hyödyntää yksinkertaista kyselykieltä tai pikavalintoja. (Jarke & Vassiliou 1985.)
- *Rutiinikäyttäjät* (clerical users) tuntevat tarkasti vastuullaan olevat työtehtävät ja hallitsevat järjestelmän näiltä osin hyvin. Rutiinikäyttäjien työtehtävien vaihtelevuus on matala ja tästä syystä heidän on tarpeen osata käyttää vain rajattua osaa järjestelmän toiminnoista. (Jarke & Vassiliou 1985.)

– *Johtotason käyttäjät* (managerial users) tarvitsevat työssään monimutkaisia, vaihtelevasisältöisiä raportteja mutta kokevat olevansa liian kiireisiä käyttääkseen aikaa järjestelmän opetteluun. Heille sopiva käyttöliittymä on valikkopohjainen, ja niiltä osin kun valikkopohjainen järjestelmä ei kykene vastaamaan kaikkiin tarpeisiin, raporttien laatiminen annetaan tietokanta-asiantuntijan tehtäväksi. (Jarke & Vassiliou 1985.)

– *Sovellusspesialisti* (application specialist) tuntee järjestelmän läpikotaisin ja tarvitsee työssään monipuolisesti eri toimintoja. Yleensä spesialistin asemaan päädytään rutiinitehtävien kautta, rutiinikäyttäjän tehtäväkuvan laajenemisen myötä. Vaikka spesialistikäyttäjä on käytössään olevan järjestelmän asiantuntija, tyypillisesti hänellä ei kuitenkaan ole ohjelmointitaustaa. (Jarke & Vassiliou 1985.)

Jarken ja Vassiloun mukaan formaalit kyselykielet soveltuvat parhaiten sovellusaluespesialistien käyttöön. Heidän mukaansa tulevaisuuden kyselykäyttöliittymien tulisi tarjota useita vaihtoehtoisia vuorovaikutustapoja, sillä yksi vuorovaikutustapa ei kata kaikkien loppukäyttäjäryhmien tarpeita. (Jarke & Vassiliou 1985.)

Elmasrin ja Navathen (2011) määritelmän mukaan loppukäyttäjät ovat tietojärjestelmän käyttäjiä, joiden työtehtävät vaativat järjestelmän tietosisällön käsittelyä eri tavoin - tiedonhakua, tietojen päivittämistä ja raportointia. He jakavat loppukäyttäjät neljään ryhmään, joita ovat satunnaiskäyttäjät, parametrikäyttäjät, edistyneet käyttäjät sekä valmisohjelmakäyttäjät.

– *Satunnaiskäyttäjien* (casual end users) tiedontarpeet ovat vaihtelevia ja epäsäännöllisiä. He tarvitsevat tietojärjestelmän sisältämiä tietoja vain silloin tällöin ja tiedontarpeissa käyttökertojen välillä ei ole toistuvuutta. Satunnaiskäyttäjät ovat tyypillisesti organisaation keski- tai ylintä johtoa, jotka ovat vuorovaikutuksessa tietojärjestelmän kanssa monipuolisen kyselykielen kautta. (Elmasri & Navathe 2011, 15.)

– *Parametrikäyttäjät* (naive or parametric end users) muodostavat suuren osan loppukäyttäjien kokonaismäärästä. Käyttäjäryhmään kuuluvien henkilöiden työtehtävät muodostuvat rutiinitoimista, jotka vaativat jatkuvaa vuorovaikutusta tietojärjestelmän kanssa. Parametrikäyttäjät eivät laadi tarvitsemiaan komentolauseita itse, vaan käyttävät järjestelmää esivalmisteltujen ja testattujen kyselyiden kautta, joihin on rakennettu oma tehtäväspesifinen käyttöliittymänsä. Parametrikäyttäjät työskentelevät esimerkiksi tilauskäsittelyn tai henkilöstöhallinnon tehtävissä. (Elmasri & Navathe 2011, 15.)

– *Edistyneiden käyttäjien* (sophisticated end users) työtehtävät ovat monimutkaisia ja niiden suorittaminen vaatii kykyä rakentaa itse työssä tarvittuja ohjelmistoja. Tähän käyttäjäryhmään voi kuulua esimerkiksi insinöörejä, analyytikkoja ja tutkijoita, jotka ovat motivoituneita tuntemaan käytössään olevat tietojärjestelmät perusteellisesti. (Elmasri & Navathe 2011, 16.)

– *Valmisohjelmakäyttäjät* (standalone users) tallentavat ja käsittelevät tietoja jotain tiettyä käyttötarkoitusta varten suunnitellun ohjelmiston avulla. Tyypillisesti valmisohjelmassa on graafinen tai valikkoihin perustuva käyttöliittymä, joka on räätälöity tukemaan ohjelmiston aiottua käyttötarkoitusta, kuten henkilökohtaista taloudenpitoa. (Elmasri & Navathe 2011, 16.)

Shneiderman (1978) jakaa loppukäyttäjät kolmeen ryhmään käyttäjien oletetun sovellusaluetietämyksen mukaan.

– *Harjaantumattomat satunnaiskäyttäjät* (nontrained intermittent users) eivät hallitse kyselykielten syntaksia ja myös käsitys tietojärjestelmäkokonaisuudesta on hatara. He tuntevat tietojärjestelmään tallennetun tiedon sovellusalueen, mutta heidän kykynsä esittää järjestelmän kannalta oikein muotoiltuja kysymyksiä on rajallinen. Järjestelmän suorakäyttö kyselykielen avulla voi tuntua tähän käyttäjäryhmään kuuluvista henkilöistä ahdistavalta. Tästä syystä harjaantumattomille satunnaiskäyttäjille suunnatun käyttöliittymän tulisi olla opastava tai vuorovaikutuksen tulisi tapahtua asiantuntijan välityksellä. (Shneiderman 1978, 422.)

– *Harjaantuneet säännölliset käyttäjät* (skilled frequent users) ovat vuorovaikutuksessa järjestelmän kanssa päivittäin. Käyttäjäryhmään kuuluvat henkilöt ovat halukkaita oppimaan tarvitsemiensa toimintojen syntaksin, mutta ovat kuitenkin kiinnostuneita ensisijaisesti omasta työstään kuin tietojärjestelmästä itsestään. Järjestelmän helppokäyttöisyys on käyttötiheyden kannalta ratkaisevassa asemassa – miellyttäväksi koettua järjestelmää käytetään usein. Käyttäjäryhmään kuuluvien henkilöiden työrooleja voivat olla esimerkiksi sihteeri, insinööri tai johtaja. (Shneiderman 1978, 423.)

– *Tietojärjestelmäammattilaisten* (professional database users) tehtävänä on tarjota muihin käyttäjäryhmiin kuuluville henkilöille pääsy tietokantaan. Käyttäjäryhmänä tietojärjestelmäammattilaiset ovat kiinnostuneet työnsä tehokkuudesta ja laadusta. (Shneiderman 1978, 423.)

Tässä työssä käsiteltävä relevanssilajittelevaan XML-tiedonhakuun tarkoitettu XIL-kieli palvelee käyttötilanteita, joissa henkilön on saatava ymmärrys suuresta, jotain rakennetta noudattavasta tekstimassasta

mahdollisimman nopeasti, ilman että tilanteessa on mahdollista odottaa tietojärjestelmäasiantuntijoiden kehittävän aineiston käsittelyä varten valikkopohjaisia työkaluja.

Seuraavaksi hahmotellaan mahdollisia XIL-kielen käyttöskenaariota. Tämän perusteella pohditaan, mihin kirjallisuudessa esitettyihin käyttäjäryhmiin käyttöskenaarioiden henkilöt kuuluvat. XIL-kielen mahdollisia käyttöskenaarioita ovat esimerkiksi:

- *Journalistinen työ*. Journalistin tehtävänä on käydä lävitse aineistomassa, joka on XML-muotoista, dokumentoimatonta raakadataa suoraan väärinkäytöksistä epäillyn julkisorganisaation asiakirjajärjestelmästä. Journalisti on saanut aineiston käyttöönsä julkisuuslakiin vedoten, ja lain kirjaimen täyttääkseen organisaatio on luovuttanut tiedot raakamuodossa, mutta ei ole vaadittua enempää yhteistyöhaluinen.
- *Organisaatiotarkastus*. Yritysanalyytikon on osana yrityksen sisäistä tarkastusta seulottava lävitse yhtiön intranet-viestinnässä käytetyn pikaviestinohjelman loki- ja viestitietoja useiden vuosien ajalta.
- *Kartoittava tutkimus*. Tutkija on saanut haltuunsa suuren määrän XML-merkattua haastatteluaineistoa ja haluaa ennen rahoitushakemuksen laadintaa varmistua aineiston hyödynnettävyydestä mielessään olevan tutkimushankkeen näkökulmasta.

Näitä käyttäjäryhmiä yhdistää käsiteltävän aineiston suuri määrä ja heterogeenisyys, voimakas aikapaine ja se, ettei käsillä olevaa aineistoa voi ymmärtää kukaan muu kuin kyseisen aihealueen asiantuntija. Tästä syystä käyttäjien on pystyttävä käsittelemään aineistoa itsenäisesti.

Suhteessa kirjallisuudessa esitettyihin loppukäyttäjäkuvauksiin XIL-kielen potentiaalisia käyttäjiä voidaan luonnehtia seuraavien käyttäjäryhmäkuvausten kautta:

- *Satunnainen loppukäyttäjä* - tiedontarpeet vaihtelevat ja järjestelmän käyttö on epäsäännöllistä (Elmasri & Navathe 2011).
- *Komentokielikäyttäjä* – valmisohjelmien oletustoiminnot eivät ole riittävän joustavia vaihtelevien työtehtävien näkökulmasta (Rockart & Flannery 1983).
- *Loppukäyttäjäohjelmoija* – työtehtävät ovat luonteeltaan analyyttisiä ja vaativat ohjelmointitaitoa (Rockart & Flannery 1983)
- *Kokenut käyttäjä* – on substanssialansa asiantuntija ja hallitsee tietojenkäsittelyn perusteet (Catarci & Santucci 1995).

– *Sovellusaluespesialisti* – tuntee käyttämänsä työkalut läpikotaisin, mutta on taustaltaan substanssialansa asiantuntija (Jarke & Vassiliou 1985).

– *Edistynyt käyttäjä* – työtehtävät vaativat ohjelmointitaitoa ja kykyä rakentaa itse työssä tarvittuja ohjelmistoja (Elmasri & Navathe 2011, 16.).

– *Harjaantunut säännöllinen käyttäjä* – on motivoitunut harjoittelemaan tarvitsemiensa työkalujen käyttöä suoriutuakseen työstään entistä paremmin (Shneiderman 1978).

XIL-kielen potentiaaliset käyttäjät eivät ole tietojenkäsittelyn ammattilaisia. Heidän ammattinimekkeensä vaihtelevat, mutta yhteisenä nimittäjänä on työn tietointensiivinen luonne: tarve käsitellä ja ymmärtää suuria määriä tekstimuotoista tietoa. Tämä käyttäjäluokka tulee kasvamaan tulevaisuudessa, sillä esimerkiksi journalistiikan ja perinteisten humanististen tieteiden piirissä on havahduttu huomaamaan mahdollisuudet, joita tietoteknisten työvälineiden aiempaa monipuolisempi hyödyntäminen voi tarjota. Eräs merkki tästä kehityksestä ovat koulutusohjelmat, joissa yhdistetään ei-teknisen alan substanssiosaaminen tietojenkäsittelytieteelliseen näkemykseen (kts. London Centre for Digital Humanities. 2011, Codrea-Rado 2012).

4.2 Ohjelmointi

Ohjelmointi voidaan ymmärtää laajassa ja suppeassa merkityksessä. Ohjelmointi suppeassa merkityksessä on määritelty toiminnaksi, jossa laaditaan joukko proseduraalisia ohjeita jonkin toiminnan suorittavista askeleista kielellä, jolla kirjoitetut ilmaukset käännetään tai tulkitaan toimivaksi sovellusohjelmaksi (Ko et al 2011).

Ymmärrettäessä ohjelmoinnin käsite laajemmin, ohjelmointina voidaan pitää myös halutun lopputuloksen kuvaamista esimerkiksi graafisten diagrammien tai sanallisten määritelmien avulla, niin ettei kuvailussa oteta kantaa siihen, mitä askeleita noudattaen lopputulokseen tulisi pyrkiä (Nardi 1993, 5). Laajasti ymmärrettynä ohjelmoinnin käsite kattaisi siis toiminnan alkaen verkkosivujen rakenteen ja ulkoasun määrittelystä XHTML ja CSS-kielen avulla, ulottuen sovellusohjelmointiin Javan tai C:n kaltaisilla yleisillä ohjelmointikielillä (Fowler 2010, Ko et al 2011).

Ko ja muut (2011) määrittelevät ohjelmoinnin toimintana ohjelman käsitteen kautta. Kon ja muiden määritelmän mukaan ohjelma on joukko laskentaan kykenevän laitteen ajettavissa tai tulkattavissa olevia kuvauksia, jotka voivat ottaa vastaan vaihtelevia syötteitä. Ohjelmoinnin käsite viittaa tämän kuvauksen kirjoittamisen prosessiin. (Ko et al 2011.)

Ohjelman ja ohjelmoinnin käsitteet laajasti ymmärrettynä liittyvät suureen kirjoon tavanomaisia ihmisen ja koneen välisiä vuorovaikutustilanteita, televisiolähetyksen tallennuksen ajastamisesta neulekoneen ohjaamiseen reikäkorttien avulla (vrt. Nardi 1993, 34). Näin ymmärrettynä ohjelmoinnin käsite kattaa myös kyselyiden kirjoittamisen, ja kyselykieliä voidaan pitää yhtenä ohjelmointikielten tyypeistä.

4.3 Deklaratiivisuus

Deklaratiivisen paradigman tulo tietojenkäsittelyyn ja relationaalisen tietomallin keksiminen liitetään yleisesti yhteen (Manthey 1990). Relaatiomallin deklaratiivinen tiedon esitys- ja kyselytapa kätki tietokannan matalan tason tietorakenteet käsitteellisesti yksinkertaisemman abstraktion sisään.

Tietokoneohjelmaa voidaan luonnehtia niin, että se on kuvaus jonkin tietyn ongelman ratkaisusta. Kuvaus voidaan antaa joko askeleittain (proseduraalisesti) tai kuvailla lopputulos (deklaratiivisesti). Proseduuri on sarja ongelman ratkaisevia askeleita tietyssä sarjallisessa järjestyksessä. Deklaratiivisessa, nonproseduraalisessa ohjelmassa ohjelmoija ei ota kantaa siihen, missä järjestyksessä ongelmanratkaisussa tulisi edetä.

Weltyn (1981) mukaan ohjelmointikielet eivät ole deklaratiivisia tai proseduraalisia absoluuttisessa mielessä, vaan kielillä on proseduraalisia ja nonproseduraalisia piirteitä suhteessa toisiinsa. Weltyn esimerkin mukaan FORTRAN-kielellä kirjoitettu ohjelma on deklaratiivinen verrattaessa sitä toiminnallisuudeltaan vastaavaan, assembly-kielellä laadittuun ohjelmaan, mutta proseduraalinen jos vertailukohtana on vastaava APL-kielinen ohjelma (Welty & Stemple 1981). Leavenoworth (1974) korostaa lisäksi nonproseduraalinen/deklaratiivinen -käsitteen aikasidonnaisuutta – käsitteen sisältö muuttuu ohjelmointikielten kehittyessä.

Leavenoworth (1974) luettelee joukon ohjelmointikielten piirteitä, jotka vähentävät kielen proseduraalisuutta. Näitä ovat esimerkiksi kokonaisia alkiojoukkoja käsittelevät koosteoperaatiot, operaatioiden vapaa järjestys ja assosiatiivinen viittaaminen ohjelman käsittelemiin arvoihin. Leavenoworthin mukaan tutkijat eivät ole yksimielisiä siitä, voidaanko näiden tai muiden syntaktisten piirteiden pohjalta kehittää yleiskäyttöinen kielen proseduraalisuuden astetta kuvaava metriikka. Welty ja Stemple (1981) esittävät yhden mahdollisen metriikan, jota muodostettaessa lasketaan vertailtavilla kielillä laadittujen toiminnallisuudeltaan identtisten ohjelmien sidottujen muuttujien lukumäärä sekä operaatioiden ja operaatioiden mahdollisten järjestysten lukumäärä. Graaumansin (2005, 27,41) mukaan tämä tapa määrittää

kielen proseduraalisuus ei ole kuitenkaan yleiskäyttöinen, vaan sidoksissa Weltyn ja Stemplen alkuperäistutkimuksissa vertailemiin kieliin.

Kielen deklaratiivisuuden on katsottu parantavan ammattimaisten ohjelmoijien tuottavuutta ja edistävän muiden kuin tietojenkäsittelyn ammattilaisten mahdollisuuksia hyödyntää kieltä työssään (Lloyd 1994). Toisaalta saatavilla oleva tutkimusnäyttö viittaa siihen, ettei deklaratiivisuuden asteen kasvattamisesta ole välttämättä oletettuja hyötyjä, vaan kielen proseduraalisista piirteistä on pikemminkin etua kaikissa muissa paitsi kaikkein yksinkertaisemmissa käyttötilanteissa. (Soloway 1981; Welty & Stemple 1981). Markstrumin (2010) mukaan on tyypillistä, että ohjelmointikieliä esittelevissä julkaisuissa esitettävät väitteet kielten paremmuudesta suhteessa aiempiin ratkaisuihin jätetään perustelematta. Käsitys deklaratiivisen ilmaisun luonnollisuudesta ja yksinkertaisuudesta verrattuna proseduraaliseen ilmaisutapaan voikin olla tutkija- ja kehittäjäyhteisön jakama uskomus, eikä niinkään tutkimusnäyttöön perustuva fakta.

4.4 Sovellusaluekielet

Ohjelmointikieliä voidaan luokitella eri tavoin. Eräs tapa on jakaa kielet yleisiin ohjelmointikieliin (general purpose languages) ja sovellusaluekieliin (domain spesific languages, myös *little languages* kts. Bentley 1986).

Sovellusaluekielet ovat ilmaisuvoimaltaan rajattuja, tietyn sovellusalueen tietojenkäsittelyongelmien ratkaisua varten luotuja formaaleja, tietokoneen tulkittavaksi tarkoitettuja kieliä (Fowler 2010, 27). Olennaisinta sovellusaluekielissä on tietoisesti rajattu ilmaisuvoima, eikä niinkään tarkka sovellusaluesuuntautuneisuus (Fowler 2010, 27).

Yleinen ohjelmointikieli tarjoaa osaavalle käyttäjälle paljon mahdollisuuksia, mutta kielen voimakas ilmaisuvoima voi tehdä siitä hankalan oppia ja käyttää. Sovellusaluekieli tarjoaa käyttäjälleen minimimäärän ominaisuuksia, joita tarvitaan sovellusalueen tiettyjen ongelmien ratkaisemisessa. Sovellusaluekielellä ei voida rakentaa kokonaista tietojärjestelmää, mutta sitä voidaan käyttää toteutettaessa jokin osa järjestelmästä. (Fowler 2010, 27.)

Tietojärjestelmien kehitystyön lisäksi sovellusaluekielet voivat palvella myös loppukäyttäjiä. Esimerkiksi toimisto-ohjelmien makrokielet ovat sovellusaluekieliä. Makrokielten käyttöalue on tarkkaan rajattu, joten niistä puuttuu yleisten ohjelmointikielten ilmaisuvoima, mutta samalla ne ovat yleisiin kieliin nähden paljon helpompia oppia (Nardi 1993, xii.)

Eräänä sovellusaluekielen tunnusmerkkinä on pidetty sitä, ettei kieli ole Turing-täydellinen. Mikäli tämä tunnuspiirre on ehdoton, kuuluvat esimerkiksi XML-dokumenttien muunnoksiin tarkoitettu XSLT-muunnoskieli ja tilastotieteellistä työtä varten kehitetty R-kieli kuuluvat yleiskielten joukkoon, sillä kummatkin kielet ovat Turing-täydellisiä[2] [3]. Yleensä sovellusaluekielissä vältetään tavanomaisten imperatiivisten kontrollirakenteiden käyttöä, eivätkä kielet tue muuttujia tai alirutiineja, kun taas R-kieli ja XSL-muunnoskieli sisältävät kaikki nämä piirteet.

Fowler (2010) korostaa sovellusaluekielen käsitteen sumeutta, sillä ohjelmointikielen käyttötilanteesta ja -tavasta riippuen sitä voidaan pitää joko sovellusaluekielenä tai yleisenä ohjelmointikielenä. Esimerkiksi XSL-kieltä voidaan pitää yleisohjelmointikielenä, jos sillä ratkaistaan jokin yleinen tietojenkäsittelytieteellinen ongelma ja sovellusaluekielenä, jos kieltä käytetään dokumenttimuunnoksen välineenä. (Fowler 2010, 30.)

Nardin (1993, 39) mukaan sovellusaluekielen tulisi olla tehtäväspesifi, siten että kielen primitiivit perustuvat sovellusalueelta tuttuihin käsitteisiin. Tehtäväspesifeistä primitiiveistä on etua etenkin silloin, kun sovellusaluekielen käyttäjiä ovat loppukäyttäjät. Tällöin loppukäyttäjät voivat suoraan kuvailla sovellusalueen ilmiöitä kielen korkean tason operaatioilla, eikä heidän tarvitse pohtia, kuinka haluttu toiminnallisuus saadaan aikaan matalan tason kontrollirakenteilla. (Nardi 1993, 39.)

4.5 Kyselykieli

Kysely on tietojärjestelmälle suunnattu tiedontarpeen ilmaus (Manning et al 2008, 5). Useimmat kyselykielet ovat tekstimuotoisia, eli kyselyt ilmaistaan tekstinä, jota kyselyn käsittelevä ohjelmisto tulkitsee (Risch 2009, 2261). Tekstimuotoisten kyselykielten ohella on olemassa myös graafisia tai valikkoihin ja komentoihin perustuvia kyselykieliä (Risch 2009, 2261).

Ulkoisesta muodostaan riippumatta kyselykielet ovat ohjelmointikieliä, kun ohjelmointikieli-käsite ymmärretään kuten edellä on kuvattu. Kyselykieli on ilmaisuvoimaltaan rajattu sovellusaluespesifinen (tässä yhteydessä käytetään myös käsitteitä sublanguage, special purpose language) ohjelmointikieli, jonka käyttötarkoitus on tiedonhaku tietokannasta (Query language. 2011, Reisner 1988). Laajimmin käytetty ja tunnetuin kyselykieli on relaatiotietokantojen SQL-kieli.

[2] `http://www.w3.org/TR/xslt`
[3] `http://http://www.r-project.org/`

Alkujaan kyselykieli-käsitteellä viitattiin vain hakuihin kykenevään kieliin (Risch 2009, 2261). Ajateltiin myös, että kyselykielen tyypillinen käyttäjä ei ole ohjelmoija, vaan jonkin muun alan ammattilainen (Reisner 1988, 258). Tämä näkökohta pyrittiin ottamaan huomioon suunnittelemalla kyselykielistä sellaisia, että järjestelmän on mahdollista tuottaa kyselyyn vastaus, kunhan tiedonhakija kuvailee, mitä tietoja ollaan hakemassa. Kyselykielistä pyrittiin suunnittemaan deklaratiivisia, korkean abstraktiotason kieliä.

Aikojen saatossa kyselykielten ilmaisuvoimaa on kasvatettu ja kyselykieli-käsitteen ala on laajennut tämän kehityksen mukana. Kyselykielen käsitettä ei ole korvattu jollain toisella, vaikka proseduraalisine laajennoksineen nykyiset kyselykielet ovat ilmaisuvoimaltaan yleisohjelmointikieliä vastaavia.

5 Kielet

Tässä pääluvussa esitellään tutkielmassa käyttäjäkokein vertailtavat kyselykielet XIL ja XQuery, sekä näiden yhteinen esivanhempi SQL. Kielten piirteiden kuvailun ohella luvussa pyritään valottamaan kielten suunnitteluhistoriaa sekä suunnittelupäätösten taustalla vaikuttaneita lähtöoletuksia.

5.1 SQL

SQL on tunnetuin ja laajimmin käytössä oleva relaatiotietokantojen kyselykieli. Alkuperäisessä, vuonna 1974 julkaistussa muodossaan kieli oli tietojärjestelmien loppukäyttäjille tarkoitettu, vuorovaikutteiseen ad hoc -käyttöön suunniteltu kieli (Biancuzzi & Warden 2009, 236).

SEQUEL, SQL-kielen esimuoto, suunniteltiin aikana, jolloin organisaatioiden johdon ja muiden kuin tietojenkäsittelyalan ammattilaisten käyttöön suunniteltuihin tietojärjestelmiin kohdistettiin suuria odotuksia. Tietointensiivisten alojen ammattilaiset, kuten kirjanpitäjät, insinöörit, arkkitehdit ja kaupunkisuunnittelijat nähtiin tietokantojen tulevina käyttäjäryhminä (McJones 2009, 17).

SEQUEL-kielen suunnittelun lähtöoletuksena oli, että suuri osa käyttäjien kirjoittamista tietokantakyselyistä tulisi olemaan verrattain yksinkertaisia. Tämä oletus perustui FORTRAN-kielen lauseiden käyttöä koskevaan tutkimukseen, jonka mukaan suurin osa FORTRAN-kielellä kirjoitetuista lauseista oli verrattain yksinkertaisia. SEQUEL-kielessä pyrittiin toteuttamaan tietokantavuorovaikutuksessa tarvittavat perusfunktiot. Tavoitteena oli yksinkertaistaa ammattimaisten ohjelmoijien työtä ja tehdä tietokannat lähestyttävämmiksi uusille käyttäjäryhmille (McJones 2009.)

Chamberlin toteaa SEQUELin seuraavan deklaratiivisen ongelmanmäärittelyn trendiä ja kielen perusrakenteen noudattavan IBM:n GIS (General Information System) -raportointityökalun komentokielen syntaksia. Darwen (2005, 3–5) näkee SQL:n komentokielijuuret ongelmallisina. Hän pitää huonona suunnittelupäätöksenä sitä, että kielen suunnittelussa noudatettiin tuolloin vallalla olleita ohjelmointi- ja komentokielten suunnittelun muotivirtauksia, sen sijaan että kielessä olisi toteutettu kaikki relaatiomallin vaatimukset.

SQL-kielellä käsitellään taulujen muodossa esitettyjä dataa. Taulut ovat nimettyjä ja koostuvat yhdestä tai useammasta sarakkeesta, joilla on nimi ja määritelty datatyyppi. Tauluun tallennetut tiedot muodostavat rivejä. Arvo, joka on tallennettu tietyn rivin tiettyyn sarakkeeseen on kyseisen sarakkeen datatyypin instanssi tai **`NULL`**-arvo, jolla merkitään puuttuvat arvot.

SQL-kyselyt koostuvat yhdestä tai useammasta kyselylohkosta. Kyselylohkot sisältävät SQL-avainsanalla alkavia lauseita. **SELECT** ja **FROM**-avainsanalla alkavat lauseet ovat kyselyn pakollisia osia, kun taas muiden avainsanojen — kuten **WHERE**, **GROUP BY** ja **HAVING** — aloittamat lauseet ovat valinnaisia. Kyselylohko voi olla myös toinen SQL-kysely, alikysely, jonka palauttamaa arvoa käytetään pääkyselyn syötteenä.

Kyselyn **SELECT**-osassa määritellään tietokantataulusta valittavat sarakkeet, **FROM**-osuudessa taulu, josta sarakkeet valitaan ja **WHERE**-osassa sarakkeiden valinnan ehdot. Tämä perusrakenne on yksi suoraan GIS-järjestelmästä peräisin olevista SQL-kielen piirteistä, jotka Darwen (2005) kritiikissään nimeää 1960–70 -lukujen tietojärjestelmäsuunnittelun muoti-ilmiöiksi. Vaikka SQL-kieli on laajentunut myöhemmissä versioissaan paljon, kielen perusrakennne on säilynyt samana (Chamberlin 2009, 2754).

Alkuperäisen SEQUEL-kielen ominaisuudet jaoteltiin kolmeen kerrokseen. Ensimmäinen kerros sisälsi käyttäjäkokeissa helposti opittaviksi todetut ominaisuudet (Reisner 1977). Nämä on tarkoitettu tietokannan satunnaiskäyttäjälle, jonka kyselykirjoittamistarpeet eivät ole monimutkaiset. Näitä ominaisuuksia olivat tietojen poiminta (simple mapping), projektio, Boolen-ehdot, sisäänrakennetut funktiot ja koosteet. Toiselle tasolle kuuluvat muut kyselyfunktiot ja lisäksi tietokannan sisällön päivittämisessä tarvittavat operaatiot. Taso kolme sisältää ominaisuudet, jotka on tarkoitettu tietokannan ylläpitäjän käyttöön. (Reisner 1977.) SQL-kielen myöhemmissä kehitysvaiheissa tason kaksi ominaisuudet ovat moninkertaistuneet, kun taas suhteessa ensimmäisen tason ominaisuuksiin kieli on säilynyt käytännöllisesti katsoen muuttumattomana.

SEQUEL/SQL-projektissa tehdyt käyttäjäkokeet edustavat kyselykielitutkimuksen pioneerityötä ja loivat menetelmäpohjan myöhemmille kyselykielten käyttäjätutkimuksille. Käyttäjäkokeiden tuottaman tiedon lisäksi kehitystyössä hyödynnettiin testikäyttäjien raportteja todellisista käyttötilanteista IBM:n asiakasyrityksissä (Chamberlin et al 1981). Myös myöhemmissä vaiheissa kielen kehitystä ovat ohjanneet ensisijaisesti sen käyttäjien pragmaattiset tarpeet (Biancuzzi & Warden 2009, 235).

Chamberlin ei näe ongelmallisena sitä, että SQL-kieltä käytetään nykyisin pääasiassa isäntäkielten sisällä tietojärjestelmien kehitystyössä, vaikka kielen suunnittelun ohjaavana periaatteena oli mahdollistaa ei-ohjelmoijien vuorovaikutus tietokantojen kanssa. Hänen mukaansa kielen alkuperäissuunnittelussa tehtiin toiveikkaita oletuksia ei-teknisten loppukäyttäjien kiinnostuksesta opetella SQL:n kaltaista kyselykieltä. Nämä oletukset osoittautuivat suurelta osin paikkansapitämättömiksi. (Biancuzzi & Warden 2009, 236.) Chamberlinin mukaan tietokantojen graafiset käyttöliittymät, taulukkolaskentaohjelmat se-

kä web-hakukoneet ovat ratkaisseet myöhemmin ne ongelmat, joita SEQUEL-kielellä koitettiin ratkaista (McJones 1997; Biancuzzi & Warden 2009,236).

SQL kasvoi jo hyvin varhaisessa vaiheessa kompleksisuustasolle, joka teki siitä ohjelmointikieleen verrattavan ja vaati käyttäjältään ohjelmoijatasoista osaamista (Biancuzzi & Warden 2009, 237). Kielen myöhemmissä versioissa lisätyt piirteet vahvistavat kehitystyön painopisteen siirtymisen loppukäyttäjien palvelemisesta ammattilaisohjelmoijien suuntaan. Proseduraaliset primitiivit lisännyt SQL/PSM -laajennos teki SQL:stä lopulta täysiverisen ohjelmointikielen.

Monet myöhemmät kyselykielet noudattavat SQL:n perusrakennetta. Kielen opittavuuden kannalta SQL-vaikutteet voivat olla eduksi, mutta toisaalta riskinä ovat sekaannukset, jos SQL:ltä syntaksin lainaava kieli käyttää jotain SQL-avainsanaa poikkeavassa merkityksessä (Graaumans 2005b, 85, 91). SQL:ltä syntaksin lainaavien kyselykielten juuret ovat pääasiassa ohjelmistoyrityksissä, eikä näitä kieliä ole testattu käyttäjillä ennen julkaisua. Ääneenlausumattomana oletuksena on, että SQL-kielen tuttuus, levinneisyys ja vakiintuneisuus ovat riittävä peruste toteuttaa uusi kyselykieli noudattaen SQL-syntaksia, sen sijaan että toteutettaisiin esimerkiksi vaihtoehtoinen syntaksi, jonka käyttökelpaisuutta verrattaisiin SQL:ään. Ohjelmistoyrityksissä on tarve saada kieli mahdollisimman nopeasti tuottavaan käyttöön ja jos kieli on syntaksiltaan tuttu, ammattitaitoinen ohjelmoija voi omaksua kielen nopeasti.

Lähivuosina esiteltyjä, syntaksin SQL:ltä lainaavia kieliä ovat esimerkiksi sisällönhallintajärjestelmien integrointia varten tarkoitettu ja laajasti käyttöön otettu CMIS-kyselykieli (McVeigh & Müller 2011), verkkopalvelujen rajapintojen tarjoamien tietojen yhdistelyä varten kehitetty YQL (Yahoo 2012), yhteisöpalvelu Facebookin FQL-kyselykieli (Facebook 2012), jolla voidaan kysellä palvelun käyttäjien profiilien sisältöjä sekä hajautettuun data-analyysiin tarkoitettu MRQL (Fegaras 2012).

Syntaksiltaan SQL:n kaltaisia kieliä on kehitetty myös tutkimustarkoituksessa. SQL:ään perustuvia tutkimuskieliä ovat esimerkiksi rakennusteollisuuden tarpeisiin kehitetty CI-SQL (Kibert & Hollister 1994), heterogeenisten tiedonlähteiden yhdistelyn mahdollistava WHIRL (Cohen 1998) ja relaatiotietokantojen sumea kyselykieli FSQL (Galindo et al 1998). Rakenteettoman, puolirakenteisen ja rakenteisen tiedon hakuun tarkoitetussa IRQL-kielessä SQL:n syntaksia on laajennettu muun muassa läheisyysoperaatiot ja relevanssilajittelun mahdollistavilla operaattoreilla (Heuer & Priebe 2000).

Kokeelliset XML-kyselykielet DSQL (Sengupta & Ramesh 2009) ja XRQL (Ykhlef 2007) noudattavat niinikään SQL-syntaksia. Nämä kielet palvelevat XML-dokumenttien tietokantatyyppistä käyttöskenaariota, eikä niitä ole tarkoitettu käytettäväksi seuraavassa luvussa esiteltävän XIL-kielen tavoin osittaistäsmäyttävän hakujärjestelmän kyselykielenä.

5.2 XIL

XIL on dokumenttiorinteituneeseen XML-tiedonhakuun tarkoitettu kyselykieli (Junkkari et al 2006). Kielen syntaksi ja ilmaisuvoima ovat lähellä SEQUEL-kieltä, SQL:n alkuperäismuotoa. Kuten edeltäneessä luvussa todettiin, SEQUEL-kielen suunnittelun tavoitteena oli luoda kyselykieli tietointensiivisten alojen ammattilaisille, joiden ei kuitenkaan oletettu olevan tietotekniikan asiantuntijoita. Tähän tavoitteeseen pyrittiin noudattamalla suunnittelussa periaatteita, joiden oletettiin vaikuttavan suotuisasti kielen käyttäjäystävällisyyteen. Kielen tuli olla lohkorakenteinen, perustua englanninkielisiin avainsanoihin ja kyselyn muotoilun tuli tapahtua ilman muuttujia tai proseduraalisia kontrollirakenteita.

SQL-kielen myöhemmissä kehitysvaiheissa näistä periaatteista on luovuttu, jotta kielen ilmaisuvoimaa on saatu kasvatettua. Toisin kuin kieltä suunniteltaessa oletettiin, sen pääasiallisia omaksujia olivat ohjelmoijat ja varsin pian kielen kaupallisen käyttöönoton jälkeen huomattiin, että kieltä vuorovaikutteisesti käyttävät ei-tekniset ammattilaiset ovat vain pieni osa käyttäjäkuntaa (Haigh 2006). Ilmaisuvoiman kasvattaminen – kielen monimutkaistaminen – on ollut perusteltua, jotta kieli palvelisi ohjelmoijien tarpeita.

XIL-kielessä on voitu sitoutua SEQUELin alkuperäismuotoon, sillä tarkoituksena ei ole ollut luoda kyselykieltä, jolla voitaisiin kattaa kaikki XML-dokumenttimuodon käyttötavat (Junkkari et al 2006). On mahdollista, että SEQUELin alkuperäismuodon asettamat rajat riittävät hyvin dokumenttityyppiselle XML-tiedonhaulle, jos oletetaan ettei tiedonhakijalla ole tarvetta yhdistellä usean eri dokumentin tietoja tai muotoilla haun tuloksia jatkokäsittelyä varten.

Listaus 3: XIL-esimerkkikysely I

```
SELECT luku/otsikko, aihe|avainsanat FROM osio/@lyhytotsikko ABOUT kyselykielet
```

XIL-kyselyiden perusmuoto noudattaa listauksen 3 mukaista rakennetta. Varattu sana **SELECT** aloittaa kyselyn. Kohta sisältää kyselyn kohde-elementit – ne XML-elementit, jotka halutaan kyselyn tuloksina. Kohta määrittää XML-puun haaran johon valintailmaus kohdistetaan. Ehtolohko koostuu **FROM**-lohkosta, johon voidaan yhdistää valinnalle lisärajoitteen antava **WHERE**-ehto. Kysely voi sisältää useita ehtolohkoja. Sekä **FROM** että **WHERE**-ehdot voivat sisältää **ABOUT**-ilmauksen, joka liittää ehtoon avainsanahaun. **ABOUT**-ilmausta käytettäessä kyselyn tulokset annetaan relevanssijärjestyksessä. Muussa tapauksessa tulosjoukon järjestys on mielivaltainen. (Junkkari et al 2006.)

Yksinkertaisimmillaan XIL-kyselyn **SELECT**-valintailmaus sisältää pilkulla erotetun listan tulosjoukkoon halutuista elementeistä (Listaus 3). Haluttaessa elementtien nimille voidaan antaa vaihtoehtoisia muotoja luettelemalla nimenmuodot pystyviivoin erotettuna. Valintailmauksessa voidaan antaa myös ehtoja sille, missä suhteessa tulosjoukkoon halutun elementin tulee olla dokumentin muihin elementteihin. Esimerkiksi, jos haun kohteena olevassa dokumentissa esiintyy otsikoita sekä osioissa että luvuissa, ja tiedonhakija on kiinnostunut ainoastaan jälkimmäisistä, haku voidaan rajata lukuotsikoihin lineaarisen polkuilmaisun avulla (Listaus 3). Lineaarisen polkuilmaisun viimeinen elementti ilmoittaa tulosjoukkoon halutun elementin ja tätä edeltävät elementit ilmoittavat vinoviivoin erotettuna halutun elementin sijainnin XML-puussa. Mikäli elementit erottavia vinoviivoja on yksi, polun peräkkäisten elementtien tulee esiintyä dokumentissa suorassa vanhempi-lapsi-suhteessa. Kahden vinoviivan erottamat elementit ovat toisiinsa nähden vanhempi-jälkeläinen-suhteessa, jolloin vinoviivojen jälkeinen elementti voi esiintyä dokumenttipuussa missä tahansa edeltävän elementin alapuolella. (Junkkari et al 2006.)

Asteriskimerkillä voidaan viitata mihin tahansa elementtiin. XML-dokumentin attribuutteja käsitellään polkuilmaisuissa elementtien tavoin, erottamalla attribuutti sen sisältämästä elementistä vinoviivalla. Verrattuna XPath-polkuilmaisuihin (kts. kohta 5.3) XIL-polkuilmaisut ovat rakenteeltaan yksinkertaisempia, sillä ne eivät voi sisältää sisäkkäisiä valintaehtoja, muuttujia tai funktiokutsuja, joita XPath puolestaan tukee. Tätä tarkoitetaan XIL-polkuilmaisujen lineaarisuudella. (Junkkari et al 2006.)

SELECT-valintailmauksen lisäksi myös kyselyn **FROM**-**WHERE**-ehtolohko voi sisältää lineaarisia polkuja. Ehtolohkon **WHERE**-osassa voidaan käyttää tavanomaisia vertailuoperaatioita tarvittaessa Boolen operaattoreilla yhdistettynä. Vertailuoperaatiot ovat sallittuja sekä elementtien että attribuuttien yhteydessä. Attribuutteja käsiteltäessä attribuutin isäntäelementin tulee seurata mukana. (Junkkari et al 2006.)

