Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

EEMIL AALTO-SETÄLÄ

Access-tietokantojen migraatio SQL Serverille

SÄHKÖ- JA AUTOMAATIOTEKNIIKAN TUTKINTO-OHJELMA
2023

TIIVISTELMÄ

Aalto-Setälä, Eemil: Access-tietokantojen migraatio SQL-Serverille
Opinnäytetyö, AMK
Tutkinto-ohjelma: sähkö- ja automaatiotekniikan koulutusohjelma
maaliskuu 2023
Sivumäärä: 40

Tämä opinnäytetyö suoritettiin Teollisuuden Voima Oyj:n toimeksiantona. Työn päätavoite oli siirtää Olkiluoto 3 ydinvoimalaitosyksikön turvallisuusautomaation tietokannat SQL Serverille, varmistaa datan eheys ja pitää siirron yhteydessä datan käytettävyys mahdollisimman hyvällä tasolla. Tietokantoja on säilytetty aiemmin Access-tiedostoissa, mutta tämä ei ole ydinvoimalaitoksen mittakaavassa pitkällä aikavälillä optimaalinen ratkaisu monista syistä.

Työssä toteutettiin datan migraatio Accessista SQL Serverille ja ylläpidettiin datan eheyttä siirron yhteydessä. Lisäksi tutkittiin eri vaihtoehtoja Access-työkalujen toimivuuden mahdollistamiseksi lähteenään SQL Serverille siirretty data.

Johtopäätöksenä todettiin, että valmiiden Access-pohjaisten työkalujen toimivuus on mahdollista säilyttää datan migraation yhteydessä huomioiden tietyt rajoitteet ja haasteet.

Avainsanat: tietokanta, SQL, Access

Abstract

Aalto-Setälä, Eemil: Migrating Access Databases to SQL Server
Bachelor's thesis
Degree programme: electrical and automation engineering
March 2023
Number of pages: 40

This thesis was commissioned by Teollisuuden Voima Oyj. The main objective was to transfer the safety automation databases of Olkiluoto 3 nuclear power plant unit to SQL Server, ensure data integrity, and maintain data usability during the migration process at the best possible level. The databases were previously stored in Access files, but this is not an optimal long-term solution for a nuclear power plant for various reasons.

The process consisted of data migration to SQL Server, maintaining data integrity during the transfer, and exploring different options to enable the functionality of Access tools with the transferred data as their source.

As a conclusion, it was found that the functionality of existing Access-based tools can be preserved during data migration, while acknowledging certain limitations and challenges.

Keywords: database, SQL, Access

SISÄLLYS

1 JOHDANTO

Olkiluoto 3:n suunnittelun, rakentamisen ja käyttöönoton aikana sen turvallisuusautomaation toiminnoista ja järjestelmistä on syntynyt suuria määriä dataa. Laitoksen tietokannoissa on miljoonia rivejä ja sarakkeita yksityiskohtaista tietoa esimerkiksi siitä, mikä kaapeli on yhdistettynä mihinkin automaatiokaappiin, ja missä huoneessa järjestelmät sijaitsevat. Tietokannoilla on tärkeä rooli laitosyksikön toimivuuden ja ydinturvallisuuden varmistamisessa.

Dataa on tähän asti säilytetty Access-tietokannoissa, ja niille on toteutettu eri kehittäjien toimesta Access-pohjaisia käyttöliittymiä. Access ei ole kuitenkaan työssä läpi käytävien asioiden vuoksi erityisen hyvä ratkaisu datan pitkäaikaiseen säilyttämiseen ydinvoimalaitoksen mittakaavassa, joten data on tarkoitus siirtää.

Työn tavoitteena on siirtää Olkiluoto 3 -laitosyksikön turvallisuusautomaation tietokantoja Accessista SQL Serverille. SQL Serverillä muun muassa suorituskyky, skaalautuvuus ja tietoturva on Accessiin verrattuna huomattavasti paremmalla tasolla. Siirron yhteydessä on tarkoitus varmistaa, että datan eheys säilyy esimerkiksi datatyyppien ja eheyssääntöjen muodossa. Tämän lisäksi tavoitteena on, että datan käytettävyys ja hyödynnettävyys olisi migraatioprosessin jälkeen mahdollisimman hyvällä tasolla.

Tietokannoille jo valmiina olevat ja aiemmin käytössä olleet käyttöliittyminä toimivat Access-työkalut ovat toiminnallisuudeltaan yksityiskohtaisia ja melko monimutkaisia. Työkalut pitävät sisällään suuren määrän tarkoin määriteltyjä toimintoja esimerkiksi kyselyiden, makrojen, moduulien ja lomakkeiden muodossa. Tästä johtuen ensisijaisena tavoitteena on saada valmiit käyttöliittymät toimimaan datan SQL Serverille siirtämisen jälkeenkin.

Työn alkuun on laadittu tietoperusta, joka muodostuu työn kahdesta ensimmäisestä kappaleesta. Ensimmäisessä teoriakappaleessa käsitellään lyhyesti työn toimeksiantajaa Teollisuuden Voima Oyj:tä, laitosyksikkö Olkiluoto 3:a ja sen perustoimintaa. Tämän jälkeen työssä perehdytään tietokantoihin, tarkemmin Accessiin ja SQL Serveriin, sekä niiden ominaisuuksiin toisiinsa verrattuna. Viimeisissä kappaleissa käsitellään työn toiminnallista osuutta yleisellä tasolla, pohditaan lopputuloksen onnistuneisuutta ja jatkokehitysvaihtoehtoja.

Työn aihe on vahvasti yhteydessä koulutusohjelmaan ja samalla työelämään, koska tietokantajärjestelmät ja erilaisten tietokantojen hallinta on keskeinen osa alan ammattilaisten vastuualueita ja työtehtäviä.

2 TEOLLISUUDEN VOIMA

Teollisuuden Voima Oyj on vuonna 1969 perustettu osakeyhtiö, jolla on Eurajoen Olkiluodossa vuoteen 2022 mennessä 44 vuotta sähköä tuottanut ydinvoimalaitos. TVO:lla on kuusi omistajayhtiötä, suurimpana niistä Pohjolan Voima Oyj. Viimeisten vuosikymmenten aikana TVO:sta on muodostunut koko yhteiskunnalle tärkeä sähköntuottaja, tuottaen kuudenneksen Suomen koko sähkönkulutuksesta pelkästään Olkiluoto 1:n ja Olkiluoto 2:n toimesta. Tuotetusta sähköstä noin puolet menee teollisuuden tarpeisiin ja toinen puoli kotitalouksille, palvelusektorille ja maatalouteen. (TVO, 2022a)

Kuva 1: Olkiluodon ydinvoimalaitokset

2.1 Olkiluoto 3

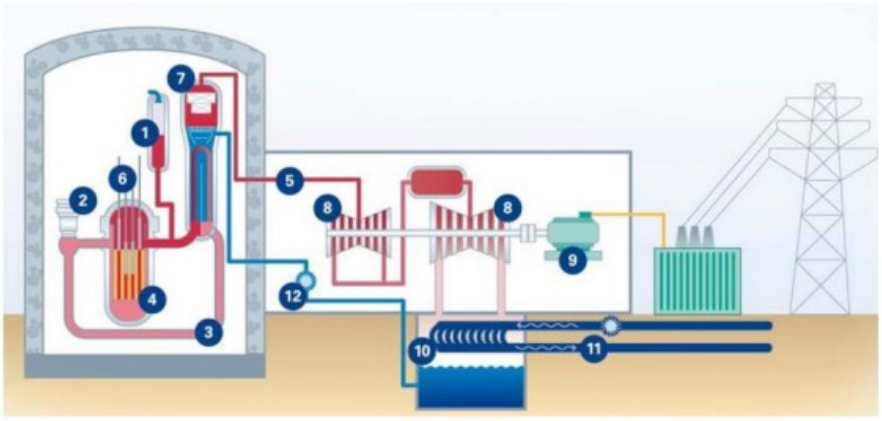
Olkiluodossa ollaan työn kirjoittamisen aikana ottamassa käyttöön lisäkapasiteetin tuomiseen ja sähkönkulutuksen suureen kasvuun vastaamaan kolmas laitosyksikkö Olkiluoto 3 (OL3). Sen polttoaineen lataus reaktoriin valmistui huhtikuussa 2021, ja ensimmäinen ydinreaktionkäynnistys tehtiin joulukuussa 2021. Yksikkö kytkettiin valtakunnan verkkoon ensimmäistä kertaa maaliskuussa 2022, ja säännöllisen sähköntuotannon on määrä alkaa koekäyttöohjelman jälkeen huhtikuussa 2023. (TVO, 2022b, s.4)