Listaus 4: XIL-esimerkkikysely II

```
SELECT kirjoittaja FROM osio WHERE kappale ABOUT kyselykielet GROUP BY osio
```

Listaus 5: XIL-esimerkkikysely III

```
SELECT UNIQUE kirjoittaja/nimi FROM osio
```

XIL-kyselyn tuloksena saadaan **SELECT**-osassa määriteltyjen elementtien lista oletusarvoisesti näiden emodokumenttien juurielementin mukaan ryhmiteltynä. Mikäli ryhmittely halutaan tehdä jonkun muun elementin mukaan, kyselyyn lisätään **GROUP BY** -operaattori (Listaus 4). Täysin ryhmittelemätön tulosjoukko saadaan aikaan **UNIQUE**-operaattorilla, joka liitetään kyselyyn **SELECT**-avainsanan jälkeen (Listaus 5).

Junkkarin ja muiden (2006) mukaan XIL-kielen ilmaisuvoima riittää esittämään INEX-tiedonhakukonferenssin testikyselyt vuosilta 2003–2006. Jos oletetaan, että INEX-testikyselyt edustavat XML-tiedonhaun realistisia käyttötapauksia, XIL-kielen ilmaisuvoima olisi riittävä myös tyypillisimpiin rakenteisen tiedonhaun käyttötarpeisiin tosielämässä.

5.3 XQuery

XQuery on tarkoitettu XML-dokumenttien yleiseksi kyselykieleksi. Se on suunniteltu käytettäväksi niin yksittäisten XML-tiedostojen kuin suurten dokumenttivarastojen käsittelyssä (Chamberlin 2003). Tyypillisimmillään kieltä oletetaan käytettävän apukielenä nykyisten tietokantakyselykielten tapaan, upotettuna muulla ohjelmointikielellä kirjoitetun koodin sekaan. Tämän tyypillisen käyttötavan lisäksi suunnittelutyössä on oletettu, että eksperttikäyttäjät saattavat satunnaisesti käyttää kieltä vuorovaikutteisesti komentoriviltä. Laajimmillaan XQuery voidaan nähdä webin yleiskielenä, jota käytetään haettaessa ja koostettaessa tietoa monista eri heterogeenisistä lähteistä (Chamberlin 2003.) XQuery on Turing-täydellinen kieli, joten tarvittaessa sitä voidaan käyttää yleisohjelmointikielen tavoin (Kepser 2004, Bamford et al 2009, Kilpeläinen 2012).

Aiotusta yleiskäyttöisyydestä huolimatta kielen suunnittelussa on vaikuttanut voimakkaimmin tietokantahakumainen käyttöskenaario. Tietokantaan tallennetut dokumentit ovat yleensä jotain tunnettua muotoa, jolloin tietokannassa olevien dokumenttien rakenne tunnetaan. Tietokantamainen käyttöskenaario on painottunut esimerkiksi siten, että haettujen dokumenttien oletetaan olevan useimmiten dataorientoituneita. Tietokanta-ajattelu näkyy myös kielen virheenkäsittelyssä – kyselyt tulkitaan täsmällisesti eikä virheellisiä kyselyitä suoriteta. Suunnittelutyön tietokantamaisen käyttötavan painotuksesta huolimatta kielen olisi tarkoitus kattaa myös dokumenttiorientoitunut käyttötapa. (Kay 2003.)

XQuery on työryhmätyönä suunniteltu kieli (Chamberlin 2003, Kay 2003). Chamberlin (2003) luonnehtii suunnittelutyön olleen kielelle asetettujen vastakkaisten tavoitteiden välisten jännitteiden selvittelyä. Chamberlinin mukaan se, että XQuery on kompromissien summa voi tehdä kielestä käyttökelpoisen ja kestävän monissa eri käyttöympäristöissä, mutta samalla kompromissihakuisuus on tehnyt kielen pitämisen pienenä, yksinkertaisena ja eleganttina vaikeaa (Kay 2003, Chamberlin 2003).

Kuten luonnolliset kielet, ohjelmointikielet – kyselykielet mukaan lukien – ovat sosiaalisia konstruktioita (Harper 2008). Suunnittelussa mukana olevat tahot tuovat työhön mukanaan taustansa ja edustamansa tradition käsitykset esimerkiksi siitä, mitä pidetään hyvänä suunnitteluna ja mitkä ratkaisut eivät

ole sallittuja (Kay 2003). Tämä huomioon ottaen XQueryn tietokantasuuntautuneisuus ei ole sattumaa – kielen suunnitelleen työryhmän jäsenten tausta on tietokantojen maailmassa ja jotkut jäsenistä olivat olleet mukana suunnittelemassa relaatio- ja oliotietokantojen kyselykieliä. Kuitenkin vain harvalla ryhmän jäsenistä oli edeltävää kokemusta XML-tekniikoista. (Kay 2003.)

XQueryn suunnitteluun otettiin vaikutteita useista aiemmista kyselykielistä, etupäässä tietokantojen mutta myöskin tiedonhaun piiristä (McJones 2009, 32-34). Kielen lähin esivanhempi on Quilt, josta on peräisin muun muassa yksi XQueryn tärkeimmistä ilmaisuista, FLWOR-rakenne. Quilt itsessään on yhdistelmä aiempien kyselykielten piirteitä. Quiltin englanninkielisiin avainsanoihin perustuva notaatio on lainaa relaatiotietokantojen kyselykieliltä, pääasiassa SQL:ltä. Monilla SQL:n ominaisuuksilla, kuten ulkoliitoksilla ja tulosjoukon ryhmittelyllä on vastineensa Quiltissa, vaikkakin ilmaistuna eri muodossa. Quiltin **`FOR-LET-WHERE-ORDER-RETURN`**-rakenne (FLWOR) on analoginen SQL-kyselyn **`SELECT-FROM-WHERE`**-lohkorakenteen kanssa. (McJones 2009, 32-34.)

FLWOR-rakenteen ohella toinen keskeinen XQueryn ominaisuus – polkuihin perustuva navigointi XML-dokumenteissa – on myös peräisin Quilt-kielestä, jonka syntaksi näiltä osin noudattaa W3C:n määrittelemää XPath-kyselykieltä. XPath käsittelee XML-dokumenttia puurakenteena ja sisältää ilmaukset puurakenteessa liikkumista sekä solmujen sisältämien arvojen käsittelyä varten. Tavanomaisissa käyttötilanteissa XPath-kyselyiden navigaatioaskeleet ilmaistaan lyhennetyssä muodossa, jolloin kyselyilmaus perushahmossaan sisältää vinoviivoilla erotetun listan XML-elementtejä. Vinoviivojen erottamat elementit voivat olla toisiinsa nähden vanhempi-lapsi tai vanhempi-jälkeläinen-suhteessa. XPath-kyselyt voivat olla sisäkkäisiä, siten että yhteen tai useampaan polkuilmaisun elementtiin on liitetty kyselyä rajaava lisäehto hakasuluin merkittynä (Listaus 6).

Listaus 6: XQuery-esimerkkikysely

```
for $joukkue in /joukkueet/joukkue[@maa="Suomi"]
  where $joukkue/voitot >= 1
  order by $joukkue/voitot descending
  return $joukkue
```

Monikäyttöisyydestään huolimatta XQueryn suosio on ollut vielä toistaiseksi vaatimatonta muihin XML-teknologioihin nähden. Verrattuna syntaksiltaan sekä käyttötarkoitukseltaan samankaltaiseen LINQ-kyselykieleen, XQuery ei ole saavuttanut vielä vastaavaa menestystä[4]. Tiedonhaun tutkimuksen piirissä

[4] `http://www.google.fi/trends/explore#q=xquery,xslt,linq,xpath`

XQueryn saavuttamaa standardiasemaa on pidetty ongelmallisena. Siitä huolimatta, että monimutkaiseksi kasvanut kieli on osoittautunut toimivan epätyydyttävästi dokumenttiorientoituneessa haussa ja parempaan lopputulokseen oletetaan päästävän aloittamalla kielen suunnittelu kokonaan alusta, aihepiirin tutkimus on ollut lähes pysähdyksissä standardoinnin valmistuttua (Lesk 2003.)

XQueryn epäoptimaalisuus dokumenttiorientoituneessa haussa on jättänyt tilaa kyselykielille, joiden suunnittelussa ei pyritä palvelemaan kaikkia nähtävissä olevia XML-dokumenttimuodon käyttötapoja (vrt. O'Keefe & Trotman 2003). Keskittymällä suunnittelussa rajattuun joukkoon käyttötapoja, kielen ilmaisuvoimaa on mahdollista rajata ja suunnata tarkoituksenmukaisesti.

6 Kyselykielten kokeellinen tutkimus

Kyselykielten kokeellinen tutkimus on osa ohjelmoinnin empiirisen tutkimuksen tutkimusperinnettä, jossa ohjelmointia tai tietojärjestelmien suunnittelutyötä tarkastellaan psykologisena ilmiönä. Ohjelmointia on pidetty kiinnostavana tutkimuskohteena yleisten psykologisten teorioiden kehittämisen näkökulkulmasta. Vastaavasti psykologisten teorioiden tuntemuksen on katsottu olevan avuksi suunniteltaessa uusia ohjelmointikieliä ja ohjelmoinnin apuvälineitä (Hoc et al 1990, 3.) Tutkimusperinteen isänä pidetään Weinbergiä (1971), jonka hahmottelemat tutkimussuunnat ovat edelleen elinvoimaisia.

Tässä luvussa esiteltävät kolme varhaista tutkimusta käsittelevät SEQUEL/SQL-kieltä. Nämä tutkimukset edustavat kyselykielitutkimuksen alkuvaiheita, ja niissä tehdyt metodologiset ratkaisut määrittäneet myöhemmän tutkimuksen suuntaa. Myös viittausmääriltään julkaisut kuuluvat vaikuttavimpien kokeellisten kyselykielitutkimusten joukkoon.[5]

Kyselykielten käyttäjätutkimuksen nykytilaan tutustutaan viimeaikaisten XML-kyselykielten käyttäjätutkimusten kautta. Käsittelemättä jätetään kyselykielitutkimuksen vaiheet 1980–1990-luvuilla, jolloin käyttäjäkokeita tehtiin etenkin oliotietokantojen kyselykieliin liittyen (kts. esim. Wu et al 1994).

Aiempien tutkimusten käsittely noudattaa pääpiirteittäin Reisnerin (1988, 260–267) katsauksen rakennetta. Huomio kiinnitetään tutkimusten koehenkilöihin, mahdollisiin opetusjärjestelyihin, varsinaiseen testitilanteeseen ja tuloksiin. Tuloksien käsittelyssä keskitytään pääasiassa seikkoihin, jotka liittyvät koehenkilöiden laatimien vastauksien oikeellisuuteen. Vastauksien oikeellisuus jonkin tunnusluvun muodossa on tulos, joka on raportoitu pääsääntöisesti kaikkien nyt tarkasteltavien tutkimusten tulososioissa. Myös muita tuloksia käsitellään lyhyesti.

6.1 Varhaisvaiheet

Reisner, Boyce & Chamberlin 1975, 1977

Reisner, Boyce ja Chamberlin vertaileva käyttäjätutkimus selvittää, soveltuvatko tarkastelun kohteena olevat SEQUEL ja SQUARE -kyselykielet suunniteltuun tarkoitukseensa, tietojenkäsittelytyökaluksi tietointensiivisten alojen ei-teknisten ammattilaisten käyttöön.

[5] Google Scholarin mukaan julkaisuihin kohdistuvat viittaukset 31.12.2012: Reisner, Boyce & Chamberlin (1975), 109 kpl; Reisner (1977), 158 kpl; Welty & Stemple (1981) 143 kpl, Thomas & Gould (1975), 168 kpl.

Testatut kielet ovat semantiikaltaan yhtenevät, siten että kielten perusoperaatiot ja tietorakenteet ovat yhdenmukaiset. Syntaksiltaan SQUARE noudattaa matemaattista notaatiota, kun taas SEQUEL perustuu englanninkielisiin avainsanoihin. Reisnerin ja muiden mukaan molemmat kielet on tarkoitettu helposti opittaviksi ja niiden hyödyntämisen tulisi olla mahdollista ilman erityistä tietojenkäsittelyn koulutusta.

Kokeeseen osallistui yhteensä 61 kanditaattivaiheen yliopisto-opiskelijaa ja kolme maisteriopiskelijaa. Koehenkilöiden yliopisto-opintojen pääaineita olivat muun muassa kirjanpito, taideaineet, matematiikka ja valtio-oppi. Reisner ja muut toteavat koehenkilöiden edustavan opintotaustansa perusteella varsin hyvin kyselykielten aiottua käyttäjäpopulaatiota.

Koehenkilöt jaettiin kahteen ryhmään ohjelmointikokemuksen perusteella. Ohjelmoijien ryhmään luettiin henkilöt, jotka olivat opintojensa aikana suorittaneet vähintään yhden ohjelmointikurssin. Muut koehenkilöt muodostivat ei-ohjelmoijien ryhmän.

Ennen koetta koehenkilöt osallistuivat kyselykieliä käsittelevään luentosaliopetukseen. Ohjelmointitaitoiselle tarjottiin opetusta yhteensä 12 oppituntia ja ei-ohjelmoijat osallistuivat opetukseen 14 oppitunnin verran. Opetus järjestettiin luokkahuoneessa siten että koehenkilöt oli jaettuna neljään opetusryhmään ohjelmointiosaamisen ja opetettavan kielen mukaan. Ohjelmointitaidottomien ryhmästä opetusjakson läpäisi 15 SEQUEL- ja 20 SQUARE-kieleen perehtynyttä opiskelijaa. Ohjelmoijien ryhmästä 18 suoritti opetusjakson SEQUEL- ja 11 SQUARE-kielellä.

Käyttäjäkokeet järjestettiin luokkaympäristössä ja vastaukset kerättiin paperilomakkeilla. Koeasetelma rakentui yhteensä kahdesta koetilaisuudesta ja opetusjakson aikana järjestetyistä kertauskokeista. Ensimmäinen koetilaisuus järjestettiin välittömästi opetusjakson päätyttyä ja toinen viikon kuluttua.

Kokeet sisälsivät kukin neljäkymmentä tehtävää, jotka selvittivät koehenkilöiden taitoa laatia kysely, joka tuottaa vastauksen luonnollisella kielellä esitettyyn kysymykseen – esimerkiksi *Etsi kaikki työntekijät, jotka ansaitsevat esimiehiään paremmin* – ja kymmenen tehtävää, joissa koehenkilöiden tuli kääntää kyselykielellä esitetty ilmaus luonnolliselle kielelle. Ensimmäisessä koetilaisuudessa koehenkilöt saivat käyttää apunaan luentoaineistoja. Toisen kokeen tarkoituksena oli selvittää, kuinka koehenkilöt hallitsevat oppimansa kyselykielen ulkomuistista, joten aineistojen käyttö ei ollut sallittua. Kokeisiin käytettävissä ollut aika oli rajattu kahteen tuntiin.

Ensimmäinen koe sisälsi 35 kyselykielen perusominaisuuksien hallintaa testannutta tehtävää ja viisi tehtävää, joissa koehenkilöiden tuli yhdistellä kielten ominaisuuksia ennalta tuntemattomilla tavoilla. Toi-

nen koe sisälsi niinikään 35 perusominaisuuksia käsitellyttä tehtävää ja viisi täytetehtävää. Tehtävät esitettiin koehenkilöille satunnaisjärjestyksessä.

Kyselykielen ilmauksista koostuva aineisto analysoitiin viisikohtaisen luokituksen avulla, kun taas luonnollisella kielellä annetut vastaukset kyselyiden ymmärtämistä selvittäneisiin tehtäviin luokiteltiin yksinkertaisesti oikeisiin ja vääriin vastauksiin. Kyselyaineiston analyysiluokitusta käsitellään tarkemmin tämän tutkielman luvussa 7.1.

Tarkasteltaessa kyselyaineistoa kaksiluokkaisesti, sijoittaen vastaukset olennaisesti oikein ja väärin -luokkiin, SEQUEL-kieltä käyttäneet ei-ohjelmoijat vastasivat oikein 65 % tehtävistä, jotka selvittivät kielen perusominaisuuksien hallintaa. Ohjelmointikokemusta omanneilla oikeiden vastauksien osuus oli 78 % kummallakin kielellä, kun taas ei-ohjelmoijien SQUARE-vastauksista 55 % oli oikein. Koehenkilöt vastasivat oikein kaikkiaan 72 % SEQUEL ja 66 % SQUARE-tehtävistä. Kyselyn ymmärtäminen -tehtävissä koehenkilöt laativat oikean kuvauksen 86 % tehtävistä. Ohjelmoijien vastauksista 91 % ja ei-ohjelmoijien 81 % oli oikein.

Tilastollisesti tarkasteltuna merkitseviä eroja syntyi ohjelmoijien ja ei-ohjelmoijien välille kummallakin kielellä. Lisäksi ero kielten välillä oli merkitsevä – SEQUEL-kieltä käyttäneet koehenkilöt suoriutuivat tehtävistä paremmin riippumatta ohjelmointikokemuksesta.

Welty & Stemple 1981

Welty & Stemple (1981) testaavat käyttäjäkokeellaan hypoteesia proseduraalisten kyselykielten paremmuudesta suhteessa deklaratiivisiin kieliin erityisesti monimutkaisia tietokantakyselyitä laadittaessa. Käyttäjäkokeeseen valitut kielet – TABLET ja SQL – ovat kumpikin ei-teknisille loppukäyttäjille tarkoitettuja relaatiotietokantojen kyselykieliä.

Käyttäjäkoe järjestettiin kaksi kertaa. Ensimmäisellä kerralla kokeeseen osallistui 72 koehenkilöä jattuna kahteen kielenmukaiseen ryhmään – 35 henkilöä osallistui kokeeseen SQL- ja 37 TABLET-ryhmässä. Koehenkilöt olivat liiketalouden pääaineopiskelijoita, joista noin puolet oli opiskellut aiemmin ohjelmointia joko BASIC- tai FORTRAN-kielillä. Ohjelmointikokemusta omaavat jaettiin tasan kieliryhmiin siten, että SQL-ryhmään osoitettiin 18 ja TABLET-ryhmään 17 henkilöä.

Toisella koekerralla yksikään 78 koehenkilöstä ei ollut ohjelmoinut aiemmin. Tämä koe oli järjestelyltään muutoin identtinen, paitsi käytetty TABLET-kielen versio erosi hivenen ensimmäisen kokeen kieliversiosta.

Koehenkilöt opiskelivat SQL- ja TABLET-kieliä itsenäisesti sekä ryhmätapaamisissa. Noin tunnin kestoisia ryhmätapaamisia järjestettiin kaikkiaan 14 kertaa. Varsinaista luento-opetusta ei järjestetty, vaan kielten opiskelun oli tarkoitus tapahtua kirjallisen aineiston pohjalta itsenäisesti, ryhmätapaamisten toimiessa harjoittelu- ja neuvontatilaisuuksina. Kirjalliset opetusaineistot olivat kummallekin kielelle identtiset.

Käyttäjäkoe sisälsi kaksi koetilaisuutta. Ensimmäinen koetilaisuus järjestettiin välittömästi opiskeluvaiheen jälkeen ja toinen kolme viikkoa myöhemmin. Kumpikin koe sisälsi 30 luonnollisella kielellä esitettyä tehtävänantoa, joihin koehenkilöiden tuli laatia vastauksena halutut tiedot palauttava kyselykielen lause. Kymmenen tehtävänantoa luokiteltiin tarvittavien kyselykielten ominaisuuksien perusteella helppojen ja loput 20 vaikeiden kyselyiden kategoriaan.

Ensimmäisessä kokeessa koehenkilöt saivat käyttää apunaan kielten opetusmateriaalia, kun taas jälkimmäisessä kokeessa koehenkilöiden oli vastattava koetehtäviin muistinvaraisesti.

Välittömästi opetusjakson jälkeen järjestetyssä kokeessa ohjelmointikokemusta omaamattomat koehenkilöt vastasivat oikein 47,5 % tehtävistä käyttäessään TABLET-kieltä ja 44,4 % osuuteen kun koehenkilöiden käyttämä kieli oli SQL. TABLET-ryhmään kuuluneet kokeneet koehenkilöt vastasivat oikein 57,4 % tehtävistä, kun taas SQL-ryhmäläisten vastauksista 54,7 % oli oikein. Erot eivät olleet tilastollisesti merkitseviä.

Muistinvaraisessa kokeessa TABLET-ryhmään kuuluneet ohjelmointitaustaiset koehenkilöt vastasivat oikein 44,7 % ja SQL-ryhmäläiset 31,7 % tehtävistä, kun taas ohjelmointitaidottomilla oikeiden vastauksien osuus oli 30,5 % TABLET-kielellä ja SQL:llä 25,9 %. Ohjelmointitaustaisten henkilöiden suoriutumisessa eri kielillä oli tilastollisesti merkitsevä ero, kun taas muut koehenkilöt suoriutuivat tehtävistä samantasoisesti kielestä riippumatta.

Thomas & Gould 1975

Thomas ja Gould tutkivat käyttäjäkokeella graafisen QBE-kyselykielen opittavuutta. OBE-kieli perustuu lomakkeisiin ja se on suunniteltu vastaamaan erityisesti ei-teknisten loppukäyttäjien tarpeisiin.

Koehenkilöt olivat 16–24-vuotiaita lukio- tai korkeakouluopiskelijoita, joilla oli hyvin vähän tai ei lainkaan tietotekniikkaan liittyvä kokemusta. Kaikkiaan käyttäjäkokeisiin osallistui 39 koehenkilöä.

Koe järjestettiin yhteensä neljä kertaa. Ensimmäiseen koetilaisuuteen osallistui neljä korkoakouluopiskelijaa, toiseen 11 korkeakouluopiskelijaa tai vastikään valmistunutta henkilöä. Kolmanteen ja neljänteen koetilaisuuteen osallistui kumpaankin 12 lukio-opiskelijaa. Lukio-opiskelijoista (23/24) oli käytettävissä opintomenestystiedot sekä älykkyystestien tulokset. Opintomenestyksen keskiarvo oli 44/197 ja IQ13-älykkyystestin mediaani 115.

Kutakin koehenkilöryhmää opetettiin erikseen luentotyyppisesti siten että opetus järjestettiin kahdessa jaksossa. Ensimmäinen opetusjakso käsitteli QBE-kielen perusominaisuuksia ja oli kestoltaan noin kaksi tuntia. Tämän jälkeen koehenkilöt osallistuivat ensimmäiseen käyttäjätestiin, joka koostui yhteensä 20 testitehtävästä. Lyhyen tauon jälkeen koehenkilöt osallistuivat toiselle opetusjaksolle, jossa koehenkilöt perehdytettiin kielen edistyneisiin ominaisuuksiin. Opetusjaksoa seurasi toinen 20-kohtainen testi, joka sisälsi sekä kielen edistyneitä ominaisuuksia että perusominaisuuksien hallintaa mitanneita tehtäviä.

Kaksi viikkoa opetusjakson jälkeen kuusi koehenkilöä osallistui kyselykielen muistamista mitanneeseen käyttäjätestiin. Koehenkilöt vastasivat aluksi 20 testitehtävään kertaamatta aiemmin oppimaansa. Tätä seurasi tunnin kestänyt kertaustilaisuus, jonka jälkeen koehenkilöt vastasivat vielä 20 kysymyksen päätöstestiin.

Opetusjaksojen aikana koehenkilöt tekivät harjoituskyselyitä opetettavalla kyselykielellä. Luennoijan lisäksi paikalla oli apuopettaja, joka tarkasti koehenkilöiden kirjoittamat harjoituskyselyt ja neuvoi pulmatilanteissa. Testien aikana apuna sai käyttää ainoastaan listausta kielen operaattoreista.

Kaikissa kolmessa testitilaisuudessa koehenkilöiden tehtävänä oli laatia kyselykielinen ilmaus luonnollisella kielellä esitetystä hakutehtävistä. Tehtäviin vastaamisen ohella koehenkilöitä pyydettiin pitämään kirjaa tehtäväkohtaisesta ajankäytöstä ja arvioimaan kunkin tehtävän kohdalla, kuinka varmoja he ovat vastauksena laatimansa kyselyn oikeellisuudesta viisiportaisella asteikolla mitattuna.

Analyysivaiheen alussa kyselyiden oikeellisuus arvioitiin kaksinapaisella oikein-väärin-asteikolla. Välittömästi opetusjakson jälkeen järjestetyssä kokeessa oikeiden vastauksien osuus koehenkilökohtaisesti vaihteli välillä 26–100 %. Keskimääräinen oikeiden vastauksien osuus oli 67 %. Virheellisistä kyselyistä 15 % oli syntaksiltaan oikein, mutta tuotti virheellisen tuloksen.

Lukio-opiskelijoiden ja korkeakouluopiskelijoiden suoriutumisessa ei ollut eroja. Korrelaatio koehenkilöiden raportoiman varmuuden ja kyselyn oikeellisuuen välillä oli voimakas, kuten myös korrelaatio lyhyen vastausajan ja raportoidun varmuuden kesken.

Viikon päästä järjestetyn uusintatestin kuusi koehenkiöä saivat ennen kertausoppituntia järjestetyssä kokeessa 53 % vastauksista oikein ja kertauksen jälkeen oikeiden vastauksien osuus kohosi 66 %:iin.

6.2 XML-kyselykielten käyttäjätutkimus

Graaumans 2005

Graaumansin väitöskirjatutkimus (2005) selvittää XML-kyselykielen syntaksin vaikutusta kielen helppokäyttöisyyteen. Tutkimus koostuu yhteensä kolmesta käyttäjäkokeesta. Ensimmäisen käyttäjäkokeen tarkoituksena on selvittää, kuinka aiempien kyselykielitutkimusten tulokset pätevät XML-kyselykielten kontekstissa (Graaumans 2005b, 65). Toisen käyttäjäkokeen tarkoituksena on tuottaa kielten tilastollisen vertailun mahdollistava aineisto sekä täsmentää ensimmäisen kokeen tuottamia tuloksia (Graaumans 2005b, 63). Kolmannella kokeella pyrittiin löytämään selityksiä kielten välillä havaituille eroille (Graaumans 2005b, 141).

Ensimmäisessä käyttäjäkokeessa selvitettiin muodostuuko SQL/XML, XQuery ja XSLT -kielten välillä eroja tarkasteltaessa koehenkilöiden suoriutumista tarkkuus (effectiveness) ja tehokkuus (efficiency) -muuttujien kautta. Tehokkuutta mitattiin koehenkilöiden tekemien operaatioiden lukumäärällä ja tarkkuutta kyselyiden oikeellisuudella kaksinapaisella oikein-väärin-asteikolla.

Ensimmäisen käyttäjäkokeeseen osallistui yhteensä kuusi koehenkilöä. Koehenkilöiden esitietoja selvittäneen taustatietokyselyn mukaan kaikki koehenkilöt olivat SQL-kielen osalta eksperttejä ja XML-teknologioiden tuntemuksen tasolta vähintään aloittelija- tai keskitasoa. Taustaltaan koehenkilöt olivat tietojenkäsittelytieteiden tutkijoita.

Koetilannetta edelsi 1–2 tunnin opiskeluvaihe, jonka aikana koehenkilöt tutustuivat kyselykieleen johdattavan aineiston avulla. Tämän lisäksi kyselykieleen oli mahdollisuus tutustua itsenäisesti vuorokauden ajan ennen koetilannetta.

Koe järjestettiin kahdessa osassa. Ensimmäisessä osassa kaikki koehenkilöt opiskelivat ja tekivät testitehtävät SQL/XML -kielellä. Toista osaa varten koehenkilöt jaettiin kahteen ryhmään, joista toisen ryhmä käytti testikielenä XSLT:tä ja toinen XQueryä.

Koehenkilöt saivat ratkaistavakseen viiden testitehtävän sarjan, joka piti sisällään kolme helppoa ja kaksi vaikeaa tehtävää. Helppoon tehtävään vastatessaan koehenkilön oli hallittava tietojen poiminta ja yksinkertaisen ehdon asettaminen. Vaikeissa tehtävissä koehenkilön oli ymmärrettävä, kuinka kielellä ilmaistaan liitokset ja ryhmittelyt.

Koehenkilöitä pyydettiin ajattelemaan ääneen koko koetilanteen ajan. Testitilanteen tapahtumien tallentamiseen käytettiin ruudunkaappausohjelmaa, videokameraa ja mikrofonia. (Graaumans 2005b, 118)

Ääneenajatteluaineisto purettiin tekstimuotoiseksi lokiksi, joka koodattiin kirjallisuuden sekä pilottitestiaineiston pohjalta laaditun kyselynmuodostusmallin avulla (Graaumans 2005b, 43). Malli kuvaa kyselyn muodostamisen sarjana takaisinkytkentöjä sisältäviä vaiheita, alkaen tiedontarpeen ymmärtämisestä ja päättyen tulosten evaluointiin. Koodauksen validiteetti tarkastettiin myöhemmin tehdyllä uudelleenkoodauksella sekä tutkimusapulaisen tekemällä rinnakkaiskoodauksella. Uusintakoodaus ja rinnakkaiskoodaus tuottivat olennaisesti saman tuloksen alkuperäisen koodauksen kanssa. Koehenkilöiden laatimien kyselyiden oikeellisuus arvioitiin oikein-väärin-asteikolla.

Kyselynmuodostusmallin kautta tarkasteltiin kyselyn muodostamisen tehokkuutta mitattuna koehenkilön kyselyn kirjoittamisen aikana tekemien operaatioiden lukumäärällä. Graaumans ei raportoi tehokkuutta kuvaavia lukuja sellaisenaan, vaan luonnehtii kielten sekä helppojen ja vaikeiden kyselyiden välisten erojen olevan huomattavia. Aktiviteettien kokonaismäärä XQueryllä on huomattavan paljon XSLT:tä ja SQL/XML:ää alempi. Erot kielten välillä yksinkertaisten kyselyiden tapauksessa ovat pienet, mutta monimutkaisemmissa kyselyissä ero XQueryn eduksi on erittäin suuri. Monimutkaisissa kyselyissä aktiviteettien keskiarvo SQL/XML:lle oli ∼270, XSLT:lle ∼180 ja XQuerylle ∼110.[6] Aktiviteettien maksimimäärissä tarkasteltuna erot ovat vielä dramaattisemmat, SQL/XML:n ja XSLT:n tapauksessa neljänsadan ja viidensadan välillä, kun taas XQueryllä vastanneilla maksimimäärä jäi alle 150:n.(Graaumans 2005b, 226).

Kuusi koehenkilöä kirjoitti yhteensä 30 SQL/XML -kyselyä, joista 18 annettiin vastauksena helppoihin ja 12 vaikeisiin tehtäviin. XQuery tai XSLT-kyselyitä kirjoitettiin 15 kieltä kohti, joista 9 helppoa ja 6 vaikeaa. Oikeiden vastauksien osuus kaikista vastauksista kieltä kohti oli SQL/XML:n tapauksessa 73 % ja XQuerylle ja XSLT:lle 87 %. Helpoista tehtävistä oikein oli SQL/XML:llä 94 %, XQueryllä 78 % ja XSLT:llä 100 % annetuista vastauksista. Vaikeat tehtävät muodostivat kielien välille suurempia eroja – oikean vastauksen tuotti 42 % SQL/XML-, 100 % XQuery- ja 67 % XSLT-kyselyistä.

Myös toisessa käyttäjäkokeessa tarkasteltiin kyselykielen käytettävyyden eri aspekteja. Tarkasteltavina muuttujina olivat tarkkuus (effectiveness), tehokkuus (efficiency) ja koehenkilön tyytyväisyys käytettyyn kyselykieleen (satisfaction). Tarkkuutta mitattiin kyselyiden oikeellisuudella asteikolla oikein – olennaisesti oikein – väärin. Tehokkuuden mittarin käytettiin aikaa, jonka koehenkilö käytti laatiessaan

[6] Luvut luettu kaaviosta.

oikean tai olennaisesti oikean vastauksen. Tyytyväisyyttä mitattiin seitsenportaisella asteikolla, joka kuvaa koehenkilön subjektiivista kokemusta kyselykielestä.

Toisen käyttäjäkokeen 74 koehenkilöä olivat XML-teknologioihin johdattavan kurssin opiskelijoita, joille käyttäjäkokeeseen osallistuminen oli kurssin vapaaehtoinen lisätehtävä. Ennen koetta koehenkilöt olivat olleet kurssilla mukana neljän viikon ajan ja osallistuneet luennoille, joiden aiheena oli ollut XML, XPath, XSLT ja XQuery. Luennoille osallistumisen lisäksi opiskelijat olivat tehneet XSLT ja XPath-harjoitustehtäviä.

Koe järjestettiin luokkahuoneympäristössä, yhteensä kuudessa erillisessä koetilaisuudessa. Kokonaisuudessaan kolme tuntia kestäneet koetilanteet aloitettiin valmisteluilla ja taustatietojen keräämisellä. Tämän jälkeen koehenkilöt tekivät XQuery-harjoitustehtäviä tunnin ajan. Koehenkilöt jotka kuuluivat XSLT-ryhmään, tekivät harjoittelutunnin viimeisen 15 minuutin aikana XSLT-harjoitustehtäviä.

Koetilanteen testiosuus aloitettiin XPath-tehtävillä, joihin vastaamiseen oli käytettävissä 15 minuuttia. Kaikki koehenkilöt vastasivat näihin tehtäviin. Tätä seurasi 90 minuuttia kestänyt varsinainen koeosuus, jossa vastattavana oli viisi koetehtävää XQuery tai XSLT -kielillä. Koetilanne päättyi loppukyselyyn. Koetta hallinnointiin selainympäristöön toteutetulla sovelluksella, jota käyttäen koehenkilöt täyttivät alku- ja loppukyselyt, harjoittelivat kyselykieliä, tarkastelivat testidokumentteja ja vastasivat testitehtäviin.

Analyysivaiheessa oikean vastauksen tuottavat kyselyt saivat kaksi pistettä, olennaisesti oikeat kyselyt yhden pisteen ja jollain tavalla virheelliset tai vastaamatta jääneet kyselyt nolla pistettä. Koehenkilöiden yleistä suoriutumista tarkasteltaessa XSLT-ryhmään kuuluneiden koehenkilöiden vastaukset saivat 0,81 pistettä ja XQuery-ryhmäläisten 1,09 pistettä. Ero on tilastollisesti merkitsevä. Vastausajan suhteen kielten välille ei muodostunut merkitseviä eroja: tehtävää kohti käytetty aika oli XSLT-ryhmäläisillä keskimäärin 11 minuuttia ja XQuery-ryhmäläisillä 9,3 minuuttia.

Kolmanteen käyttäjäkokeeseen osallistui yhteensä 33 vapaaehtoista koehenkilöä, joista 27 oli osallistunut myös aiempaan käyttäjäkokeeseen. Mainitut 27 koehenkilöä olivat kanditaattivaiheen opiskelijoita ja loput kuusi opintojensa maisterivaiheessa.

Koeaineisto koostettiin useista eri lähteistä. Mukaan valittiin tehtäviä, jotka vaativuudeltaan olisivat voineet esiintyä alkeistason XML-kurssin aineistoissa. Tehtävien monimutkaisuus laskettiin luvussa 7.3 esiteltävän menetelmän avulla. (Graaumans 2005b). Tehtäviä laadittiin kaikkiaan 128 kappaletta.

Koeympäristönä käytettiin XML Testbed-ohjelmistoa. Ne koehenkilöt, jotka olivat osallistuneet XSLT/XQuery-käyttäjäkokeeseen, käyttivät tässä kokeessa samaa kieltä kuin aiemmin. Muut kuusi koehenkilöä saivat itse valita käyttämänsä kielen. Kaikkiaan 19 koehenkilöä vastasi tehtäviin XSLT-kielellä ja 14 XQueryllä. Jokainen koehenkilö sai vastattavakseen 12–18 tehtävää satunnaisessa järjestyksessä. Tehtäviin pyydettiin vastaamaan esittämisjärjestyksessä ja ohittamaan tehtävä, mikäli se oli liian vaikea ratkaista.

Koetilanne kesti kokonaisuudessaan kolme tuntia. Ensimmäisen 40 minuutin ajan koehenkilöt kertasivat käyttämäänsä kyselykieltä, tutustuivat testiympäristöön ja -aineistoon. Koehenkilöt saivat halutessaan ajaa kokeilukyselyitä kyselykäyttöliittymässä tämän harjoittelujakson aikana. Opiskelua seurasi kaksi tuntia kestänyt koejakso.

Koehenkilöiden suoritumista kyselytehtävissä tarkasteltiin kyselykohtaisesti laskemalla kunkin kyselytehtävän vastauksien keskimääräinen oikeellisuus ja käytetty aika. Keskiarvotarkasteluissa käytettiin koehenkilöiden kokemusta kuvaavalla taustamuuttujalla korjattuja lukuja. Taustamuuttujan muodostamisessa tarvitut tiedot kerättiin verkkolomakkeen avulla koetilanteen alussa.

XSLT-vastauksien ($n = 62$) keskimääräinen oikeellisuus oli 0,62 ja XQuery-vastauksien 0,78. Tarkastelut ajan suhteen tehtiin ainoastaan niiden kyselyiden osalta, joiden keskimääräinen oikeellisuus oli nollaa suurempi – eli vähintään yksi koehenkilö oli antanut tehtävään oikean vastauksen. Tällaisiin XSLT-tehtäviin ($n = 57$) vastaamiseen koehenkilöiltä kului keskimäärin 9,6 minuuttia ja XQuery-tehtäviin ($n = 58$) 5,5 minuuttia.

Sengupta & Ramesh 2009

DSQL on Senguptan ja Rameshin (2009) suunnittelema XML-tietokantojen täystäsmäyttävä kyselykieli, joka noudattaa relaatiotietokantojen SQL-kyselykielen syntaksia. DSQL on tarkoitettu vuorovaikutteisia käyttötilanteita varten. Tästä syystä kielen ilmaisuvoimaa on tietoisesti rajattu estämään käyttäjän toimet, jotka tietokantaympäristössä voisivat saattaa järjestelmän epävakaaseen tilaan. Senguptan ja Rameshin mukaan DSQL soveltuu käytettäväksi esimerkiksi välikielenä ilmaisuvoimaltaan tehokkaamman XQueryn kanssa, sillä DSQL-kyselyt ovat käännettävissä vastaaviksi XQuery-kyselyiksi.

Sengupta ja Ramesh tutkivat käyttäjäkokeen avulla, onko DSQL-kielellä mahdollista laatia XML-tietokantakyselyitä XQuery-kieleen verrattuna nopeammin, niin että DSQL-kyselyt sisältävät vertailukohtaansa nähden vähemmän virheitä.

Koehenkilöt olivat opintojensa loppuvaiheessa olevia tietojärjestelmätieteen opiskelijoita, joilla oli aiempaa kokemusta ohjelmoinnista sekä tietokannoista. Osallistumispalkkiona koehenkilöt saivat yhden ylimääräisen opintopisteen. Koehenkilöt sijoitettiin satunnaisesti DSQL ja XQuery-ryhmiin. Kokeeseen osallistuneista henkilöistä osa jätti vastaamatta tehtäviin, jolloin lopulta analyysikelpoisia vastauksia saatiin DSQL-ryhmästä 21 ja XQuery-ryhmästä 25 koehenkilöltä.

Koeaineiston ja tehtävien pohjana käytettiin kahta W3C:n XML-kyselykielten vaatimusmäärittelyssä kuvattua käyttötapausta, joista toinen koskee tietokantatyyppisen XML-aineiston käsittelyä ja toinen dokumenttityyppistä aineistoa. Tehtäviä laadittiin yhteensä kymmenen kappaletta, viisi kumpaakin aineistotyyppiä kohti. Aineistoihin liittyvän järjestysefektin välttämiseksi kummankin kieliryhmän koehenkilöt jaettiin satunnaisesti kahteen ryhmään ensiksi esitettävän aineistotyypin mukaan.

Koetilanteen alussa koehenkilöille opetettiin 35 minuutin ajan kyselyiden kirjoittamista joko DSLQ tai XQuery-kielillä. Opetusmateriaaleihin kuului kielen mahdollista käyttötilannetta kuvaava skenaario ja joukko kielten ominaisuuksia esitteleviä esimerkkikyselyitä. Kaikki koetehtävissä tarvittavat rakenteet esiintyivät esimerkkikyselyissä. Esimerkit ja kyselyiden kuvaukset olivat kummallekin kielelle identtiset. Opetusjakson jälkeen koetilanne jatkui välittömästi varsinaisilla testitehtävillä. Koehenkilöt saivat pitää opetusaineiston hallussaan koko koetilanteen ajan.

Tehtävät esitettiin vaikeusjärjestyksessä yksi kerrallaan verkkolomakkeella, jota käytettiin myös tallentamaan koehenkilöiden vastaukset. Koejärjestelyn kuvauksesta ei selviä, oliko koehenkilöiden mahdollista ajaa laatimiaan kyselyitä aidossa hakujärjestelmässä vai oliko käytetty verkkolomake yksinomaan tallennuskäyttöliittymä. Sengupta ja Ramesh eivät myöskään mainitse, oliko koetilanteen tai tehtäväkohtainen aika rajattu.

Tulokset esitetään virheluokkaan kuuluvien virheiden keskiarvoina per kirjoitettu kysely. Tuloksien tilastollista merkitsevyyttä tarkastellaan ANOVA-keskiarvovertailulla. Merkitseviä eroja kielten välille DSQL:n eduksi esiintyi esimerkiksi kyselyn kohdistamisessa oikeaan dokumentin osaan ja tulokseksi haluttavien elementtien valinnassa.

Kyselyn laatimiseen käytetty aika oli merkitsevästi lyhyempi DSQL:ää käytettäessä sekä dokumentti- että tietokantatyyppisen aineiston tapauksessa. Dokumenttityyppisellä aineistolla DSQL-vastauksen kirjoittamiseen kului noin 300 sekuntia ja XQuery:llä vastatessa 450 sekuntia. Tietokantatyyppistä aineistolla DSQL-kyselyn kirjoittaminen vaati 147 sekuntia ja XQueryllä 210 sekuntia.

Weiand 2010

Weiand (2010) on väitöskirjatyössään tutkinut semanttisten wikien kyselykieliä. Verrattuna tavanomaisiin wikeihin, joiden sisältö on pääasiassa rakenteetonta tekstiä, semanttisissa wikeissä tekstisisällön oheen voidaan tallentaa myös rakenteista tietoa ja tähän rakenteeseen voidaan kohdistaa hakuja.

Weiand on kehittänyt työssään loppukäyttäjille tarkoitetun KWQL-kyselykielen, jonka suunnitteluperiaatteena on ollut vapaan avainsanahaun ja tietokantatyyppisen tarkan haun piirteiden yhdistäminen. KWQL-kielen lisäksi Weiand on suunnitellut KWQL:ään perustuvan ja ilmaisuvoimaltaan vastaavan visuaalisen kyselykielen – visKWQL:n. Keskeinen osa Weiandin väitöskirjaa on näiden kielten kokeellinen vertailu käyttäjäkokeiden avulla, joiden tarkoituksena oli selvittää, kuinka koehenkilöt suhtautuvat KWQL ja visKWQL-kyselykieliin ja kuinka he oppivat nämä kielet.

Weiandin mukaan KWQL-kieli on pyritty suunnittelemaan siten, että kyselyiden monimutkaisuus kasvaa tarvittavan ilmaisuvoiman mukaan. Näin kielen piirteiden oppiminen voi tapahtua vähitellen kieltä käytettäessä. Rakenne-ehdot eivät ole kyselyissä välttämättömiä, sillä paljaat hakusanat ovat oikeamuotoisia kyselyitä. Tarvittaessa kyselykielellä voidaan kuitenkin ilmaista hakuehtoja perinteisiin tietokantakieliin verrattavalla tarkkuudella. (Weiand 2010, 130.)

Weiandin järjestämään käyttäjäkokeeseen osallistui yhteensä 21 henkilöä, jotka rekrytoitiin yliopiston tietojenkäsittelytieteiden opiskelijoiden verkkofoorumilta ja tietojenkäsittelytieteiden luennoilta. Koehenkilöistä 16 oli tietojenkäsittelytieteiden tai mediatekniikan opiskelijoita, neljä muiden pääaineiden opiskelijoita ja yksi tietojenkäsittelytieteiden tutkija. Kaikki koehenkilöt olivat vapaaehtoisia ja he saivat osallistumisestaan 35 euron palkkion. Koetta varten koehenkilöt jaettiin satunnaisesti kahteen ryhmään siten että toisen ryhmän jäsenten tuli vastata tehtäviin KWQL- ja toisen visKWQL-kielellä. (Weiand 2010, 183.)

Ennen koetilannetta koehenkilöitä pyydettiin vastaamaan kyselyyn, jossa heidän tuli arvioida ymmärrystään yhdeksästä kokeen kannalta relevantista käsitteestä, kuten XML-tekniikoihin ja semanttiseen webiin liittyvistä käsitteistä. Tämän lisäksi koehenkilöitä pyydettiin listaamaan tuntemansa ohjelmointi- ja kyselykielet. Taustatietojen perusteella koehenkilöt jaettiin noviisi- ja eksperttiryhmiin. Henkilöistä, joilla ei ollut lainkaan kyselykielikokemusta tai jotka eivät tunteneet vähintään neljää taustatietolomakkeella mainittua käsitettä, muodostettiin noviisikäyttäjien ryhmä. Loput koehenkilöistä muodostivat eksperttikäyttäjien ryhmän. (Weiand 2010, 183.)

Koetilanteeseen oli varattuna kokonaisuudessaan 90 minuuttia aikaa, josta puoli tuntia käytettiin kyselykieleen ja koeympäristön tutustumiseen, 45 minuuttia tehtäväsarjaan, joka käsitteli kyselyiden kirjoittamista ja 15 minuuttia kyselyiden lukemista käsitelleeseen tehtävään. Koetilanteen tehtäväosuus aloitettiin ja päätettiin kyselyillä, joilla selvitettiin koehenkilöiden saamia vaikutelmia käytetyistä kielistä. Koeaineistona käytettiin Simpsons-televisiosarjan maailmaa käsittelevää julkista wikiä, josta oli luotu rakenteistettu kopio tutkimusympäristönä käytettyyn semanttiseen wikiin. (Weiand 2010, 183.)

Kyselyn kirjoittaminen -tehtävässä koehenkilöt saivat vastatakseen luonnollisella kielellä esitettyjä kysymyksiä, joihin vastauksena tuli antaa kyselykielellä laadittu kysely sekä kyselyn palauttamien sisältöjen otsikot tai palautettujen sisältöyksiköiden lukumäärä. Tehtäväsarjaan kuului yhteensä kymmenen tehtävää, jotka esitettiin vaikeusjärjestyksessä. Kyselyn ymmärtäminen -tehtävässä koehenkilöille annettiin tulkittavaksi kuusi visKWQL- tai KWQL-kyselyä. Vastauksessa koehenkilöiden tuli kuvailla, mitä sisältöjä tehtävänannon kysely palauttaa. (Weiand 2010, 184.)

Kyselyn kirjoittaminen -tehtävään annettua vastausta pidettiin oikeana, jos kysely palauttaa halutut sisällöt tai vaihtoehtoisesti halutut sisällöt suuremman tulosjoukon osana. Kaikkiaan koehenkilöt vastasivat 62 % tehtävistä oikein. Kaikista noviisikäyttäjien vastauksista oikeiden vastauksien osuus oli 44 % ja eksperttikäyttäjien 85 %. Kyselykielikohtaisesti tarkasteltuna noviisikäyttäjien KWQL-vastauksista oikein oli 53 % ja visKWQL-vastauksista 36 %. Eksperttikäyttäjien tapauksessa oikeita vastauksia oli 78 % KWQL:n ja 93 % visKWQL:n osalta. (Weiand 2010, 196.)

Kyselyn ymmärtäminen -tehtävään annetun vastauksen oikeellisuus määriteltiin arvioimalla, onko kyselyn tarkoitus ymmärretty oikein ja sisältääkö vastaus kaikki kyselyn valintaehdot. Kokonaisuudessaan koehenkilöt ymmärsivät kyselyn oikein 82 % tehtävistä. Noviisikäyttäjien vastauksista oli oikein 73 % ja eksperttien 92 %. Kyselykielikohtaisesti oikeiden vastauksien osuus oli KWQL:ää käyttäneillä noviiseilla 79 % ja visKWQL:iä käyttäneillä 67 %. Eksperttien tapauksessa kyselykieli ei vaikuttanut ymmärtämiseen, vaan kyselyistä ymmärrettiin oikein 92 % kummallakin kielellä. (Weiand 2010, 202.)

Kummankaan kokeen tapauksessa tuloksille ei ole tehty tilastollista testausta, vaan ne esitetään yksinomaan frekvenssitaulukkoina.

6.3 Yhteenvetoa tutkimuksista

SQL-syntaksia noudattavaa kieltä käyttäen ohjelmointitaidottomat koehenkilöt vastasivat oikein 44,4–65 % tehtävistä, kun taas ohjelmoinnissa harjaantuneet onnistuivat 54,7–78 % tehtävistä.[7] Muuta kuin SQL:n kaltaista kieltä käytettäessä ohjelmointitaidottomien tai vain vähäistä kokemusta omaavien koehenkilöiden laatimien oikeiden vastauksien osuus oli 47,5–79 %[8] kun taas ohjelmointitaitoisten vastauksista oikein oli 57–92 %[9].

Koehenkilöt perehtyivät testeissä käytettyihin kyselykieliin moniviikkoisilla kursseilla (Graaumans: 2005, kokeet 1–2, Reisner 1977) tai vain lyhyesti ennen koetilannetta (Weiand 2010, Sengupta & Ramesh 2009, Graaumans:2005, koe 3).

Kuten tyypillisesti missä tahansa kokeellisissa tutkimuksissa, tutkimusten koehenkilöt ovat opintojensa eri vaiheessa olevia yliopisto-opiskelijoita tai korkeakoulujen opetus- tai tutkimushenkilökuntaa. Varhaisissa kokeissa koehenkilöiksi on rekrytoitu pääasiassa ei-teknisten alojen opiskelijoita, joiden pääaineita ovat olleet esimerkiksi liiketalous (Welty & Stemple 1981), yhteiskuntatieteet, matematiikka tai taideaineet (Reisner et al 1975). Käsiteltyjen XML-kyselykielten käyttäjätutkimusten koehenkilöt ovat sitä vastoin kaikki taustoiltaan tietotekniikkaan liittyvien alojen edustajia. Graaumansin (2005) koehenkilöt olivat joko tietojenkäsittelytieteiden tutkijoita tai opintojensa alkuvaiheessa olevia alan opiskelijoita. Senguptan ja Rameshin (2009) kokeisiin osallistui valmistumisvaiheessa olevia tietojärjestelmätieteiden opiskelijoita, ja myös Weiand (2010) rekrytoi koehenkilönsä tietojenkäsittelytieteiden opiskelijoiden joukosta.

Koehenkilön ekspertti-, edistynyt- tai ohjelmoijastatuksen kriteerit ovat sidoksissa tutkimusten julkaisuajankohtaan. Varhaisissa kokeissa koehenkilön ohjelmoijastatukseen on riittänyt osallistuminen yhdelle ohjelmointikurssille (Reisner et al 1975, Welty & Stemple 1981) kun taas Graaumansin (2005) ja Weiandin (2010) käyttämät kriteerit ovat moniuloitteisemmat ja niissä otetaan huomioon käytyjen ohjelmointikurssien määrän lisäksi koehenkilöiden ymmärrys aihepiirin peruskäsitteistä. Oletettavasti tietotekniikan arkipäivästymisen myötä ero ohjelmoijiksi luokiteltujen ja ei-ohjelmoijien välillä on kaventunut. Nykypäivän korkeakouluopiskelijan – tyypillisen tutkimusten koehenkilön – keskimääräi-

[7] Reisner 1977: 78 (SEQUEL/SQL); Welty & Stemple 1981: 54,7 % (SQL); Graaumans 2005: 73 % (SQL/XML). Graaumans raportoi erikseen vastaukset helppoihin ja vaikeisiin tehtäviin – helpoista tehtävistä 94 % ja vaikeista 42 % oli oikein

[8] Reisner 1977: 55 % (SQUARE); Welty & Stemple 1981: 47,5 % (TABLET); Thomas & Gould 1975: 67 % (QBE); Weiand 2010; 79 % (KWQL), 67 % (visKWQL).

[9] Reisner 1977: 78 % (SQUARE); Welty & Stemple 1981: 57 % (TABLET); Graaumans 2005: 97 % (XQuery ja XSLT); Weiand (2010): 92 % (visKWQL)

nen tietotekninen osaamistaso voi olla lähempänä 1970-luvun tutkimuksissa ohjelmoijiksi luokiteltujen kuin ei-ohjelmoijien osaamista.

XIL-kieleltä odotettava suoriutumisen perustaso voidaan ottaa SQL-syntaksia noudattavien kielten koetuloksista. Ohjelmointiin harjaantumattomien koehenkilöiden tulisi saada oikein 44,4–65 % vastauksista kun taas ohjelmointitaitoisten voidaan olettaa vastaavan oikein 54,7–78 % osuuteen tehtävistä. Aiemmista tutkimustuloksista saatava perustaso on suuntaa antava, sillä vaikka XIL ja SQL ovat syntaksiltaan samankaltaiset, erot käsitellyn datan rakenteessa (hierarkkinen vs. relationaalinen), koejärjestelyissä ja koehenkilöiden taustoissa tekevät tutkimuksellisesti pätevän vertailun mahdottomaksi.

7 Koeasetelmat

Luvun aloittaa yhteenveto kyselykielten kokeellisen tutkimisen menetelmistä. Pääosin osuus perustuu edellisessä pääluvussa esiteltyihin Reisnerin (1975) sekä Weltyn ja Stemplen (1981) tutkimuksiin. Menetelmien esittelyn jälkeen kuvaillaan tässä tutkielmassa toteutettujen käyttäjäkokeiden järjestelyt sekä kokeiden tulokset.

7.1 Kyselykielten tutkimisen menetelmiä

Tehtävätyypit

Kyselykielten testaamista varten on kehitetty erilaisia tehtävätyyppejä, joista Reisner (1981) mainitsee kuusi erilaista (Taulukko 1).

Taulukko 1: Kyselykielitutkimuksissa käytettyjä tehtävätyyppejä

Tehtävä	Kuvaus
Kyselyn kirjoittaminen	Koehenkilöille annetaan luonnollisellä kielellä muotoiltu tehtävä ja pyydetään häntä laatimaan kysely testattavalla kyselykielellä.
Kyselyn lukeminen	Koehenkilöille annetaan testattavalla kyselykielellä laadittu kysely ja pyydetään esittämään luonnollisella kielellä kyselyn sisältö.
Kyselyn tulkkaus	Koehenkilöille annetaan testattavalla kyselykielellä laadittu kysely ja paperille tulostettu tietokanta. Koehenkilöiden tulee tulkata kysely ja poimia tietokannasta kyselyn osoittamat tiedot.
Kysymyksen ymmärtäminen	Koehenkilöille annetaan luonnollisella kielellä laadittu kysymys ja paperille tulostettu tietokanta. Koehenkilöiden tulee etsiä tietokannasta kysymyksessä pyydetyt tiedot.
Muistiinpainaminen	Koehenkilöitä pyydetään painamaan mieleensä tietokannan rakenne ja luomaan testitilanteessa tietokanta muistikuvansa pohjalta.
Ongelmanratkaisu	Koehenkilöille annetaan ongelma ja tietokanta, ja pyydetään luomaan luonnollisella kielellä kysymyksiä joiden avulla tietokannasta voidaan hakea ongelmaan ratkaisu.

Kyselyn kirjoittaminen -tehtävätyyppi on ollut kyselykielten tutkimuksissa kaikkein käytetyin. Tämä ei ole yllättävää, sillä kyselykielten aitojen käyttäjien työtehtävissä kyselykielten käyttö on juuri ensisijaisesti kyselyjen kirjoittamista (Srinivasan & Irwin 2006). Hyvä tosielämävastaavuus on myös kyselyn lukeminen tehtävätyypillä. Ammattiohjelmoijat joutuvat ohjelmakoodia ylläpitäessään lukemaan muuhun ohjelmakoodiin upotettua kyselykieltä ja tällöin kyselyiden ymmärrettävyys nopeuttaa työskentelyä. Ongelmanratkaisukielenä kyselykieltä käyttävät tallentavat itsensä ja työyhteisön käyttöön usein tarvittavia kyselyitä esimerkiksi erilaisissa raportointiympäristöissä. Vaikka ad hoc -kyselyt eivät olisi rakenteeltaan monimutkaisia, helposti luettavalla kielellä laaditut kyselyt soveltuvat paremmin yhteiskäyttöön sillä ad hoc -kyselyitä ei juurikaan dokumentoida.

Analyysitavoista

Koehenkilöiden kyselynkirjoitustehtäviin laatimat vastaukset muodostavat tutkimusaineiston, joka on perusolemukseltaan laadullinen tekstiaineisto. Useimmiten kyselykielitutkimusten tulokset halutaan esittää kuitenkin määrällisessä muodossa, erilaisten tunnuslukujen joukkoina. Laadullinen aineisto on siis voitava muuntaa analyysia ja tulosten raportointia varten määrälliseen muotoon. Kyselykielten lauseista koostuvan aineiston määrällistämistä varten on kehitetty useita lähestymistapoja, joita käsitellään seuraavassa tarkastelemalla niitä kolmen päätyypin kautta. Kaikkien lähestymistapojen tavoitteena on luoda objektiivinen mittari sille, kuinka hyvin koehenkilö on onnistunut laatimaan kyselykielen ilmaisun luonnollisella kielellä annetun tehtävänannon pohjalta, eli kuinka virheettömiä koehenkilöiden laatimat kyselyt ovat.

Kyselyiden laadinnassa tehdyt virheet voidaan jakaa syntaktisiin ja semanttisiin virheisiin. Syntaktiset virheet rikkovat kyselykielen kieliopin sääntöjä. Syntaktisesti väärin muotoiltu kysely ei tuota tulosta, vaan kyselyn suorittaminen pysähtyy virheilmoitukseen. Hyvin suunniteltu kyselykielen tulkki voi opastaa käyttäjää korjaamaan kyselyn syntaktiset virheet. (Smelcer 1995.)

Mikäli kyselykielitutkimus on suoritettu aidossa tietokantaympäristössä ja koehenkilön on sallittu korjata kyselykäyttöliittymän raportoimat virheet, tutkimusaineistossa ei pitäisi olla lainkaan syntaktisesti virheellisiä kyselyitä. Tällöin kaikki aineiston kyselyt ovat syntaktisesti oikeita ja tuottavat jonkin tuloksen. Jos tulos ei ole tehtävänannon mukainen, kysely on semanttisesti virheellinen. Semanttiset virheet ovat syntaktisia virheitä vakavampia, sillä toisin kuin syntaktisia virheiden tapauksessa, kyselykielen tulkki on vaikeaa saada varoittamaan semanttisista virheistä. (Smelcer 1995.)

Virheiden luokittelusta ja analyysistä

Reisnerin (1975) laatima kyselyvirheiden jäsennys sisältää viisi luokittelukategoriaa. Taulukko 2 listaa kyselyvirheluokat nousevassa vakavuusjärjestyksessä.

Taulukko 2: Reisnerin (1975) kyselyvirheluokat

Virhekoodi	Selite
C	Oikea vastaus
D	Pieni virhe haetussa datassa. Liittyvät haettuun dataan, ei kieleen.
M	Pieni syntaksivirhe
S	Sisältövirhe. Syntaksiltaan oikea kysely, joka tuottaa väärän tuloksen.
F	Muotovirhe. Kyselyn perushahmo on väärä.

Welty ja Stemple (1981) laajentavat Reisnerin luokittelua neljällä lisäluokalla ja nimeävät jotkin luokittelun kategoriat uudelleen. Järjestyksessä neljään ensimmäiseen luokkaan kuuluvat vastaukset kuuluvat yläkategoriaan *käytännöllisesti oikeat vastaukset* (essentially correct answers). Muut viisi luokkaa muodostavat *virheellisten vastauksien* yläkategorian (Taulukko 3). Luokista tehtiin toistensa poissulkevia siten, että jos vastauksessa oli useita virheitä, se luokiteltiin alimman esiintyneen virheluokan mukaan (Welty & Stemple 1981). Virhe-esiintymät laskettiin itsenäisinä. Jos koehenkilö kirjoitti toistuvasti jonkin kielen avainsanan väärin, nämä laskettiin omina esiintyminään (Reisner et al 1975).

Yen ja Scamell (1993) käyttivät SQL ja QBE -kyselykieliä vertailevassa tutkimuksessaan Reisnerin luokittelua miltei alkuperäismuodossaan, täsmentäen *pieni syntaksivirhe* -luokan kuvausta. Yenin ja Scamellin mukaan tämä luokka koostuu kyselyistä, joissa on joko *sanamuotovirhe* tai *välimerkkivirhe*. Sanamuotovirheellä tarkoitetaan kirjoitusvirhettä esimerkiksi tietokantataulun nimessä tai sarakkeessa, kun taas välimerkkivirheisiin luetaan esimerkiksi puuttuvat puolipisteet tai sulkumerkit.

Huolimattomuusvirheiltä näyttävät kyselyiden sanamuoto- ja välimerkkivirheet voivat toistuessaan olla merkki jostain kielessä olevasta ongelmasta. Siksi pienet sanamuoto- ja välimerkkivirheet on hyödyllistä luokitella yksityiskohtaisesti. Reisner (1981) jakoi pienet virheet aliluokkiin *yksikkö-monikkovirhe*, *kirjoitusvirhe*, *synonyymivirhe*, *lainausmerkkivirhe* ja *muu välimerkkivirhe*.

Moniluokkaisen virheluokittelun ensimmäisenä esitellyt Reisner (1975) ei kerro periaatteita, jotka ovat ohjanneet luokittelun laadintaa. Ei ole selvää, onko Reisnerin luokitus ja myöhemmät luokituksen versiot luonteeltaan aineisto- vai teorialähtöisiä – teorialähtöisellä tarkoittaen tässä sitä onko kyselyaineis-

Taulukko 3: Welty & Stemplen (1981) kyselyvirheluokat

Virheluokka	Selite
Oikea vastaus	Kysely on virheetön.
Pieni syntaksivirhe	Kyselyssä on pieni syntaksivirhe, joka on automaattisesti kyselykäyttöliittymän korjattavissa.
Pieni operandivirhe	Kyselyssä on pieni virhe sen käsittelemien tietojen määrittelyssä, kuten esimerkiksi kirjoitusvirhe tietokantataulun sarakkeen nimessä.
Pieni sisältövirhe	Kysely on syntaktisesti oikea, mutta tuottaa väärän tuloksen siksi, että koehenkilö on ymmärtänyt tehtävän ongelmanasettelun joiltain osin väärin.
Korjauskelpoinen	Kyselyssä on virhe, joka on korjattavissa kyselykäyttöliittymän antaman palautteen avulla.
Sisältövirhe	Kysely on syntaktisesti oikein, mutta tuottaa vastauksen joka ei vastaa tehtävänantoa.
Syntaksivirhe	Kyselyssä on selkeä syntaksivirhe.
Puutteellinen	Kysely ei ota huomioon kaikkia tehtävänannon vaatimuksia.
Ei vastausta	Koehenkilö ei laatinut kyselyä.

ton ensimmäistä analyysia on ohjannut jokin aineistoa koskeva oletus. Aineistolähtöistä lähestymistapaa ovat soveltaneet tietokannan tietomallin laadintaa tutkineet Batra ja muut (1990), jotka laativat virheidenluokitteluskeeman pilottitestin tuottaman aineiston pohjalta.

Yksityiskohtainen kyselyvirheluokittelu tuottaa kyselykielten kehittämistä tukevaa tietoa. Esimerkiksi syntaksivirheitä koskevan tiedon avulla voidaan parantaa kyselykäyttöliittymän virheenkäsittelyä niin että kyselykäyttöliittymä saadaan korjaamaan tyypillisimmät käyttäjien tekemät virheet ja tarjoamaan käyttäjälle kontekstisidonnaisia ohjeita.

Mikäli tutkimuksen tavoitteena on kuitenkin vain vertailla kyselykieliä, yksinkertaista oikein-väärinluokittelua voidaan pitää riittävänä luokittelutarkkuutena. Chan ja Wei (1996) vertailivat tutkimuksessaan kyselykielitutkimuksissa käytettyjen testikyselyiden pisteytysjärjestelmien vaikutusta siihen, kuinka vertailukelpoisia eri pisteytysjärjestelmien mukaan tehdyt arviot kyselykielten helppokäyttöisyydestä ovat. Vertailua varten kyselyaineiston virheet eriteltiin 36-kohtaisen virheluokituksen avulla. Virheluokkia olivat muun muassa *Puuttuva GROUP BY*, *Välimerkkivirhe*, *Kirjoitusvirhe* ja *Ylimääräinen WHERE*. Virheiden erittelyn jälkeen kyselyt pisteytettiin käyttäen viittä erilaista pisteytysjärjestelmää. Joukossa oli yksityiskohtaisia pisteytysjärjestelmiä, joissa otettiin huomioon kyselyissä tehtyjen virheiden vakavuusaste sekä yksinkertaisia kaksiluokkaisia pisteytysjärjestelmiä, joiden puitteissa kyselyt olivat pisteytyksen näkökulmasta joko täysin virheellisiä tai täysin virheettömiä. (Chan 1996.)

Chanin ja Wein mukaan käytetyllä pisteytysjärjestelmällä ei ollut vaikutusta siihen, millaisia tilastollisia eroja vertailtavien kielten välille syntyy. Näyttäisi siis siltä, että yksityiskohtainen virheluokittelu ei tarjoa mitään etua tutkimuksissa, joissa ei ole erityistä suunnittelutieteellistä tavoitetta. Yksinkertaista oikein-väärin-luokittelua on soveltanut esimerkiksi Topi ja muut (2004), jotka tarkastelivat tehtäväkompleksisuuden ja aikarajojen vaikutusta kyselynkirjoitustehtävistä suoriutumiseen.

Reisner-Welty-tyyppisessä virheluokittelussa arvioitava kysely voi kuulua vain yhteen luokittelukategoriaan. Mikäli kyselyssä on useita virheitä, se luokitellaan vakavimman esiintyneen virheen mukaan (Welty & Stemple 1981, Reisner et al 1975).

De ja muut (2001) käyttivät tutkimuksessaan virheluokitusta, jossa jokainen kyselyn sisältämä virhe luokitellaan erikseen, eikä kyselyitä sijoiteta virhe-esiintymien perusteella Reisner-Welty-tyyppisiin luokkiin. Yhteensä erilaisia virhetyyppejä tunnistettiin 26, jotka ryhmiteltiin kahdeksaan kyselyvirheiden pääluokkaan. Yksittäinen kysely voi kuulua virhe-esiintymiensä perusteella yhteen tai useampaan pääluokkaan (De et al 2001.)

Chan ja muut (1993) sekä De ja muut (2001) muodostavat tutkimuksissaan käsityksen kyselyn oikeellisuudesta arvioimalla, kuinka monta askelta virheellisen kyselyn korjaaminen syntaktisesti ja semanttisesti oikeaan muotoon vaatii ja kuinka vaativa tarvittava korjaus on. Tätä arviota varten kyselyistä laskettiin yksittäiset virheet, joista ei kuitenkaan muodostettu yhtenäisiä virhekategorioita. Chanin ja Wein mukaan vertailtavat kielet poikkesivat luonteeltaan niin paljon, ettei kielille yhteisen virheluokittelun laatiminen ollut mielekästä (Chan et al 1993).

Kyselyiden virheiden erittely on kyselyaineiston analyysin ensimmäinen askel. Analyysin toisessa vaiheessa kyselyt pisteytetään ja laadullinen kyselyaineisto saa numeerisen muodon. Tällöin tilastolliset vertailut esimerkiksi kyselykielten ja kyselytyyppien välillä tulevat mahdollisiksi. Käytetyn pisteytysmenetelmän valinta ei ole tutkimuksen luotettavuuden kannalta kriittinen teko, sillä pisteytysmenetelmällä ei näyttäisi olevan vaikutusta kielten välille muodostuviin eroihin (Chan 1996).

Reisnerin (1975) sekä Weltyn ja Stemplen (1981) pisteytysjärjestelmässä virheluokituksen *käytännöllisesti oikeat vastaukset* -yläkategoriaan kuuluvat vastaukset saivat yhden pisteen. Muihin vastauskategorioihin kuuluvat vastaukset saivat nolla pistettä.

Yen ja Scamell (1993) käyttävät tutkimuksessaan Reisner-Welty-virheluokitusta, mutta pisteyttävät kyselyt käyttäen hienojakoisempaa pisteytysjärjestelmää. Täysin oikeille vastauksille annettiin kolme pis-

tettä, kirjoitusvirheitä sisältäville kyselyille kaksi pistettä ja operaattorivirheisille kyselyille yksi piste [10].

Muoto- tai sisältövirheitä sisältävät kysymykset saivat nolla pistettä. Batran ja muiden (1990) tutkimuksessa sovellettiin viisiportaista arviointia asteikolla yhdestä nollaan. Pistevähennykset tapahtuivat kolmiportaisen virheluokittelun – pieni, keskitasoinen, suuri – mukaan neljäsosapisteen askel kerrallaan.

Chanin ja muiden (1993) sekä Den ja muiden (2001) tutkimuksissa pisteytys tapahtui ilman ennalta asetettua pisteytyskeemaa arvioijien kyselyistä muodostaman kokonaisnäkemyksen pohjalta. Chanin ja Wein tutkimuksessa pisteitä annettiin asteikolla yhdestä viiteen, De ja muut (2001) antoivat pisteitä yhdestä seitsemään. Bowen ja muut (2003) tutkimuksessa kyselyitä ei pisteytetty lainkaan, vaan kyselyistä tunnistettujen virheiden lukumäärää käytettiin suoraan tilastollisten analyysien lähtökohtana.

Kyselynkirjoittamistehtävien tuottaman aineiston analyysiin on olemassa useita lähestymistapoja, jotka on luettavissa karkeasti kahteen päätyyppiin. Reisner-Welty-lähestymistavassa kyselylle etsitään paikka luokituksessa sen sisältämän vakavimman virheen mukaan. Chan-lähestymistavassa etsitään kyselyiden virheet, mutta kyselyitä ei ryhmitellä virhekategorioihin. Eri lähestymistapoja edustavien tutkimusten (Chan et al 1993, Yen & Scamell 1993, Chan 1996, De et al 2001, Bowen et al 2003, Topi et al 2004) yhteinen piirre on vähintään kahden toisistaan riippumattoman luokittaja-pisteyttäjähenkilön käyttö. Reisnerin (1975) sekä Weltyn ja Stemplen (1981) raportoinnista ei käy ilmi, analysoiko yksi vai useampi henkilö.

7.2 Koe 1: Kyselyiden intuitiivinen ymmärtäminen

Koejärjestelyt

Kokeeseen osallistui kymmenen Tampereen yliopiston opiskelijaa. Koehenkilöt rekrytoitiin sähköpostitse pääasiassa Tietokantojen perusteet -kurssille osallistuneiden henkilöiden joukosta. Osallistujista neljän pääaine oli tietojenkäsittelytiede, kahden informaatiotutkimus, ja oppiaineista bioinformatiikka, kansantaloustiede, käännöstiede ja vuorovaikutteinen teknologia oli osallistujia yksi henkilö per aine.

[10] Operaattorivirheitä olivat esimerkiksi virheet GROUP BY, HAVING, ja ORDER BY -ilmaisuissa tai vertailuoperaattoreiden käyttövirheet.

Pääaineesta riippumatta koehenkilöitä voidaan luonnehtia vähintään noviisitasoisiksi ohjelmoijiksi. Kaikki osallistujat yhtä lukuunottamatta olivat osallistuneet Tietokantojen perusteet -kurssille. Lausekielinen ohjelmointi -kurssille oli osallistunut kahdeksan koehenkilöä. Tietokantaohjelmointi-kurssin oli käynyt neljä koehenkilöä ja Tietorakenteet-kurssin viisi. Nämä kurssit ovat aineopintotasoisia.

Koehenkilöistä kuusi oli ohjelmoinut työn tai harrastuksen puitteissa opintojen ulkopuolella. XML-teknologioihin oli tutustunut oma-aloitteisesti kaksi koehenkilöä ja SQL-kieleen viisi koehenkilöä. Yksittäinen koehenkilö, joka ei ollut osallistunut Tietokantojen perusteet -kurssille, oli kuitenkin perehtynyt SQL-kieleen oma-aloitteisesti.

Koe järjestettiin yhteensä neljä kertaa. Ensimmäisellä kerralla kokeeseen osallistui yksi, toisella kerralla kaksi, kolmannella kerralla kuusi ja neljännellä kerralla yksi henkilöä. Koetilaisuuksien järjestelyt olivat identtiset. Koetilanteen aluksi koehenkilölle kuvailtiin testin kulku ja tarkoitus. Heille kerrottiin, että tarkoituksena on vertailla kahden XML-kyselykielen intuitiivista ymmärtämistä, mutta heille ei kuitenkaan paljastettu mistä kielistä on kyse. Jättämällä mainitsematta tutkittavien kielten nimet pyrittiin vähentämään koehenkilöiden mahdollisten kyselykieliä koskevia ennakkoasenteiden vaikutusta kokeen kulkuun.

Tämän jälkeen koehenkilöille annettiin luettavaksi yhden A4-arkin mittainen johdatus XML-dokumenttimuotoon sekä kokeessa käytettävät XML-aineistot (LIITE 1). Koehenkilöitä pyydettiin ilmoittamaan, kun he ovat tutustuneet jaettuun materiaaliin käsityksensä mukaan riittävän hyvin ja ovat valmiita kokeeseen. Koehenkilöille kerrottiin, että he saavat pitää jaetun aineiston hallussaan testitilanteen ajan.

Järjestysefektin välttämiseksi testilomakkeet jaettiin testattava kieli kerrallaan siten että koko koehenkilöjoukosta kaikkiaan kuusi henkilöä täytti aluksi XIL-lomakkeen ja neljä XQuery-lomakkeen. Kokeeseen käytetty aika kirjattiin ylös lomakekohtaisesti.

Kokeeseen käytettävissä olevaa aikaa ei oltu rajattu ja koetilanteesta saattoi poistua välittömästi vastaamisen jälkeen – koehenkilön näin halutessa myös kesken koetilanteen. Koetilaisuuden alussa koehenkilöille kuitenkin kerrottiin, että aikaa kuluu arviolta tunti. Koetilanteen lopuksi koehenkilöitä pyydettiin täyttämään taustatietolomake (LIITE 1). Osallistumispalkkioksi varatut elokuvaliput annettiin koehenkilöille heidän poistuessaan tilaisuudesta.

Analyysi ja tulokset

Lomakeaineisto tallennettiin analyysiä varten XML-tietokantaan yhdessä koehenkilöiden taustatietojen kanssa. Aineiston analyysissa edettiin kieli ja testitehtävä kerrallaan niin että kaikkien koehenkilöiden vastaukset kuhunkin tehtävään otettiin tarkasteluun yhtäaikaisesti ja vastauksia verrattiin kyseisen tehtävän mallivastaukseen. Vertaillen tapahtuvan analyysin tarkoituksena oli löytää ja luokitella vastauksista ne tavat, joilla koehenkilöiden laatimat kyselykuvaukset poikkeavat mallivastauksista. Työhypoteesina oli, että koehenkilöt onnistuisivat kuvailemaan virheettömämmin XIL-kielellä kirjoitettujen kyselyiden toiminnan SQL-tietämyksensä perusteella.

Analyysi kumosi työhypoteesin, sillä koehenkilöt onnistuivat kuvailemaan kaikkien testikyselyiden toiminnan käytännöllisesti katsoen virheettä käytetystä kielestä riippumatta. Kuvailuissa oli yksittäisiä virheitä, joiden ei voida katsoa johtuvan käytetystä kyselykielestä, vaan ajatusvirheen tai väärinymmärryksen kaltaisista satunnaistekijöistä. Yksittäisiä, huolimattomuusvirheiden kaltaisia erehdyksiä matemaattisen vertailuoperaation suunnasta teki yhteensä kolme koehenkilöä. Tehtävänannon vastaisia vastauksia oli yhteensä kuusi – nämä kaikki olivat yhden koehenkilön vastauksia XQuery-tehtäviin. Vastauksien perusteella voidaan olettaa, että jonkin verran ohjelmointikokemusta omaavat henkilöt kykenevät ymmärtämään helpohkoja XML-kyselykielellä kirjoitettuja kyselyitä, ilman että kielen deklaratiivisuuden asteella olisi vaikutusta koehenkilöiden suoriutumiseen tehtävästä.

Kourallinen vastauksia sisälsi piirteitä, joiden perusteella voidaan esittää varovaisia arveluja XIL-kielen mahdollisista ongelmakohdista. Kaksi koehenkilöä tulkitsi **`GROUP BY`** -ryhmittelyoperaattorin tarkoittavan tulosjoukon järjestämistä SQL:n **`ORDER BY`** -operaattorin tapaan. Myös SQL sisältää **`GROUP BY`** -operaattorin, mutta sen semantiikka ei ole XIL:n kanssa yhteneväinen. **`GROUP BY`** -operaattoria tarvitaan SQL:ssä esimerkiksi **`SUM`**, **`MIN`** ja **`MAX`**-koostefunktioita käytettäessä yhdistämään rivejä, jotka jakavat keskenään arvon **`GROUP BY`** -operaattorille annetunssa kentässä. Koska testikyselyissä ei käytetty koostefunktioita, on mahdollista että SQL:n **`GROUP BY`** -operaattorin tunteneet koehenkilöt päättelivät XIL:n **`GROUP BY`**:n välttämättä merkitsevän jotain muuta ja olettivat operaattorin merkityksen olevan SQL:n **`ORDER BY`**:tä vastaavan. Kolme koehenkilöä huomautti puuttuvasta **`FROM`**-operaattorista, joka SQL-kielessä on pakollinen.

Tähän mennessä tarkastelussa ei ole otettu huomioon aikaa, jonka koehenkilöt käyttivät kääntäessään kyselykielen ilmaisuja luonnolliselle kielelle. On mahdollista, että vaikka koehenkilöt suoriutuivat testitehtävistä yhtä hyvin riippumatta siitä, noudattiko käytetty kieli heille entisestään tutun SQL-kyselykielen

syntaksia, syntaksiltaan monimutkaisemman ja hahmoltaan tuntemattomamman XQuery-kielen tapauksessa koehenkilöt joutuivat käyttämään kyselyiden tulkintaan enemmän aikaa. Taulukko 4 listaa koehenkilöiden testitehtäviin käyttämän ajan sekä tiedon vastausjärjestyksestä.

Taulukko 4: Koehenkilöiden käyttämä aika

	Aika (min)		
Koehenkilö	XIL	XQuery	Järjestys
1	23	22	XIL
2	37	20	XIL
3	18	20	XIL
4	19	17	XIL
5	25	32	XIL
6	16	14	XIL
7	15	33	XQuery
8	18	18	XQuery
9	12	21	XQuery
10	22	31	XQuery

Taulukosta huomataan, että koehenkilön numero 2 XIL-tehtäviin käyttämä aika poikkeaa silmiinpistävän paljon muiden koehenkilöiden ajoista. Koska kyseessä voi olla ajanotossa tehty virhe, tämän koehenkilön vastaukset suljettiin pois myöhemmistä tarkasteluista.

Testilomakkeet jaettiin koehenkilöille siten, että koko koehenkilöjoukosta kaikkiaan kuusi henkilöä täytti aluksi XIL-lomakkeen ja neljä XQuery-lomakkeen. Kun aineistosta on poistettu edellä mainitun yhden koehenkilön vastaukset, aineistossa on viiden XQuery- ja neljän XIL-tehtävillä aloittaneen koehenkilön vastaukset. Nollahypoteesina oli, ettei vastausjärjestyksellä ole tilastollisesti merkitsevää vaikutusta tehtäviin käytettyyn aikaan. Vastausjärjestyksen vaikutusta käytettyyn aikaan testattiin Mann-Whitneyn U-testillä. XIL-vastauksille testin tulokseksi saatiin $W = 15{,}5$; $p = 0{,}2187$ ja XQuery-vastauksille $W = 6$; $p = 0{,}4127$. Testin tulosten perusteella lomakejärjestyksellä ei ole tilastollisesti merkitsevää vaikutusta tehtäviin käytettyyn aikaan.