OL3 tilattiin avaimet käteen -periaatteella konsortiolta, jonka muodostavat AREVA GmbH, AREVA NP SAS ja Siemens AG. Projekti työllisti suurimmillaan noin 4500 henkilöä kansainvälisesti, vuoden 2022 lopulla noin 1500. Laitosyksikkö tuottaa vuodessa noin 12 TWh, joka kattaa noin 14 prosenttia Suomen sähköntarpeesta. Tämä vähentää säännöllisen sähköntuotannon aikana Suomen sähköntuontia noin 60 prosentilla, ja samalla päästöttömän sähköntuotannon osuus kasvaa 87 prosentista 90 prosenttiin. (TVO, 2022b, s.4)

Laitosyksikkö perustuu moderniin teknologiaan ja sisältää edistyksellisiä turvallisuusominaisuuksia. Erityisesti vakavien onnettomuuksien estämiseen ja hallintaan, tuotannon tehokkuuteen ja taloudellisuuteen on kiinnitetty yksikön suunnittelussa ja toteutuksessa huomiota. Laitoksen hyötysuhde, noin 37 %, onkin noin neljä prosenttia suurempi kuin Olkiluodon olemassa olevilla laitosyksiköillä alun perin. Olkiluoto 3:n käyttöiän on suunniteltu olevan vähintään 60 vuotta säännöllisen sähköntuotannon alkamisesta. (TVO, 2022b, s.5)

2.1.1 OL3 perustoiminta

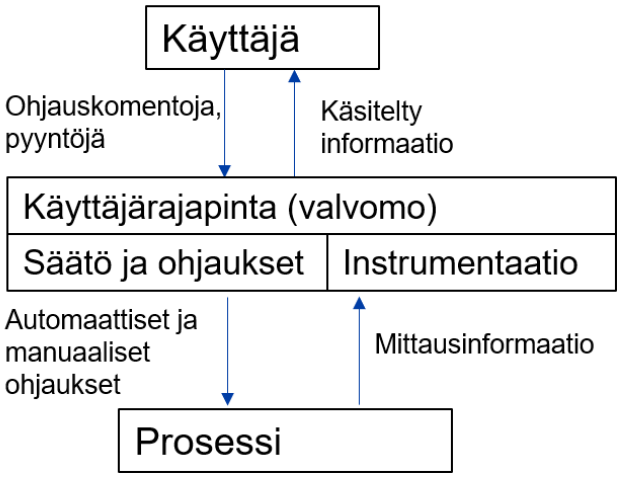
OL3 on EPR-tyyppinen (European Pressurized Water Reactor) painevesilaitos, jonka nettoteho on 1600 MW. Painevesilaitoksessa on lämmönsiirtoon kaksi erillistä piiriä. Paineistimen avulla korkeassa paineessa pidettävä vesi kiertää pääkiertopumppujen avulla primääripiirissä ja luovuttaa reaktorin tuottaman lämmön sekundääripiirille höyrystimessä. Reaktorin tehoa säädetään säätöelementeillä. Sekundääripiirin paine on selvästi primääripiirin painetta pienempi, joten sekundääripiirin vesi kiehuu höyrystimessä. Höyrystimessä syntynyt vesihöyry pyörittää turbiinia. Turbiini pyörittää samalle akselille kytkettyä generaattoria, joka tuottaa sähköä. Turbiinista tuleva höyry jäähdytetään takaisin vedeksi lauhduttimessa meriveden avulla. Lauhdevesi syötetään takaisin höyrystimeen syöttövesipumpulla, ja lämmennyt merivesi palautetaan mereen (Kuva 2). (TVO, 2022b, s.5)



Kuva 2: OL3 toimintaperiaate (TVO, 2022b, s.4)

2.1.2 OL3 automaatio

OL3 laitosyksikön automaatiojärjestelmien suunnittelussa on painotettu turvallisuutta ja käytön joustavuutta. Laitoksen säätö- ja ohjausjärjestelmät on täysin automatisoitu koeteltua digitaalitekniikkaa käyttäen, ja perinteinen langoitettu tekniikka toimii varmennuksena. Automaatioon kuuluu käyttäjärajapinta, automaatiojärjestelmä ja prosessirajapinta. Käyttäjärajapinnan toimintaan kuuluu prosessidatan ja hälytysten näyttäminen ja tallennus, manuaaliset ohjaukset ja automaattisten säätimien asetusarvot (Kuva 3). (TVO, 2022b, s.49)

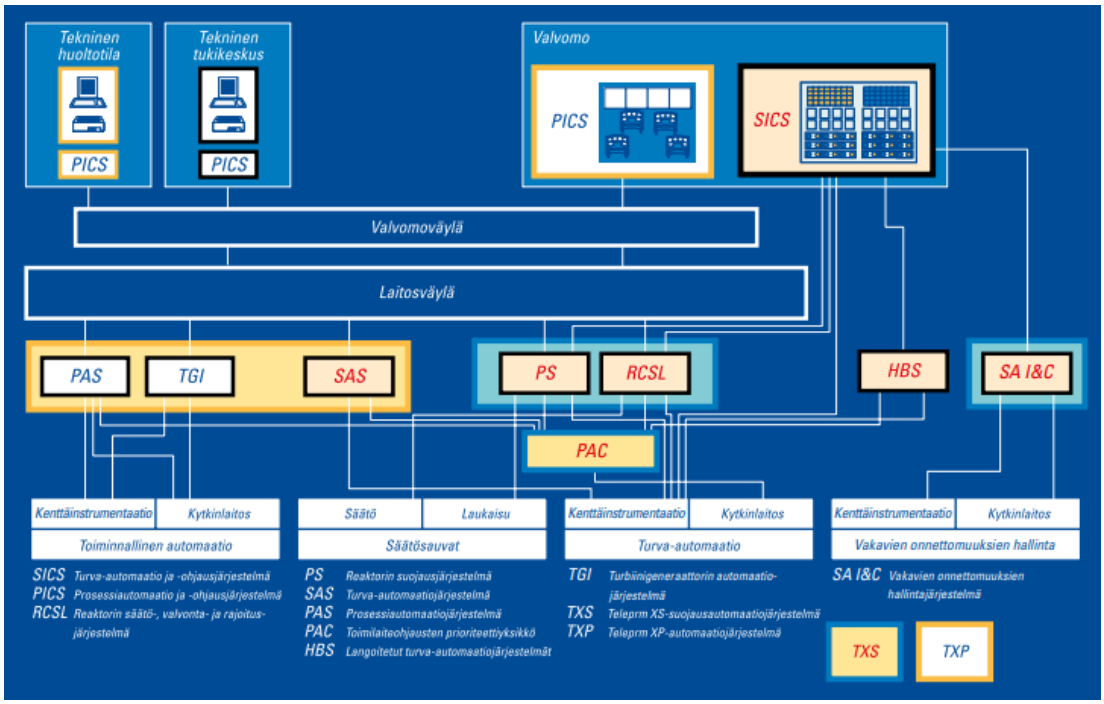


Kuva 3: OL3:n Automaatiojärjestelmä yksinkertaistetusti

OL3:ssa on poikkeustilanteiden varalta neljä toisiaan korvaavaa rinnakkaista turvallisuusjärjestelmää. Tämä tarkoittaa käytännössä, että useat eri laitteet ja järjestelmät voivat hoitaa samaan aikaan yhtä tehtävää. Näin ollen minkään yksittäisen laitteen vikatilanne ei vaaranna laitosyksikön turvallisuutta. Lisäksi turvallisuusjärjestelmät sijaitsevat neljässä erillisessä rakennuksessa samanaikaisen vioittumisen riskin pienentämiseksi. (TVO, 2022b, s.49)

Poikkeustilanteita hallinnoidaan OL3:ssa pääsääntöisesti digitaalisilla järjestelmillä. Reaktorin automatisoidut turvallisuustoiminnot antavat henkilöstölle 30 minuuttia aikaa korjaavien toimintojen suunnitteluun. Turvallisuuden takaamiseksi eri tilanteissa digitaalisten järjestelmien rinnalle on asennettu erillinen analoginen järjestelmä, jonka kautta laitosyksikön tilaa voidaan tarpeen vaatiessa seurata korjaavien toimenpiteiden tekemiseksi. (TVO, 2022b, s.49)

OL3:n automaatioarkkitehtuuri, joka on havainnollistettu kuvassa 4, toimii syvyyssuuntaisesti turvallisuuden takaamiseksi. Olosuhteissa, joissa käyttöparametrit ovat normaalien puitteissa, prosessiautomaatiojärjestelmät ylläpitävät ja ohjaavat laitoksen toimintaa. Tilanteessa, jossa normaalit käyttöparametrit on ylitetty, rajoitusjärjestelmät korjaavat laitoksen tilan takaisin normaaliksi siten, että suojausjärjestelmiä ei tarvitsisi käynnistää. Jos parametrit kuitenkin ylittävät jonkun suojausjärjestelmän kynnysarvoista, reaktorisuojausjärjestelmä käynnistää automaattisesti tarvittavat turvallisuustoiminnot. (TVO, 2022b, s.49)



Kuva 4: OL3 Automaatiojärjestelmien arkkitehtuuri (TVO, 2022b, s.48)

Edellä mainittujen digitaalisten järjestelmien menetyksen varalle on kaikista muista automaatiotoiminnoista riippumaton langoitettu turvajärjestelmä, jossa tarvittavat ohjaustoimenpiteet on mahdollista toteuttaa käsin. Vakavien onnettomuuksien varalta laitoksella on vielä kaikista edellä mainituista riippumaton vakavien reaktorionnettomuuksien automaatiojärjestelmä, joka antaa välttämätöntä tietoa laitoksen tilasta ja onnettomuuden vaikutuksista. (TVO, 2022b, s.49)

3 TIETOKANNAT

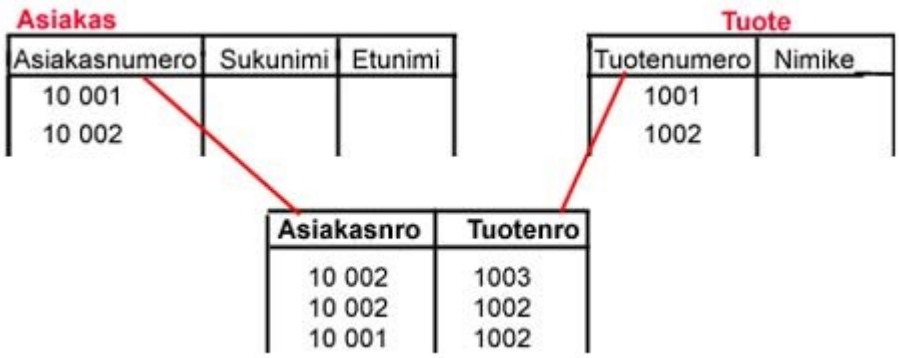
Tietokanta on organisoitu kokoelma rakenteellista tietoa tai dataa, joka on tarkoitettu suurien tietomäärien säilyttämiseen niin, että useampikin käyttäjä voi samanaikaisesti tarkastella ja hallinnoida tietoa tehokkaasti. Tietokannoissa tietoa noudetaan yleensä kyselykielillä, joista laajasti käytetyin ja tunnetuin on SQL (Structured Query Language).

3.1 Relaatiotietokanta

Suuressa osassa tietokantoja säilytetään dataa niin, että datapisteet liittyvät toisiinsa eli ovat relaatioissa toisiinsa nähden. Näitä relaatiomalliin perustuvia tietokantoja kutsutaan relaatiotietokannoiksi. Relaatiotietokannoissa esitetään tieto riveillä ja sarakkeilla.

Relaatiomalli on Edgar F. Coddin ajatuksiin perustuva matemaattinen teoria, joka on tietokannan fyysisestä toteutuksesta riippumaton. Tieto tallennetaan relaatiotietokannassa yhteen tiettyyn sijaintiin ja tieto voidaan hakea relaatioiden avulla tarvittaessa helposti. (Sarja, 2006)

Jos relaatiotietokannassa olisi alla olevan esimerkin (Kuva 5) mukaisesti taulut 'Asiakas' ja 'Tuote', niin asiakkaan 10001 sukunimen muuttuessa tarvitsee ainoastaan muuttaa tiedot 'Asiakas' -tauluun. Jos tieto olisi listattu yksinkertaisessa luettelomuodossa, tiedot olisi joutunut muuttamaan jokaisen ostetun tuotteen kohdalle. Esimerkki on yksinkertainen, mutta tämän vuoksi relaatiotietokanta on hyödyllinen ja suosittu keino tiedon säilyttämiseen. (Sarja, 2006)



Kuva 5: Relaatiotiomallinen tietokanta (Sarja, 2006)

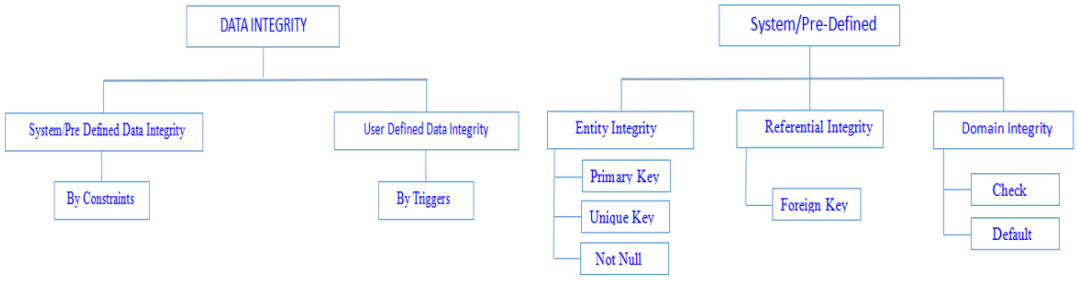
Relaatiotietokannoissa skeema määrittelee tietojen rakenteen, taulujen attribuutit ja niiden väliset suhteet. Se kuvaa, miten tietokannan tiedot on järjestetty ja millaisia sääntöjä niille on olemassa. Tämä sisältää datatyypit, taulujen väliset viiteavaimet ja mahdolliset yksilöllisyysrajoitteet. Skeeman avulla voidaan varmistaa tietokannan eheys. (IBM, 2021)

3.1.1 Datan eheys

Coddin ensimmäisten relaatiomallia käsittelevien julkaisujen jälkeen hän käsitteli tulevissa artikkeleissaan mallille ensimmäisiä eheyssääntöjä. Niiden mukaan relaatiotietokannassa tulee noudattaa tiettyjä sääntöjä, joista tärkeimmät ovat viite-eheys ja entiteetin eheys. (Doorn & Rivero, 2001, s. 8)