Kun lomakejärjestyksen mahdollinen vaikutus vastausaikoihin poissuljettiin, vastausaikojen mahdollisten erojen voidaan olettaa johtuvan käytetystä kielestä, joka oli koeasetelman riippumaton muuttuja. Yksinomaan keskiarvoja tarkastelemalla kielten välillä näyttäisi olevan eroa: XIL-tehtävien vastausaikojen mediaani on 18,5 minuuttia, kun taas XQuery-tehtävien kohdalla mediaani on 20,5 minuuttia. Havaittujen erojen tilastollinen merkitsevyys testattiin Wilcoxonin sijalukujen merkkitestillä, joka on riippuvien otosten t-testin nonparametrinen vastine. Testin mukaan vastausaikojen ero ei ole tilastollisesti merkitsevä, $W = 7$, $p = 0{,}139$.

Kokeen tulosten perusteella kyselykielen intuitiiviseen ymmärrettävyyteen ei näyttäisi vaikuttavan se, että kieli muistuttaa syntaksiltaan läheisesti jotain koehenkilön ennestään tuntemaa kyselykieltä. Koe kumosi työhypoteesin, jonka mukaan SQL-kielen alkeet taitavat koehenkilöt tulkitsevat virheettömämmin ja nopeammin XIL-kielellä kirjoitettuja kyselyitä, kuin sisällöltään identtisiä XQuery-kyselyitä.

Tuloksia tulkittaessa on kuitenkin otettava huomioon se, että SQL-kielen alkeiden lisäksi enemmistö koehenkilöistä hallitsi vähintään perusteet jostain yleiskäyttöisestä ohjelmointikielestä. Koehenkilöistä kahdeksan oli osallistunut ohjelmoinnin perusteet Java-kielellä opettavalle Lausekielinen ohjelmointi -kurssille. Aineopintotasoisia ohjelmointiopintoja oli suorittanut noin puolet koehenkilöistä (Tietorakenteet-kurssi, 5 koehenkilöä; Tietokantaohjelmointi, 4 koehenkilöä). Koehenkilöiden taustan perusteella voidaan siis olettaa, että kaikki koehenkilöt ovat jossain vaiheessa opintohistoriaansa altistuneet proseduraalisessa ohjelmoinnissa tarvittaville käsitteille ja ajattelutavoille.

Ohjelmointitaitoa tutkineiden Solowayn ja Ehrlichin (1984) mukaan ohjelmoijien ohjelmointitietämys on jäsentynyt yleiskäyttöisiksi rakenteiksi, joita he kutsuvat ohjelmointimenetelmiksi (programming plans) ja ohjelmointidiskurssin säännöiksi (rules of programming discourse). Ohjelmointimenetelmät ja -säännöt ovat ohjelmointirakenteiden stereotyyppisiä, skeeman kaltaisia käyttötapoja ja riippumattomia käytetystä ohjelmointikielestä. Ne voivat kuitenkin sitoutua johonkin tiettyyn ohjelmointiparadigmaan. (Soloway & Ehrlich 1984.) Aiemmissa opinnoissa omaksutut proseduraalisen paradigman ohjelmointimenetelmät ja -säännöt voivat osaltaan selittää koehenkilöiden odottamattoman hyvää menestystä XQuery-tehtävissä.

Silloin kun luonnolliselle kielelle tulkittavat kyselyt ovat koeasetelmassa käytettyjen kaltaisia – lyhyitä ja rakenteeltaan yksinkertaisia – vähäinenkin proseduraalinen ohjelmointikokemus näyttäisi antavan valmiuden lukea ja ymmärtää proseduraalisia piirteitä sisältäviä kyselyitä. Näissä käyttötilanteissa puhtaan deklaratiivisella kielellä ei näyttäisi olevan merkittävää etua proseduraaliisia piirteitä sisältäviin kieliin nähden. Lisätutkimuksella voitaisiin selvittää, olisiko kielen deklaratiiivisuudesta etua monimutkaisemmissa käyttötilanteissa tai täysin ohjelmointitaidottomien käyttäjien työkaluna.

7.3 Koe 2: Kyselyiden kirjoittaminen

Koejärjestelyt

Kokeeseen osallistui 39 Tampereen yliopiston opiskelijaa, joille kyselykielikokeeseen osallistuminen oli osa keväällä 2011 järjestetyn XML-tiedonhaku ja kyselykielet -kurssin suoritusta. Kurssi kuluu sekä tie-

tojenkäsittelytieteiden että informaatiotutkimuksen ja interaktiivisen median syventäviin opintoihin. Opiskelijan suuntautumisvaihtoehdosta riippuen kurssi on joko pakollinen tai vapaavalintainen osa opintoja.

Kurssiin kuului yhteensä kahdeksan luentokertaa, joista osa käsitteli yleisesti XML-tiedonhakuun liittyviä kysymyksiä ja osa XML-kyselykieliä. XPath ja XIL-kieliä käsiteltiin kumpaakin yhdellä luentokerralla, kun taas XQuery-kieleen perehdyttiin kahden luentokerran verran. Luennot olivat vapaaehtoinen osa kurssisuoritusta. Kyselykielikokeen 39:stä koehenkilöstä XPath-luennolle osallistui 13 henkilöä, ensimmäiselle XQuery-luennolle 18, toiselle XQuery-luennolle 10 ja XIL-luennolle 17 henkilöä.

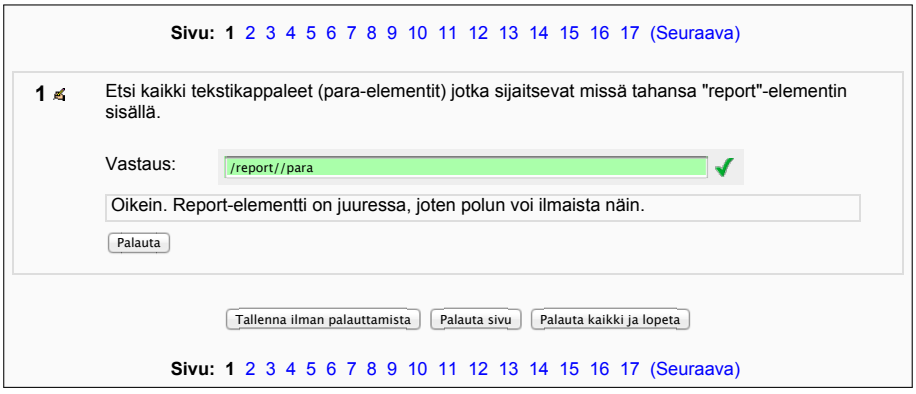
Luentojen ohella kurssin ohjelmaan kuului sekä vapaaehtoisia että pakollisia harjoituksia. Vapaaehtoiset harjoitukset harjaannuttivat opiskelijoita XPath- ja XQuery-työkalujen käyttöön palvelinympäristössä ja valmistivat heitä kurssin pakollisen harjoitustyön laadintaan. Vapaaehtoisista harjoituksista oli mahdollista saada palautetta kerran viikossa järjestetyissä harjoitustilaisuuksissa.

Pakolliset harjoitukset toteutettiin kokonaisuudessaan verkossa. Verkkoharjoituksia varten perustettiin Moodle-oppimisympäristö, jonka Quiz-moduulia[11] käytettiin harjoitusten laadintaan. Harjoitustehtävien lisäksi kurssilaiset vastasivat Moodle-ympäristössä kyselykielikokeen taustakysymyksiin, jotka toteutettiin Questionnaire-moduulin[12] avulla. Kurssilaisten ajankäytön seurantaa varten Moodleen asennettiin Course Dedication -moduuli, jonka avulla tehtiin suuntaa antavia arvioita kurssilaisen tehtäväkokonaisuuskohtaisesta ajankäytöstä[13].

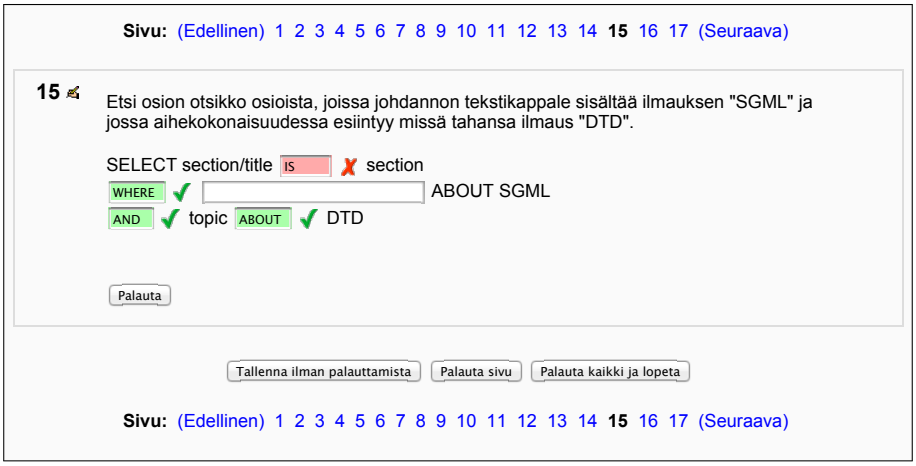
Kurssin luennot olivat tiistaisin, jolloin julkaistiin myös kyseisen viikon verkkotehtävät. Aluksi verkkotehtäville asetettiin viikon vastausaika, mutta viimeisen tehtäväsarjan yhteydessä vastausaikarajoitus purettiin ja kaikki tehtäväsarjat avattiin vastattaviksi ensimmäiseen koetilaisuuteen saakka.

Verkkoharjoitustehtäviä julkaistiin yhteensä neljän viikon ajan. Ensimmäinen sarja verkkoharjoituksia julkaistiin, kun kurssi oli ollut käynnissä kaksi viikkoa. Ensimmäisten pakollisten viikkoharjoitusten aiheena oli XPath, toisten ja kolmansien XQuery ja neljänsien XIL. XQuery ja XIL-harjoitukset olivat tehtävänannoiltaan identtiset.

[11] http://docs.moodle.org/24/en/Quiz_module
[12] http://docs.moodle.org/24/en/Questionnaire_module
[13] https://bitbucket.org/ciceidev/moodle_block_dedication



Kuva 3: XPath-tehtävä Moodlessa



Kuva 4: XIL-tehtävä Moodlessa

Pakollisia harjoitusten XPath-osiossa tehtävänä oli kirjoittaa XPath-ilmaus, joka palauttaa tehtävänannossa mainitut tiedot (Kuva 3). Tehtävien vastauksia ei ajettu XPath/XQuery -kyselyprosessorissa, vaan tehtävien tarkastus ja palautteen anto perustuivat säännöllisillä lausekkeilla tehtyyn merkkijonotäsmäytykseen. XQuery ja XIL-tehtävissä opiskelijoiden tehtävänä oli täydentää annettu XQuery tai XIL-lause siten että se tuottaa tehtävänannon mukaisen vastauksen (Kuva 4).

Koska XIL-kielestä ei ollut saatavilla monen yhtäaikaisen koehenkilön käyttöön sopivaa toteutusta ja testi oli voitava järjestää samaan aikaan kurssitentin kanssa, testi täytyi toteuttaa paperilomakepohjaisesti (LIITE 2). Koeaineisto luotiin löyhästi Graaumansin käyttämän testiaineiston pohjalta siten että testidokumenttien rakenne säilytettiin kutakuinkin alkuperäisenä, mutta tekstisisältö vaihdettiin. Testiaineistoon kuului kaksi dokumenttia, joista toisen sisältö oli peräisin Aleksis Kiven Seitsemän veljestä

-romaanin ensiluvusta ja toisen Wikileaks-sähkeaineiston katkelmista[14]. Kirja-aineisto edusti löyhemmin rakenteista dokumenttityyppiä ja sähkeaineisto tietokantamaista dokumenttityyppiä.

Koetehtävät laadittiin Graaumansin testitehtäväaineiston pohjalta (Graaumans 2005a). Tehtäviä laadittiin yhteensä 48 per kieli, joista testilomakkeelle valittiin kieltä kohden 15 tehtävää. Testitehtävät olivat kummallakin kielellä identtiset. Testitehtävät esitettiin lomakkeella tehtävien monimutkaisuusjärjestyksessä.

Testitehtävien monimutkaisuus laskettiin Graaumansin esittelemän menetelmän avulla. Graaumansin monimutkaisuus-käsite viittaa tunnuslukuun, joka saadaan laskemalla testitehtävän malliratkaisun sisältämien XML-elementtien ja -attribuuttien sekä kyselykielen avainsanojen määrä. (Graaumans 2005b, 144, 148, 152).

Testi järjestettiin yhteensä neljä kertaa. Ensimmäinen testitilaisuus järjestettiin välittömästi kurssin luento-osuuden päätyttyä. Tähän tilaisuuteen osallistui seitsemän henkilöä. Toinen ja kolmas testitilaisuus järjestettiin kurssitenttien yhteydessä mutta tenttisalista erillisessä tilassa. Toiseen testitilaisuuteen osallistui kymmenen ja kolmanteen 21 henkilöä. Neljäs testitilaisuus järjestettiin yksittäiselle opiskelijalle, joka oli estynyt osallistumasta testiin aiemmin.

Kaikki neljä testitilaisuutta olivat käytännön järjestelyiltään identtiset. Saapuessaan testitilaisuuteen koehenkilölle annettiin kummankin testattavan kielen lomakkeet yhteen nidottuna siten että puolessa jaetuista lomakkeista ensimmäisenä vastattavana oli XIL-tehtävät ja toisessa puolessa XQuery-tehtävät. Näin toimien pyrittiin vähentämään järjestysefektiä. Koehenkilöt saattoivat kuitenkin vastata tehtäviin parhaaksi katsomassaan järjestyksessä, sillä heitä ei erikseen pyydetty noudattamaan lomakejärjestystä. Koko koehenkilöjoukosta 20 henkilöä sai testilomakkeen, jossa XIL-kielen tehtävät olivat ensimmäisenä ja 19 henkilöä XQuery-tehtävät ensimmäisenä sisältävän lomakkeen. Lomakkeen ohella koehenkilöt saivat tehtävissä käytetyn aineiston.

Testitehtävälomakkeen yhteyteen oli liitetty ohjeita ja vastausesimerkin sisältävä saate (LIITE 2). Koehenkilöitä pyydettiin kirjoittamaan vastauslomakkeeseen nimensä, jota tarvittiin yhdistämään vastaukset Moodle-ympäristössä täytetyn taustatietolomakkeen tietoihin. Testitilaisuudesta pidettiin pöytäkirjaa, johon kirjattiin testitehtävälomakkeessa oleva juokseva numero ja aika, jolloin koehenkilö on saanut ja palauttanut lomakkeen.

[14]Tästä eteenpäin aineistoista käytetään nimityksiä kirja-aineisto ja sähkeaineisto.

Koeasetelman kannalta olisi ollut hyödyllistä pitää testattavien kielten lomakkeet erillään, jolloin olisi voitu kontrolloida vastausjärjestystä ja seurata kielikohtaista ajankäyttöä. Käytännön syistä tähän ei kuitenkaan ryhdytty, sillä lomakkeiden jakelu koetilanteessa olisi tehnyt tilaisuudesta rauhattoman. Vaihtoehtona olisi ollut myös kontrolloida koeaikaa niin, että kaikkia koehenkilöitä olisi vaadittu olemaan läsnä koetilaisuuteen ennälta määrätyn ajan, joka olisi jaettu tasan testattavien kielten kesken. Koska kyselykielikokeeseen osallistuminen oli pakollinen osa kurssisuoritusta, tutkimuseettisistä syistä tämä vaihtoehto katsottiin epäsopivaksi.

Taustamuuttujat

Taustatietojen valossa koehenkilöt olivat verrattain homogeeninen joukko. Koehenkilöiden enemmistö ($n = 29$) opiskeli pääaineenaan tietojenkäsittelyä. Muita koehenkilöiden opiskelemia pääaineita olivat vuorovaikutteinen teknologia ($n = 3$) sekä informaatiotutkimus ja interaktiivinen media ($n = 3$). Yhden koehenkilön pääaine jäi tuntemattomaksi taustatietojen puuttumisen vuoksi.

Omaehtoisesti SQL-kielen kanssa oli työskennellyt 24 koehenkilöä (62 %). Itsenäisesti opintojen ulkopuolella harrastuksena tai työhön liittyen ohjelmoi 28 koehenkilöä (72 %). XML-teknologioiden kanssa oli työskennellyt omaehtoisesti 19 koehenkilöä (49 %).

Koehenkilöiden opintotaustassa ei ollut juurikaan vaihtelua. Tietokantojen perusteet -kurssille oli osallistunut 34 henkilöä (87 %), joista jokainen oli suorittanut kurssin hyväksytysti. Lausekielinen ohjelmointi -kurssille oli osallistunut 36 henkilöä, joista vain yhdellä kurssisuoritus oli jäänyt kesken. Tietorakenteet ja algoritmit -kurssin oli suorittanut hyväksytysti 30 opiskelijaa ja neljä opiskelijaa oli osallistunut kurssille, mutta jättänyt kurssisuorituksen kesken. Tietokantaohjelmointi-kurssin oli käynyt 37 opiskelijaa, joista 33 oli vienyt kurssin loppuun saakka.

Harrastuneisuus ja opintotaustatietoihin pohjaten koehenkilöille laskettiin kokemuspisteet-summamuuttuja, jonka laskennassa painotettiin omaehtoista harrastuneisuutta ja vaativimpia kursseja[15]. Normalisoimalla kokemuspisteet välille 0–1 saatiin luku, joka nimettiin kokemusindeksiksi. Koehenkilöiden kokemusindeksin keskiarvo on 0,682 ja keskihajonta 0,305.

[15]Kokemuspisteiden laskennassa kurssiosallistumisista annettiin yksi piste kurssia kohti, kuten myöskin kunkin kurssin hyväksytystä suorittamisesta. Kaksi pistettä annettiin omaehtoisesta ohjelmointiharrastuksesta sekä XML ja SQL-kokemuksesta. Kokemuspisteiden maksimi on tällöin 14 pistettä.

Kyselyiden monimutkaisuus

Testitehtävien ja kielten välisten vertailujen mahdollistamiseksi intutitiivisesti ymmärrettävälle mutta epämääräiselle monimutkaisuuden käsitteelle tulee löytää yksiselitteinen esitysmuoto. Tässä tutkielmassa testitehtävien monimutkaisuus määritetään Graaumansin (2005b) esittelemän menetelmän avulla. Testitehtävien mallivastausten monimutkaisuus ilmaistaan Halsteadin (1977) *vaikeus*-tunnusluvulla.

Graaumans kuvaa XML-tiedonhakutehtävien monimutkaisuutta tunnusluvulla, joka saadaan laskemalla testitehtävän malliratkaisun sisältämien XML-elementtien ja -attribuuttien sekä näiden saamien arvojen määrä. Mikäli kysely sisältää alikyselyitä, mainittujen kyselyn osien lukumäärä kerrotaan kahdella. (Graaumans 2005b, 144, 148, 152.) Testitehtävän monimutkaisuus on kyselykielestä tai käytetystä dokumentista riippumaton suure.

Halsteadin (1977) ohjelmistometriikat ovat joukko ohjelmakoodista laskettavia tunnuslukuja, joiden avulla voidaan luonnehtia esimerkiksi ohjelman tai algoritmin jollain kielellä laaditun toteutuksen abstraktiotasoa tai toteutuksen tarvittavaa aikaa. Kyselykieliin Halsteadin ohjelmistometriikoita ovat soveltaneet esimerkiksi Chan ja muut (1994), Borthick ja muut (2001) sekä Bowen ja muut (2003).

Vaihtoehtoisia tapoja kyselyiden monimutkaisuuden määrittämiselle ovat esittäneet muun muassa Chan (1999) ja Orman (1991). Chanin (1999) lähestymistavassa lasketaan kyselyn loogisten koodirivien, eli kyselyn sisältämien lauseiden lukumäärä. Esimerkiksi SQL-kysely `SELECT nimi FROM tyontekija WHERE osasto = 1` sisältää kolme loogista koodiriviä: `SELECT kappale`, `FROM luku` ja `WHERE osasto = 1`. Chanin mukaan koodiriveihin perustuva mittari soveltuu erityisen hyvin kyselykielten tutkimiseen, sillä aiempien tutkimustulosten perusteella rivien lukumäärään perustuvien mittarien voidaan olettaa antavan tarkkoja tuloksia juuri silloin, kun tarkastelun kohteena oleva ohjelma on kyselykieli-ilmauksien tapaan verrattain lyhyt.

Ormanin (1991) mukaan kyselykielen ja kyselyiden kompleksisuuden tarkastelu vaatii mittarin, joka on riippumaton tarkasteltavana olevan kielen ilmauksista. Tätä tarkoitusta varten Orman kehittää tarkasteltavia kieliä alemmalla abstraktiotasolla olevan referenssikielen, jonka ilmaisuvoima riittää kattamaan kaikki tarkastelun kohteena olevien kielten operaatiot. Referenssikieli on suunniteltu siten, että sen ilmauksien pituus on lineaarisessa suhteessa kyselyiden monimutkaisuuteen. Kyselyn monimutkaisuus on sen referenssikielellä ilmaistujen operaatioiden kompleksisuuden summa. (Orman 1991.)

Tässä tutkielmassa kyselyiden monimutkaisuutta kuvaava Halsteadin *vaikeus*-tunnusluku lasketaan seuraavan esimerkin osoittamalla tavalla (Borthick et al 2001, Verifysoft Technology GmbH 2010).

Listaus 7: XIL-esimerkkikysely IV

```
SELECT subject FROM cable WHERE year=2009
```

Halstead-tunnuslukuja laskettaessa ohjelmakoodi tulkitaan sarjaksi merkkijonoja, ja merkkijonot luokitellaan kuuluvaksi joko operaattoreiden tai operandien luokkiin. Listauksen 7 merkkijonot **`SELECT`**, **`FROM`**, **`WHERE`** ja = ovat operaattoreita ja `subject`, `cable`, `year` ja `2009` operandeja.

Tämän jälkeen lasketaan uniikkien operaattoreiden (n_1) ja operandien (n_2) lukumäärä sekä näiden esiintymien kokonaissummat (N_1 ja N_2). Näitä lukuja käytetään seuraavien perustunnuslukujen pohjana.

Ohjelman pituus (length) on ohjelman sisältämien operaattoreiden ja operandien kokonaissumma.

$$N = N_1 + N_2 \tag{1}$$

Sanaston koko (vocabulary size) on uniikkien operaattoreiden ja operandien kokonaissumma.

$$n = n_1 + n_2 \tag{2}$$

Laajuus (program volume) kuvaa ohjelmatoteutuksen kokoa, mitattuna suoritettujen operaatioiden ja käsiteltyjen operandien lukumäärällä.

$$V = Nlog_2(n) \tag{3}$$

Vähimmäislaajuus (potential volume) kuvaa tiiviimmän mahdollisen ohjelmatoteutuksen kokoa, jossa n^* viittaa toteutuksessa tarvittavaan vähimmäissanastoon.

$$V^* = n^* log_2(n^*) \tag{4}$$

Sekä XIL ja XQuery -kielten vähimmässanaston koko on kaksi – yksinkertaisin mahdollinen XIL tai XQuery -kysely sisältää yhden operandin ja yhden operaattorin (vrt. Borthick 2001) Vastaavasti SQL-kielen vähimmäissanaston koko on viisi, sillä lyhin mahdollinen SQL-kysely sisältää yhteensä viisi operaattoria tai operandia: **`SELECT`** ja **`FROM`** -avainsanat, valittavan sarakkeen, kohdetaulun nimen ja puolipisteen, joka päättää kyselyn.

Ohjelman toteutuskielen mukainen laajuus (program level) -tunnusluku kuvaa ohjelmatoteutuksessa käytetyn ohjelmointikielen abstraktiotason vaikutusta ohjelman laajuuteen. Toteutuskielen mukainen laajuus on vähimmäislaajuuden ja laajuuden osamäärä.

$$L = \frac{V^*}{V} \tag{5}$$

Vaikeus (difficulty) on ohjelman toteutuskielen mukaisen laajuuden käänteisluku.

$$D = \frac{1}{L} \tag{6}$$

Kaava Halsteadin vaikeus-tunnusluvun laskemiseksi voidaan johtaa edellä kuvatun pohjalta seuraavasti:

$$\begin{aligned} D &= \frac{1}{L} \\ &= \frac{1}{\frac{V^*}{V}} \\ &= \frac{V}{V^*} \\ &= \frac{N log_2(n)}{n^* log_2(n^*)} \end{aligned}$$

Taulukko 5: Testitehtävien malliratkaisujen Graaumans-monimutkaisuus (C) ja Halstead-vaikeus (D)

Tehtävä	C	D_{XIL}	D_{XQuery}
1	1	0,17	0,17
2	2	0,69	0,69
3	3	1,34	1,69
4	3	1,34	1,69
5	3	1,34	3,28
6	3	1,34	2,46
7	4	2,07	6,18
8	4	2,07	4,46
9	5	2,73	3,43
10	5	2,58	1,69
11	6	3,43	2,46
12	6	3,58	6,89
13	6	3,58	8,8
14	8	5,38	8,42
15	12	8,08	5,58

Taulukko 5 listaa testitehtävien Graaumans-monimutkaisuuden (C), jonka rinnalla esitetään XIL ja XQuery-malliratkaisujen D-arvot laskettuna yllä esitetyn kaavan mukaan. Liite 3 sisältää tehtävien malliratkaisut sekä näiden operaattori- ja operandikategorisoinnit.

Halstead-tunnuslukujen käyttökelpoisuutta kohtaan on esitetty esimerkiksi mittausteoriaan pohjautuvaa kritiikkiä. Ongelmakohtia on osoitettu muun muassa perustunnuslukujen laskentatapojen määrittelyssä sekä mitta-asteikoissa (Al-Qutaish & Abran 2005). Nämä ongelmat korostuvat tarkasteltavien ohjelmien koon kasvaessa, joten lyhyiden kyselykieli-ilmausten tapauksessa Halstead-tunnuslukujen käyttö voi olla niiden puutteista huolimatta mielekästä.

Analyysi

Analyysiä varten taustatieto- ja kyselyaineisto syötettiin XML-tietokantaan[16]. Ensimmäisessä analyysivaiheessa vastauksien syntaksinmukaisuus ja mahdolliset semanttiset virheet tarkastettiin ajamalla kyselyt kielten tulkeissa. Syntaksinmukaiset, mutta virheellisen tuloksen tuottaneet kyselyt merkittiin aineistoon tulosvirheellisiksi. Tuloksettomat syntaksivirheelliset kyselyt merkittiin aineistoon syntaksivirheellisiksi.

[16] `http://www.basex.org`

Ensimmäisessä analyysivaiheessa käytetty kyselyiden luokittelu oli kaksijakoinen, jota hienojakoistettiin toisessa ja kolmannessa analyysivaiheessa. Toinen analyysivaihe eteni aineistolähtöisesti siten, että aineisto käytiin testitehtäväjärjestyksessä läpi ja kyselyt kategorisointiin niiden sisältämien virheiden mukaisiin luokkiin. Luokittelussa ei käytetty valmista luokituskaavaa, vaan luokat pyrittiin erottamaan aineistosta tutkimalla kunkin vastaukseksi annetun kyselyn kohdalla, millaisia virheitä kysely näyttäisi sisältävän ja kuinka nämä virheet voisi nimetä.

Kukin vastaukseksi annettu kysely saattoi kuulua yhteen tai useampaan virheluokkaan. Näin toimien voitiin päästä selville virheiden kokonaismäärästä, joka olisi jäänyt paljastumatta, mikäli virheluokat olisivat olleet toistensa poissulkevia. Oli tavanomaista, että kyselyissä ei oltu tehty ainoastaan yhtä, vaan useita erityyppisiä virheitä.

Luokittelu tapahtui iteratiivisesti, niin että aineistoa käytiin analyysivaiheen aikana läpi useaan otteeseen. Joidenkin testitehtävien kohdalla yksi iteraatio antoi tyydyttävän lopputuloksen, kun taas runsaasti erilaisia syntaksivirheitä sisältäneiden vastauksien tapauksessa iteraatiokierroksia vaadittiin useita.

Toisen analyysivaiheen synnyttämä virheluokitus antoi yleiskuvan kyselyaineiston sisältämistä virheistä. Luokitus ei ollut kuitenkaan vielä riittävän hienojakoinen, jotta sen perusteella olisi voinut esittää XIL-kieltä koskevia kehitysehdotuksia sellaisella tarkkuudella, että näitä ehdotuksia voitaisiin käyttää ohjelmistokehitystyön vaatimusmäärittelyssä. Tästä syystä virheluokitusta tarkennettiin XIL-kyselyaineiston osalta kolmannessa analyysivaiheessa, jossa toisessa analyysivaiheessa luotuja luokkia purettiin aliluokiksi. Aliluokkien avulla pyrittiin erottamaan pääluokkien yleisten virhetyyppien sisältä virhetyypin variaatiot omiksi ryhmikseen. Kaikkia pääluokkia ei purettu aliluokkiin, vaan tarkasteluun otettiin luokat joissa virheen variaatioita oli kaikkein runsaimmin ja jotka olivat kooltaan kaikkein suurimpia luokkia. Pääluokitusperusteena olevan virheen variaatio tai luokan koko ei ollut aliluokittelun syy kaikissa tapauksissa, vaan virheluokkia jaettiin aliluokkiin myös silloin, jos luokka vaikutti jakautuvan selkeästi osiin jonkin virheellisiä vastauksia yhdistävän piirteen perusteella, ja tämä piirre oli mahdollista nimetä kuvaavasti.

Toisen ja kolmannen analyysivaiheen tuottamat virheluokitukset on esitetty XIL-kielen osalta taulukoissa 6 ja 7 sekä XQuery-kielen osalta taulukossa 8. XIL-aineistossa esiintyvien virheiden kolme suurinta pääluokkaa ovat *kielilaina* ($n = 118$), *polkuvirhe* ($n = 113$) ja *attribuuttivirhe* ($n = 80$). XIL-aineiston virheiden aliluokista suurimpia ovat *viittausvirhe* ($n = 74$), *attribuutti ilman elementtiä* ($n = 54$) ja *ABOUT*

korvattu contains-avainsanalla vaihtelevaa syntaksia noudattaen ($n = 47$). XQuery-kyselyiden suurimpia virheluokkia ovat *muu muotovirhe* ($n = 115$), *viittausvirhe* ($n = 94$) ja *virhe tulosten muotoilussa* ($n = 40$).

Virheiden absoluuttisten määrien perusteella ei yksin voida tehdä päätelmiä siitä, mitkä virheet ovat sellaisia, jotka kantavat kielen kehittämisen kannalta olennaista tietoa. On mahdollista, että jonkin virheluokan absoluuttisesti suuren virhemäärän takana on yksittäinen epäonninen koehenkilö, joka on muistanut jonkin kielen ominaisuuden väärin ja siksi toistanut jonkin virheen jokaisessa vastauksessaan. Tällaisten epäonnesta johtuvien virheiden sijaan on kiinnitettävä huomiota virheisiin, joiden absoluuttinen määrä on suuri siitä syystä, että verrattain moni koehenkilö on tehnyt kyseisen virheen.

Taulukko 6: XIL-päävirheiden toistuvuus (n) ja kokonaismäärä ($\sum$)

Virheluokka	n	$\sum$
Attribuuttivirhe	26	80
Viittausvirhe	11	22
Polkuvirhe	33	113
Puuttuva osa	3	3
Puuttuva data	13	24
Kielilaina	25	118
Muu syntaksivirhe	9	11
Muotovirhe	14	26
Sisältövirhe	5	7
Muu sisältövirhe	4	4
Kielisekaannus	3	29

Taulukko 7: XIL-alivirheiden toistuvuus (n) ja kokonaismäärä ($\sum$)

Virheluokka	n	$\sum$
ABOUT korvattu contains-avainsanalla vaihtelevaa syntaksia noudattaen	12	47
ABOUT korvattu EXISTS-avainsanalla	1	5
ABOUT puuttuu	1	4
Attribuutti ilman elementtiä	18	54
Attribuutin käsittely elementtinä	8	16
ABOUT korvattu contains-avainsanalla XIL-syntaksia noudattaen	2	8
GROUP BY korvattu	5	5
Puuttuva attribuutin erotinmerkki	5	8
Puuttuva erotinmerkki	2	6
SELECT korvattu	1	15
SELECT puuttuu	1	8
Viittausvirhe	29	74
Viittausvirhe kyselyn WHERE-osassa	4	4
Virheellinen viittaus attribuuttiin	1	4
XPath-kielilaina	10	32
XPath-syntaksin mukainen contains()	5	11
XQuery-kielilaina	1	9

Taulukko 8: XQuery-virheiden toistuvuus ja kokonaismäärä (XQuery)

Virheluokka	n	$\sum$
Kielilaina	6	16
Muu muotovirhe	30	115
Muu pieni syntaksivirhe	10	21
Muu syntaksivirhe	17	49
Polkuvirhe	31	94
Predikaattilause	14	36
Puutteellinen ratkaisu	7	8
Puuttuvat lainausmerkit	2	2
Useita elementtejä tuottava ehto	9	12
Virhe tietojen määrittelyssä	11	18
Virhe tuloksen muotoilussa	17	40

Kuten edellä on todettu, kyselyt voivat kuulua kerrallaan yhteen tai useampaan pää- ja alivirheluokkaan. Suurimmalla kielisekaannus-päävirheluokalla ei ole lainkaan alivirheluokkia. Kahden seuraavaksi suurimman XIL-päävirheluokan kanssa yleisimmin esiintyneitä alivirheitä ovat päävirheluokittain: polkuvirhe – viittaus ($n = 74$), attribuutin käsittely elementtinä ($n = 16$) ja attribuuttivirhe – attribuutti ilman elementtiä ($n = 54$), attribuutti ilman erotinmerkkiä ($n = 8$).

Seuraava luku esittelee yksityiskohtaisesti analyysin tuottaman virheluokituksen. Koska tutkielman tarkoituksena on tuottaa XIL-kielen kehittämisessä tarvittavaa tietoa, analyysissa keskitytään erityisesti XIL-virheiden kuvailuun. XQueryn osalta virheluokkien kuvaus pidetään yleisemmällä tasolla.

XIL-virheluokat

Attribuuttivirhe

Koehenkilöt tekivät yhteensä 80 attribuutteihin liittyvää virhettä. Kaikkiaan 26 koehenkilöä (67 % koehenkilöistä) teki vähintään yhden attribuuttivirheen. Attribuuttivirhe-virheluokka koostuu kolmesta alivirhetyypistä, joita ovat *attribuutti ilman elementtiä*, *attribuutin käsittely elementtinä* ja *puuttuva attribuutin erotinmerkki* -virheet.

XIL-kieliopissa attribuutti ilmaistaan samalla tavalla kuin XPathissa siten että attribuutti esiintyy aina emoelementtinsä seurassa. XPathissa attribuutti voi esiintyä predikaatin sisällä myös ilman emoelementtiä. Predikaatin sisällä ilman emoelementtiä esiintyvä attribuutti liittyy aina siihen elementtiin, jonka yhteyteen predikaatti on liitetty.

XIL-kieliopin mukaisessa kyselyssä elementtiin vinoviivalla yhdistetty attribuutti palauttaa ne elementit jälkeläisineen, joista kyselyssä mainittu attribuutti löytyy. Esimerkiksi XIL-kysely **SELECT** `kappale /@tyyli=”puheenvuoro”` palauttaa kaikki dokumentin kappale-elementit jälkeläisineen. XIL-lauseen kanssa ulkoisesti yhdenmukainen XPath-lause `//kappale/@tyyli=”puheenvuoro”` palauttaa sen sijaan totuusarvon siitä, esiintyykö kyseinen elementti-attribuuttipari dokumentissa. Esimerkin mukaiselle XIL-lauseelle yhtenevä XPath-lause olisikin `//kappale[@tyyli=”puheenvuoro”]`.

XPath:in tuntevalle käyttäjälle XIL:n poikkeava polkuilmaisun tulkinta aiheuttaa sekaannuksia. Käyttäjän voi olla vaikea muistaa, että ilmaus joka XPath:issa tuottaa totuusarvon, palauttaakin XIL:in tapauksessa elementtejä. Tämän XIL-ilmauksen hallinnan vaikeus näkyy testitehtäviin kolme, neljä ja yhdeksän annetuista vastauksista. Vastatakseen näihin tehtäviin oikein koehenkilön tuli ymmärtää XPath- ja XIL-polkuilmausten eroavan toisistaan edellä kuvatulla tavalla. Tehtäviin kolme ja neljä annettin kumpaankin kolme oikeaa vastausta ja tehtävään yhdeksän vain yksi oikea vastaus. Vähintään yhden kerran polkuilmauksen laati oikein kaikkiaan neljä koehenkilöä. Kaikkien koehenkilöiden joukosta vain yksi henkilö vastasi oikein kaikkiin kolmeen tehtävään.

XIL-XPath polkuilmausten eron ohella koehenkilöille aiheutti vaikeuksia XIL ja SQL-syntaksien eroavaisuus kyselyn **FROM**-osan osalta. XIL-kieliopin mukaan kyselystä on sallittua jättää pois **FROM**-osa, kun taas SQL:ssä ja alkuperäisessä SEQUELissa **FROM**-osa on pakollinen osa kyselyä. Aineiston perusteella voidaan olettaa, että koehenkilöt käsittävät XIL:n noudattavan SQL:n syntaksia myös **FROM**-osan osalta, niin että **FROM** on aina pakollinen osa kyselyä. Esimerkiksi testitehtävissä yksi ja kaksi, joissa **FROM**-osan poisjättäminen oli mahdollista, näin teki kummassakin tehtävässä vain kahdeksan koehenkilöä. Tehtävään yksi antoi oikean tuloksen tuottavan vastauksen 30 koehenkilöä ja tehtävään kaksi 12 koehenkilöä.

Attribuutti ilman elementtiä. Attribuutin sisällyttäminen kyselyyn ilman elementtiä oli kaikkein useiten esiintynyt attribuuttivirhe ja lukumäärältään kaikkein yleisin virhe koko aineistossa (55 esiintymää). Näissä kyselyissä ilman elementtiä esiintyvä attribuutti saattoi liittyä joko kyselyn **SELECT** tai **FROM**-osaan (Taulukot 9, 10 ja 11).

SQL:ssä **WHERE**-osan predikaatilla viitataan **FROM**-lauseessa annettuun relaatioon, johon voidaan viitata joko piste-erotteisella lyhenteellä tai jättämällä taulun nimi kokonaan pois. Piste-erotteista lyhennettä käytetään, jos SQL-lauseessa on sovellettu alikyselyitä. On mahdollista, että koehenkilöt käsittelivät XIL-kyselyissä **WHERE**-lauseeseen attribuuttia samaan tapaan kuin SQL:ssä käsitellään taulujen sarakkeiden nimiä.

Taulukko 9: Ilman elementtiä esiintyvien attribuuttien viittauskohteet

Virheluokka	Sijainti	n	$\sum$
Attribuutti liittyy FROM-osaan	4, 9, 11, 15	14	30
Attribuutti liittyy SELECT-osaan	3, 4, (15)	16	21

Attribuuttien ja taulujen sarakkeiden nimien ymmärtäminen samankaltaisiksi voi osaltaan vaikuttaa siihen, miksi koehenkilöt ovat pääsääntöisesti sijoittaneet kyselyyn liittyvän ehdon WHERE-lauseeseen ilman elementtiä esiintyvänä attribuuttina, sen sijaan että attribuutti olisi liitetty suoraan kyselyn **SELECT** -osaan. Näin on tehty erityisesti kyselyissä, joissa **FROM**-osan poisjättäminen on ollut mahdollista. SQL-kyselyissä **SELECT**-osassa luetellaan, mitä tietoalkioita kyselyn halutaan palauttavan, kun taas XIL-kyselyissä **SELECT**-osaan voidaan liittää haettavien elementtien oheen myös ehtoja attribuuttien muodossa.