Relaatiotietokannoissa on suotavaa, että jokaisessa taulussa on rivit toisistaan yksilöllisesti erottava pääavain, ja entiteetin eheys tarkoittaa sitä, että mikään taulun pääavaimen osa ei saa sisältää tyhjäarvoja (null). Kahden eri tietokantataulun välillä toisiinsa liittyviin tietoihin taas viitataan määrittämällä viiteavain, ja viite-eheys tarkoittaa sitä, että jokaisella erilaisella viiteavaimen arvolla tulee olla identtinen pääavain tietokannan jossain toisessa taulussa. Muussa tapauksessa viiteavaimen arvon on oltava tyhjä. (Doorn & Rivero, 2001, s. 8)

Lisäksi relaatiotietokantoihin on olemassa domainin eli kohdealueen, ja sarakekohtaisia ehtoja, jotka viittaavat siihen mitä tietyt attribuutit sallivat arvoikseen. Myös käyttäjän eli tietokannan ylläpitäjien on mahdollista määrittää eheyssääntöjä, joita voi toimeenpanna esimerkiksi SQL triggereillä. Triggerit ovat T-SQL-koodia, ja ne voidaan asettaa suorittumaan automaattisesti, vastauksena toiseen määriteltyyn tapahtumaan tietokannassa. Kun eheyssäännöt ovat olemassa, käytössä oleva DBMS pakottaa niiden noudattamisen automaattisesti, ja antaa virheilmoituksia, kun niitä ollaan rikkomassa. (Doorn & Rivero, 2001, s. 9) Eheyden määrittelemisen osa-alueet voidaan esittää kuvan 6 mukaisesti.

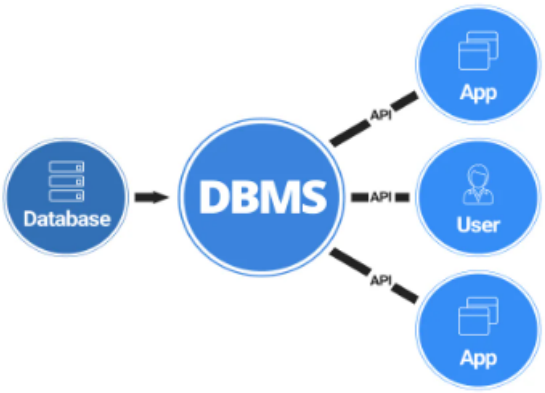


Kuva 6: Datan eheyden osa-alueita (Ranwa, 2022)

Eheys on keskeinen osa tietokantojen toimintaa ja luotettavuutta. Eheyssääntöjen noudattamisella varmistetaan, että tietokannat pysyvät johdonmukaisina ja virheettöminä, vähentäen samalla virheellisen datan olemassaolon riskiä. Tämä on erityisen tärkeää monimutkaisissa järjestelmissä kuten ydinvoimalaitoksissa, joissa useat sovellukset ja käyttäjät käyttävät tietoja. Lisäksi noudattamalla edellä mainittuja eheyssääntöjä voidaan helpottaa tietojen hallintaa ja analysointia, sekä parantaa tietokantojen suorituskykyä ja tehokkuutta. (Doorn & Rivero, 2001, s. 9)

3.1.2 DBMS

DBMS (database management system) eli tietokannan hallintajärjestelmä tarkoittaa ohjelmaa, jonka avulla on mahdollista säilyttää ja prosessoida suuria määriä dataa rakenteellisessa formaatissa. DBMS:ät toimivat tietokantakokonaisuuksissa käyttäjän ja tietokannan välisenä rajapintana (Kuva 7).



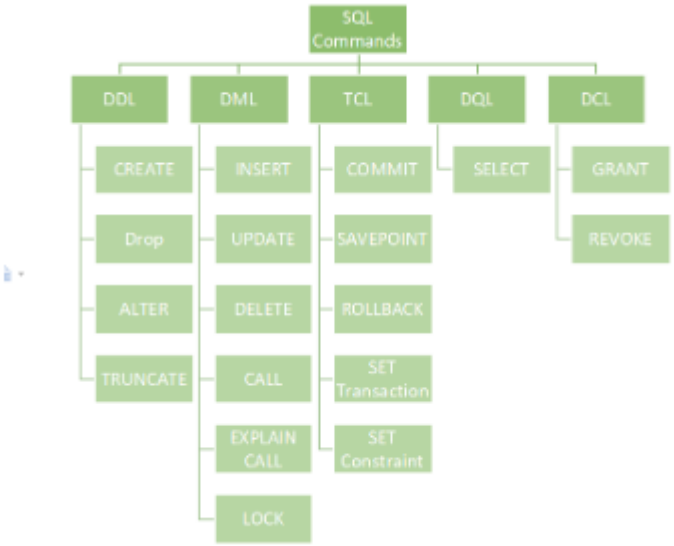
Kuva 7: DBMS:n toimintakokonaisuus (PrepInsta, n.d.)

Hallintajärjestelmät tallentavat datan relaatiomallin mukaisesti riveistä ja sarakkeista koostuviin tauluihin, joissa jokainen sarake edustaa tiettyä datan attribuuttia. Samalla hallintajärjestelmä varmistaa, että tiedot tallentuvat tietokantoihin johdonmukaisesti ja relaatioalgebran sääntöjä noudattaen. Suosituimpia DBMS:iä on Post-greSQL, Oracle Database, MySQL, SQLite, sekä tässä työssä käytössä olevat Microsoft SQL Server ja Microsoft Access. (Wiech, 2020)

3.2 SQL

SQL (Structured Query Language) on kaupallinen IBM:n 1970-luvulla kehittämä ohjelmointikieli relaatiotietokantojen hallintaan. Se tarjoaa tehokkaat työkalut esimerkiksi datan lisäämiseen, poistamiseen, päivittämiseen ja hakemiseen tietokannan hallintajärjestelmässä viiden eri kategorian komennoilla (Kuva7). SQL:ää voidaan käyttää myös muihin tarkoituksiin, kuten data-analytiikassa ja tietokantojen toiminnan optimoinnissa. (AWS, 2023)

SQL on laajasti hyvin laajasti käytössä, ja monet kaupalliset ja avoimen lähdekoodin tietokannanhallintajärjestelmät tukevat sitä. SQL tarjoaa standardikielen datan hakemiseen ja hallintaan tietokannassa, mikä yksinkertaistaa koodin jakamisesta ja yhteistyötä projekteissa. SQL-komennot voidaan jakaa viiteen kategoriaan kuvan 8 mukaisesti.



Kuva 8: SQL komentoja kategorioittain (GeeksforGeeks, 2019)

3.3 SQL Server

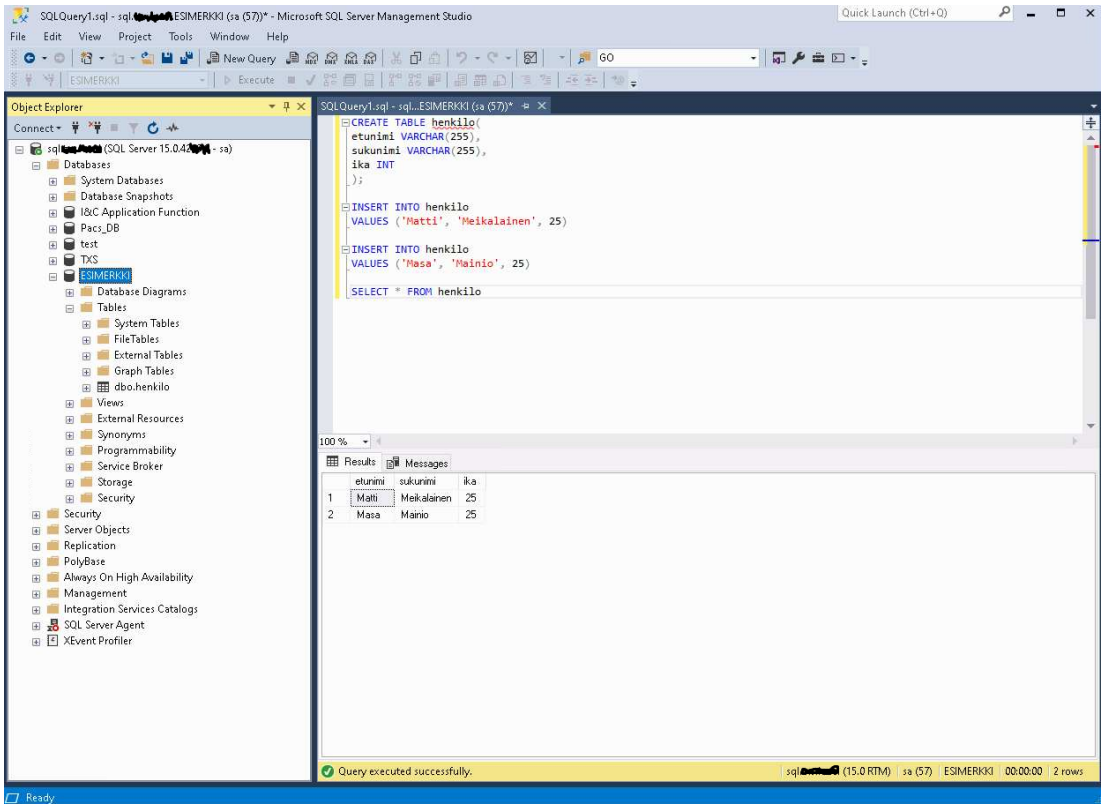
Microsoftin SQL Server on relaatiotietokannoille tarkoitettu hallintajärjestelmä, eli RDBMS. Sen ensimmäinen versio ilmestyi vuonna 1989 ja tästä lähtien siitä on julkaistu useita eri versioita, joista tuorein ja tässä työssä käytössä ollut on SQL Server 2019. SQL Server on suunniteltu kilpailemaan suoraan MySQL:n