Taulukko 10: SELECT-osaan viittaava attribuutti

Numero	Referenssi	Vastausesimerkkejä
3.	SELECT kappale/@tyyli=puheenvuoro	SELECT kappale where /@tyyli=”puheenvuoro” SELECT kappale FROM kirja where @tyyli=”puheenvuoro” SELECT //kappale where @tyyli=”puheenvuoro”
4.	SELECT cable/@classification=SECRET	SELECT sahke FROM sahkeet WHERE @luokitus =”SECRET” SELECT //sahke FROM sahkeet WHERE @luokitus=”SECRET” SELECT sahke WHERE @luokitus=”SECRET”
11.	SELECT osio/johdanto FROM luku WHERE osio/@lyhytotsikko ABOUT Kouluunlähtö	select //osio/johdanto from kirja where //@lyhytotsikko=”Kouluunlähtö”
15.	SELECT tilanne/otsikko, tilanne/avainsana FROM kirja WHERE tilanne/@tunniste=tilanne1 OR tilanne/@tunniste=tilanne2	select //tilanne where /@tunniste=”tilanne1” or /@tunniste=”tilanne2”return /otsikko and /avainsana

Taulukko 11: FROM-osaan viittaava attribuutti

Numero	Referenssi	Vastausesimerkkejä
4.	SELECT cable/@classification=SECRET	select sahke from // where @luokitus=”SECRET” select * from //sahke where @luokitus=”SECRET”
11.	SELECT osio/johdanto FROM luku WHERE osio/@lyhytotsikko ABOUT Kouluunlähtö	SELECT johdanto FROM osio where /@lyhytotsikko=”Kouluunlähtö” select johdanto from osio where @lyhytotsikko=”Kouluunlähtö” select ./johdanto from //osio where /@lyhytotsikko=”Kouluunlähtö”
9.	SELECT vastaanottaja/nimi FROM sahke/@luokitus=”CONFIDENTIAL”	SELECT vastaanottaja/nimi FROM sahke where @luokitus=”confidential” select ./vastaanottaja/nimi from //sahke where ./@luokitus=”confidential” select nimi from sahke where @luokitus=”CONFIDENTIAL”
15.	SELECT tilanne/otsikko, tilanne/avainsana FROM kirja WHERE tilanne/@tunniste=tilanne1 OR tilanne/@tunniste=tilanne2	SELECT otsikko, avainsana FROM tilanne where /@tunniste=”tilanne1”OR /@tunniste=”tilanne2” SELECT otsikko, avainsana FROM tilanne where @tunniste=”tilanne1”OR @tunniste=”tilanne2” SELECT otsikko, avainsana FROM tilanne where @tunniste=”tilanne1|tilanne2” select ./otsikko ./avainsana from //tilanne where ./@tunniste=”tilanne2”or ./@tunniste=”tilanne2”

Attribuutin käsittely elementtinä. Attribuutin käsittely elementtinä -virhe esiintyi aineistossa yhteensä 16 kertaa, kahdeksan eri koehenkilön vastauksissa. Kyselyssä viitataan attribuuttiin ilman attribuutin @-etumerkkiä (Taulukko 12).

Virhetilanne voitaisiin välttää väljentämällä kyselyn tulkintaa siten että attribuutteja ja elementtejä käsitellään samanarvoisina tietoalkioina. Virheen välttämisessä auttaa myös, mikäli kyselykäyttöliittymässä on ennustava tekstinsyöttö, jonka opastamana käyttäjä välttää sisällyttämästä kyselyyn elementtejä tai attribuutteja, joita ei löydy indeksoiduista dokumenteista. Mikäli elementtinä käsiteltyyn attribuuttiin liitetään @-etumerkki, useimmissa tapauksissa kysely muuttuu muotoon, jossa kysely on sijoitettavissa virheluokkaan *attribuutti ilman elementtiä*

Taulukko 12: Esimerkkejä kyselyistä, joissa on attribuutin käsittely elementtinä -virhe

Numero	Referenssi	Virheellinen vastaus
9	SELECT vastaanottaja/nimi FROM sahke/@luokitus=”CONFIDENTIAL”	SELECT sahke//vastaanottaja/nimi where luokitus=”CONFIDENTIAL”
11	SELECT osio/johdanto FROM luku WHERE osio/@lyhytotsikko ABOUT Kouluunlähto	select johdanto from osio where lyhytotsikko=”Kouluunlähtö”

Puuttuva attribuutin erotinmerkki. Aineistossa kahdeksan kertaa esiintyvän puuttuva attribuutin erotinmerkki -virheen teki viisi koehenkilöä. Kyselyssä viitataan attribuuttiin ilman vinoviivaa, joka erottaa emoelementin attribuutti-osasta.

Viittausvirhe

Viittausvirhe esiintyy aineistossa 11 koehenkilön vastauksissa, yhteensä 22 kertaa. Viittausvirheellisessä kyselyssä oleva **`WHERE`**-lause on rakennettu niin, että kysely palauttaa kyselyn kohteena olevan elementin ohessa kaikki sen sisarelementit, vaikka **`WHERE`**-lauseen tarkoituksena on rajoittaa tulosjoukkoa (Taulukko 13).

Tyypillisesti viittausvirhe johtuu siitä, että kyselyn **`FROM`**-osaan on valittu liian ylhäällä dokumentin puurakenteessa sijaitseva elementti. **`FROM`**-osaltaan väärin laadittu kysely palauttaa kaikki haetun elementin sisaret, kun yksikin sisar täyttää **`WHERE`**-lauseessa annetun ehdon. Viittausvirheitä esiintyi myös tilanteissa, joissa kyselyn tulee kohdistua dokumentin puurakenteessa tietyllä tasolla sijaitsevaan elementtiin ja elementti esiintyy samannimisenä myös muilla tasoilla. Esimerkiksi otsikot olivat testiaineistossa tällaisia elementtejä.

Polkuvirhe

Polkuvirhe on aineistossa kaikkein yleisin virhetyyppi. Miltei kaikki koehenkilöt tekivät yhden tai useamman polkuvirheen – virhe esiintyy 33:n koehenkilön vastauksissa yhteensä 113 kertaa. Polkuvirheellisessä kyselyssä on polkurakenne, jolle ei löydy vastaavuutta dokumentissa. Koehenkilön laatima polkurakenne voi esimerkiksi viitata attribuuttiin kuin kyseessä olisi viittaus elementtiin (Taulukko 14).

Taulukko 13: Esimerkkejä viittausvirheellisistä kyselyistä

Numero	Referenssi	Viittausvirheellinen vastaus
2	SELECT luku/otsikko	SELECT otsikko FROM kirja/luku
3	SELECT kappale/@tyyli=puheenvuoro	SELECT kappale FROM kirja WHERE kappale/@tyyli=”puheenvuoro”
4	SELECT sahke/@luokitus=SECRET	SELECT sahke FROM sahkeet WHERE sahke/@luokitus=”SECRET”
8	SELECT otsikko, avainsana FROM tilanne ABOUT eukko	SELECT otsikko, avainsana FROM kirja where tilanne ABOUT ”eukko”
13	SELECT otsikko FROM osio WHERE johdanto ABOUT aamu AND tilanne ABOUT kosioretki	SELECT osio/otsikko FROM kirja where johdanto ABOUT ”aamu”AND tilanne ABOUT ”kosioretki”

Kohdistamisvirhe kyselyn **FROM**-osassa johtaa usein siihen, että kyselyn **WHERE**-osassa viitataan **FROM**-osassa mainittuun elementtiin. Lause voitaisiin tulkita siten että viitataan elementtiin itseensä tai jälkeläisiin. Polkuvirheellisessä kyselyssä käyttäjän kyselyssä viitataan usein **WHERE**-osassa elementtiin, joka on nimetty kyselyn **FROM**-osassa. Elementtiä yritetään hakea elementin itsensä alta.

Taulukko 14: Esimerkkejä polkuvirheellisistä kyselyistä

Numero	Referenssi	Polkuvirheellinen vastaus
3	SELECT kappale/@tyyli=puheenvuoro	select kappale where kappale/@tyyli=”puheenvuoro”
9	SELECT vastaanottaja/nimi FROM sahke/@luokitus=”CONFIDENTIAL”	select vastaanottaja/nimi from sahke where sahke@luokitus=”confidential”
11	SELECT osio/johdanto FROM luku WHERE osio/@lyhytotsikko ABOUT Kouluunlähto	select johdanto from osio where osio/lyhytotsikko about Kouluunlähtö

Kielilaina

Kielilaina-virhe esiintyi aineistossa kaikkiaan 118 kertaa, yhteensä 25 koehenkilön vastauksissa.

Kielilaina-virheellisessä kyselyssä on yhdistetty yhden tai useamman ohjelmointi- tai kyselykielen piirteitä (Taulukko 15). Tavallisimpia kielilainoja ovat XPath-predikaattilauseet sekä XPath/XQuery:n contains-funktion käyttö XIL:n ABOUT-ilmauksen sijaan. Tyypillisimmillään contains-funktio esiintyy lauseissa kuin se olisi osana XPath-ilmaisua. Jossain tapauksissa ABOUT-avainsana on korvattu XPath/XQueryn contains tai SQL:n EXISTS avainsanoilla siten että kysely noudattaa muutoin XIL:n kielioppia.

Taulukko 15: Esimerkkejä kielilainoista

Numero	Referenssi	Kielilainallinen vastaus
4	SELECT sahke/@luokitus=SECRET	select //sahke[@luokitus=”SECRET”]
5	SELECT otsikko FROM osio ABOUT laulu	SELECT osio/otsikko FROM kirja WHERE contains(osio, ”laulu”)
13	SELECT otsikko FROM osio WHERE johdanto ABOUT aamu AND tilanne ABOUT kosioretki	SELECT osio/otsikko FROM kirja where contains(osio/johdanto, ”aamu”) AND contains(osio/tilanne, ”kosioretki”)
15	SELECT tilanne/otsikko, tilanne/avainsana FROM kirja WHERE tilanne/@tunniste=tilanne1 OR tilanne/@tunniste=tilanne2	select //tilanne where /@tunniste=”tilanne1”or /@tunniste=”tilanne2”return /otsikko and /avainsana

Muotovirhe

Muotovirhe esiintyi aineistossa 14 koehenkilön vastauksissa yhteensä 26 kertaa. Muotovirheellisiä kyselyitä ovat kyselyt, jotka sisältävät syntaksivirheen, mutta virhe ei esiinny samassa muodossa systemaattisesti läpi aineiston. Muotovirheet ovat kyselyiden sekalaisia syntaksivirheitä (Taulukko 16).

Taulukko 16: Esimerkkejä muotovirheellisistä kyselyistä

Numero	Referenssi	Muotovirheellinen vastaus
6	SELECT subject ABOUT war FROM cable	select otsikko where exists="war"
12	SELECT jakelu FROM sahke GROUP BY sahke WHERE lahetetty/vuosi<2009	SELECT jakelu FROM sahke WHERE lahetetty/vuosi/text()<2009 ORDER BY sahke
15	SELECT tilanne/otsikko, tilanne/avainsana FROM kirja WHERE tilanne/@tunniste=tilanne1 OR tilanne/@tunniste=tilanne2	SELECT (//tilanne/otsikko, //tilanne/avainsana) FROM /kirja WHERE //tilanne/@tunniste=(tilanne1|tilanne2)

Kielisekaannus

Kolme koehenkilöä laati yhden tai useamman vastauksen jollain muulla kuin testitehtävälomakkeen ohjeistamalla kyselykielellä. Yhteensä tällaisia vastauksia oli 29 kappaletta. Käytetty kieli saattoi olla tunnistettavissa joksikin olemassaolevaksi kysely- tai ohjelmointikieleksi tai vastaukset oli laadittu kielellä, jota käytettiin johdonmukaisesti, mutta lauseet eivät noudattaneet minkään tunnetun ohjelmointi- tai kyselykielen kielioppia (Taulukko 17).

Taulukko 17: Esimerkkejä kielisekaannuksista

Numero	Referenssi	Kielisekaannus
3	SELECT kappale/@tyyli=puheenvuoro	//kappale/@tyyli="puheenvuoro"
5	SELECT otsikko FROM osio ABOUT laulu	SELECT jakelu FROM sahke WHERE lahetetty/vuosi/text()<2009 ORDER BY sahke
10	SELECT tilanne/otsikko, tilanne/avainsana FROM kirja	`for t in //tilanne return [t/otsikko, t/avainsana]`

XQuery-virheluokat

XQuery-kyselyiden suurimpia virheluokkia ovat *muu muotovirhe* ($n = 115$), *polkuvirhe* ($n = 94$) ja *virhe tulosten muotoilussa* ($n = 40$).

Kielilaina-virhe esiintyi aineistossa kuuden koehenkilön vastauksissa yhteensä 16 kertaa. Virheen sisältävissä vastauksissa esiintyi jonkin muun kyselykielen avainsana tai rakenne (Taulukko 18, 1. rivi).

Muu muotovirhe esiintyi aineistossa 115 kertaa, yhteensä 30 koehenkilön vastauksissa. Muut muotovirheet ovat kyselyiden sekalaisia virheitä, joiden muoto vaihtelee yksittäisen koehenkilön vastausten ja koehenkilöiden välillä (Taulukko 18, 2. rivi).

Taulukko 18: XQuery-virheluokat ja vastausesimerkit

Virheluokka	Vastausesimerkki	Referenssi
Kielilaina	//sahke[lahetetty/vuosi < 2009]/jakelu order by sahke	for $sahke in //sahke[lahetetty/vuosi/text()<2009] return {$sahke/jakelu}
Muu muotovirhe	//sahke(/lahetetty/vuosi<2009/)/jakelu//text()	for $sahke in //sahke[lahetetty/vuosi/text()<2009] return {$sahke/jakelu}
Muu pieni syntaksivirhe	kappale/[@tyyli=”puheenvuoro”]	//kappale[@tyyli=”puheenvuoro”]
Muu syntaksivirhe	for $o = //osio where $o[.,contains(”laulu”)] return $o/otsikko	for $a in //osio where contains ($a, ”laulu”) return $a/otsikko
Polkuvirhe	//sahke[/julkaistu/vuosi=2009]/otsikko	/sahkeet/sahke[.//vuosi=”2009”]/otsikko
Predikaattilause	//sahke/@luokitus=”secret”	//sahke[@luokitus = ”SECRET”]
Puutteellinen ratkaisu	//otsikko	//tilanne/(otsikko|avainsana)
Puuttuvat lainausmerkit	//sahke[./@luokitus=SECRET]	//sahke[@luokitus = ”SECRET”]
Useita elementtejä tuottava ehto	//osio[contains(.//*,”laulu”)]/otsikko	//osio[contains(.,”laulu”)]/otsikko
Virhe tietojen määrittelyssä	//osio[lyhytotsikko[(contains(.,”kouluunlähtö”)]/johdanto]	//osio[@lyhytotsikko=”Kouluunlähtö”]/johdanto
Virhe tuloksen muotoilussa	for $n in tilanne return $n/otsikko, $n/avainsana	//tilanne[@tunniste=”tilanne1”or @tunniste=”tilanne2”]/(otsikko|avainsana)

Eri asteisia syntaksivirheitä esiintyi aineistossa yhteensä 70 kertaa. Syntaksivirheet ovat aineistossa johdonmukaisesti esiintyviä virheitä, jotka eivät kuitenkaan ole tunnistettavaksi jonkin muun kyselykielen piirteiksi. *Pieniä syntaksivirheitä* näistä oli 21 ja *muita syntaksivirheitä* 49. Syntaksivirheen tulkittiin olevan pieni, mikäli sen automaattinen korjaaminen kyselykäyttöliittymässä katsottiin mahdolliseksi (Taulukko 18, rivit 3 ja 4).

Predikaattilause-virhe esiintyy aineistossa 14 koehenkilön vastauksissa, kaikkiaan 36 kertaa. Virheen sisältävä kysely tuottaa totuusarvon siitä, esiintyykö kyselyyn sisällytetty merkkijono kyselyn polkuilmaisun osoittamassa kohdassa dokumenttia. Tyypillisesti virhe on tehty tilanteissa, joissa kyselyllä on ollut tarkoitus palauttaa predikaatissa määrätyn ehdon mukainen kohta dokumentista. Koehenkilön muotoilema kysely palauttaa kuitenkin tiedon siitä, esiintykö ehdon mukainen kohta dokumentissa halutun dokumentin kohdan sijaan (Taulukko 18, 6. rivi).

Puuttuvat lainausmerkit -virheen sisältävästä kyselystä puuttuivat merkkijonojen vertailuoperaatiossa tarvittavat lainausmerkit, kyselyn muodon ollessa muutoin syntaksinmukainen. Virhe esiintyi aineistossa kaksi kertaa yhteensä kahden koehenkilön vastauksissa (Taulukko 18, 6. rivi).

Puutteellisesti ratkaistu kysely ei palauta kaikkia tehtävänannon mukaisia tietoja. Aineistossa puutteellisia kyselyitä esiintyi yhteensä kahdeksan kertaa, seitsemän eri koehenkilön vastauksissa (Taulukko 18, 6. rivi).

Useita elementtejä tuottava ehto -virhe esiintyi yhdeksän koehenkilön vastauksissa yhteensä 12 kertaa. Koehenkilö on laatinut kyselyn, jossa yritetään tehdä vertailuoperaatio useaan XML-dokumentin kohtaan viittaavan XPath-ilmauksen ja jonkin merkkijonoarvon välillä. Vertailun kontekstissa XPath-ilmauksen

tulee viitata vain yhteen kohtaan dokumentissa, kun taas koehenkilön vastauksessa ilmaus täsmää useaan dokumentin kohtaan (Taulukko 18, 6. rivi).

Polkuvirheen sisältävässä kyselyssä on polkuilmaus, jolle ei ole vastinetta kohdedokumenteissa. Tyypillisesti virheellisessä polkuilmauksessa käsiteltiin dokumentin attribuutteja elementteinä tai viitataan dokumenttipuussa liian ylös tai alas, polun muodon olessa muutoin oikeellinen. Virhe esiintyi aineistossa 94 kertaa, 31 koehenkilön vastauksissa (Taulukko 18, 6. rivi).

Tietojen määrittelyvirheen sisältävä kysely palauttaa tehtävänantoon nähden virheellisiä tietoja. Tyypillisesti koehenkilö on tehnyt virheen kyselyn predikaatin muotoilussa ja tästä syystä kyselyn tuottama tulos ei ole odotettu. Virhe esiintyi aineistossa 18 kertaa, 11 koehenkilön vastauksissa (Taulukko 18, 6. rivi).

Tuloksen muotoiluvirhe esiintyi 17 koehenkilön vastauksissa yhteensä 40 kertaa. Tyypillisesti virheen tehnyt koehenkilö on unohtanut kyselyn return-osasta sulkumerkit, jotka vaaditaan mikäli kyselyssä halutaan palauttaa useita arvoja (Taulukko 18, 6. rivi).

Kielten vertailua

Analyysin tuloksena syntyneet kyselyvirheluokitukset eivät ole yhteismitalliset, ja kukin kysely voi kuulua yhteen tai useampaan virheluokkaan. Koska kielten syntaksit eroavat toisistaan, myös kyselyiden kirjoittamisessa tehdyt virheet ovat kullekin kielelle ominaisia – yksinkertaiset kirjoitusvirheet poislukien. Jotta kielten vertailu olisi mahdollista, tarvitaan kieliriippumaton tapa järjestellä analysoidut kyselyt toistensa poissulkeviin pääluokkiin. Pääluokituksena käytetään Weltyn ja Stemplen (1981) esittämää kyselyvirheluokitusta (Taulukko 19). Welty ja Stemple jakavat luokituksensa kahteen pääluokkaan, joista neljä ensimmäistä kategoriaa kuuluu olennaisesti oikea vastaus -pääluokkaan ja loput viisi virheellinen vastaus -pääluokkaan.

Luokittelupäätökset tehtiin kyselyiden virheluokkasijaintien pohjalta. Kyselyiden sijoittaminen olennaisesti oikea vastaus -pääluokan kategorioihin oli varsin suoraviivaista. Virheellinen vastaus -pääluokan korjauskelpoinen-kategorian tapauksessa luokittelupäätökset vaativat kyselykohtaista harkintaa. Luokittelupäätös tehtiin arvioimalla, olisiko kyselyn sisältämien virheiden yhdistelmä korjattavissa ohjelmallisin keinoin, esimerkiksi muuttamalla kielen tulkkia, kielioppia tai dokumenttien indeksointitapaa. Jakoperuste korjauskelvottomiin ja korjauskelpoisiin kyselyihin oli siis ennemminkin intuitiivinen, kuin

Taulukko 19: Weltyn ja Stemplen kyselyvirheluokat

Virheluokka	Selite
Oikea vastaus	Kysely on virheetön.
Pieni syntaksivirhe	Kyselyssä on pieni syntaksivirhe, joka on automaattisesti kyselykäyttöliittymän korjattavissa.
Pieni operandivirhe	Kyselyssä on pieni virhe sen käsittelemien tietojen määrittelyssä, kuten esimerkiksi kirjoitusvirhe tietokantataulun sarakkeen nimessä.
Pieni sisältövirhe	Kysely on syntaktisesti oikea, mutta tuottaa väärän tuloksen siksi, että koehenkilö on ymmärtänyt tehtävän ongelmanasettelun joiltain osin väärin.
Korjauskelpoinen	Kyselyssä on virhe, joka on korjattavissa kyselykäyttöliittymän antaman palautteen avulla.
Sisältövirhe	Kysely on syntaktisesti oikein, mutta tuottaa vastauksen joka ei vastaa tehtävänantoa.
Syntaksivirhe	Kyselyssä on selkeä syntaksivirhe.
Puutteellinen	Kysely ei ota huomioon kaikkia tehtävänannon vaatimuksia.
Ei vastausta	Koehenkilö ei laatinut kyselyä.

johonkin tiettyyn virheiden yhdistelmään perustuva. Luokkarajaa korjauskelpoinen ja syntaksivirhe -luokkien välillä ei siksi voida pitää ehdottoman selkeänä, vaan luokittelupäätökset eivät oletettavasti olisi täysin samat, mikäli toinen henkilö luokittelisi kyselyt uudelleen.

Koska yhtenä tutkimustavoitteena oli löytää XIL-kielen kehittämistä palvelevaa tietoa, kyselyiden korjauskelpoisuutta pohdittiin huolellisesti erityisesti tältä kannalta. Tästä syystä on mahdollista, että vakavuudeltaan samankaltaisen syntaksivirheen sisältävä XQuery-kysely on tulkittu luokittelussa syntaksivirheelliseksi, kun taas XIL-kysely on luokiteltu huolellisemman harkinnan jälkeen korjauskelpoiseksi. XIL-kielen päävirheluokkien koostumusta kuvaavasta taulukosta voidaan kuitenkin havaita, että korjauskelpoinen-päävirheluokan muodostuu suurelta osin attribuuttivirheellisiksi luokitelluista kyselyistä, joista attribuutti ilman elementtiä -alivirheen sisältävät kyselyt muodostavat suuren osan. Tämä virhe esiintyy vain XIL-kielellä kirjoitetuissa kyselyissä. XQuery-vastausten virheiden kanssa samankaltaisia ja oletettavasti samankaltaisin tekniikoin ratkaistavissa olevia virheitä oli alle puolessa korjauskelpoisista XIL-kyselyistä. Samankaltaisia virheitä olivat esimerkiksi lievät syntaksivirheet tai virheelliset polkuviittaukset. Suurin osa XIL-kyselyiden korjauskelpoinen-luokituspäätöksistä perustui siis kyselyn sisältämään attribuutti ilman elementtiä -alivirheeseen, eikä näiden kyselyiden kohdalla ollut riskiä samankaltaisen virheen sisältävän XQuery-kyselyn luokittelusta epäedullisesti syntaksivirheellinen-luokkaan.

Koehenkilöt kirjoittivat kieltä kohden yhteensä 585 kyselyä. XIL-kielellä kirjoitetuista kyselyistä olen-

naisesti oikea vastaus -pääkategoriaan luokiteltiin 133 kyselyä ja virheellisiksi 452 kyselyä. XQuery-vastauksista olennaisesti oikein oli 199 ja virheellisiä 386 kyselyä (Taulukot 20 ja 21).

Taulukko 20: Olennaisesti oikeat vastaukset

		Oikea vastaus		Pieni syntaksivirhe		Pieni operandivirhe		Pieni sisältövirhe		∑	
Kysely	C	XIL	XQuery	XIL	XQuery	XIL	XQuery	XIL	XQuery	XIL	XQuery
1	1	30	34	0	2	0	0	0	0	30	36
2	2	12	30	0	1	0	0	0	0	12	31
3	3	3	15	0	1	0	0	0	0	3	16
4	3	3	13	0	2	0	1	0	0	3	16
5	3	6	12	0	0	0	0	0	0	6	12
6	3	13	21	0	0	0	1	0	0	13	22
7	4	11	4	0	1	0	1	0	1	11	7
8	4	8	1	0	0	0	0	0	0	8	1
9	5	1	14	0	1	0	1	0	0	1	16
10	5	25	6	0	0	0	0	0	0	25	6
11	6	4	16	0	1	0	0	0	0	4	17
12	6	1	8	0	1	0	0	0	0	1	9
13	6	8	0	0	1	0	0	0	0	8	1
14	8	7	4	0	1	0	0	0	0	7	5
15	12	1	4	0	0	0	0	0	0	1	4
∑		133	182	0	12	0	4	0	1	133	199

Taulukko 21: Virheelliset vastaukset

		Korjauskelpoinen		Sisältövirhe		Syntaksivirhe		Puutteellinen		Ei vastausta		∑	
Kysely	C	XIL	XQuery	XIL	XQuery	XIL	XQuery	XIL	XQuery	XIL	XQuery	XIL	XQuery
1	1	2	2	1	0	2	1	0	0	4	0	9	3
2	2	1	5	19	0	2	2	1	0	4	1	27	8
3	3	20	5	8	11	4	6	0	0	4	1	36	23
4	3	17	5	9	11	5	5	0	0	5	2	36	23
5	3	4	6	9	1	15	17	0	1	5	2	33	27
6	3	4	5	2	0	15	10	0	0	5	2	26	17
7	4	3	15	11	4	8	10	0	1	6	2	28	32
8	4	2	2	7	0	15	29	0	1	7	6	31	38
9	5	20	8	5	1	6	9	0	0	7	5	38	23
10	5	0	0	4	3	3	24	0	1	7	5	14	33
11	6	13	5	10	3	5	11	0	0	7	3	35	22
12	6	5	7	11	0	12	13	0	1	10	9	38	30
13	6	2	4	5	0	16	28	0	0	8	6	31	38
14	8	1	7	5	1	17	16	0	1	9	9	32	34
15	12	18	0	4	2	8	24	0	1	8	8	38	35
∑		112	76	110	37	133	205	1	7	96	61	452	386

Olennaisesti oikein -pääkategoriaan luokiteltujen kyselyiden joukon pääosan muodostavat testitehtäviin 1–7 annetut vastaukset. Tämän joukon ulkopuolelta, tehtäviin 7–15 annettujen vastauksien joukosta, erottuvat tehtäviin 9–11 laaditut kyselyt, joissa tehtävään yhdeksän kirjoitettiin XQuery-kielellä 16 olennaisesti oikeaa vastausta ja XIL-kielellä vain yksi. Tehtävän 10 kohdalla XIL-vastauksista oli oikein 25 ja XQuery-vastauksista kuusi, kun taas tehtävässä 11 määrät ovat neljä ja 17, XQueryn eduksi. Olennaisesti oikein -pääkategoriaan luokitellut kyselyt olivat suurelta osin täysin oikein. Vain XQuery-vastauksien joukkoon kuului *pieni syntaksivirhe* ($n = 12$), *pieni operandivirhe* ($n = 4$) tai *pieni sisältövirhe*

($n = 1$) -luokiteltuja kyselyitä

Syntaksivirheelliset kyselyt ovat *virheellinen vastaus* -pääkategorin suurin kyselyvirheiden luokka kummankin kielen tapauksessa. Syntaksivirhe-luokkaan kuului 133 XIL-kyselyä ja 205 XQuery-kyselyä. Muiden luokkien kohdalla luokkien suuruusjärjestys riippuu kielestä. XIL-kielen kohdalla virheluokat ovat laskevassa järjestyksessä *korjauskelpoinen* ($n = 112$), *sisältövirhe* ($n = 110$), *ei vastausta* ($n = 96$) ja *puutteellinen* ($n = 1$). XQueryn tapauksessa luokkien järjestys on *korjauskelpoinen* ($n = 76$), *ei vastausta* ($n = 61$), *sisältövirhe* ($n = 37$) ja *puutteellinen* ($n = 7$).

Taustamuuttujat ja suoriutuminen

Tässä luvussa tarkastellaan testitehtäväsuoriutumisen ja koehenkilöiden esitietämyksen, opiskeluaktiivisuuden sekä kurssimenestyksen yhteyttä. Lisäksi selvitetään, oliko vastauslomakkeiden täyttöjärjestyksellä tai kokeeseen käytetyllä ajalla vaikutusta testävästä suoriutumiseen. Käytettyjä taustamuuttujia ovat kokemusindeksi (kts. luku 7.3), tieto ohjelmointiharrastuksesta, koeaika, opiskeluaika sekä kurssin kokonaisarvosana.

Taustamuuttujista lomakejärjestys sekä ohjelmointiharrastuneisuus ovat kaksiluokkaisia. Tarkastelun yhdenmukaistamiseksi jatkuvat taustamuuttujat – kokemusindeksi, koeaika ja opiskeluaika – kaksiluokkaistettiin, käyttäen jakopisteenä muuttujien keskiarvoa. Näin koehenkilöistä muodostuu kunkin muuttujan suhteen kaksi ryhmää, joiden testitehtäväsuoriutumista kuvaavat liitteen 4 taulukot. Ryhmien välisten erojen tilastollinen merkitsevyys selvitettiin Mann Whitney U-testillä. Järjestysasteikollisen kurssiarvosana-taustamuuttujan sekä testitehtävistä suoriutumisen yhteyttä tarkasteltiin Kendallin järjestyskorrelaation τ_b avulla. Koska kurssiarvosana on tulkittavissa myös suhdeasteikolliseksi muuttujaksi, tau_b:n ohella yhteyden mittana käytetään myös tavanomaista Pearsonin tulomomenttikorrelaatiota *r*.

Vaikka muuttujien kaksiluokkaistaminen on yleinen käytäntö, sitä vastaan on esitetty voimakasta kritiikkiä. MacCallum (2002) vetää kritiikkiä yhteen sekä havainnollistaa käytännön haitallisuuden: useimmissa tilanteissa kaksiluokkaistamisesta on yksinomaan negatiivisia seurauksia. MacCallumin mukaan luokittelun mahdollistaman keskiarvovertailun sijaan analyysimenetelmänä olisi parempi käyttää regressiota sekä korrelaatiotarkasteluja, jolloin muuttujan hienovaraisen vaihtelun kantamaa informaatiota ei tarvitse hävittää luokittelulla.

Tässä tutkielmassa jatkuvien muuttujien kaksiluokkaistamisen perusteena on jakaumataulukoiden esitysmuodon yhdenmukaistaminen valmiiksi kaksiluokkaisten muuttujien suhteen tehtävien vertailujen kanssa. Muuttujien kaksiluokkaistamista koskeva kritiikki otetaan huomioon testaamalla jatkuvien taustamuuttujien sekä tehtäväsuoritumisen Pearson-korrelaatio sen lisäksi että kaksiluokkaistettujen taustamuuttujien vaikutusta selvitetään U-testillä.

Tilastollisesti merkitseviä tai melkein merkitseviä eroja vertailuryhmien välille ilmeni vertailuissa koeaika- ja opiskeluaika-taustamuuttujien suhteen. Jatkuvien muuttujien korrelaatiotarkasteluiden sekä U-testauksen tulokset ovat samansuuntaiset. Testien tulosten (LIITE 4) pohjalta voidaan todeta:

- Yli koeajan keskiarvon aikaa käyttäneet jättivät enemmän oikeita XIL-vastauksia (Liite 4, taulukko 9: $U = 125{,}5$; $p = 0{,}073$).

  Oikein vastaamisen ja koeajan välillä vallitsee kohtalainen positiivinen korrelaatio (Liite 4, taulukko 10: $r = 0{,}34$; $r^2 = 0{,}11$; $p = 0{,}03$).
- Alle koeajan keskiarvon aikaa käyttäneet jättivät enemmän tyhjiä XQuery-vastauksia (Liite 4, taulukko 9: $U = 248{,}5$; $p = 0{,}042$). Vastaamatta jättämisen ja koeajan välillä vallitsee kohtalainen negatiivinen korrelaatio (Liite 4, taulukko 10:$r = -0{,}48$; $r^2 = 0{,}23$; $p = 0{,}002$).
- Keskiarvoa enemmän opiskelleet tekivät vähemmän korjauskelvottomia XQuery-virheitä (Liite 4, taulukko 13: $U = 120{,}5$; $p = 0{,}046$). Korjauskelpoisten vastausten määrän sekä opiskeluajan välillä vallitsee kohtalainen positiivinen korrelaatio (Liite 4, taulukko 14: $r = 0{,}33$; $r^2 = 0{,}10$; $p = 0{,}04$).
- Keskiarvoa vähemmän opiskelleet jättivät enemmän tyhjiä XQuery-vastauksia (Liite 4, taulukko 13: $U = 266$; $p = 0{,}009$). Vastaamatta jättämisen ja opiskeluajan välillä vallitsee kohtalainen negatiivinen korrelaatio (Liite 4, taulukko 14: $r = -0{,}37$; $r^2 = 0{,}14$; $p = 0{,}02$).
- XQueryllä vastaamisen aloittaneet vastasivat enemmän oikein XQuery-tehtäviin (Liite 4, taulukko 17: $U = 127$; $p = 0{,}076$).
- Omaehtoisesti ohjelmointia harrastaneet jättivät enemmän oikeita XQuery-vastauksia (Liite 4, taulukko 20: $U = 80$; $p = 0{,}02$).
- Oikeiden XIL-vastausten lukumäärän ja kurssiarvosanan välillä vallitsee kohtalainen positiivinen Kendall ($\tau_b = 0{,}41$; $p = 0{,}003$) ja Pearson ($r = 0{,}51$; $p = 0{,}002$) -korrelaatio (Liite 4, taulukko 1).
- Oikeiden XQuery-vastausten lukumäärän ja kurssiarvosanan välillä vallitsee kohtalainen positiivinen Pearson ($r = 0{,}36$; $p = 0{,}03$) ja heikko Kendall ($\tau_b = 0{,}25$; $p = 0{,}03$) -korrelaatio (Liite 4, taulukko 2).

– Tyhjien XQuery-vastausten lukumäärän ja kurssiarvosanan välillä vallitsee kohtalainen negatiivinen Pearson (r = -0,31; p = 0,06) -korrelaatio (Liite 4, taulukko 2).

Tilastollisten testien tulokset vahvistavat intuitioon perustuvan oletuksen siitä, että kurssiin voimakkaammin sitoutuneet opiskelijat ovat koehenkilöinä toimiessaan suoriutuneet koetehtävistä paremmin, kuin opiskelijat joiden sitoutuminen on ollut keskimääräistä heikompaa. Odottamaton ero muodostui lomakejärjestyksen mukaan muodostettujen ryhmien välille – XQueryllä vastaamisen aloittaneet suoriutuivat paremmin XQuery-tehtävistä kuin koehenkilöt jotka aloittivat vastaamisen XIL:llä. Ryhmien välinen ero ei kuitenkaan ole tilastollisesti kovinkaan merkitsevä, mutta alittaa kuitenkin 10 % riskitason. Koehenkilöiden aiemmalla kokemuksella ei ollut tilastollisesti merkitsevää vaikutusta, joka oli myös odottamaton tulos.

Halstead-vaikeus ja suoriutuminen

Tässä luvussa tarkastellaan regressioanalyysin avulla luvussa 7.3 käsitellyn kyselyiden referenssitoteutusten Halstead-vaikeuden sekä testitehtäviin laadittujen oikeiden vastausten lukumäärän välistä yhteyttä. Työhypoteesina on, että oikeiden vastauksien lukumäärän ja Halstead-vaikeuden välillä vallitsee negatiivinen yhteys – mitä vaikeampi kysely on, sitä vähemmän koehenkilöt ovat laatineet oikeita vastauksia.

Regressioanalyysissa tutkitaan yhden tai useamman selittävän muuttujan vaikutusta selitettävään muuttujaan sekä tämän vaikutuksen voimakkuutta. Tästä vaikutussuhteesta pyritään luomaan tilastollinen esitys, regressiomalli. Regressiomallin rakentamiseen on useita tapoja, joista seuraavassa on valittu käyttöön näistä yksinkertaisin: yhden selittäjän lineaarinen regressio. Käytössä on vain yksi selittävä muuttuja – Halstead-vaikeus – ja vaikutussuhteen selittävän ja selitettävän muuttujan välillä oletetaan olevan lineaarinen. (Draper & Smith 1998, 15, 19–20.) Selitettävänä muuttujana on kutakin Halstead-vaikeustasoa edustaviin kyselyihin annettujen oikeiden vastauksien lukumäärä. Taulukot 22 ja 23 esittelevät regressioanalyysien tulokset.

Regressiomallin β -kerroin kuvaa, kuinka suuren muutoksen selittävän muuttujan kasvu yhdellä saa aikaan selitettävässä muuttujassa. Kertoimen hyvyyttä mallissa testataan t-testillä, jota käytettäessä mitattujen arvojen poikkeaman mallin ennustearvosta oletetaan noudattavan t-jakaumaa.

Studentin t-jakauman mukaan 13 vapausasteella t-arvon tulee olla itseisarvoltaan $\geqslant$ 2,16, jotta 5 % riskitasolla voidaan hylätä olettamus siitä, että regressiokertoimen todellinen arvo on nolla. XIL:n tapauk-

sessa *t*:n arvoksi saatiin -1,78, jonka itseisarvo ylittää niukasti 10 % riskitason *t*-arvon 1,770. XQuery:lle *t*-arvo on -4,39 joka riittää ylittämään 0,01 % riskitason.

Regressiomallin *F*-testi (taulukossa F-arvo) testaa nollahypoteesia, jonka mukaan regressiomallin kaikki kertoimet ovat nollia (Draper & Smith 1998, 38–39). XIL-regressiomallille F on 3,168 joka ylittää 10 % riskitason rajan 3,13 vapausasteilla 1 ja 13 (Taulukko 22). XQueryn regressiomallin F on 19,3, joka riittää 1 % riskitason saavuttamiseen (Taulukko 23). Kummankin regressiomallin regressiokertoimen arvo eroaa nollasta tilastollisesti merkitsevästi.

Regressiomallin ennustevoimaa voidaan luonnehtia *F*-testin tuloksen perusteella. Kokemusperäisten arvioiden mukaan F-arvon tulisi ylittää valittu riskitaso vähintään nelinkertaisesti, jotta mallia voitaisiin käyttää ennustetarkoituksissa (Draper & Smith 1998, 244). XIL-regressiomallille tämä raja jää saavuttamatta ($p = 0{,}098$), kun taas XQuerylle saatu tulos $p = 0{,}0007$ ylittää 5 % riskitason rajan selvästi.

R^2-luku kuvaa regressiomallin selitysosuutta. Luku saadaan laskemalla selitettävän muuttujan mitattujen arvojen sekä mallin tuottamien ennustearvojen korrelaation neliö (Draper & Smith 1998, 33). Mitä suurempi selitysosuus on, sitä suurempi osa regressiomallin selitettävien muuttujien vaihtelusta voidaan selittää selittävien muuttujien vaihtelun kautta. Korjattu R^2 saadaan ottamalla huomioon laskennassa muuttujien määrä ja otoskoko. Selitysosuus ilmaistaan tyypillisesti prosentteina, jolloin R^2 arvon 0,3 saavan regressiomallin todetaan selittävän 30 % havaitusta vaihtelusta.

XIL-regressiomallin korjattu R^2 on 0,1 joka on varsin vähän – vaihtelusta jää selittämättä 90 %. Tuntemattomat tekijät vaikuttavat vastauksien oikeellisuuteen enemmän kuin kyselyiden Halstead-vaikeus. Sekoittavia tekijöitä voivat olla esimerkiksi aineiston käsittelyssä tehty virhe, koehenkilön ajatusvirhe tai motivaatioon liittyvät seikat. XQueryn kohdalla mallin korjattu R^2 on 0,6, joten mallin selitysosuus on selvästi XIL-regressiomallia parempi. Kaikilla koehenkilöillä oli ohjelmointikokemusta proseduraalisista kielistä, jolloin kyselyn Halstead-vaikeuden ennustevoima kyselyn oikeamuotoisuudelle tulee paremmin esille kuin XIL:n tapauksessa, jossa koehenkilöille vaikeuksia aiheutti kielen syntaksin hallinta. Kuten luvussa 7.3 todettiin, merkittävä osa virheellisistä XIL-vastauksista oli virheellisiä johdonmukaisella tavalla – koehenkilöt olettivat kielen toimivan eri tavoin, kuin sen testattu toteutus.

Edellä käsiteltyjen F ja R^2 -lukujen ohella estimaatin keskivirhe on regressiomallin onnistuneisuutta kuvaava tunnusluku. Estimaatin virheellä tarkoitetaan yksittäisen mitatun havainnon etäisyyttä regressiosuorasta, kun taas estimaatin keskivirhe on näiden keskihajonta. Mitä suurempi estimaatin keskivirhe on, sitä pienempi on regressiomallin selitysvoima. XIL-regressiomallin estimaatin keskivirhe on 8,

mikä on varsin suuri suhteutettuna selitettävän muuttujan mahdolliseen arvoalueesen (0–39). XQueryn mallin estimaatin keskivirhe on 6,6, joka myös on vaihteluväli huomioiden kohtalaisen suuri. Näin myöskään XQuery-regressiomalli ei ennusta kovinkaan hyvin odotettavissa olevaa oikeiden vastauksien määrää kyselyn Halstead-vaikeuden perusteella, vaikka rohkaisevat F ja R^2 -tunnuslukujen arvot antoivatkin näin olettaa.

Taulukko 22: XIL-regressiotarkastelu

	Regressiokerroin	*t*-arvo	merkitsevyys
(Vakio)	13,9 **	3,982	**
β	-1,9 †	-1,78	*
N	15		
R^2	0,2		
Korjattu R^2	0,1		
F-testi	3,168		†
Estimaatin keskivirhe	8,0		

Merkitsevyystasot: † $p <, 10$; * $p <, 05$; ** $p <, 01$; *** $p <, 001$.

Taulukko 23: XQuery-regressiotarkastelu

	Regressiokerroin	*t*-arvo	merkitsevyys
(Vakio)	23,1 ***	7,62	***
β	-2,8 ***	-4,39	***
N	15		
R^2	0,6		
Korjattu R^2	0,6		
F-testi	19,31		***
Estimaatin keskivirhe	6,6		

Merkitsevyystasot: † $p <, 10$; *$p <, 05$; **$p <, 01$; ***$p <, 001$.

8 XIL-kielen kehitysehdotuksia

Luvun aloittaa yhteenveto koetulosten analyysin yhteydessä esitetyistä kehitysehdotuksista. Koetuloksiin perustuvia kehitysehdotuksia täydennetään tutkimuskirjallisuuteen nojautuvilla huomioilla niistä kyselykielten ja hakujärjestelmien piirteistä, jotka parantaisivat vuorovaikutteisessa käyttötilanteessa tehtyjen hakujen tuloksellisuutta.

8.1 Koetuloksiin perustuvia kehitysehdotuksia

XIL-kielen kehittämisen kannalta mielenkiintoisinta tietoa kantavat ne virheelliset kyselyt, joista suuri osa suhteessa muihin virkeluokkiin kuuluu korjauskelpoinen-luokkaan. Tällaisia ovat tehtäviin 3, 4, 9 ja 11 kirjoitetut kyselyt (Taulukko 24).

Taulukko 24: XIL-mallivastaukset testitehtäviin, joissa korjauskelpoinen-luokkaan kuuluvien vastauksien osuus virheellisistä vastauksista on suuri

Kysely	Mallivastaus
3	`SELECT kappale/@tyyli=puheenvuoro`
4	`SELECT cable/@classification=SECRET`
9	`SELECT to/name FROM cable/@classification=CONFIDENTIAL`
11	`SELECT osio/johdanto FROM luku WHERE osio/@lyhytotsikko ABOUT Kouluunlähto`

Mallivastauksista voidaan nähdä, että kaikkien näiden tehtävien tapauksessa koehenkilön on tullut hallita XIL-kielen tapa käsitellä attribuutteja ja elementtien hierarkkisia suhteita. Tehtävässä 11 koehenkilön on tullut lisäksi osata **WHERE**-lauseen käyttö.

Taulukot 25 ja 26 esittävät yhteenvedon tehtävissä 3, 4, 9 ja 11 tehdyistä virheistä. Päävirheet listaavasta taulukosta (Taulukko 25) huomataan, että tehtäviin laadittujen kyselyiden virheistä suurin osa kuuluu joko attribuuttivirheiden tai polkuvirheiden joukkoon. Alivirheet listaava taulukko (Taulukko 26) vahvistaa, että kyselyissä tehdyt virheet liittyvät juuri niihin XIL-kielen ominaisuuksiin, joiden osaamista nyt tarkastelun kohteena olevat kyselyt mittasivat.

Taulukko 25: Päävirheiden jakautuminen XIL-kyselyissä 3, 4, 9 ja 11

Alivirhe	Kyselyn numero			
	3	4	9	11
Attribuuttivirhe	18	15	16	10
Datavirhe	2	4	3	3
Kielilaina	4	5	4	3
Kielisekaannus	3	3	2	1
Muotovirhe	2		3	1
Polkuvirhe	10	12	11	12
Sisältövirhe	1	1		
Viittausvirhe	3	6	4	2

Taulukko 26: Alivirheiden jakautuminen XIL-kyselyissä 3, 4, 9 ja 11

Alivirhe	Kyselyn numero			
	3	4	9	11
Attribuutti ilman elementtiä	13	11	10	8
Attribuutin käsittely elementtinä	3	3	5	3
Puuttuva attribuutin erotin	1	2	2	2
SELECT korvattu	1	1	1	1
SELECT puuttuu	1	1	1	
Viittausvirhe		7	5	7
Viittausvirhe kyselyn WHERE-osassa	4			
Virheellinen viittaus attribuuttiin	1	1	1	
XPath-kielilaina	1	2	2	2
XQuery-kielilaina			1	

Koetulosten perusteella XIL-kielen syntaksin sukulaisuus SQL ja XPath -kyselykielille ei helpota oikeamuotoisten kyselyiden kirjoittamista. Erityisesti attribuutteja sisältävissä kyselyissä XIL:n samankaltainen mutta yhteensopimaton syntaksi SQL:n kanssa näyttäisi pikemminkin vaikeuttavan kuin helpottavan oikeellisten kyselyiden kirjoittamista. Mikäli XIL-kielen tavoitteena on mahdollistaa XML-kyselykielen nopea omaksuminen SQL:n hallitseville käyttäjille, saatujen tulosten perusteella kieltä tulisi viedä lähemmäksi SQL-syntaksia.

Kyselyiden ymmärtämistä selvittäneessä kokeessa jotkut koehenkilöistä kommentoivat XIL-syntaksissa sallittua **FROM**-osan poisjättöä. **FROM**-osa on SQL-syntaksissa pakollinen osa kyselyä. Suuri osa virheellisistä kyselyistä sisälsi ilman elementtiä esiintyviä attribuuttteja. Tämän virheen taustalla voi olla XIL:n SQL:stä poikkeava tulkinta sallia kyselyä rajaavat ehdot myös kyselyn **SELECT**-osassa, polkuilmaisun

muodossa. SQL-kyselyissä ehdot asetetaan kyselyn **WHERE**-lohkoissa ja **SELECT**-osassa luetellaan kyselyn tuloksiksi haluttavat tietoalkiot.

Ratkaisun attribuutteihin liittyviin virheisiin voisi tuoda kyselyn syntaksin ja tulkinnan muuttamisen siten, että ilman elementtiä esiintyvät attribuutit sallitaan. Tällöin kyselyn **WHERE**-lohkossa luetellut attribuutit tulkittaisiin **SELECT**-osassa annettuja elementtejä rajaaviksi ehdoiksi. Vielä lähemmäksi SQL-syntaksia päästäisiin, mikäli attribuutteja ja elementtejä käsiteltäisiin samankaltaisina tietoalkioina. Tällöin kyselyä muodostettaessa ei tarvitse tietää, esiintykö haluttu tieto tai rajaava ehto dokumentin elementeissä vai attribuuteissa.

Sekaannukset XPath-kielen kanssa voisivat olla vältettävissä, mikäli polkuilmaisuissa erotinmerkkinä käytettäisiin vinoviivan sijaan pistettä. SQL-syntaksissa pistettä käytetään erottamaan taulujen ja sarakkeiden nimiä.

Koehenkilöiden vastauksissa yleiset elementtien hierarkkisiin suheisiin liittyvät virheet ovat yleisiä myös verrattain harjaantuneiden käyttäjien laatimissa kyselyissä. Tiedonhaun tutkijoiden laatimien XPath-kyselyiden virheistä tätä tyyppiä edustavat virheet muodostivat valtaosan (O'Keefe & Trotman 2003). Elementtien muodostaman hierarkian hallintaa voisi helpottaa, jos polkuilmaisuuun sisällytettyjen elementtien tulkittaisiin oletusarvoisesti olevan vanhempi-jälkeläinen-suhteessa nykyisen vanhempi-lapsi-tulkinnan sijaan.

8.2 Muihin tutkimuksiin perustuvia kehitysehdotuksia

Reisnerin (1977, 244) antamat SEQUEL-kielen ja hakukäyttöliittymän kehitysehdotukset voivat olla päteviä ja hyödyllisiä myös vuorovaikutteisen XML-haun tapauksessa. Reisner suosittaa, että SEQUEL-hakukäyttöliittymässä otettaisiin käyttöön oikeinkirjoituksen tarkastus, synonyymisanasto sekä tietokantataulujen ja -sarakkeiden nimien osia täsmäyttävä toiminnallisuus. Näyttöä näiden toiminnallisuuksien hyödystä ei kuitenkaan ole, sillä Reisnerin ehdottamia toiminnallisuuksia ei ole toteutettu SEQUEL/SQL-kielen myöhemmissä versioissa. Oletettavasti tähän syynä on se, että vastoin kielen suunnittelun lähtöoletuksia, SEQUEL/SQL-kielen pääasiallisia omaksujia eivät olleetkaan tietokantojen sisältämiä tietoja ad hoc -tyyppisesti tarvitsevat tietointensiivisten alojen ammattilaiset, vaan päätoimiset ohjelmoijat, jotka käyttivät SQL-kieltä sovelluskehityksen työvälineenä.

Reisnerin kehitysehdotukset olisivat parantaneet SEQUEL/SQL-kielen käyttökokemusta etenkin vuorovaikutteisessa käyttötilanteissa, joiden osuus kaikista käyttötilanteista oli käytännössä paljon pienempi kuin kielen kehittäjät alkujaan olettivat. Vuonna 1975 tehdyn kyselytutkimuksen mukaan noin 40 %

yrityksistä, jotka olivat hankkineet käyttöönsä erityisesti ei-teknisiä käyttäjiä varten suunniteltuja tietokantatuotteita, käyttivät tietokantoja ainoastaan ammattilaisohjelmoijien avulla (Haigh 2006). Oletettavasti samansuuntainen tilanne vallitsi myös ensimmäisen SQL-kieltä tukeneen System R -järjestelmän tullessa markkinoille hieman myöhemmin, vuonna 1977. Noin 20 vuotta tästä, vuonna 1993, tehdyn kyselytutkimuksen mukaan vain noin 14 % kyselyyn vastanneista SQL-kielen käyttäjistä saattoi luonnehtia loppukäyttäjiksi ja lähes 60 % vastaajista kuului ohjelmistoammattilaisten joukkoon. (Lu et al 1993.)

Alkujaan Reisnerin muotoilemat kehitysehdotukset ovat sittemmin toistuneet muita kyselykieliä käsittelevien tutkimusten yhteydessä kyselykielten ja hakujärjestelmien ominaisuuksina, joiden oletetaan parantavan vuorovaikutteisten hakujen tuloksellisuutta. Seuraavaksi käsitellään näitä sekä muita XML-tiedonhaun tuloksellisuutta parantavia ominaisuuksia XIL-kielen kehittämisen näkökulmasta.

Tiedonhaun olemuksellisen epätarkkuuden huomioiminen XML-tiedonhaussa

Kyselyssä ilmaistujen rakenteiden täsmällisestä tulkinnasta saatava hyöty ei ole selvä vuorovaikutteisessa XML-tiedonhaussa. Suuren, heterogeenisen XML-dokumenttikokoelman kohdalla on epärealistista olettaa, että tiedonhakija olisi perillä dokumenttien rakenteesta. Jos käyttäjän laatimiin kyselyihin suhtaudutaan niin että ne ovat tiedontarpeen täsmällisiä ilmaisuja, ollaan tekemisissä kyselykielen ohjelmointityyppisen käyttötapauksen kanssa, jossa kyselyn laatija tietää etukäteen dokumenttien rakenteen. Tällöin kyselyn palauttaman tiedon muoto ja rakenne ovat etukäteen selvillä ja jatkokäsittelyn kannalta on tärkeää, että palautetut elementit vastaavat täsmällisesti kyselyssä asetettuja ehtoja.

Jos kyselyyn suhtaudutaan sen sijaan käyttäjän tiedontarpeen epätäsmällisenä ilmauksena, kyselyn tuloksellisuuden arviointi on tehtävä sen suhteen, tyydyttävätkö palautetut dokumentit tai dokumentin osat tiedontarpeen. Tällöin olennaista ei ole, vastaavatko kyselyssä ilmaistut rakenteet täsmällisesti palautettujen tietojen rakenteita. Kampsin mukaan XML-tiedonhaussa saattaisi olla hyödyllistä käsitellä kyselyissä ilmaistuja rakenteita pikemminkin haettavan tiedon rakennetta koskevina vihjeinä, kuin ehdottomina hakuvaatimuksina (Kamps et al 2006, 433). Huomio on lähes sama, kuin mitä Reisner (1977) esitti aiemmin SEQUEL-kieleen liittyen – tietokannan hakujärjestelmän tulisi sietää variaatiota taulujen ja sarakkeiden nimissä ja palauttaa tuloksia, vaikka kyselyssä esitetyt rakenne-ehdot eivät vastaisikaan täysin tietokannan rakennetta.

Fuhrin ja Kain (2001) mukaan tiedonhakuun tarkoitetussa XML-kyselykielissä tulisi huomioida tiedonhaun olemuksellinen epätarkkuus. Dokumenttien sisällön ohella osittaustäsmäytys tulisi ulottaa myös

dokumentin rakenteisiin. Esimerkiksi sillä, löytyykö haettu tieto dokumentin attribuuteista vai elementeistä ei ole välttämättä merkitystä useimmissa käyttötilanteissa.

Liu ja muut (2005) toteavat XML-kokoelmien rakenteiden olevan useimmiten tiedonhakijoille liian monimutkaisia ymmärtää kokonaan. Tiedonhakijoilla saattaa olla jokin käsitys dokumenttien rakenteista, mutta tämä tietämys ei ole täydellistä. Tästä huolimatta useimpien XML-hakukoneiden suunnittelun lähtöoletuksena on kuitenkin ollut, että käytettäessä sisältö ja rakenne -tyyppistä hakua dokumentin rakenne on tunnettu ja siksi kyselyssä annetut rakenne-ehdot tulkitaan tiukasti. Tiedonhaun tuloksellisuuden kannalta voisi olla kuitenkin parempi, jos rakenne-ehtojen tulkintaa väljennettäisiin. Näin toimien epätäydellisen rakennetietämyksen hyödyntäminen kyselyn muotoilussa olisi mahdollista, mutta erehdys rakenne-ehtojen määrittelyssä ei johtaisi haun epäonnistumiseen, kuten tapahtuu rakenne-ehtoja tiukasti tulkittaessa.

Epätäydellinen tietämys dokumenttien rakenteesta voi myös tarkoittaa tilannetta, jossa tiedonhakija olettaa tuntevansa dokumentin rakenteen haun kohteena olevien dokumenttien genren perusteella – esimerkiksi tieteellisen artikkelin voidaan olettaa sisältävän ainakin kirjoittajien nimet, otsikon, abstraktin ja lähdeluettelon. Dokumenteissa nämä tiedot eivät välttämättä ole kuitenkaan nimetty tiedonhakijan olettamalla tavalla. Esimerkiksi tekijän mukaan hakiessaan tiedonhakija saattaa olettaa kirjoittajan nimen olevan `writer`-kentässä, kun dokumenteissa tekijätieto onkin tallennettu elementtiin nimeltä `author`. Reisner (1977) tunnisti tämän ongelman SEQUEL-hakujen taulujen ja sarakkeiden nimissä ja ehdotti ratkaisuksi synoonyymisanaston käyttöönottoa. XML-tiedonhaun kontekstissa ongelmaan on viitattu tunnistenimiepävarmuuden (tag name ambiguity) käsitteellä (Li et al 2004). Kuten Reisner (1977) myös Li ja muut (2004) näkevät sovellusaluespesifin synonyymisanaston olevan toimiva ratkaisu tähän ongelmaan.

Kyselyksi muotoillun tiedontarpeen olemuksellinen epätarkkuus voidaan ottaa huomioon hakujärjestelmässä eri tavoin. Hakujärjestelmä voi lähtökohtaisesti pitää kyselyä tiedontarpeen epätarkkana ilmaisuna ja sisällyttää tuloksiin automaattisesti myös muiden kuin kyselyssä eksplisiittisesti mainittujen elementtien ja attribuuttien sisältöjä. Tämä on esimerkiksi Liun ja muiden (2005) järjestelmässä sovellettu tapa tulkita NEXI-kyselyitä.

Sen sijaan, että hakujärjestelmä automaattisesti olettaisi kyselyn olevan epätarkka tiedontarpeen ilmaus, kyselykieleen voidaan sisällyttää primitiivejä, joiden avulla tiedonhakija voi osoittaa epätarkkoina pitämänsä kyselyn osat. Fuhrin ja Kain (2001) XIRQL-kielessä aaltomerkillä (∼) ilmaistaan, että kyselyssä

nimetty tietoalkio voi olla joko elementti tai attribuutti. Li ja muut (2004) ja muut laajentavat XQuery-kieltä funktioilla, jotka lisäävät kieleen mahdollisuuden kuvailla epätarkkoja rakenne-ehtoja. Campi ja muut (2009) ovat toteuttaneet vastaavia funktioita XPath-kieleen.

Proseduraaliset piirteet kyselykielissä

Tutkijat eivät ole yksimielisiä siitä, tulisiko satunnaisille käyttäjille suunnatuissa ohjelmointi- tai kyselykielissä pyrkiä mahdollisimman korkeaan deklaratiivisuuden asteeseen. Solowayn (1982) näkemyksen mukaan satunnaisille käyttäjille olisi hyödyllistä opettaa proseduraalista kyselykieltä, sillä aiemman tutkimuksen perusteella proseduraaliset kielet näyttäisivät olevan helpompia oppia. Lisäksi proseduraalisen kielen oppimisen oheisvaikutuksena on havaittu suotuisia muutoksia henkilöiden yleisissä ongelmanratkaisutaidoissa. (Soloway 1982.)

Solowayn tekemät kokeet paljastivat, että täysin ohjelmointitaidottomat henkilöt laativat ongelmanratkaisutehtäviin deskriptiivisiä ratkaisukuvauksia, kun taas ohjelmoinnin alkeet hallitsevat henkilöt turvautuivat proseduraalisiin ratkaisumalleihin. Vaikeammissa tehtävissä deskriptiiviset kuvaukset johtivat selvästi useammin virheellisiin ratkaisuihin, kun taas proseduraalisia ratkaisumalleja käyttäneet suoriutuivat tehtävistä paremmin. Soloway huomauttaa, että tämä havainto on linjassa Weltyn ja Stemplen (1981) nonproseduraalisia ja proseduraalisia kyselykieliä vertailleen koeasetelman tulosten kanssa: tehtävien kompleksisuuden kasvaessa proseduraaliset ratkaisut tuottivat useammin oikean tuloksen. (Soloway 1982.)

Soloway ei kiistä deklaratiivisten kielten hyötyjä, mutta suosittaa että deklaratiiviseen paradigmaan perehdyttäisiin vasta proseduraalisen ajattelutavan oppimisen jälkeen (Soloway 1982). Varsinkin jos proseduraalista ongelmanratkaisukykyä pidetään perustavanlaatuisena taitona lukemisen tapaan (vrt. Mateas 2005), saattaisi olla hyödyllistä, että ensikosketus formaalien kielten maailmaan tapahtuisi proseduraalisen kielen kautta.

Lloyd (1994) edustaa vastakkaista kantaa. Hänen mukaansa proseduraalinen ohjelmointitapa vaikuttaa negatiivisesti ohjelmoijien tuottavuuteen ja estää ohjelmointitaidon leviämisen sovelluskehittäjien ammattikunnan ulkopuolelle. Lloydin mukaan ohjelmointikielten kehitys- ja tutkimustyössä tulisi pyrkiä kehittämään vahvan deklaratiivisia kieliä, joita käyttäessään ohjelmoija voisi keskittää huomionsa ohjelman logiikkaan ja välttää ohjelman kontrollivuon hallintaan liittyvät kysymykset.

Manthey (1990) tähdentää, ettei pyrkimys kyselykielen deklaratiivisuuteen saisi olla itsetarkoitus. Hänen mukaansa hyvin suunnitellut proseduraaliset ominaisuudet parantavat kyselykielen käytettävyyttä

etenkin monimutkaisissa kyselyissä. Manthey kiistää kirjallisuudessa esitetyt väitteet proseduraalisen ja deklaratiivisen lähestymistavan yhteensopimattomuudesta. Hän näkee lupaavimpana kehityssuuntana kyselykieli- ja tietokantaparadigmat, joissa yhdistetään sekä deklaratiivisia- että proseduaalisia piirteitä.

Proseduraalisia piirteitä omaavasta kyselykielestä olisi hänen mukaansa hyötyä etenkin sovelluskehitystyössä, jossa tarvittavat kyselyt ovat usein paljon monimutkaisempia kuin vuorovaikutteisissa käyttötilanteissa kirjoitetut. Manthey toteaa puhtaan deklaratiivisten kyselykielten palvelevan hyvin yksinkertaisia ad hoc -käyttötilanteita, mutta tekevän monimutkaisempien kyselyiden ilmaisusta hankalaa. Tämä toteamus on linjassa Reisnerin ja myöhempien tutkimustulosten kanssa – esimerkiksi alikyselyt tuottavat koehenkilöille vaikeuksia riippumatta koehenkilön ohjelmointikokemuksen määrästä, kun taas yksinkertaisten, yhden **`SELECT-FROM-WHERE`** -lohkon kyselyiden laatiminen onnistuu myös ei-teknisiä aloja edustavilta koehenkilöiltä.

Mantheyn toive proseduraalisia ja deklaratiivisia piirteitä yhdistävästä kyselykielestä voidaan nähdä toteutuneen esimerkiksi XQuery ja LINQ-kyselykielelissä. W3C:n johdolla kehitettävä XQuery ja Microsoftin NET-ohjelmistokehykseen läheisesti liittyvä LINQ ovat funktionaalisia, laskennallisesti täydellisiä kyselykieliä. LINQ-kieli on sovellusaluespesifi, .NET-ohjelmistokehyksen kanssa yhteensopivien ohjelmointikielten laajennos, kun taas XQuery soveltuu käytettäväksi itsenäisenä ohjelmointikielenä tai SQL:n tapaan upotettuna jollain muulla kielellä kirjoitetun ohjelmakoodin joukkoon. (Box & Hejlsberg 2007, Buxton et al 2011.)

Listaukset 8 ja 9 havainnollistavat tapaa, jolla XQuery ja LINQ-kielissä on yhdistetty proseduraalisten ja deklaratiivisten kielten piirteitä. Listauksista käy myös ilmi XQuery- ja LINQ-syntaksien samankaltaisuus.

Listaus 8: XQuery-esimerkkikysely

```
for $joukkue in /joukkueet/joukkue
  where $joukkue/voitot >= 1
  order by $joukkue/voitot descending
  return $joukkue
```

Listaus 9: LINQ-esimerkkikysely

```
var voittajat =
  from joukkue in joukkueet
    where joukkue.voitot >= 1
    orderby joukkue.voitot descending
    select joukkue;
```

Esimerkkikyselyiden tapa käyttää muuttujia sekä toistorakennetta ovat kyselykielten proseduraalisia piirteitä, kun taas valinta- ja järjestysehdot annetaan käyttäen deklaratiivista ilmaisutapaa. Kokonaisuudessaan esimerkkikyselyiden hahmo on lähellä tavanomaista englannin kieltä, ja muuttujan käsitteen tunteva henkilö voi lukea kyselyt luonnollisen kielen lauseiden tapaan ilman, että hänen tarvitsee tuntea käytettyjä kyselykieliä. Näin voidaan olettaa, sillä kyselykielten intutiivista ymmärtämistä selvittäneen kokeen perusteella (Luku 7.2) ohjelmoinnin alkeet hallitsevan henkilön voitiin todeta ymmärtävän ongelmitta yksinkertaisia proseduraalisia rakenteita sisältäviä kyselyitä ilman, että käytetty kyselykieli oli hänelle ennestään tuttu.

Puhtaan deklaratiivisessa ilmaisussa pysyminen ei näyttäisi tuottavan etua kyselykielen satunnaiskäyttäjille. Sitävastoin näyttö proseduraalisen ajattelutavan hallinnan ja yleisen ongelmanratkaisutaidon välisestä myönteisestä syy-yhteydestä puhuu proseduraalisia piirteitä sisältävien kyselykielen puolesta. Tällöin kyselykielten opettamisen kautta voitaisiin vaikuttaa suotuisasti myös muilla elämänaloilla hyödyllisiin ongelmanratkaisutaitoihin (Soloway 1982).

Ammattimaista ohjelmointia vastaavissa tilanteissa kyselykielen proseduraalisten piirteiden on todettu helpottavan kyselyiden kirjoittamista (Welty & Stemple 1981, Soloway 1982, Manthey 1990). Vaikka tässä voi olla kyse osittain myös ohjelmoijien henkilökohtaisista mieltymyksistä ja tavoista (vrt. Manthey 1990), vaikutus on silti todellinen. Kielen proseduraalisuus voi olla eduksi myös kielen oppimisessa silloin kun kielen käyttäjällä ei ole edeltävää ohjelmointikokemusta (Welty & Stemple 1981).

Koska kyselykielen proseduraalisten piirteiden kokonaisvaikutus näyttäisi olevan positiivinen, XIL-kielen myöhemmissä versioissa voisi olla hyödyllistä harkita, onko olemassa perusteita pitäytyä puhtaan deklaratiivisessa ilmaisutavassa, vai olisiko esimerkisi alikyselyiden tuki viisainta toteuttaa muuttujien avulla, joka olisi askel proseduraaliseen suuntaan.

Hakujärjestelmään tallennetun tietämyksen hyödyntäminen

Hakujärjestelmän sisältämän tiedon hyödyntämisestä hakuvuorovaikutuksen parantamisessa on esitetty jo varhaisissa kyselykielitutkimuksissa. Esimerkiksi Reisner (1977) ehdotti tietokantaskeeman käyttöä kyselyjen muotoilun tukena, sillä merkittävä osa kyselyn kirjoittamisessa tehdyistä virheistä oli käyttäjätestauksessa paljastunut yksinkertaisiksi syntaksivirheiksi taulujen nimissä. XML-tiedonhaun kontekstissa esimerkiksi Erdmann ja Studer (2001), Fuhr (2001) ja Mandreoli (2004) ovat esittäneet ratkaisuja, jotka käyttävät XML-skeeman, dokumenttityyppimäärittelyn tai muuta hakujärjestelmän sisältämää lisäinformaatiota kyselyn uudelleenmuotoilussa tai hakuvuorovaikutuksen tukena.

XML-tiedonhaun yhteydessä skeeman tai dokumenttityyppimäärittelyn hyödytämisestä ongelmallista tekee kuitenkin se, ettei skeema- tai dokumenttityypimäärittely ole pakollinen osa XML-dokumenttia (Bray et al 1998). Tagarellin ja Grecon (2010) mukaan skeemojen käyttö voisi yksinkertaistaa XML-dokumenttien käsittelyä, mutta koska skeematieto puuttuu heidän mukaansa suuresta osasta käytännön sovellutuksissa kohdatusta XML:stä, näin ei kuitenkaan käytännössä voida tehdä. Toisaalta kuitenkin Mlynkovan ja muiden (2006) tutkimuksessa dokumenttikeskeistä merkintätapaa noudattaneista tutkimusaineiston dokumenteista ($n = 6691$) 93,7 % sisälsi viittauksen dokumenttityyppimäärittelyyn ja 57,8 % XML-skeemaan. Ilman minkäänlaista merkkaustapa- tai arvoaluemäärittelyä oli vain 6,3 % dokumenteista (Mlynkova et al 2006). Tätä varhaisemissa tutkimuksissa (Mignet et al 2003) dokumenttityyppimäärittely löytyi 48 %:sta dokumenteista, joista 92 % oli viittauksia nykyään jo käytöstä poistuneeseen WAP-mobiilisisältöstandardiin. Vain 0,09 % dokumenteista viittasi XML-skeemaan. (Mignet et al 2003.)

Mlynkovan ja muiden (2006) sekä Mignetin ja muiden (2003) tulokset eivät ole suoraan vertailukelpoisia, sillä tutkimuksissa käytettiin erityyppisiä aineistoja. Mignet ja muut keräsivät aineiston avointa verkkoa haravoimalla, kun taas Mlynkovan ja muiden aineisto on peräisin käsin valitusta joukosta XML-muodossa sisältöä jakelevia sivustoja. Koska XML-kyselykieliä ei ole pääsääntöisesti tarkoitettu hakujen tekemiseen avoimesta verkosta, kyselykielten suunnittelussa on perustellumpaa olettaa haun kohteena olevien dokumenttien sisältävän viittauksen skeemaan, kuin tehdä vastakkainen oletus. Lisäksi vastakkaista oletusta puoltava Mignetin ja muiden saama tulos skeemaa tai DTD:tä noudattavan aineiston alle 50 % osuudesta voi olla nykyään toinen myös avoimesta verkosta haravoidun aineiston tapauksessa.

Sen sijaan, että kyselykielen suunnittelun ensisijaisena lähtökohtana pidettäisiin skenaariota, jossa tiedonhakijan tulee tehdä hakuja skeemattomiin dokumentteihin, edellä käsitellyn perusteella suunnitte-

lupäätökset voitaisiin tehdä vastaamaan tilanteeseen, jossa haun kohteena on eri skeemoja noudattavia samaan sovellusalueeseen liittyviä dokumentteja. Tällöin tiedonhakija voi tuntea dokumenttien skeemat, mutta ei välttämättä tapaa, jolla dokumentit noudattavat kutakin skeemaa, poislukien pakollisiksi määritellyt dokumentin osat. Tämäntyyppisiä aineistoja ovat esimerkiksi kielitieteellisessä tutkimuksessa käytetyt tekstikorpukset, joissa sovellettuja XML-skeemoja ovat muun muassa TEI[17], XCES[18] ja NITF[19].

Erdmann ja Studer (2001) esittelevät hakujärjestelmän, jossa ontologian muotoon tallennettua tietoa XML-dokumenttien sovellusalueesta käytetään kyselyiden uudelleenmuotoilun tukena. Perusajatukseltaan kyseessä on vastaava kuin varhaisissa kyselykielitutkimuksissa ehdotetut synonyymisanastoihin ja tesauruksiin perustuvat ratkaisut – tehtyä kyselyä laajennetaan hakujärjestelmään tallennetun lisäinformaation pohjalta. Tällöin käyttäjä voi tehdä hakuja käyttäen kyselyn rakenne-ehtoina tietoalkioiden nimiä, jotka eivät esiinny samassa muodossa järjestelmään tallennetussa aineistossa. Lisäksi käyttäjälle palautettaviin tuloksiin voidaan sisällyttää alkuperäiseen kyselyyn täsmäämätöntä, mutta sovellusaluekontekstin perusteella relevanttia aineistoa. (Erdmann & Studer 2001.)

Fuhr ja Großjohann (2001) ehdottavat skeematiedon laajamittaista hyödyntämistä XML-tiedonhaussa. Heidän näkemyksensä mukaan relevanssilajitteleva XML-hakujärjestelmä on mielekästä suunnitella käyttäen lähtö-olettamana dokumenttien skeemanmukaisuutta. Tällöin tiedonhakijan avuksi voidaan toteuttaa erilaisia kyselyn muotoilua helpottavia toimintoja. Samoin mahdollistuu semanttisesti sumea haku. Kuten Erdmannin ja Studerin (2001) ontologioihin perustuvassa ratkaisussa, tiedonhakija voi muotoilla kyselynsä käyttäen myös muita kuin dokumenteista löytyviä rakenteita. Eräänä skeematiedon sovelluskohteena Fuhr ja Großjohann mainitsevat myös maantieteelliseen ja ajalliseen läheisyyteen perustuvat haut. (Fuhr & Grosjohann 2001.)

Mandreoli ja muut (2004) ovat kehittäneet hakujärjestelmän, jossa heterogeeniseen dokumenttikokoelmaan tehdyt kyselyt uudelleenmuotoillaan skeematiedon pohjalta. Järjestelmään syötetyt XQuery-kyselyt uudelleenmuotoillaan dokumenttikokoelmaan kuuluvien dokumenttien rakenteita vastaaviksi käyttäen apuna Wordnet-tietokannasta haettua tietoa tietoalkioiden nimien semanttisista suhteista. (Mandreoli et al 2004.)

Vaikka skeematiedon hyödyntämisestä olisi kiistattomia hyötyjä, käytännössä näin ei kuitenkaan voida aina tehdä. Dokumentti voi viitata skeemaan, mutta sisältää skeeman ulkopuolisia rakenteita, mikäli

[17] http://www.tei-c.org/
[18] http://www.xces.org/
[19] http://www.iptc.org/site/News_Exchange_Formats/NITF/

dokumentin käsittelyvaiheissa ei huolehdita jatkuvasta skeemanmukaisuudesta. Tämän lisäksi on huomioitava, että käsittelyn kohteena oleva dokumenttikokoelma on voinut kasvaa orgaanisesti, ilman rakenteiden etukäteissuunnittelua. Esimerkiksi tämän tutkielman aineistoja hallittiin XML-tietokannassa, jossa olevien dokumenttien rakenteita täydennettiin ja muokattiin useita kertoja tutkielman aineiston analyysin aikana. Dokumenttien skeemattomuuden ansioista muutoksia oli mahdollista tehdä nopeasti eri työvaiheissa esiin nousseiden tarpeiden mukaan. XML-dokumenttien ad hoc -hakua tukisi parhaiten järjestelmä, joka sallisi haun skeemattomista dokumenteista, mutta skeematiedon ollessa saatavana hyödyntäisi sitä hakuvuorovaikutuksen parantamisessa.

9 Yhteenveto

Suunnittelutieteellisellä tutkimuksella on käytännönläheinen päämäärä. Tässä tutkielmassa tarkoituksena oli tuottaa suunnittelu- ja kehitystyölähtöistä tutkimustapaa noudattaen toteutetun XIL-kielen jatkokehitystä tukevaa tietoa. Erityisesti XIL-kieltä koskevaa tietoa tuotettiin käyttäjäkokeiden avulla, kun taas kirjallisuuteen perehtyen luotiin ymmärrystä, jonka voi katsoa päteväksi minkä tahansa ad hoc -kyselykielen tapauksessa.

Monien tutkimuskirjallisuudessa esitettyjen kehitysehdotusten kohdalla voitiin todeta kehitysehdotuksen esiintyneen jossain muodossa jo varhaisten kyselykielitutkimusten loppupäätelmissä. Kokeellista näyttöä kehitysehdotusten toimivuudesta ei kuitenkaan ole saatavissa.

Oletus deklaratiivisen ilmaisutavan paremmuudesta suhteessa proseduraalisia piirteitä sisältäviin kieliin osoittautui osin perusteettomaksi. Pyrkimyksestä puhtaan deklaratiiviseen ilmaisuun voidaan todeta olevan hyötyä vain kaikkein yksinkertaisimpien kyselyiden tapauksessa. Muissa käyttötilanteissa kyselykielen proseduraaliset piirteet voivat parantaa kielen käytettävyyttä.

Tutkimuskirjallisuuden pohjalta ohjelmointiin harjaantumattomien koehenkilöiden odotettiin vastaavan oikein 44–65 % osuuteen testitehtävistä, kun taas ohjelmointitaitoisten odotettiin yltävän 55–78 % suoritustasoon.

Kyselyiden kirjoittaminen -kokeeseen osallistuneet koehenkilöt laativat kieltä kohden yhteensä 585 kyselyä. XIL-kielellä kirjoitettiin yhteensä 133 ja XQueryllä 199 oikeaa vastausta. Oikeiden vastauksien prosenttiosuudet jäivät varsin kauas odotetusta suoritustasosta, sillä XIL-vastauksista 23 % ja XQuery-vastauksista 34 % oli oikein. Kyselyiden ymmärtämistä selvittäneessä kokeessa kielten välillä ei havaittu tilastollisesti merkitsevää eroa.

Mikäli XIL-kieleen toteutetaan tässä tutkielmassa esitetyt muutokset, suuntaa antavana suoriutumistasona voidaan käyttää prosenttiosuutta, joka saadaan laskemalla yhteen oikeiden vastauksien sekä korjauskelpoisiksi luokiteltujen vastauksien määrät. Tällöin XIL-kieltä käyttävän koehenkilön voidaan olettaa saavan oikein vähintään 42 % laatimistaan kyselyistä. XQuery-kyselyistä tulisi olla oikein noin 50 % olettaen että esitetyt muutokset voitaisiin tehdä myös XQuery-kieleen.

Vaikka korjattunakin XIL-kielellä saavutettava suoritustaso näyttäisi jäävän taakse vertailukohtana käytetystä XQuerystä, tämän perusteella ei kuitenkaan voida todeta kielen olevan aiottuun käyttötarkoituk-

seensa kelpaamaton. Tässä tutkielmassa saatujen tuloksien perusteella koehenkilöiden tausta, kuten ohjelmointikokemus ja ajankäytöllä sekä kurssiarvosanalla mitattu motivaatio, vaikuttivat suoriutumiseen vähäistä merkittävämmin.

Koetilanteessa keskiarvoa etemmän aikaa käyttäneet jättivät enemmän oikeita XIL-vastauksia ja oikeiden vastauksien lukumäärä ja kurssiarvosana korreloivat kohtalaisesti kummankin kielen tapauksessa. Koetilanteessa tai opiskellessa keskiarvoa vähemmän aikaa käyttäneet jättivät enemmän tyhjiä XQuery-vastauksia.

Koehenkilöiden taustan vaikutus näkyi myös regressioanalyysin tuloksissa. XIL-regressiomallin tapauksessa kyselyiden Halstead-vaikeus selitti vain 10 % havaitusta vaihtelusta.

Tuntemattomien tekijöiden vaikutuksen voimakkuutta ei voi erotella, mutta motivaation kaltaisten taustatekijöiden lisäksi regressiomallin selitysosuutta oletettavasti laskee johdonmukaisella tavalla virheellisten vastauksien suuri määrä – ajatusvirheet, joita koehenkilöt tekivät muotoillessaan kyselyitä. Kielen testattu toteutus toimi eri tavalla, kuin koehenkilöt olettivat.

Tehdyt ajatusvirheet voidaan tulkita merkiksi siitä, ettei uuden XML-kyselykielen suunnittelu käyttäen lähtökohtana olemassaolevia kieliä ole välttämättä hyvä suunnitteluratkaisu. Vaikka XIL-kielen syntaksi muistuttaa SQL ja XPath-kyselykieliä, eroavuuksia on kuitenkin niin paljon, etteivät koehenkilöt voineet suoraan soveltaa näihin kieliin liittyvää tietämystään. Jotta syntaksin samankaltaisuudesta olemassaolevan kielen kanssa olisi etua, uuden kielen tulisi olla lähtökohtana käytetyn kielen yhteensopiva laajennos, ei murre.