ja Oraclen kanssa, jotka ovatkin yhdessä SQL Serverin kanssa kolme selvästi suosituinta tietokannan hallintajärjestelmää. (Peterson, n.d.)

SQL Server tukee avoimen lähdekoodin standardi SQL:ää, mutta siihen lukeutuu myös Microsoftin omakehitteinen versio SQL-kyselykielestä, eli T-SQL. T-SQL on käytännössä lisäosa standardi SQL:ään, tarjoten samojen perusominaisuuksien lisäksi hyödyllisiä laajennuksia. T-SQL suorittaa koodia palanen kerrallaan loogisesti ja rakenteellisessa järjestyksessä, niin sanotusti lohkoissa, kun taas standardi SQL:ssä koodi suoritetaan rivi kerrallaan. (Clark, 2021)

3.3.1 SQL Server Management Studio

Microsoftin SQL Server Management Studio (SSMS) on integroitu hallintaympäristö SQL infrastruktuurille. SSMS:lla voi hakea ja ylläpitää dataa sekä konfiguroida, hallinnoida, ja kehittää kaikkia SQL Server-ympäristön muita komponentteja. SSMS:n perusnäkymässä on vasemmassa reunassa Object Explorer (Kuva 9), josta löytyy muun muassa kaikki SQL Server-instanssin tietokannat ja niiden taulut, taulujen datatyypit, avaimet, proseduurit, funktiot ja muut ominaisuudet. (Microsoft, 2023a)



Kuva 9: SSMS käyttöliittymä ja yksinkertainen esimerkki SQL-lauseista

Object Explorerin yläpuolella on alasvetovalikko, jossa on kaikki instanssiin kuuluvat tietokannat. Näistä valitaan SQL-kyselyä kirjoittaessa mitä tietokantaa vasten kysely suoritetaan. Valikosta on valittavissa käyttäjätietokantojen lisäksi myös järjestelmätietokannat master, model, msdb ja tempdb. Järjestelmätietokannat sisältävät metatietoja ja asetuksia SQL Server instanssista. Kyselyitä suorittaessa on oltava huolellinen, ettei ne tule suoritetuksi järjestelmätietokannoissa, koska tämä johtaa mahdollisesti merkittäviin ongelmiin instanssin tietokantojen toiminnassa.

3.3.2 Proseduuri

Proseduurit ovat SQL-lauseiden joukkoja, jotka tallennetaan tietokantaan yhdeksi kokonaisuudeksi ja voidaan tallentamisen jälkeen suorittaa toistuvasti tehokkaasti.

Proseduurit omaavat useita etuja yksinkertaisiin SQL-komentoihin verrattuna. Ne parantavat tietokannan suorituskykyä vähentämällä tiedon määrää, joka on

lähetettävä edestakaisin clientin eli asiakasohjelman ja palvelimen välillä. Lisäksi proseduurit ovat turvallisempia kuin perinteiset SQL-lauseet, ja mahdollistavat tehokkaamman helpomman koodin uudelleenkäytön ja ylläpidon. Proseduuri luodaan SQL:ssä ’CREATE PROCEDURE’ -komennolla (Kuva 10).

```
CREATE  PROCEDURE proseduurin_nimi
AS
sql_lauseet
GO;

EXEC proseduurin_nimi;
```

Kuva 10: Proseduurin luominen ja suorittaminen

3.3.3 Change Data Capture (CDC) ja Change Tracking (CT)

Muutosten seuranta on suurissa tietokannoissa tärkeää, ja SQL Serverissä on useita keinoja sen toteuttamisen. Yksi näistä on CDC (Change Data Capture), joka tallentaa tietokantatauluihin tulleet muutokset. Sen tallentamaa yksinkertaista relaatioperusteisessa muodossa olevaa muutostietoa voi hyödyntää monipuolisesti eri käyttötarkoituksissa, kuten auditoinnissa, datan synkronoinnissa ja datan migraatiossa. CDC on metodina tehokas, mutta voi aiheuttaa huomattavasti lisää prosessointia, hidastaen koko järjestelmän toimintaa. (Microsoft, 2023b)

Vaihtoehtoisesti muutosten seurannassa voi käyttää Change Trackingia (CT), joka on ominaisuuksiltaan ja toiminnallisesti yksinkertaisempi vaihtoehto kuin CDC. CT tallentaa muutoksista tietoa, kuten mihin kohtaan muutos tuli ja oliko muutos datan lisäys, poistaminen vai päivittäminen. Se vie huomattavasti vähemmän prosessointikapasiteettia palvelimelta CDC:hen verrattuna, mutta dataan tulleita muutosten yksityiskohtia se ei tallenna. (Microsoft, 2023b)

3.4 Microsoft Access

Toinen työssä käytössä ollut DBMS on Microsoft Access. Microsoftin Jet Database Engineen pohjautuva Access julkaistiin vuonna 1992, ollen tuohon

aikaan ainoa massamarkkinoille tarkoitettu tietokannan hallintajärjestelmä satavilla Windowsille. (Terra, 2023)

Access on ennen kaikkea helppo keinon säilyttää ja hallinnoida dataa. Se mahdollistaa käyttäjille helpon tietokantasovellusten luomisen kustomoiduilla kyselyillä, tauluilla, lomakkeilla ja raporteilla. Access on parhaimmillaan edullinen tapa toteuttaa yhden käyttäjän tai pienen työryhmän tietokantasovelluksia. Sen puutteet tulevat esiin, kun käyttäjä- ja datamäärä lisääntyy. Access ei ole aito palvelinpohjainen DBMS, vaan tiedostopohjainen työpöytätietokanta, minkä vuoksi se ei ole otollinen ratkaisu laajempiin usean käyttäjän sovellusratkaisuihin.

Kun suuri määrä käyttäjiä työskentelee Accessissa, ongelmia voi aiheutua muun muassa datan turvallisuuden, skaalautuvuuden, luotettavuuden ja ylläpidettävyyden kanssa. Kun tietokanta kasvaa datan ja käyttäjien määrässä, usein on kannattavaa siirtyä Accessista muihin järjestelmiin, kuten SQL Serveriin. (FMS, n.d.)

3.4.1 Access -kyselykieli

Tietokantojen migraatiota Accessista SQL serverille tehdessä on huomioitavaa, että Accessissa käytössä oleva kyselykieli on syntaksiltaan hieman erilainen kuin standardi SQL. Näin ollen Accessiin tehtyjä kyselyitä on muokattava SQL serveriin sopivaan muotoon, jos tavoitteena on suorittaa kyselyiden prosessointi SQL Serverin puolella.

Monista syntaksieroista esimerkkejä ovat merkkijonojen erittelyyn Accessissa käytössä olevat heittomerkit (" ") ja SQL:n lainausmerkit (' ') ja korvausmerkki, joka kirjoitetaan Accessissa asteriskilla (*) ja SQL:ssä prosenttimerkkinä (%). Lisäksi on useita funktioita, joita kutsutaan eri funktionimillä Accessissa ja standardi SQL:ssä. Varsinkin SQL koodin on oltava syntaksiltaan virheetön ja kieliopiltaan täysin oikein, jotta suoritus menee läpi ilman virheviestejä. Kuvassa 11 on esimerkkejä kyselykielien eroavaisuuksista.

	Access(VBA)	SQL Server
Hakee merkkijonon '.' taulusta ja palauttaa sen sijainnin numeerisella arvolla	`inStr(1,T02_Geraete.Einbauplatz,'.')`	`CHARINDEX('.', T02_Geraete.Einbauplatz)`
Hakee minkä tahansa merkkijonon, johon sisältyy merkkijono 'CS'	`(T20_KS.Schrank) Like '*CS*'`	`(T20_KS.Schrank) Like '%CS%'`

Kuva 11: Access ja SQL Kyselykielien eroavaisuuksia

3.4.2 Access Form

Suurimmalla osalla työssä siirrettävistä tietokannoista on tehty käyttöliittymä Accessin Formeilla eli lomakkeilla. Formit ovat Accessiin kuuluvia tietokanta-objekteja, joiden luominen mahdollistaa helpon, nopean ja mukautettavissa olevan tavan lisätä, muokata tai hakea tietokantoihin tallennettua tietoa. (Microsoft, n.d.a)

Formien toiminnot tapahtuvat taustalla pääasiassa Accessin moduulien avulla, jotka ovat kokoelmia käyttäjän määrittämiä, VBA-koodilla kirjoitettuja toimintoja, aliohjelmia ja globaaleja muuttujia. (Microsoft, n.d.a)

3.5 Access-SQL Server tietokantayhteys

Accessin ja SQL Serverin välille on mahdollista muodostaa yhteys yhteistä rajapintaa käyttämällä, hyödyntäen näin molempien DBMS:ien vahvuuksia samanaikaisesti, ja mahdollistaen Accessin lomakkeiden ja muiden käyttöliittymään kuuluvien objektien käytön lähteenään SQL Serverillä sijaitseva data.

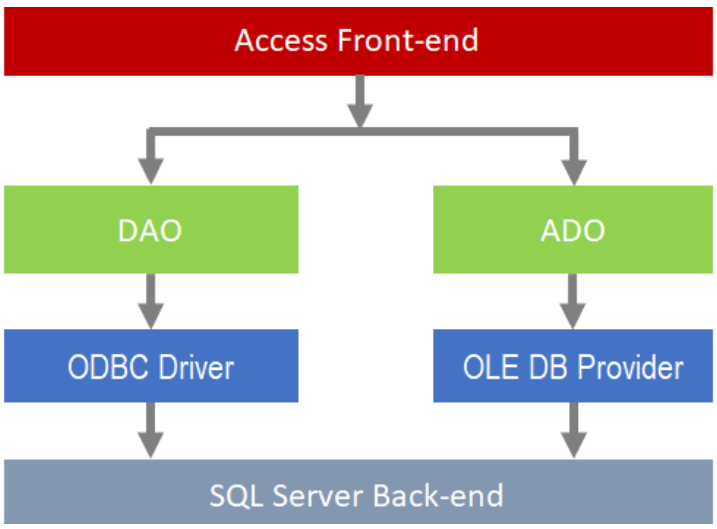
3.5.1 ODBC ja OLE DB

ODBC ja OLE DB ovat ohjelmointirajapintoja, jotka tarjoavat keinon päästä käsiksi eri datalähteisiin, kuten tietokantoihin tallennettuihin tietoihin.

Molemmat tekniikat mahdollistavat SQL Serveriin tallennetun datan käsittelyn Accessista yhteysmerkkijonon (connection string) avulla. Yhteysmerkkijono välittää konfiguraatiotietoja suoraan tietokantaan, kuten palvelimen sijainnin, tietokannan nimen ja suojausmenetelmän. (Microsoft, n.d.b)

ODBC ajureita käyttäessä tyypillisin tapa yhteyden muodostamiseen on DSN tiedosto, mutta vaihtoehtoisesti yhteyden voi määrittää VBA-koodin avulla. Tässä työssä käytettiin pääasiassa DSN tiedostoja. ODBC vaatii toimiakseen saman version ODBC-ajurit asennettuna sekä sille työasemalle, jossa Accessia ajetaan, että SQL Serverille, jolla data sijaitsee. OLE DB on uudempi rajapinta, ja se ei vaadi lähdetiedostojen käyttöä. (Devart, n.d.)

Access käyttöliittymän ja SQL Serverin välillä on yhteyden muodostamiseksi vielä rajapinnat ADO ja DAO, riippuen siitä kumpi ohjelmointirajapinnoista on käytössä. ADO on yhteydessä OLE DB:hen, joka toimii ’välikätenä’ ADO:n ja SQL Serverin välillä. Vastaavasti ODBC toimii DAO:n ja SQL Serverin välillä (Kuva 12).



Kuva 12: Tiedonsiirto Access -sovelluksen ja SQL Server -tietokannan välillä (Microsoft, n.d.b)

3.5.2 DSN

DSN on ODBC:n yhteydessä käytössä oleva termi, joka viittaa tallennettuun konfiguraatioon määritellystä tietokannasta. DSN:ää käytetään

tietokantayhteyksien muodostamiseen ODBC datalähteiden välillä, ja siihen on tallennettu tietoa määritellystä tietokannasta. Se on toisin sanoin ODBC yhteyksiin viittaava symbolinen nimitys, johon on tallennettu tietokantayhteyden määritelmät yhteyden muodostamista varten, kuten tietokannan nimi, ajuri, hakemisto, käyttäjätunnus ja salasana. (Microsoft, n.d.c)

ODBC datanlähteen tyyppejä on useita; User DSN, System DSN, ja tässä toteutuksessa käytössä ollut File DSN, jossa DSN eli yhteyden määrittelyt on tallennettu tekstitiedostoon. Toimiakseen File DSN vaatii, että siihen sisältyy kaikki tiedot, joita tarvitaan haluttuun datanlähteeseen yhdistämiseen. Lisäksi oikea versio ODBC ajurista on oltava asennettuna paikallisesti, jota puhtaalla SQL Server toteutuksella ei ole olemassa oletuksena. (Microsoft, n.d.c)

3.6 Migraatioarkkitehtuurin toteutusvaihtoehdot

Access tietokantojen migraatioprojektia aloittaessa on huomionarvoista, että migraation voi toteuttaa monella eri tasolla pelkän datan siirtämisen ja kokonaan uusien sovellusten ja hallintaratkaisuiden tekemisen välillä.