Kielten testaminen paperilomakkeella rajoittaa saatujen tulosten hyödynnettävyyttä XIL-kielen kehittämisen apuna. Vuorovaikutteiseen käyttöön tarkoitetuissa hakukäyttöliittymissä voi olla kyselyn muotoilua helpottavia ominaisuuksia, kuten syntaksintarkastus ja XML-dokumenttipuun osia täydentävä ennakoiva tekstinsyöttö, jotka voivat estää tutkimusaineistossa runsaimmin esiintyneiden virheiden – attribuutti ja viittausvirheiden – tapahtumisen. Saadut tulokset kuvaavat testatuilla kielillä saavutettavaa suoriutumistasoa käyttötilanteessa, jossa tiedonhakija joutuu muotoilemaan kyselyn muistinvaraisesti ilman käyttöliittymän tai dokumentaation tukea. Mikäli XIL-kieltä halutaan kehittää tässä tutkielmassa esitettyjen kehitysehdotusten pohjalta, käyttäjäkokeiden tulokset tulisi validoida uusintatestillä mahdollisimman todenmukaisessa käyttöliittymässä.

Lähteet

Al-Qutaish, R. & Abran, A. (2005). An analysis of the design and definitions of Halstead's metrics. *Proceedings of the 15th International Workshop on Software Measurement (IWSM 2005)*. 337–352. (s. 64)

Bamford, R., Borkar, V., Brantner, M., Fischer, P. M., Florescu, D., Graf, D., Kossmann, D., Kraska, T., Muresan, D., Nasoi, S. & Zacharioudakis, M. (2009). XQuery reloaded. *Proceedings of the VLDB Endowment* 2, 1342–1353.
<http://www.vldb.org/pvldb/2/vldb09-1078.pdf> (s. 30)

Batra, D., Hoffler, J. A. & Bostrom, R. P. (1990). Comparing representations with relational and EER models. *Communications of ACM* 33, 126–139.
<http://doi.acm.org/10.1145/75577.75579> (s. 50, 52)

Bentley, J. (1986). Programming pearls: little languages. *Communications of ACM* 29, 711–721.
<http://doi.acm.org/10.1145/6424.315691> (s. 22)

Biancuzzi, F. & Warden, S. (2009). *Masterminds of programming: conversations with the creators of major programming languages*. O'Reilly. (s. 25, 26, 27)

Borthick, A., Bowen, P. L., Jones, D. R. & Tse, M. H. K. (2001). The effects of information request ambiguity and construct incongruence on query development. *Decision Support Systems* 32(1), 3 – 25.
<http://dx.doi.org/10.1016/S0167-9236(01)00097-5> (s. 61, 62)

Bowen, P. L., Ferguson, C. B., Lehmann, T. H. & Rohde, F. H. (2003). Cognitive style factors affecting database query performance. *International Journal of Accounting Information Systems* 4(4), 251–273.
<http://dx.doi.org/10.1016/j.accinf.2003.05.002> (s. 52, 61)

Box, D. & Hejlsberg, A. (2007). LinQ: .NET language-integrated query. *MSDN Developer Centre* .
<http://msdn.microsoft.com/en-us/library/bb308959.aspx> (s. 89)

Bray, T., Paoli, J. & Sperberg-McQueen, C. M. (1998). Extensible markup language (XML) 1.0. W3C Recommendation REC-XML-19980210.
<http://www.w3.org/TR/1998/REC-xml-19980210> (s. 2, 91)

Buxton, S., Case, P. & Rys, M. (2011). XQuery and XPath full text 1.0 requirements. W3C Working Group Note 25.
<http://www.w3.org/TR/xquery-full-text-requirements/> (s. 3, 89)

Campi, A., Damiani, E., Guinea, S., Marrara, S., Pasi, G. & Spoletini, P. (2009). A fuzzy extension of the XPath query language. *Journal of Intelligent Information Systems* 33, 285–305.
<http://dx.doi.org/10.1007/s10844-008-0066-3> (s. 88)

Catarci, T. & Santucci, G. (1995). Are visual query languages easier to use than traditional ones? An experimental proof. *Proceedings of the HCI'95 Conference on People and Computers X*. 323–338.
<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.45.4510> (s. 16, 19)

Chamberlin, D. (2003). Influences on the design of XQuery. Teoksessa Katz, H. (toim.) *XQuery from the Experts: A Guide to the W3C XML Query Language*. Addison-Wesley, 81–141. (s. 3, 30)

Chamberlin, D. (2009). SQL. Teoksessa Liu, L. & Özsu, M. T. (toim.) *Encyclopedia of Database Systems*. Springer, 2753–2760. (s. 26)

Chamberlin, D. D., Astrahan, M. M., Blasgen, M. W., Gray, J. N., King, W. F., Lindsay, B. G., Lorie, R., Mehl, J. W., Price, T. G., Putzolu, F., Selinger, P. G., Schkolnick, M., Slutz, D. R., Traiger, I. L., Wade, B. W. & Yost, R. A. (1981). A history and evaluation of System R. *Commununications of ACM* 24(10), 632–646.
<http://doi.acm.org/10.1145/358769.358784> (s. 26)

Chamberlin, D. D. & Boyce, R. F. (1974). SEQUEL: a structured english query language. *Proceedings of the 1974 ACM SIGFIDET (SIGMOD) Workshop on Data description, access and control*. 249–264.
<http://dx.doi.org/10.1145/800296.811515> (s. 2)

Chan, H. (1996). Effect of grading schemes on outcomes in query writing experiments. *Interacting with Computers* 8(1), 7–12.
<http://dx.doi.org/10.1016/0953-5438(95)01021-1> (s. 50, 51, 52)

Chan, H., Lim, L., Lim, L. & Loh, V. (1994). Evaluation of query languages with software science metrics. *Proceedings of 1994 IEEE Region 10's 9th Annual International Conference. Frontiers of Computer Technology (TENCON'94)*. 516–520.
<http://dx.doi.org/10.1109/TENCON.1994.369247> (s. 61)

Chan, H. C. (1999). The relationship between user query accuracy and lines of code. *International Journal of Human-Computer Studies* 51(5), 851–864.
<http://dx.doi.org/10.1006/ijhc.1999.0264> (s. 61)

Chan, H. C., Wei, K. K. & Siau, K. L. (1993). User-database interface: the effect of abstraction levels on query performance. *MIS Quarterly* 17(4), 441–464.
<http://dx.doi.org/10.2307/249587> (s. 51, 52)

Codrea-Rado, A. (2012). What can journalism learn from computer science? columbia journalism school. TOW center for digital journalism.
<http://towcenter.org/what-can-journalism-learn-from-computer-science/> (s. 20)

Cohen, W. W. (1998). Integration of heterogeneous databases without common domains using queries based on textual similarity. *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*. SIGMOD '98, New York, NY, USA: ACM, 201–212.
<http://doi.acm.org/10.1145/276304.276323> (s. 27)

Darwen, H. (2005). Having a blunderful time or wish you were where.
<http://www.dcs.warwick.ac.uk/~hugh/TTM/HAVING-A-Blunderful-Time.html> (s. 25)

De, P., Sinha, A. P. & Vessey, I. (2001). An empirical investigation of factors influencing object-oriented database querying. *Information Technology and Management* 2(1), 71–93.
<http://dx.doi.org/http://dx.doi.org/10.1023/A:1009934820999> (s. 51, 52)

DeRose, S. (1997). Navigation, access, and control using structured information. *American Archivist* 60(3), 298–309.
<http://www.jstor.org/stable/40294439> (s. 10, 11)

London Centre for Digital Humanities., U. C. (2011).
<http://www.ucl.ac.uk/dh> (s. 20)

Draper, N. R. & Smith, H. (1998). *Applied regression analysis*. 3. painos. Wiley Series in Probability and Statistics, Wiley-Interscience. (s. 80, 81)

Elmasri, R. & Navathe, S. (2011). *Fundamentals of database systems*. 6. painos. Addison-Wesley. (s. 11, 12, 17, 18, 19, 20)

Erdmann, M. & Studer, R. (2001). How to structure and access XML documents with ontologies. *Data & Knowledge Engineering* 36(3), 317–335.
<http://dx.doi.org/10.1016/S0169-023X(00)00048-3> (s. 92)

Euroopan Komissio (2011). *Avoin data. Innovoinnin, kasvun ja läpinäkyvän hallinnon moottori.*
<http://ec.europa.eu/information_society/policy/psi/docs/pdfs/opendata2012/open_data_communication/fi.pdf> (s. 1)

Facebook (2012). Facebook query language (FQL). .
<https://developers.facebook.com/docs/reference/fql/> (s. 27)

Fegaras, L. (2012). MRQL: A Map-Reduce Query Language.
<http://lambda.uta.edu/mrql/description.html> (s. 27)

Fowler, M. (2010). *Domain-specific languages*. 1. painos. Addison-Wesley Signature Series, Addison-Wesley.
<http://martinfowler.com/books/dsl.html> (s. 20, 22, 23)

Fuhr, N. & Grosjohann, K. (2001). XIRQL: A query language for information retrieval in XML documents. *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '01, 172–180.
<http://doi.acm.org/10.1145/383952.383985> (s. 86, 87, 92)

Galindo, J., Medina, J., Pons, O. & Cubero, J. (1998). A server for fuzzy SQL queries. Teoksessa Andreasen, T., Christiansen, H. & Larsen, H. (toim.) *Flexible Query Answering Systems*. Lecture Notes in Computer Science, Springer Berlin Heidelberg, 164–174.
<http://dx.doi.org/10.1007/BFb0055999> (s. 27)

Glushko, R. J. & McGrath, T. (2005). *Document engineering: analyzing and designing documents for business informatics and web services*. MIT Press. (s. 8, 9)

Graaumans, J. (2005a). A collection of XML documents and query tasks. UU-CS-2005-038, Institute of information and computing sciences, Utrecht University. (s. 59)

Graaumans, J. (2005b). *Usability of XML query languages*. Väitöskirja, Instituut voor Informatica en Informatiekunde, Universiteit Utrecht.
<http://igitur-archive.library.uu.nl/dissertations/2005-1018-200002/> (s. 27, 38, 39, 40, 59, 61)

Gray, J., Liu, D. T., Nieto-Santisteban, M., Szalay, A., DeWitt, D. J. & Heber, G. (2005). Scientific data management in the coming decade. *SIGMOD Record* 34(4), 34–41.
<http://doi.acm.org/10.1145/1107499.1107503> (s. 1)

Grün, C., Holupirek, A. & Scholl, M. H. (2007). Visually exploring and querying XML with BaseX. *Proceedings of Database Systems for Business, Technology, and Web (BTW 2007)*. 629–632.
<http://nbn-resolving.de/urn:nbn:de:bsz:352-opus-118154> (s. 3)

Haigh, T. (2006). A veritable bucket of facts. origins of the data base management system. *SIGMOD Record* 35, 33–49.
<http://doi.acm.org/10.1145/1147376.1147382> (s. 28, 86)

Halstead, M. H. (1977). *Elements of software science*. Operating and Programming Systems Series, Elsevier. (s. 61)

Harper, R. (2008). Position paper: practical foundations for programming languages. *SIGPLAN Notices* 43, 71–73.
<http://doi.acm.org/10.1145/1480828.1480843> (s. 30)

Heuer, A. & Priebe, D. (2000). Integrating a query language for structured and semi-structured data and IR techniques. *Proceedings of 11th International Workshop on Database and Expert Systems Applications (DEXA'00)* , 703.
<http://dx.doi.org/10.1109/DEXA.2000.875102> (s. 27)

Hey, A. & Trefethen, A. (2003). The data deluge: an e-science perspective. UK e-science core programme.
<http://eprints.soton.ac.uk/257648/1/The_Data_Deluge.pdf> (s. 1)

Hey, T., Tansley, S. & Tolle, K. (toim.) (2009). *The fourth paradigm: data-intensive scientific discovery*. Microsoft Research.
<http://research.microsoft.com/en-us/collaboration/fourthparadigm/4th_paradigm_book_complete_lr.pdf> (s. 1)

Hoc, J., Green, T., Samurçay, R. & Gilmore, D. (1990). *Psychology of programming*. Academic Press.
<http://www.cl.cam.ac.uk/teaching/1011/R201/> (s. 33)

Howe, B. & Cole, G. (2010). SQL is dead; long live SQL: lightweight query services for ad hoc research data. *4th Microsoft eScience Workshop*.
<http://escience.washington.edu/sites/default/files/msr_publication.pdf> (s. 1, 2)

Howe, B. & Halperin, D. (2012). Advancing declarative query in the long tail of science. *IEEE Data Engineering Bulletin* 35(3), 16–26.
<http://sites.computer.org/debull/A12sept/tail.pdf> (s. 1, 2)

Jarke, M. & Vassiliou, Y. (1985). A framework for choosing a database query language. *ACM Computing Surveys* 17(3), 313–340.
<http://dx.doi.org/http://doi.acm.org/10.1145/5505.5506> (s. 16, 17, 20)

Junkkari, M., Arvola, P. & Kekäläinen, J. (2006). Grammatical approach to XML information retrieval query languages. Report A-2006-5, University of Tampere, Department of Computer Science. (s. 3, 6, 28, 29, 30)

Kamps, J., Marx, M., Rijke, M. & Sigurbjörnsson, B. (2006). Articulating information needs in XML query languages. *ACM Transactions of Information Systems* 24, 407–436.
<http://doi.acm.org/10.1145/1185877.1185879> (s. 86)

Kay, M. (2003). XQuery, XPath, and XSLT. Teoksessa Katz, H. (toim.) *XQuery from the Experts: A Guide to the W3C XML Query Language*. Addison-Wesley, 145–183. (s. 30, 31)

Kepser, S. (2004). A simple proof for the Turing-completeness of XSLT and XQuery. *Proceedings of Extreme Markup Languages 2004*.
<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.71.8846> (s. 30)

Kibert, C. J. & Hollister, K. C. (1994). An enhanced construction specific SQL. *Automation in Construction* 2(4), 303 – 312.
<http://dx.doi.org/10.1016/0926-5805(94)90006-X> (s. 27)

Kilpeläinen, P. (2012). Using XQuery for problem solving. *Software: Practice and Experience* 42(12), 1433–1465.
<http://dx.doi.org/10.1002/spe.1140> (s. 30)

Ko, A. J., Abraham, R., Beckwith, L., Blackwell, A. F., Burnett, M. M., Erwig, M., Scaffidi, C., Lawrance, J., Lieberman, H., Myers, B. A., Rosson, M. B., Rothermel, G., Shaw, M. & Wiedenbeck, S. (2011). The state of the art in end-user software engineering. *ACM Computing Surveys* 43(3), 21.
<http://dx.doi.org/10.1145/1922649.1922658> (s. 20)

Leavenoworth, B. M. & Sammet, J. E. (1974). An overview of nonprocedural languages. *Proceedings of the ACM SIGPLAN Symposium on Very High Level Languages*. 1–12.
<http://doi.acm.org/10.1145/800233.807040> (s. 21)

Lesk, M. (2003). Asilomar in Lowell - May 5 2003. The Lowell database research self-assessment meeting.
<http://research.microsoft.com/en-us/um/people/gray/Lowell/Lesk_Notes_new.doc> (s. 32)

Li, Y., Yu, C. & Jagadish, H. V. (2004). Schema-free XQuery. *Proceedings of the 13th International Conference on Very large data bases*. VLDB '04, 72–83.
<http://www.vldb.org/conf/2004/RS2P3.PDF> (s. 87, 88)

Liu, S., Chu, W. W. & Shahinian, R. (2005). Vague content and structure (VCAS) retrieval for document-centric XML collections. *Proceedings of the International Workshop on the Web and Databases*. 79–84.
<http://www.cobase.cs.ucla.edu/tech-docs/sliu/WebDB05.pdf> (s. 87)

Lloyd, J. W. (1994). Practical advantages of declarative programming. *Proceedings of the Joint Conference on Declarative Programming (GULP-PRODE'94)*.
<ftp://clip.dia.fi.upm.es/pub/papers/PARFORCE/second_review/D.WP3.1.M2.3.ps.Z> (s. 22, 88)

Lu, H., Chan, H. C. & Wei, K. K. (1993). A survey on usage of SQL. *SIGMOD Record* 22(4), 60–65. <http://doi.acm.org/10.1145/166635.166656> (s. 86)

MacCallum, R. C., Zhang, S., Preacher, K. J. & Rucker, D. D. (2002). On the practice of dichotomization of quantitative variables. *Psychological Methods* 7(1), 19–40. <http://dx.doi.org/10.1037//1082-989X.7.1.19> (s. 78)

Mandreoli, F., Martoglia, R. & Tiberio, P. (2004). Approximate query answering for a heterogeneous XML document base. Teoksessa Zhou, X., Su, S., Papazoglou, M., Orlowska, M. & Jeffery, K. (toim.) *Web Information Systems – WISE 2004*. Lecture Notes in Computer Science, 337–351. <http://dx.doi.org/10.1007/978-3-540-30480-7_35> (s. 91, 92)

Manning, C. D., Raghavan, P. & Schütze, H. (2008). *Introduction to information retrieval*. 1. painos. Cambridge University Press. <http://nlp.stanford.edu/IR-book/> (s. 23)

Manthey, R. (1990). Declarative languages - paradigm of the past or challenge of the future? *Proceedings of the 1st International East-West Database Workshop*. Lecture Notes in Computer Science, Springer Verlag, 1–16. <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.226.1792> (s. 21, 88, 90)

Markstrum, S. (2010). Staking claims: a history of programming language design claims and evidence: a positional work in progress. *Proceedings of Evaluation and Usability of Programming Languages and Tools (PLATEAU '10)*. ACM, 7:1–7:5. <http://doi.acm.org/10.1145/1937117.1937124> (s. 22)

Mateas, M. (2005). Procedural literacy: Educating the new media practitioner. Teoksessa *On the Horizon: Special Issue on Future Strategies for Simulations, Games and Interactive Media in Educational and Learning Contexts*. 2005. <http://dm.lcc.gatech.edu/~mateas/publications/MateasOTH2005.pdf> (s. 88)

McJones, P. (1997). The 1995 SQL reunion: People, projects, and politics. . <http://www.mcjones.org/System_R/SQL_Reunion_95/> (s. 2, 27)

McJones, P. (2009). Oral history of Donald Chamberlin. . <http://www.computerhistory.org/collections/accession/102702111> (s. 25, 31)

McVeigh, R. & Müller, F. (2011). Content management interoperability services (CMIS) version 1.0 OASIS standard. Oasis standard incorporating approved errata 01, OASIS Content Management Interoperability Services (CMIS) TC.
<http://docs.oasis-open.org/cmis/CMIS/v1.0/errata-01/os/cmis-spec-v1.0-errata-01-os-complete.html> (s. 27)

Mignet, L., Barbosa, D. & Veltri, P. (2003). The XML web: a first study. *Proceedings of the 12th international conference on World Wide Web*. WWW '03, ACM, 500–510.
<http://doi.acm.org/10.1145/775152.775223> (s. 91)

Mlynkova, I., Toman, K. & Pokorny, J. (2006). Statistical analysis of real XML data collections. *Proceedings of the 13th International Conference on Management of Data (COMAD'06)*. 20–31. (s. 91)

Nardi, B. A. (1993). *A small matter of programming: perspectives on end user computing*. MIT Press. (s. 20, 21, 22, 23)

O'Keefe, R. A. & Trotman, A. (2003). The simplest query language that could possibly work. *Proceedings of the 2nd INEX Workshop*. 167–174.
<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.59.2314> (s. 32, 85)

Orman, L. (1991). Complexity of database languages. *Information Systems* 16(2), 169–184.
<http://dx.doi.org/10.1016/0306-4379(91)90013-Y> (s. 61)

Panko, R. R. & Port, D. N. (2012). End user computing: the dark matter (and dark energy) of corporate IT. *Proceedings of the 45th Hawaii International Conference on System Science (HICSS)*. 4603–4612.
<http://dx.doi.org/10.1109/HICSS.2012.244> (s. 1)

Peffers, K., Tuunanen, T., Rothenberger, M. & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems* 24(3), 45–77.
<http://dx.doi.org/10.2753/MIS0742-1222240302> (s. 2, 3, 5, 6, 7)

Poikola, A., Kola, P. & Hintikka, K. A. (2010). *Julkinen data: Johdatus tietovarantojen avaamiseen*. Liikenne- ja viestintäministeriö, Helsinki.
<http://www.scribd.com/doc/28845102/Julkinen-data> (s. 1)

Query language. (2011). *Encyclopædia Britannica Online*.
<http://www.britannica.com/EBchecked/topic/487090/query-language> (s. 23)

Reisner, P. (1977). Use of psychological experimentation as an aid to development of a query language. *IEEE Transactions of Software Engineering* 3(3), 218–229.
<http://dx.doi.org/10.1109/TSE.1977.231131> (s. 26, 85, 86, 87)

Reisner, P. (1981). Human factors studies of database query languages: A survey and assessment. *ACM Computing Surveys* 13(1), 13–31.
<http://dx.doi.org/http://doi.acm.org/10.1145/356835.356837> (s. 47, 49)

Reisner, P. (1988). Query languages. Teoksessa Helander, M. (toim.) *Handbook of Human-Computer Interaction*, 1. painos. 257–280. (s. 23, 24, 33)

Reisner, P., Boyce, R. F. & Chamberlin, D. D. (1975). Human factors evaluation of two data base query languages: SQUARE and SEQUEL. *Proceedings of the AFIPS '75 National Computer Conference and Exposition*. 447–452.
<http://doi.acm.org/10.1145/1499949.1500036> (s. 45, 47, 49, 51, 52)

Risch, T. (2009). Query language. Teoksessa Liu, L. & Özsu, M. T. (toim.) *Encyclopedia of Database Systems*. 2260–2261.
<http://www.springerreference.com/docs/html/chapterdbid/64049.html> (s. 23, 24)

Rockart, J. F. & Flannery, L. S. (1983). *The management of end user computing: a research perspective*. 100, Massachusetts Institute of Technology.
<http://hdl.handle.net/1721.1/48508> (s. 14, 15, 19)

Rushkoff, D. (2010). *Program or be programmed: Ten commands for a digital age*. OR Books. (s. 2)

Rys, M. (2003). Full-text search with XQuery: A status report. Teoksessa Blanken, H., Grabs, T., Schek, H.-J., Schenkel, R. & Weikum, G. (toim.) *Intelligent Search on XML Data*. Lecture Notes in Computer Science, 39–57.
<http://dx.doi.org/10.1007/978-3-540-45194-5_3> (s. 12)

Sengupta, A. & Ramesh, V. (2009). Designing document SQL (DSQL): An accessible yet comprehensive ad-hoc querying frontend for XQuery. *Journal of Database Management* 20(4), 26–53.
<http://dx.doi.org/10.4018/jdm.2009062502> (s. 27)

Shneiderman, B. (1978). Improving the human factors aspect of database interactions. *ACM Transactions of Database Systems* 3(4), 417–439.
<http://dx.doi.org/10.1145/320289.320295> (s. 18, 20)

Smelcer, J. B. (1995). User errors in database query composition. *International Journal of Human-Computer Studies* 42(4), 353–381.
<http://dx.doi.org/10.1006/ijhc.1995.1017> (s. 48)

Soloway, E. (1982). Kill two birds with one stone: teach procedural query languages. *Proceedings of the ACM '82 conference*. ACM '82.
<http://doi.acm.org/10.1145/800174.809784> (s. 88, 90)

Soloway, E. & Ehrlich, K. (1984). Empirical studies of programming knowledge. *IEEE Transactions of Software Engineering* 10(5), 595–609.
<http://dx.doi.org/10.1109/TSE.1984.5010283> (s. 56)

Sperberg-McQueen, C. M. (2005). Xml and semi-structured data. *ACM Queue* 3(8), 34–41.
<http://doi.acm.org/10.1145/1103822.1103834> (s. 11, 12)

Srinivasan, A. & Irwin, G. (2006). Communicating the message: translating tasks into queries in a database context. *IEEE Transactions on Professional Communication* 49(2), 145–159.
<http://dx.doi.org/10.1109/TPC.2006.875075> (s. 48)

Tagarelli, A. & Greco, S. (2010). Semantic clustering of XML documents. *ACM Transactions of Information Systems* 28(1), 1–56.
<http://doi.acm.org/10.1145/1658377.1658380> (s. 91)

Topi, H., Valacich, J. S. & Hoffer, J. A. (2004). The effects of task complexity and time availability limitations on human performance in database query tasks. *International Journal of Human-Computer Studies* 62(3), 349–379.
<http://dx.doi.org/10.1016/j.ijhcs.2004.10.003> (s. 51, 52)

Tuomisto, S. (2011). *XIL-tiedohakujärjestelmän suunnittelu ja toteutus*. Pro gradu -tutkielma, Tampereen yliopisto. (s. 6)

Vainio, J. (2012). *XILtoSQL - hierarkkisten kyselyiden semantiikka relaatiotietokannassa*. Pro gradu -tutkielma, Tampereen yliopisto.
<http://urn.fi/urn:nbn:fi:uta-1-22430> (s. 6)

Verifysoft Technology GmbH (2010). Halstead metrics.
<http://www.verifysoft.com/en_halstead_metrics.html> (s. 61)

Weiand, K. (2010). *Keyword-based querying for the social semantic web*. Väitöskirja, Ludwig-Maximilians-Universität München.
<http://edoc.ub.uni-muenchen.de/12671/> (s. 43, 44)

Weinberg, G. M. (1971). *The psychology of computer programming*. Van Nostrand Reinhold. (s. 33)

Welty, C. & Stemple, D. W. (1981). Human factors comparison of a procedural and a nonprocedural query language. *ACM Transactions of Database Systems* 6(4), 626–649.
<http://doi.acm.org/10.1145/319628.319656> (s. 21, 45, 47, 49, 51, 52, 75, 88, 90)

Wu, C., Chan, H., Teo, H. & Wei, K. (1994). An experimental study of object-oriented query language and relational query language for novice users. *Journal of Database Management* 5(4), 16–27. (s. 33)

Yahoo (2012). YQL guide.
<http://developer.yahoo.com/yql/guide/yql_set.pdf> (s. 27)

Yen, M. Y.-M. & Scamell, R. W. (1993). A human factors experimental comparison of SQL and QBE. *IEEE Transactions on Software Engineering* 19(4), 390–409.
<http://dx.doi.org/10.1109/32.223806> (s. 49, 51, 52)

Ykhlef, M. (2007). Recursive SQL-like query language for XML. *The 9th International Conference on Information Integration and Web-based Applications Services (iiWAS'2007)*. 235–245.
<http://hdl.handle.net/123456789/15530> (s. 27)

1. Mikä on pääaineesi?

   _____

2. Minä vuonna olet aloittanut yliopisto-opinnot?

   _____

3. Oletko osallistunut *Tietokantojen perusteet* -kurssille tai osallistut kurssille tällä hetkellä?
   A. Kyllä
   B. Ei
4. Oletko osallistunut *Lausekielinen ohjelmointi* -kurssille tai osallistut kurssille tällä hetkellä?
   A. Kyllä
   B. Ei
5. Oletko osallistunut *Tietokantaohjelmointi* -kurssille tai osallistut kurssille tällä hetkellä?
   A. Kyllä
   B. Ei
6. Oletko osallistunut *Tietorakenteet* -kurssille tai osallistut kurssille tällä hetkellä?
   A. Kyllä
   B. Ei
7. Oletko osallistunut *XML-tiedonhaku ja -kyselykielet* -kurssille tai osallistut kurssille tällä hetkellä?
   A. Kyllä
   B. Ei
8. Ohjelmoitko työksesi tai harrastuksena opintojesi ulkopuolella?
   A. Kyllä
   B. Ei
9. Oletko työskennellyt XML-dokumenttien parissa opintojesi ulkopuolella?
   A. Kyllä
   B. Ei
10. Oletko ohjelmoinut SQL-kielellä opintojesi ulkopuolella?
    A. Kyllä
    B. Ei

Ohjeita

Listauksen 1 XML-dokumentissa esimerkiksi `henkilot`, `henkilo`, `etunimi`, `sukunimi` ja `biografia` ovat *elementtejä*, kun taas `kutsumanimi`, `numero` ja `nykyinen` ovat *attribuutteja*. Elementit voivat esiintyä sisäkkäin; esimerkkidokumentissamme `henkilo` -elementit sijaitsevat `henkilot` elementin sisällä ja `etunimi`-elementit puolestaan `henkilo` -elementin sisällä.

XML-dokumenttien osiin voi viitata *polkurakenteilla*. Polkurakenteissa elementit erotetaan toisistaan `/` -merkillä. Jos merkkejä on vain yksi, tällä tarkoitetaan että etsitty elementti on välittömästi toisen elementin sisällä. Esimerkiksi `/henkilot/henkilo` polkurakenne viittaa esimerkkidokumentin kahteen `henkilo` -elementtiin. Jos merkkejä on kaksi, tällä viitataan kaikkiin elementteihin toisen elementin sisällä. Esimerkiksi `/henkilot//etunimi` viittaa `etunimi`-elementteihin missä tahansa `henkilot` -elementin sisällä.

- `/henkilot/henkilo/biografia` Etsii kaikki biografiat
- `//etunimi[@kutsumanimi="true"]` tai `//etunimi/@kutsumanimi="true"` Etsii kaikki kutsumanimet
- `//henkilo/sukunimi[@nykyinen="true"]` tai `//henkilo/@nykyinen="true"` Etsii kaikki nykyiset sukunimet.
- `/henkilot/henkilo[@numero=2]` Etsii henkilön numero 2 kaikki tiedot

Listaus 1: Esimerkkidokumentti

```
<?xml version="1.0" encoding="UTF-8"?>
<henkilot>
    <henkilo numero="1">
        <etunimi kutsumanimi="true">Matti</etunimi>
        <etunimi>Teppo</etunimi>
        <sukunimi>Meikäläinen</sukunimi>
        <ammatti>opiskelija</ammatti>
        <biografia>Olen opiskellut yliopistolla neljä vuotta
        ja aion valmistua vuoden päästä.</biografia>
    </henkilo>
    <henkilo numero="2">
        <etunimi kutsumanimi="true">Maija</etunimi>
        <etunimi>Miia</etunimi>
        <sukunimi nykyinen="true">Meikäläinen</sukunimi>
        <sukunimi>Mallikas</sukunimi>
        <ammatti>sairaanhoitaja</ammatti>
        <biografia>Valmistuin ammattikorkeakoulusta vuosi sitten
        ja olen siitä eteenpäin työskennellyt Hatanpään terveysasemalla.
        </biografia>
    </henkilo>
</henkilot>
```

Tehtäväesimerkkejä

Kuvaile suomen kielellä, mitä kyselykielellä kirjoitettu lause käsityksesi mukaan tekee. Esimerkkejä:

1. `SELECT sukunimi FROM kirjoittaja WHERE etunimi = W.`

   Esimerkkivastaus: Etsi sukunimi kirjailijalle, jonka etunimi on W.

2. ```
   for $p in //johdanto/tekstikappale
       return $p
   ```

   Esimerkkivastaus: Etsi kaikki johdannon tekstikappaleet.
   ```

Testitehtävät

Kuvaile suomen kielellä, mitä kyselykielellä kirjoitettu lause käsityksesi mukaan tekee.

1. `SELECT tekstikappale`

   ○

   .....
   .....
   .....
   .....

2. `SELECT kappale/otsikko WHERE kappale/@numero = 1`

   ○

   .....
   .....
   .....
   .....

3. `SELECT otsikko FROM kirja/@vuosi < 1999`

   ●

   .....
   .....
   .....
   .....

4. `SELECT otsikko, avainsana FROM aihekokonaisuus ABOUT documentation`

   ○

   .....
   .....
   .....
   .....

5. `SELECT otsikko, hinta GROUP BY kirja FROM kirja`

   ●

   .....
   .....
   .....
   .....

6. **SELECT** otsikko
   WHERE kirja/@vuosi > 1991 **AND** verkkosivu = bstore1.example.com

   ●

   .....
   .....
   .....
   .....

7. **SELECT** otsikko, vinkki
   GROUP BY kirja
   FROM hinnat **WHERE** hinta = 69.95

   ●

   .....
   .....
   .....
   .....

8. **SELECT** otsikko **FROM** aihekokonaisuus
   WHERE avainsana = Markup **AND** tekstikappale **ABOUT** descriptive markup

   ○

   .....
   .....
   .....
   .....

9. **SELECT** otsikko **FROM** osio
   WHERE johdanto/tekstikappale **ABOUT** SGML **AND** aihekokonaisuus **ABOUT** DTD

   ○

   .....
   .....
   .....
   .....

10. **SELECT** aihekokonaisuus/otsikko, aihekokonaisuus/avainsana
    GROUP BY aihekokonaisuus **FROM** raportti
    WHERE aihekokonaisuus/@tunniste = top2 **OR** aihekokonaisuus/@tunniste = top3

    ○

    .....
    .....
    .....
    .....

Testitehtävät

Kuvaile suomen kielellä, mitä kyselykielellä kirjoitettu lause käsityksesi mukaan tekee.

1. ```
   for $p in raportti//tekstikappale
       return $p
   ```

   ○

   .....

   .....

   .....

   .....

2. ```
   for $a in //kappale[@numero = 1]
     return $a/otsikko
   ```

   ○

   .....

   .....

   .....

   .....

3. ```
   for $a in //kirja
     where $a/@vuosi < 1999
     return $a/otsikko
   ```

   ●

   .....

   .....

   .....

   .....

4. ```
   for $a in //aihekokonaisuus
     where contains($a, "documentation")
     return($a/otsikko, $a/avainsana)
   ```

   ○

   .....

   .....

   .....

   .....

5.
```
for $b in //kirja
    $t in $b/otsikko,
    $a in $b//hinta
    return
    <kirja>
        {$t}
        {$a}
    </kirja>
```

● .....

.....

.....

.....

.....

6.
```
for $t in //kirja
  where $t/@vuosi > 1991 and $t/kauppa/verkkosivu = "bstore1.example.com"
  return $t/otsikko
```

● .....

.....

.....

.....

.....

7.
```
for $a in //kirja
  where $a/kauppa/hinta = "65.95"
  return
  <result>
    {$a/otsikko}
    {$a/vinkki}
  </result>
```

● .....

.....

.....

.....

.....

8.
```
for $t in //aihekokonaisuus
    $p in $t//tekstikappale,
    $k in $t/avainsana,
  where contains($p, "descriptive markup")
    and contains($k, "Markup")
  return $t/otsikko
```

○ .....

.....

.....

.....

.....

9.
```
for $s in //osio,
      $p in $s/johdanto/tekstikappale,
      $t in $s/aihekokonaisuus
  where contains($p, "SGML")
    and contains($t, "DTD")
  return $s/otsikko
```

○

.....

.....

.....

.....

```
for $a in //aihekokonaisuus
  where $a/@tunniste = "top2" or $a/@tunniste = "top3"
  return { $a/otsikko, $a/avainsana }
```

○

.....

.....

.....

.....

Listaus 1: Kirjakauppa-aineisto (●)

```
<?xml version="1.0" encoding="utf-8"?>
<hinnat>
  <kirja vuosi="1992" tunniste="b1">
    <otsikko>Advanced Programming in the Unix environment</otsikko>
    <kauppa>
      <verkkosivusto>bstore2.example.com</verkkosivusto>
      <hinta>65.95</hinta>
    </kauppa>
    <kauppa>
      <verkkosivusto>bstore1.example.com</verkkosivusto>
      <hinta>65.95</hinta>
    </kauppa>
    <kauppa>
      <verkkosivusto>www.bol.com</verkkosivusto>
      <hinta>45.95</hinta>
    </kauppa>
    <vinkki>
      <viittaus kohde="b2" />
    </vinkki>
  </kirja>
  <kirja vuosi="1994" tunniste="b2">
    <otsikko>TCP/IP Illustrated</otsikko>
    <kauppa>
      <verkkosivusto>bstore2.example.com</verkkosivusto>
      <hinta>65.95</hinta>
    </kauppa>
    <kauppa>
      <verkkosivusto>bstore1.example.com</verkkosivusto>
      <hinta>65.95</hinta>
    </kauppa>
    <vinkki>
      <viittaus kohde="b1" />
    </vinkki>
  </kirja>
  <kirja vuosi="2000" tunniste="b3">
    <otsikko>Data on the Web</otsikko>
    <kauppa>
      <verkkosivusto>bstore2.example.com</verkkosivusto>
      <hinta>34.95</hinta>
    </kauppa>
    <kauppa>
      <verkkosivusto>bstore1.example.com</verkkosivusto>
      <hinta>39.95</hinta>
    </kauppa>
  </kirja>
  <kirja vuosi="1999" tunniste="b4">
    <otsikko>The Economics of Technology and Content for Digital
    TV</otsikko>
    <kauppa>
      <verkkosivusto>www.amazon.com</verkkosivusto>
      <hinta>34.95</hinta>
    </kauppa>
    <kauppa>
      <verkkosivusto>www.bol.com</verkkosivusto>
      <hinta>69.95</hinta>
    </kauppa>
  </kirja>
</hinnat>
```

Listaus 2: SGML-aineisto (○)

```
<?xml version="1.0" encoding="utf-8"?>
<raportti>
  <otsikko>Getting started with SGML</otsikko>
  <kappale numero="1">
    <otsikko>Getting to know SGML</otsikko>
    <kirjoittaja>
      <sukunimi>Stevens</sukunimi>
      <etunimi>W.</etunimi>
    </kirjoittaja>
    <johdanto>
      <tekstikappale>While SGML is a fairly recent technology, the use of
      markup in computer-generated documents has
      existed for a while.</tekstikappale>
    </johdanto>
    <osio lyhytotsikko="What is markup?">
      <otsikko>What is markup, or everything you always wanted to
      know about document preparation but were afraid to
      ask?</otsikko>
      <johdanto>
        <tekstikappale>Markup is everything in a document that is not
        content. The traditional meaning of markup is the manual
        marking up of typewritten text to give
        instructions for a typesetter or compositor about how to
        fit the text on a page and what typefaces to use. This kind
        of markup is known as
        procedural markup.</tekstikappale>
      </johdanto>
      <aihekokonaisuus tunniste="top1" numero="1">
        <otsikko>Procedural markup</otsikko>
        <avainsana>Markup</avainsana>
        <tekstikappale turvallisuus="u">Most electronic publishing systems today
        use some form of procedural markup. Procedural markup codes
        are good for one presentation of the information.</tekstikappale>
      </aihekokonaisuus>
      <aihekokonaisuus tunniste="top2" numero="2">
        <otsikko>Generic markup</otsikko>
        <avainsana>Markup</avainsana>
        <tekstikappale>Generic markup (also known as descriptive markup)
        describes the purpose of the text in a document. A basic
        concept of generic markup is that the content of a document
        must be separate from the style. Generic markup allows for
        multiple presentations of the information.</tekstikappale>
      </aihekokonaisuus>
      <aihekokonaisuus tunniste="top3" numero="3">
        <otsikko>Drawbacks of procedural markup</otsikko>
        <avainsana>Markup</avainsana>
        <tekstikappale turvallisuus="u">Industries involved in technical
        documentation increasingly prefer generic over
        procedural markup schemes. When a company
        changes software or hardware systems, enormous data
        translation tasks arise, often resulting in errors.</tekstikappale>
      </aihekokonaisuus>
    </osio>
    <osio lyhytotsikko="What is SGML?">
      <otsikko>What is SGML in the grand scheme of the universe, anyway?</otsikko>
      <johdanto>
        <tekstikappale>SGML defines a strict markup scheme with a syntax for
        defining document data elements and an overall framework
        for marking up documents.</tekstikappale>
        <tekstikappale>SGML can describe and create documents that are not
        dependent on any hardware, software, formatter, or
        operating system. Since SGML documents conform to an
        international standard, they are portable.</tekstikappale>
      </johdanto>
    </osio>
  </kappale>
</raportti>
```

Nimi: _____

Vastauksiasi voidaan käyttää kyselykielitutkimuksen aineistona. Aineistoa käsitellään nimettömänä. Tällä lomakkeella pyydettyä nimeäsi käytetään vain yhdistämään verkkotehtävien yhteydessä kerätyt tiedot koetehtävävastauksiin. Tämän jälkeen aineistosta poistetaan kaikki henkilötiedot.