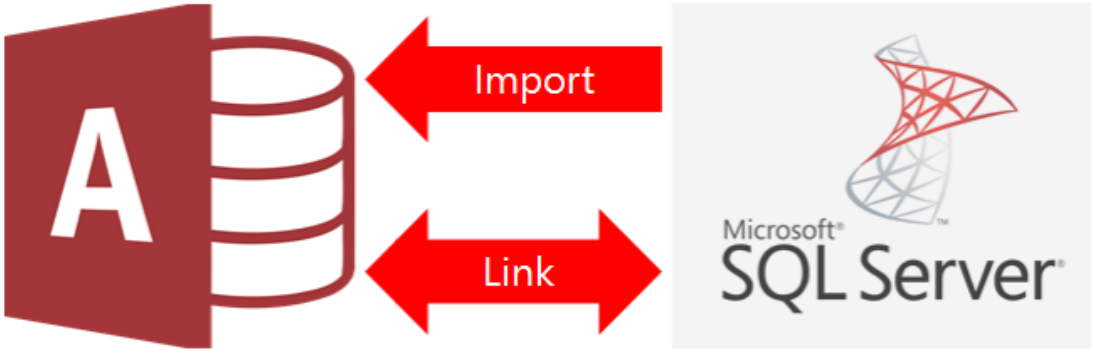
Siirto on teoriassa mahdollista suorittaa niin, että Accessissa jo valmiina olevaa logiikkaa eli esimerkiksi tauluratkaisuja, kyselyitä, lomakkeita, raportteja, makroja ja moduuleja voidaan hyödyntää siirron jälkeenkin dataa linkittämällä SQL palvelimen ja Access sovelluksen välillä. Näin tekemällä on mahdollista hyödyntää Accessin helppoutta ja SQL Serverin luotettavuutta, turvallisuutta ja skaalautuvuutta samanaikaisesti. Suurimpana hyötynä tässä menetelmässä on se, että vanhat ja usein toimiviksi todetut sovellukset ja niihin toteutetut logiikat toimivat ja niiden käyttöä voidaan jatkaa ilman resurssien käyttöä uusiin toteutuksiin. (FMS, n.d.)

Kokonaisessa uudelleentoteutuksessa datan siirron lisäksi tehdään kokonaan uusi käyttöliittymä toimintoineen, jos sellaiselle on tarvetta. Käyttöliittymän toteuttamiseen on monia eri vaihtoehtoja; esimerkiksi .NET on SQL Serverin kanssa hyvin yhteensopiva uuden käyttöliittymän kehittämiseen, tarjoten

muun muassa hyvätasoista koodinhallintaa useille käyttäjille ja monipuolisia ohjelmointikomponentteja.

3.7 Datan linkitys SQL:n ja Accessin välillä

Accessin ja SQL:n välille on mahdollista luoda yhteys linkittämällä data SQL Serveristä Accessiin. Linkitys on kaksisuuntainen, ja synkronoi SQL server tietokantaan tehdyt muutokset Accessiin ja toisinpäin. Ilman linkitystä dataa tuodessa Access luo siitä kertaluontoisen kopion, ja dataan tehdyt muutokset eivät synkronoidu (Kuva 13). (Microsoft, n.d.d)



Kuva 13: Access-SQL yhteys (Microsoft, n.d.d)

Linkityksen jälkeen SQL palvelimella olevaa dataa voi siis tarkastella ja muokata Accessin kautta. Lisäksi linkityksien avulla Accessin käyttäjäystävällisyyttä on mahdollista hyödyntää 'front-end' -sovelluksena SQL palvelimella olevalle datalle. Linkitys oli tarkoituksenmukaista tässä työssä, koska tietokannoilla oli valmiita Access-pohjaisia käyttöliittymiä, joiden käyttöä oli mahdollista jatkaa SQL palvelimelle siirretyn datan kanssa linkitysten kautta.

3.7.1 Läpivientikysely

Accessissa on linkitykseen liittyen mahdollisuus suorittaa myös läpivientikyselyitä. Läpivientikysely on kysely, joka lähettää komentoja suoraan tietokantapalvelimelle, tässä tapauksessa SQL-Serverille, prosessoi kyselyn kokonaisuudessaan palvelimen puolella ja palauttaa tulokset Accessiin. Pienten

datamäärien kanssa läpivientikyselystä saatava hyöty on minimaalinen, mutta kun datamäärät kasvavat, läpivientikysely on prosessoinnin mahdollisimman suuren nopeuden ja tehokkuuden varmistamiseksi hyvä ratkaisu.

Linkitykseen verrattuna kyselyn suorittaminen läpivientinä on hyödyllistä, koska suoraan linkitettyyn tauluun kyselyä suoritettaessa, koko datamäärä tuodaan palvelimelta 'clientin' puolelle suodatukseen, joka tuottaa turhaa prosessointia. Läpivientikyselyä suoritettaessa kysely taas ohjataan suoraan palvelimelle, prosessointi tapahtuu siellä ja ainoastaan data, joka läpäisee kyselyn seulan, palautuu 'clientille'. (Microsoft, n.d.e)

Accessiin tottuneella läpivientikyselyjen haasteena voi olla se, että Access ei tarjoa läpivientikyselyitä tehdessä lainkaan apua esimerkiksi koodin värityksellä tai oikeinkirjoituksen tarkistamisella, toisin kuin SSMS. Tämän lisäksi läpivientikyselyitä tehdessä on osattava kirjoittaa SQL Serverin omaa T-SQL kieltä, sillä Accessin SQL kyselykieli ei toimi niissä. Syntaksi on toki melko samanlainen, joten toisinaan Access-kyselyn läpivientiä SQL:ään tehdessä koodia ei tarvitse muuttaa lainkaan.

3.8 NoMachine

Työssä käytettiin NoMachinen työaseman virtualisointisovellusta palvelimille yhteyden muodostamiseen. Sovelluksen graafisesta käyttönäkymästä otetaan yhteys SUSE Linux-palvelimelle, jonka komentokehotteesta taas otetaan yhteys ’rdesktop’ -komennolla SQL Serveriin, johon data siirrettiin ja suunnitteluympäristön työasemille, joihin Access käyttöliittymät sijoitettiin.

4 TOTEUTUS

4.1 Lähtökohta

Ydinvoimalaitoksen OL3 automaation tietokantoja on säilytetty Access-tietokannoissa, koska laitoksen toimittaja Areva on aikoinaan valinnut sen datalle alustaksi. Nyt tietokannat olisi kuitenkin siirrettävä Accessista SQL Serverille, koska se on huomattavasti joustavampi ja luotettavampi datan säilytysmenetelmä. Lisäksi SQL Server on tietoturvanäkökulmasta merkittävästi parempi ratkaisu, ja dataan tulleiden muutosten seuranta ja varmuuskopiointi on mahdollista toteuttaa sujuvammin SQL palvelimella Accessiin verrattuna.