Mikäli et halua, että vastauksiasi käytetään tutkimusaineistona, voit kieltää sen allekirjoittamalla kieltolomakkeen vastauksia palauttaessasi.

Vastaa seuraaviin tehtäviin XIL-kielellä

1. Laadi kysely, joka hakee kaikki kirjan kappaleet. ○

```
SELECT kappale
```

2. Laadi kysely, joka hakee lukuotsikot. ○

```
SELECT luku/otsikko
```

3. Laadi kysely, joka hakee kaikki kappaleet, joiden tyyli on *puheenvuoro*. ○

```
SELECT kappale/@tyyli=puheenvuoro
```

4. Laadi kysely, joka hakee kaikki salaisiksi luokitellut (secret) sähkeet. ●

```
SELECT cable/@classification=SECRET
```

5. Laadi kysely, joka hakee osion otsikon kun osiossa mainitaan missä tahansa sana *laulu*. ○

```
SELECT otsikko FROM osio ABOUT laulu
```

6. Laadi kysely, joka hakee sähkeen otsikon jossa mainitaan sana *war*. ●

```
SELECT subject ABOUT war FROM cable
```

7. Laadi kysely, joka hakee otsikot vuonna 2009 lähetetyistä tai julkaistuista sähkeistä. ●

```
SELECT subject FROM cable WHERE year=2009
```

8. Laadi kysely, joka hakee otsikon ja avainsanat tilanteille, jotka sisältävät ilmauksen *eukko*. ○

```
SELECT otsikko, avainsana FROM tilanne ABOUT eukko
```

9. Laadi kysely, joka hakee luottamuksellisten (confidential) sähkeiden vastaanottajien nimet. ●

```
SELECT to/name FROM cable/@classification="CONFIDENTIAL"
```

10. Laadi kysely, joka hakee tilanteiden otsikot ja avainsanat. ○

```
SELECT tilanne/otsikko, tilanne/avainsana FROM kirja
```

11. Laadi kysely, joka hakee osion johdannon kun osion lyhytotsikko on *Kouluunlähtö*. ○

```
SELECT osio/johdanto FROM luku WHERE osio/@lyhytotsikko ABOUT Kouluunlähtö
```

12. Laadi kysely, joka hakee aiemmin kuin 2009 lähetettyjen sähkeiden jakelutiedot. Ryhmitä tulokset sähkeen mukaan. ●

```
SELECT header FROM cable GROUP BY cable WHERE sent/year<2009
```

13. Laadi kysely, joka etsii otsikon osiosta, jonka johdannossa mainitaan *aamu* ja jonka tilanteessa mainitaan missä tahansa *kosioretki*. ○

```
SELECT otsikko FROM osio WHERE johdanto ABOUT aamu AND tilanne ABOUT kosioretki
```

14. Laadi kysely, joka hakee otsikot sähkeistä jotka on lähetetty ennen vuotta 2009 ja joiden vastaanottajan nimessä mainitaan *WASHDC*. ●

```
SELECT subject FROM cable
WHERE sent/year=2009 AND to/name ABOUT WASHDC
```

15. Laadi kysely, joka hakee otsikon ja avainsanan tilanteille, joiden tunniste on *tilanne1* tai *tilanne2*. ○

```
SELECT tilanne/otsikko, tilanne/avainsana GROUP BY tilanne FROM kirja WHERE
    tilanne/@tunniste=tilanne1 OR tilanne/@tunniste=tilanne2
```

Vastaa seuraaviin tehtäviin XPath/XQuery-kielellä

1. Laadi kysely, joka hakee kaikki kirjan kappaleet. ○

```
kirja//kappale

for $p in kirja//kappale return $p
```

2. Laadi kysely, joka hakee lukuotsikot. ○

```
/kirja/luku/otsikko
//luku/otsikko

for $t in //luku/otsikko return $t
```

3. Laadi kysely, joka hakee kaikki kappaleet, joiden tyyli on *puheenvuoro*. ○

```
//kappale[@tyyli="puheenvuoro"]

for $p in //kappale where $p/@tyyli="puheenvuoro" return $p
```

4. Laadi kysely, joka hakee kaikki salaisiksi luokitellut (secret) sähkeet. ●

```
//cable[@classification = "SECRET"]

for $b in //cable where $b/@classification = "SECRET" return $b
```

5. Laadi kysely, joka hakee osion otsikon kun osiossa mainitaan missä tahansa sana *laulu*. ○

```
//osio[contains(.,"laulu")]/otsikko

for $a in //osio where contains ($a, "laulu") return $a/otsikko

for $a in //osio[contains (., "laulu")] return $a/otsikko
```

6. Laadi kysely, joka hakee sähkeen otsikon jossa mainitaan sana *war*. ●

```
//sahke/otsikko[contains(.,"war")]
//cable/subject[contains(., "war")]
```

7. Laadi kysely, joka hakee otsikot vuonna 2009 lähetetyistä tai julkaistuista sähkeistä. ●

```
/sahkeet/sahke[julkaistu/vuosi="2009" or lahetetty/vuosi="2009"]/otsikko
```

8. Laadi kysely, joka hakee otsikon ja avainsanat tilanteille, jotka sisältävät ilmauksen *eukko*. ○

```
//tilanne[contains(.,"eukko")]/(otsikko|avainsana)

for $a in //tilanne where contains ($a, "eukko") return {$a/otsikko} {$a/avainsana
    }
```

9. Laadi kysely, joka hakee luottamuksellisten (confidential) sähkeiden vastaanottajien nimet. ●

```
//sahke[@luokitus="CONFIDENTIAL"]/jakelu/vastaanottaja/

for $a in //cable where $a/@classification="CONFIDENTIAL" return $a//to/name

//cable[@classification="CONFIDENTIAL"]//to/name
```

10. Laadi kysely, joka hakee tilanteiden otsikot ja avainsanat. ○

```
//tilanne/(otsikko|avainsana)

let $a:=//tilanne return ($a/otsikko, $a/avainsana)

for $a in //tilanne return ($a/otsikko, $a/avainsana)
```

11. Laadi kysely, joka hakee osion johdannon kun osion lyhytotsikko on *Kouluunlähtö*. ○

```
//osio[@lyhytotsikko="Kouluunlähtö"]/johdanto
```

12. Laadi kysely, joka hakee aiemmin kuin 2009 lähetettyjen sähkeiden jakelutiedot. Ryhmitä tulokset sähkeen mukaan. ●

```
  for $sahke in //sahke where $sahke/lahetetty/vuosi<2009 return <sahke>{$sahke/
      jakelu}</sahke>

for $x in //sahke[lahetetty/vuosi/text()<2009] return $x/jakelu
```

13. Laadi kysely, joka etsii otsikon osiosta, jonka johdannossa mainitaan *aamu* ja jonka tilanteessa mainitaan missä tahansa *kosioretki*. ○

```
for $t in //osio
where contains($t/johdanto,'aamu') and contains($t/tilanne,'kosioretki')
return $t/otsikko
```

14. Laadi kysely, joka hakee otsikot sähkeistä jotka on lähetetty ennen vuotta 2009 ja joiden vastaanottajan nimessä mainitaan *WASHDC*. ●

```
//sahke[lahetetty/vuosi < 2009 and contains(./jakelu/vastaanottaja/nimi, "WASHDC")
    ]/otsikko

for $c in //cable where $c/sent/year<2009 and contains(.//to,"WASHDC") return $c/
    subject
```

15. Laadi kysely, joka hakee otsikon ja avainsanan tilanteille, joiden tunniste on *tilanne1* tai *tilanne2*. ○

```
//tilanne[@tunniste="tilanne1" or @tunniste="tilanne2"]/(otsikko|avainsana)

let $a := //tilanne where $a[@tunniste="tilanne1"] or $a[@tunniste="tilanne2"]
    return ($a/otsikko, $a/avainsana)

for $a in //tilanne where $a/@tunniste ="tilanne1" or $a/@tunniste ="tilanne2"
    return { $a/otsikko, $a/avainsana }
```

Listaus 1: Sähkeaineisto (●)

```
<sahkeet>
    <sahke luokitus="CONFIDENTIAL" numero="561">
        <tunnistekoodi>09TALLINN317</tunnistekoodi>
        <toimipiste>Embassy Tallinn</toimipiste>
        <lahetetty>
            <vuosi>2009</vuosi>
            <kuukausi>10</kuukausi>
            <paiva>30</paiva>
        </lahetetty>
        <julkaistu>
            <vuosi>2010</vuosi>
            <kuukausi>12</kuukausi>
            <paiva>17</paiva>
        </julkaistu>
        <otsikko>Estonian FM Visit to Belarus; Lukashenko Goes On (and On and</otsikko>
        <jakelu>
            <tarkastussumma>VZCZCXRO4717</tarkastussumma>
            <lahettaja>
                <nimi>AMEMBASSY TALLINN</nimi>
                <koodi>9103</koodi>
            </lahettaja>
            <vastaanottaja>
                <nimi>RUEHC/SECSTATE WASHDC</nimi>
                <koodi>9880</koodi>
            </vastaanottaja>
        </jakelu>
    </sahke>
    <sahke luokitus="SECRET" numero="562">
        <tunnistekoodi>07TALLINN366</tunnistekoodi>
        <toimipiste>Embassy Tallinn</toimipiste>
        <lahetetty>
            <vuosi>2007</vuosi>
            <kuukausi>6</kuukausi>
            <paiva>4</paiva>
        </lahetetty>
        <julkaistu>
            <vuosi>2010</vuosi>
            <kuukausi>12</kuukausi>
            <paiva>6</paiva>
        </julkaistu>
        <otsikko>ESTONIA'S CYBER ATTACKS: WORLD'S FIRST VIRTUAL WAR</otsikko>
        <jakelu>
            <tarkastussumma>VZCZCXRO4489</tarkastussumma>
            <lahettaja>
                <nimi>AMEMBASSY TALLINN</nimi>
                <koodi>9103</koodi>
            </lahettaja>
            <vastaanottaja>
                <nimi>RUEHC/SECSTATE WASHDC</nimi>
                <koodi>9880</koodi>
            </vastaanottaja>
        </jakelu>
    </sahke>
    <sahke luokitus="SECRET" numero="563">
        <tunnistekoodi>09TALLINN114</tunnistekoodi>
        <toimipiste>Embassy Tallinn</toimipiste>
        <lahetetty>
            <vuosi>2009</vuosi>
            <kuukausi>4</kuukausi>
            <paiva>27</paiva>
        </lahetetty>
        <julkaistu>
            <vuosi>2010</vuosi>
            <kuukausi>12</kuukausi>
            <paiva>6</paiva>
        </julkaistu>
        <otsikko>ESTONIA'S PESSIMISTIC APPROACH TO RUSSIA</otsikko>
    </sahke>
    <sahke luokitus="SECRET" numero="564">
        <tunnistekoodi>09TALLINN373</tunnistekoodi>
        <toimipiste>Embassy Tallinn</toimipiste>
        <lahetetty>
            <vuosi>2009</vuosi>
            <kuukausi>12</kuukausi>
            <paiva>16</paiva>
        </lahetetty>
        <julkaistu>
            <vuosi>2009</vuosi>
            <kuukausi>12</kuukausi>
            <paiva>31</paiva>
        </julkaistu>
        <otsikko>Estonia Pleased with NATO Contingency Planning Plan</otsikko>
        <jakelu>
            <tarkastussumma>VZCZCXYZ0000</tarkastussumma>
            <lahettaja>
                <nimi>AMEMBASSY TALLINN</nimi>
                <koodi>9103</koodi>
            </lahettaja>
            <vastaanottaja>
                <nimi>RUEHC/SECSTATE WASHDC</nimi>
                <koodi>0289</koodi>
            </vastaanottaja>
            <kopio>
                <nimi>RUEHNO/USMISSION USNATO</nimi>
                <koodi>0010</koodi>
            </kopio>
        </jakelu>
    </sahke>
</sahkeet>
```

Listaus 2: Kirja-aineisto (○)

```
<?xml version="1.0" encoding="utf-8"?>
    <kirja>
      <otsikko>Seitsemän veljestä</otsikko>
      <julkaisuvuosi>1890</julkaisuvuosi>
      <luku numero="1">
        <kirjailija>
          <sukunimi>Kivi</sukunimi>
          <etunimi>Aleksis</etunimi>
        </kirjailija>
        <otsikko>Ensimmäinen luku</otsikko>
        <johdanto>
          <kappale>Jukolan talo, eteläisessä Hämeessä, seisoo erään mäen pohjaisella rinteellä, liki Toukolan kylää. Sen
          läheisin ympäristö on kivinen tanner, mutta alempana alkaa pellot, joissa, ennenkuin talo oli häviöön mennyt,
          aaltoili teräinen vilja.--Ja tämä on niiden seitsemän veljen koto, joiden elämänvaiheita tässä nyt käyn
          kertoilemaan.</kappale>
        </johdanto>
      </luku>
      <luku numero="2">
        <otsikko>Toinen luku</otsikko>
        <johdanto>
          <kappale>On tyyni syyskuun aamu. Kaste kiiltää kedolla, sumu kiiriskelee kellastuneiden lehdistöjen tutkaimilla
          ja haihtuu lopulta korkeuteen. Tänä aamuna ovat veljet nousneet ylös kovin äkeinä ja äänettöminä, pesneet
          kasvonsa, harjanneet tukkansa ja pukeutuneet pyhävaatteisinsa. Sillä tänä päivänä olivat he päättäneet lähteä
          lukkarin luoksi kouluun.</kappale>
        </johdanto>
        <osio lyhytotsikko="Kouluunlähtö">
          <otsikko>Veljekset harkitsevat kouluunlähtöä, mutta tulevat toisiin aatoksiin</otsikko>
          <johdanto>
            <kappale>Syövät he nyt aamuistansa Jukolan pitkän, honkaisen pöydän ääressä, ja näkyy heille maittavan ruskeat
            herneet, ehkei ollut heidän muotonsa iloinen, vaan kiusan karmeus väikkyi heidän kulmakarvoillansa; aatos
            kouluretkestä, johon heidän kohta tulee lähteä, on matkaan-saattanut tämän. Mutta atrioittuansa, eivät he
            kuitenkaan rientäneet heti matkaan, vaan istuivat vielä hetkeksi levähtämään. Pirtin eteläisen akkunan ääressä
            istuu Juhani, katsahdellen ylös kiviseen mäkeen ja tuuheaan männistöön, josta haamoitti muorin tönö
            punapielisellä ovellansa.</kappale>
          </johdanto>
          <tilanne tunniste="tilanne1" numero="1">
            <otsikko>Veljekset ja Venla</otsikko>
            <avainsana>kosioretki</avainsana>
            <kappale tyyli="puheenvuoro">JUHANI. Venla tuolla astelee pitkin polkua, ja onpa hänen käymisensä
            nopsa.</kappale>
            <kappale tyyli="puheenvuoro">AAPO. Ja eilen piti niin äitin kuin tyttärenkin lähtemän sukulaistensa luoksi
            Tikkalaan nauriita liistimään ja puolaimia poimiskelemaan, viipyäkseen siellä aina myöhään syksyyn.</kappale>
            <kappale tyyli="puheenvuoro">JUHANI. Aina myöhään syksyyn? Minä tulen kovin levottomaksi. Kaiketi lähtevät he;
            mutta Tikkalassa on tänä vuonna renki, joka on pulski poika ja suuri vekkuli, ja sinne menisi pian meidän
            kaikkein toivo. kappalestahan siis tehdä tuossa tiimassa se merkillinen temppu, tehdä kysymys, kaikkein
            kysymysten kysymys. Mennään siis likalta kysymään, tahtoisiko mielensä taipua ja sydämmensä syttyä.</kappale>
          </tilanne>
          <tilanne tunniste="tilanne2" numero="2">
            <otsikko>Venlan luona</otsikko>
            <avainsana>Venla</avainsana>
            <avainsana>äiti</avainsana>
            <avainsana>kosioretki</avainsana>
            <kappale tyyli="puheenvuoro">JUHANI. Ken aukaisi ovea? Venlako?</kappale>
            <kappale>Astuivat he muorin matalaan mökkiin, Juhani edellä, silmät pöllöllään ja tukka pystyssä kuin
            piikkisian harjakset, ja toiset seurasivat häntä uskollisesti, vakaasti kantapäissä. Niin astuivat he sisään,
            ja Eero kimmautti oven heidän jälkeensä kiinni, mutta itse jäi hän ulkopuolelle, istui alas kedolle, huulilla
            sukkela myhäilys.</kappale>
            <kappale>Mutta eukko, jonka huoneessa viisi veljestä nyt seisoo kosiomiehinä, on reipas ja vireä eukko; hän
            käyttelee elinkeinoksensa kananhoitoa ja marjannoukkimista. Suvin ja syksyin keikkuu hän ahkerasti kantoisilla
            ahoilla, mansikka- ja puolain-töyräillä, tyttärensä Venlan kanssa. Kauniiksi kutsuttiin neitoa. Hänen hiuksensa
            olivat ruosteenkarvaiset, katsanto viekas ja terävä, suu myös sulava, ehkä melkein liian leveä.Tämän kaltainen
            oli veljesten lempilintu männistön suojassa.</kappale>
          </tilanne>
        </osio>
      </luku>
      <luku numero="3">
        <otsikko>Kolmas luku</otsikko>
        <osio lyhytotsikko="Seitsemän miehen voima">
          <otsikko>Toukolan veljesten laulu Jukolan veljeksille</otsikko>
          <johdanto>
            <kappale>Veljesten näin haastellessa, läheni heitä joukko Toukolan poikia, mutta eipä juuri niin kohteliaasti
            ja hyväntahtoisesti kuin Jukolaiset vartoivat. Olivatpa jotenkin päissään, ja miellytti heitä nyt hieman
            ilvehtiä veljesten kanssa ja lauloivat heidän edessään nykyään sepitetyn laulun, jonka olivat nimittäneet:
            Seitsemän miehen voima. Niinpä he, Kissalan Aapelin puhaltaissa, likenivät koulumiehiä, laulain seuraavalla
            tavalla:</kappale>
          </johdanto>
          <kappale tyyli="laulu">Kiljukoon nyt kaikkein kaula, Koska mielin virren laulaa Voimasta seitsemän miehen. Tähtiä
          kuin otavassa, Poikia on Jukolassa, Laiskanpulskeja jallii. Juho pauhaa, pirtti roikkaa; Hän on talon aika poika,
          Ankara Poika-Jussi. Tuomas seisoo niinkuin tammi, Koska saarnaa Aaprahammi, Jukolan Salomon suuri. Simeoni,
          liuhuparta, Valittaa se ihmisparka, Syntinen, saatana, kurja. Simeoni herneet keittää, Timo sekaan rasvat
          heittää, Patahan kuohuvaan sylkee. Lauri-poika metsäss häärii, Katselevi puita väärii, Mäyränä nummia tonkii.
          Viimein tulee hännän huippu, Pikku-Eero, liukas luikku, Jukolan tiuskea rakki. Siinä onpi velisarja, Jalo
          niinkuin sonnikarja, Voimalla seitsemän miehen.</kappale>
        </osio>
      </luku>
    </kirja>
```

1 XIL-esimerkkiratkaisujen Halstead-vaikeus

Taulukko 1: 1. Laadi kysely, joka hakee kaikki kirjan kappaleet.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	kappale	1
N_1	1	N_2	1
η_1	1	η_2	1

Taulukko 2: 2. Laadi kysely, joka hakee lukuotsikot.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	luku	1
/	1	otsikko	1
N_1	2	N_2	2
η_1	2	η_2	2

Taulukko 3: 3. Laadi kysely, joka hakee kaikki kappaleet, joiden tyyli on puheenvuoro.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	kappale	1
/	1	@tyyli	1
=	1	puheenvuoro	1
N_1	3	N_2	3
η_1	3	η_2	3

Taulukko 4: 4. Laadi kysely, joka hakee kaikki salaisiksi luokitellut (secret) sähkeet.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	sahke	1
n	1	classification	1
/	1	SECRET	1
N_1	1	N_2	1
η_1	1	η_2	1

Taulukko 5: 5. Laadi kysely, joka hakee osion otsikon kun osiossa mainitaan missä tahansa sana laulu

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	otsikko	1
FROM	1	osio	1
ABOUT	1	laulu	1
N_1	3	N_2	3
η_1	3	η_2	3

Taulukko 6: 6. Laadi kysely, joka hakee sähkeen otsikon jossa mainitaan sana war.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	otsikko	1
ABOUT	1	war	1
FROM	1	sahke	1
N_1	3	N_2	3
η_1	3	η_2	3

Taulukko 7: 7. Laadi kysely, joka hakee otsikot vuonna 2009 lähetetyistä tai julkaistuista sähkeistä.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	otsikko	1
FROM	1	sahke	1
WHERE	1	year	1
=	1	2009	1
N_1	3	N_2	3
η_1	3	η_2	3

Taulukko 8: 8. Laadi kysely, joka hakee otsikon ja avainsanat tilanteille, jotka sisältävät ilmauksen eukko.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	otsikko	1
0	1	avainsana	1
FROM	1	tilanne	1
ABOUT	1	eukko	1
N_1	4	N_2	4
η_1	4	η_2	4

Taulukko 9: 9. Laadi kysely, joka hakee luottamuksellisten (confidential) sähkeiden vastaanottajien nimet.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	to	1
/	2	name	1
FROM	1	sahke	1
=	1	@luokitus	1
		CONFIDENTIAL	1
N_1	5	N_2	5
η_1	4	η_2	5

Taulukko 10: 10. Laadi kysely, joka hakee tilanteiden otsikot ja avainsanat.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	tilanne	2
/	2	otsikko	1
	1	avainsana	1
FROM	1	kirja	1
N_1	5	N_2	5
η_1	4	η_2	4

Taulukko 11: 11. Laadi kysely, joka hakee osion johdannon kun osion lyhytotsikko on Kouluunlähtö.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	osio	2
/	2	johdanto	1
FROM	1	luku	1
WHERE	1	@lyhytosikko	1
ABOUT	1	Kouluunlähtö	1
N_1	6	N_2	6
η_1	5	η_2	5

Taulukko 12: 12. Laadi kysely, joka hakee aiemmin kuin 2009 lähetettyjen sähkeiden jakelutiedot. Ryhmitä tulokset sähkeen mukaan.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	header	1
FROM	1	sahke	2
GROUP BY	1	sent	1
WHERE	1	year	1
/	1	2009	1
<	1		
N_1	6	N_2	6
η_1	6	η_2	5

Taulukko 13: 13. Laadi kysely, joka etsii otsikon osiosta, jonka johdannossa mainitaan aamu ja jonka tilanteessa mainitaan missä tahansa kosioretki.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	otsikko	1
FROM	1	osio	1
WHERE	1	johdanto	1
ABOUT	2	aamu	1
AND	1	tilanne	1
		kosioretki	1
N_1	6	N_2	6
η_1	5	η_2	6

Taulukko 14: 14. Laadi kysely, joka hakee otsikot sähkeistä jotka on lähetetty ennen vuotta 2009 ja joiden vastaanottajan nimessä mainitaan WASHDC.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	otsikko	1
FROM	1	sahke	1
WHERE	1	sent	1
/	2	year	1
=	1	2009	1
AND	1	to	1
ABOUT	1	name	1
		WASHDC	1
N_1	8	N_2	8
η_1	7	η_2	8

Taulukko 15: 15. Laadi kysely, joka hakee otsikon ja avainsanan tilanteille, joiden tunniste on tilanne1 tai tilanne2.

Operaattori	$\sum$	Operandi	$\sum$
SELECT	1	tilanne	5
/	4	otsikko	1
	1	avainsana	1
GROUP BY	1	kirja	1
FROM	1	@tunniste	2
WHERE	1	tilanne1	1
=	2	tilanne2	1
OR	1		
N_1	12	N_2	12
η_1	8	η_2	7

2 XQuery-esimerkkiratkaisujen Halstead-vaikeus

Taulukko 16: 1. Laadi kysely, joka hakee kaikki kirjan kappaleet.

Operaattori	$\sum$	Operandi	$\sum$
//	1	kappale	1
N_1	1	N_2	1
η_1	1	η_2	1

Taulukko 17: 2. Laadi kysely, joka hakee lukuotsikot.

Operaattori	$\sum$	Operandi	$\sum$
//	1	luku	1
/	1	otsikko	1
N_1	2	N_2	2
η_1	2	η_2	2

Taulukko 18: 3. Laadi kysely, joka hakee kaikki kappaleet, joiden tyyli on puheenvuoro.

Operaattori	$\sum$	Operandi	$\sum$
//	1	kappale	1
[]	1	@tyyli	1
=	1	puheenvuoro	1
”	1		
N_1	4	N_2	3
η_1	4	η_2	3

Taulukko 19: 4. Laadi kysely, joka hakee kaikki salaisiksi luokitellut (secret) sähkeet.

Operaattori	$\sum$	Operandi	$\sum$
//	1	sahke	1
[]	1	@luokitus	1
”	1	SECRET	1
=	1		
N_1	4	N_2	3
η_1	4	η_2	3

Taulukko 20: 5. Laadi kysely, joka hakee osion otsikon kun osiossa mainitaan missä tahansa sana laulu.

Operaattori	$\sum$	Operandi	$\sum$
//	1	osio	1
[]	1	laulu	1
()	1	otsikko	1
.	1		
0	1		
contains	1		
/	1		
”	1		
N_1	8	N_2	3
η_1	8	η_2	3

Taulukko 21: 6. Laadi kysely, joka hakee sähkeen otsikon jossa mainitaan sana war.

Operaattori	$\sum$	Operandi	$\sum$
//	1	sahke	1
/	1	otsikko	1
[]	1	war	1
()	1		
.	1		
”	1		
N_1	6	N_2	3
η_1	6	η_2	3

Taulukko 22: 7. Laadi kysely, joka hakee otsikot vuonna 2009 lähetetyistä tai julkaistuista sähkeistä.

Operaattori	$\sum$	Operandi	$\sum$
/	5	sahkeet	1
[]	1	sahke	1
=	2	julkaistu	1
”	2	vuosi	2
or	1	lahetetty	1
		otsikko	1
		2009	2
N_1	11	N_2	9
η_1	5	η_2	7

Taulukko 23: 8. Laadi kysely, joka hakee otsikon ja avainsanat tilanteille, jotka sisältävät ilmauksen eukko.

Operaattori	$\sum$	Operandi	$\sum$
//	1	tilanne	1
/	1	eukko	1
[]	1	otsikko	1
contains	1	avainsana	1
()	2		
.	1		
0	1		
”	1		
|	1		
N_1	10	N_2	4
η_1	9	η_2	4

Taulukko 24: 9. Laadi kysely, joka hakee luottamuksellisten (confidential) sähkeiden vastaanottajien nimet.

Operaattori	$\sum$	Operandi	$\sum$
//	1	sahke	1
/	3	@luokitus	1
[]	1	CONFIDENTIAL	1
”	1	jakelu	1
=	1	vastaanottaja	1
N_1	7	N_2	5
η_1	5	η_2	5

Taulukko 25: 10. Laadi kysely, joka hakee tilanteiden otsikot ja avainsanat.

Operaattori	$\sum$	Operandi	$\sum$
//	1	tilanne	1
/	1	otsikko	1
|	1	avainsana	1
()	1		
N_1	4	N_2	3
η_1	4	η_2	3

Taulukko 26: 11. Laadi kysely, joka hakee osion johdannon kun osion lyhytotsikko on Kouluunlähtö.

Operaattori	$\sum$	Operandi	$\sum$
//	1	osio	1
/	1	@lyhytotsikko	1
[]	1	Kouluunlähtö	1
=	1	johdanto	1
”	1		
N_1	5	N_2	4
η_1	5	η_2	4

Taulukko 27: 12. Laadi kysely, joka hakee aiemmin kuin 2009 lähetettyjen sähkeiden jakelutiedot. Ryhmitä tulokset sähkeen mukaan.

Operaattori	$\sum$	Operandi	$\sum$
for	1	$sahke	3
in	1	sahke	1
//	1	lahetetty	1
where	1	vuosi	1
/	3	2009	1
<	1	<sahke>	1
return	1	jakelu	1
	1	</sahke>	1
N_1	10	N_2	10
η_1	8	η_2	8

Taulukko 28: 13. Laadi kysely, joka etsii otsikon osiosta, jonka johdannossa mainitaan aamu ja jonka tilanteessa mainitaan missä tahansa kosioretki.

Operaattori	$\sum$	Operandi	$\sum$
for	1	$t	4
in	1	osio	1
//	1	johdanto	1
where	1	aamu	1
contains	2	tilanne	1
()	2	kosioretki	1
/	3	otsikko	1
”	2		
and	1		
return	1		
N_1	15	N_2	10
η_1	10	η_2	7

Taulukko 29: 14. Laadi kysely, joka hakee otsikot sähkeistä jotka on lähetetty ennen vuotta 2009 ja joiden vastaanottajan nimessä mainitaan WASHDC.

Operaattori	$\sum$	Operandi	$\sum$
//	1	sahke	1
[]	1	lahetetty	1
/	5	vuosi	1
<	1	2009	1
and	1	jakelu	1
contains	1	vastaanottaja	1
()	1	nimi	1
.	1	WASHDC	1
0	1	otsikko	1
”	1		
N_1	14	N_2	9
η_1	10	η_2	9

Taulukko 30: 15. Laadi kysely, joka hakee otsikon ja avainsanan tilanteille, joiden tunniste on tilanne1 tai tilanne2.

Operaattori	$\sum$	Operandi	$\sum$
//	1	tilanne	1
[]	1	@tunniste	2
”	2	tilanne1	1
=	2	tilanne2	1
or	1	otsikko	1
/	1	avainsana	1
()	1		
|	1		
N_1	10	N_2	7
η_1	8	η_2	6

Taulukko 1: Kurssiarvosanan ja kyselyvastausluokkien korrelaatio (XIL)

	Pearson		Kendall	
	r	p	τ_b	p
Oikea vastaus	0,51	0,002**	0,41	0,003**
Ei vastausta	-0,23	0,17	-0,28	0,06†

Merkitsevyystasot: † p<.10; * p<.05; ** p<.01; *** p<.001.

Taulukko 2: Kurssiarvosanan ja kyselyvastausluokkien korrelaatio (XQuery)

	Pearson		Kendall	
	r	p	tau	p
Oikea vastaus	0,36	0,03*	0,25	0,03*
Ei vastausta	-0,31	0,06†	-0,29	0,05*

Merkitsevyystasot: † p<.10; * p<.05; ** p<.01; *** p<.001.

Taulukko 3: Kokemus alle kokemusindeksin keskiarvon (0,682)

Luokka	%		n	
	XQuery	XIL	XQuery	XIL
Oikea vastaus	27,4	20	74	54
Pieni syntaksivirhe	0,74	0	2	0
Pieni operandivirhe	0	0	0	0
Pieni sisältövirhe	0	0	0	0
Korjauskelpoinen	13,7	23,33	37	63
Sisältövirhe	7,03	20,74	19	56
Syntaksivirhe	41,8	26,67	113	72
Puutteellinen	0	0,37	2	1
Ei vastausta	8,5	8,89	23	24

Taulukko 4: Kokemus yli kokemusindeksin keskiarvon (0,682)

Luokka	%		n	
	XQuery	XIL	XQuery	XIL
Oikea vastaus	35,6	25,67	107	77
Pieni syntaksivirhe	3,33	0	10	0
Pieni operandivirhe	1,33	0	4	0
Pieni sisältövirhe	0,33	0	1	0
Korjauskelpoinen	13	15,6	39	47
Sisältövirhe	6	17,67	18	53
Syntaksivirhe	30,6	20	92	60
Puutteellinen	1,67	0	5	0
Ei vastausta	8	21	24	63

Taulukko 5: Kokemuksen ja vastausluokkien tilastollinen tarkastelu

	XIL		XQuery	
	U	p	U	p
Oikea vastaus	153,5	0,307	143,5	0,193
Korjauskelpoinen	225	0,325	210	0,574
Sisältövirhe	196	0,875	220	0,35
Syntaksivirhe	205	0,6761	227	0,3
Puutteellinen	200	0,329	162,5	0,253
Ei vastausta	176,5	0,645	201	0,718

Taulukko 6: Kokemusindeksin ja eri vastausluokkiin annettujen vastausten lukumäärän korrelaatio

Luokka	r	
	XIL	XQuery
Oikea vastaus	0,21	0,25
Korjauskelpoinen	-0,25	0,04
Sisältövirhe	0,05	-0,08
Syntaksivirhe	-0,02	-0,20
Puutteellinen	-0,11	0,23
Ei vastausta	0,02	-0,16

Taulukko 7: Koeaika alle koeajan keskiarvon

Luokka	%		n	
	XQuery	XIL	XQuery	XIL
Oikea vastaus	26,0	18,6	78	56
Pieni syntaksivirhe	0,3	0,0	1	0
Pieni operandivirhe	0,3	0,0	1	0
Pieni sisältövirhe	0,0	0,0	0	0
Korjauskelpoinen	11,7	21,0	35	63
Sisältövirhe	7,0	17,3	21	52
Syntaksivirhe	35,3	21,7	106	65
Puutteellinen	1,0	0,0	3	0
Ei vastausta	18,3	21,3	55	64

Taulukko 8: Koeaika yli koeajan keskiarvon

Luokka	%		n	
	XQuery	XIL	XQuery	XIL
Oikea vastaus	35,2	28,1	95	76
Pieni syntaksivirhe	4,1	0,0	11	0
Pieni operandivirhe	1,1	0,0	3	0
Pieni sisältövirhe	0,4	0,0	1	0
Korjauskelpoinen	14,1	16,2	38	44
Sisältövirhe	5,6	20,7	15	56
Syntaksivirhe	35,9	22,9	97	62
Puutteellinen	1,5	0,3	4	1
Ei vastausta	2,2	11,5	6	31

Taulukko 9: Koeajan ja vastausluokkien tilastollinen tarkastelu

	XIL		XQuery	
	U	p	U	p
Oikea vastaus	125,5	0,073†	154	0,327
Korjauskelpoinen	211	0,539	188,5	1
Sisältövirhe	155	0,339	200	0,739
Syntaksivirhe	192	0,942	179	0,786
Puutteellinen	178,5	0,3036	174	0,539
Ei vastausta	212,5	0,4139	248,5	0,042*

Taulukko 10: Koeajan ja eri vastausluokkiin annettujen vastausten lukumäärän korrelaatio

Luokka	r	
	XIL	XQuery
Oikea vastaus	0,34	0,27
Korjauskelpoinen	-0,03	-0,05
Sisältövirhe	0,12	0,01
Syntaksivirhe	-0,02	0,13
Puutteellinen	0,14	0,03
Ei vastausta	-0,20	-0,48

Taulukko 11: Opiskeluaika alle opiskeluajan keskiarvon (124 min)

Luokka	%		n	
	XQuery	XIL	XQuery	XIL
Oikea vastaus	33,3	20,7	90	56
Pieni syntaksivirhe	3,0	0,0	8	0
Pieni operandivirhe	0,0	0,0	0	0
Pieni sisältövirhe	0,0	0,0	0	0
Korjauskelpoinen	10,0	20,4	27	55
Sisältövirhe	5,9	16,3	16	44
Syntaksivirhe	29,6	19,6	80	53
Puutteellinen	0,7	0,4	2	1
Ei vastausta	17,4	22,6	47	61

Taulukko 12: Opiskeluaika yli opiskeluajan keskiarvon (124 min)

Luokka	%		n	
	XQuery	XIL	XQuery	XIL
Oikea vastaus	29,2	24,4	92	77
Pieni syntaksivirhe	1,3	0,0	4	0
Pieni operandivirhe	1,3	0,0	4	0
Pieni sisältövirhe	0,3	0,0	1	0
Korjauskelpoinen	15,6	18,1	49	57
Sisältövirhe	6,7	21,0	21	66
Syntaksivirhe	39,7	25,4	125	80
Puutteellinen	1,6	0,0	5	0
Ei vastausta	4,4	11,1	14	35

Taulukko 13: Opiskeluajan ja vastausluokkien tilastollinen tarkastelu

	XIL		XQuery	
	U	*p*	*U*	*p*
Oikea vastaus	167	0,523	202,5	0,734
Korjauskelpoinen	224	0,34	120,5	0,046*
Sisältövirhe	165,5	0,495	159,5	0,342
Syntaksivirhe	182,5	0,84	149	0,249
Puutteellinen	200	0,329	182	0,751
Ei vastausta	209,5	0,5	266	0,009**

Taulukko 14: Opiskeluajan ja eri vastausluokkiin annettujen vastausten lukumäärän korrelaatio

Luokka	*r*	
	XIL	XQuery
Oikea vastaus	0,09	-0,04
Korjauskelpoinen	0,01	0,33
Sisältövirhe	0,24	-0,18
Syntaksivirhe	0,04	0,31
Puutteellinen	-0,02	0,03
Ei vastausta	-0,20	-0,37

Taulukko 15: Lomakejärjestys: XIL-lomake ensin

Luokka	%		n	
	XQuery	XIL	XQuery	XIL
Oikea vastaus	25,3	19,3	76	58
Pieni syntaksivirhe	2,7	0,0	8	0
Pieni operandivirhe	0,0	0,0	0	0
Pieni sisältövirhe	0,0	0,0	0	0
Korjauskelpoinen	14,3	16,0	43	48
Sisältövirhe	4,7	17,0	14	51
Syntaksivirhe	37,7	28,0	113	84
Puutteellinen	1,0	0,0	3	0
Ei vastausta	14,3	19,7	43	59

Taulukko 16: Lomakejärjestys: XQuery-lomake ensin

Luokka	%		n	
	XQuery	XIL	XQuery	XIL
Oikea vastaus	37,2	26,3	106	75
Pieni syntaksivirhe	1,4	0,0	4	0
Pieni operandivirhe	1,4	0,0	4	0
Pieni sisältövirhe	0,4	0,0	1	0
Korjauskelpoinen	11,6	22,5	33	64
Sisältövirhe	8,1	20,7	23	59
Syntaksivirhe	32,3	17,2	92	49
Puutteellinen	1,4	0,4	4	1
Ei vastausta	6,3	13,0	18	37

Taulukko 17: Lomakejärjestyksen ja vastausluokkien tilastollinen tarkastelu

	XIL		XQuery	
	U	*p*	*U*	*p*
Oikea vastaus	140,5	0,165	127	0,076†
Korjauskelpoinen	161	0,417	179,5	0,773
Sisältövirhe	152	0,299	159,5	0,342
Syntaksivirhe	234	0,21	216	0,468
Puutteellinen	180	0,33	178,5	0,642
Ei vastausta	215,5	0,3758	197,5	0,8098

Taulukko 18: Ohjelmointi harrastuksena: ei harrastuneisuutta.

Luokka	%		n	
	XQuery	XIL	XQuery	XIL
Oikea vastaus	19,3	16,6	29	25
Pieni syntaksivirhe	0,67	0	1	0
Pieni operandivirhe	0	0	0	0
Pieni sisältövirhe	0	0	0	0
Korjauskelpoinen	12	18	18	27
Sisältövirhe	9,3	20,6	14	31
Syntaksivirhe	46,67	29,3	70	44
Puutteellinen	1,3	0	2	0
Ei vastausta	10,7	15,3	16	23

Taulukko 19: Ohjelmointi harrastuksena: harrastaa ohjelmointia.

Luokka	%		n	
	XQuery	XIL	XQuery	XIL
Oikea vastaus	36,2	25,2	152	106
Pieni syntaksivirhe	2,6	0	11	0
Pieni operandivirhe	0,9	0	4	0
Pieni sisältövirhe	0,2	0	1	0
Korjauskelpoinen	13,8	19,76	58	83
Sisältövirhe	5,4	18,6	23	78
Syntaksivirhe	32,14	21	135	88
Puutteellinen	1,2	0,23	5	1
Ei vastausta	7,3	15,23	31	64

Taulukko 20: Ohjelmointiharrastuksn ja vastausluokkien tilastollinen tarkastelu

	XIL		XQuery	
	U	p	U	p
Oikea vastaus	111,5	0,186	80	0,02*
Korjauskelpoinen	142	0,716	132	0,491
Sisältövirhe	129,5	0,449	184,5	0,291
Syntaksivirhe	162	0,81	185,5	0,327
Puutteellinen	148,5	0,569	154,5	1
Ei vastausta	169,5	0,555	190	0,175