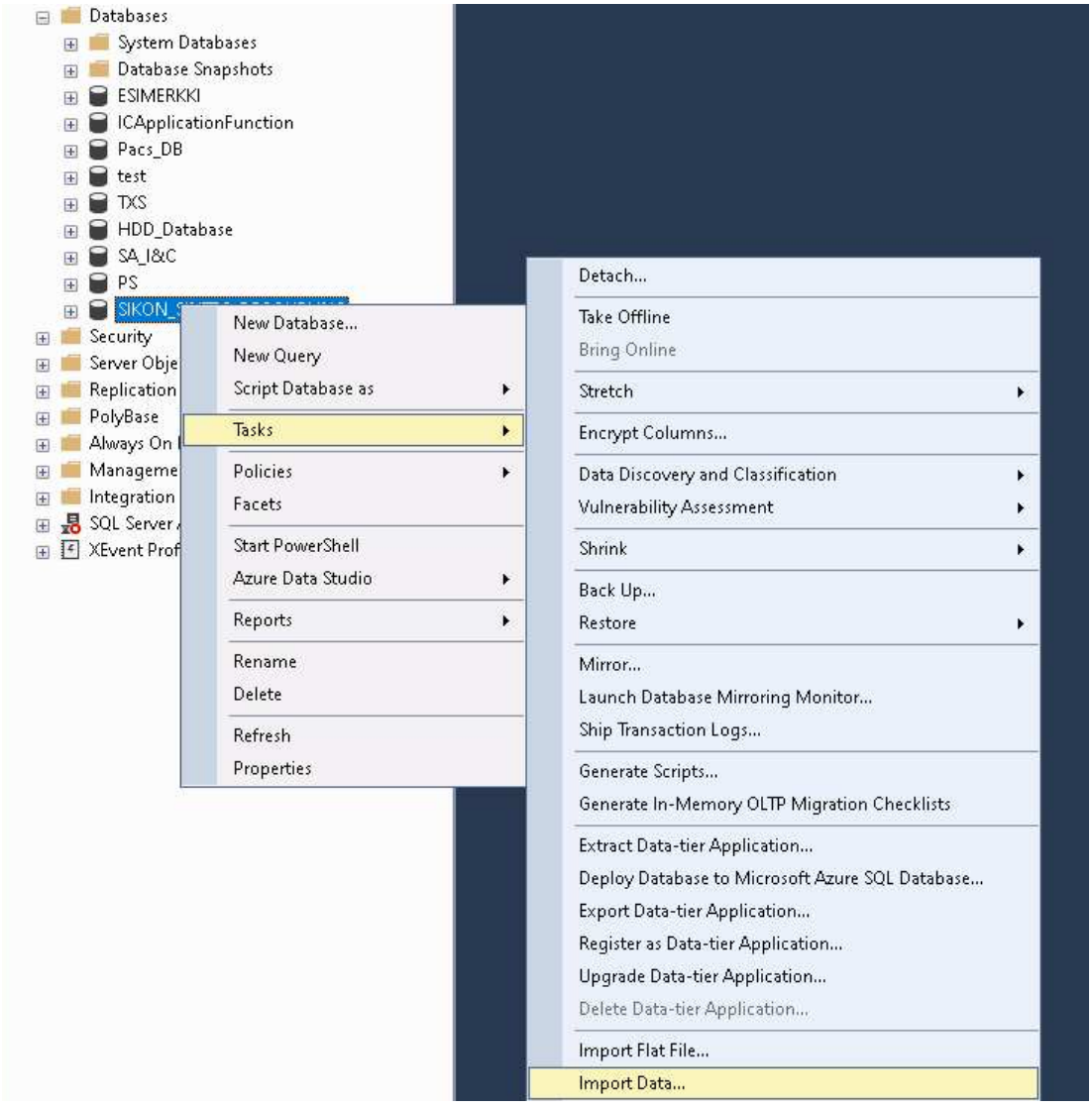
Accessissa on myös joitain rajoitteita, joita SQL:ssä ei käytännössä ole. Tietokantoja on esimerkiksi jouduttu pilkkomaan pienempiin osiin Accessin maksimikapasiteetin rajallisuuden vuoksi. SQL:n puolella näitä pilkottuja tietokantoja on mahdollista taas yhdistää suuremmiksi tietokannoiksi, mikä yksinkertaistaa niiden hallintaa ja parantaa datan käytettävyyttä.

Työn tarkoitus ja tavoite on siirtää Olkiluoto 3 -laitosyksikön turvallisuusautomaation Access tietokannat SQL Serverille. Tietokannat siirretään varmistaen samalla, että data säilyy virheettömänä ja eheänä datatyyppien ja eheyssääntöjen kannalta. Lisäksi datalle valmiina olevat työkalut on tarkoitus saada toimimaan mahdollisimman hyvin, ja selvittää uusien työkalujen ja käyttöliittymien tarvetta. Uusia ratkaisuja mietitään varsinkin sellaisille tietokannoille, joissa valmiita työkaluja ja niiden kyselyitä, makroja, moduuleita ja lomakkeita ei saada toimimaan datan siirtämisen jälkeen.

4.2 Data Import SQL Serverille

SQL Server Management Studion tarjoamiin lisäosiin kuuluu datan migraation työkaluksi tarkoitettu 'Import and Export Wizard'. Työkalun avulla voi siirtää suuria määriä dataa tehokkaasti valitsemalla datan lähteen, siirrettävät taulut, tai kirjoittamalla kyselyitä datan purkamiseen valitusta lähteestä.

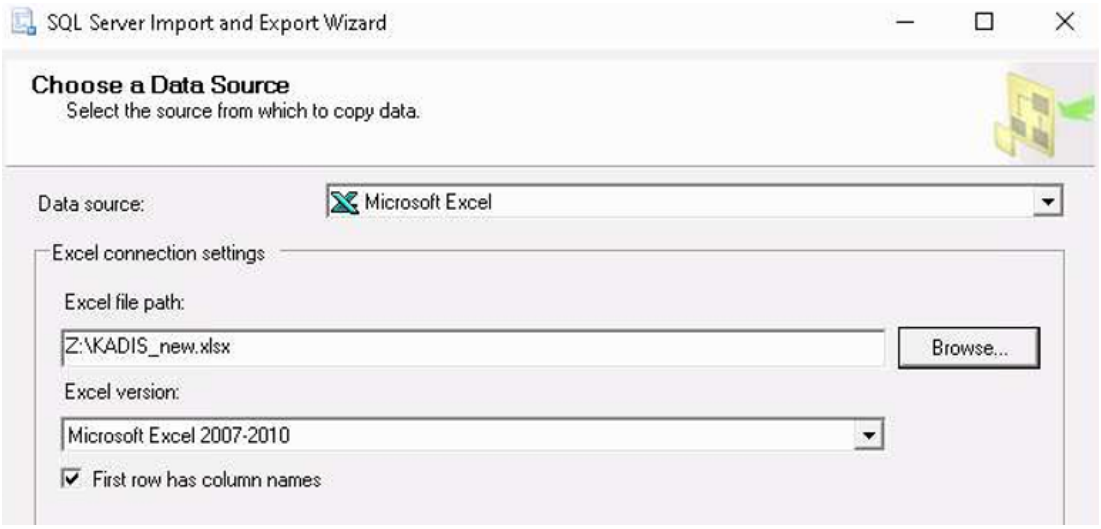
Import and Export Wizardin käynnistämiseksi tulee ensiksi valita Object Explorerista tietokanta, johon siirrettävä data on tarkoitus sijoittaa. Haluttua tietokantaa hiiren oikealla painikkeella klikkaamisen jälkeen valitaan Tasks, jonka kautta avautuvasta valikosta Import Data, jos datan lähde on rakenteellinen, tai 'Import Flat File', jos siirrettävä data on rakenteetonta, esimerkiksi tekstitiedostona (Kuva 14).



Kuva 14: Import and Export Wizardin käynnistäminen

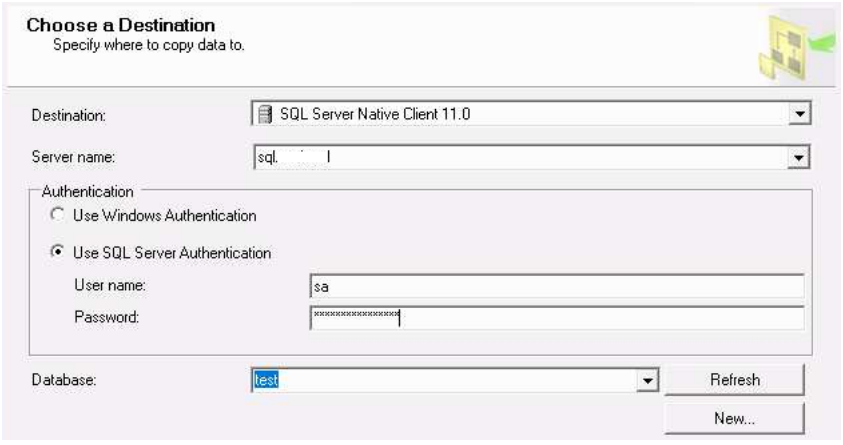
Import and Export Wizardin avauduttua työkalu kysyy ensimmäiseksi siirrettävän datan lähdettä. Lähde voi olla esimerkiksi Access tiedosto, toinen SQL Server tietokanta tai Excel taulukko. Access- tai Excel-tiedostoa importoidessa

tiedosto täytyy olla siirrettynä työasemalle, jolla SQL Server instanssi sijaitsee. Toiselta SQL palvelimelta siirrettäessä yhteyden voi muodostaa suoraan palvelimelta toiselle SQL Server autentikoinnin avulla. Tässä kohtaa tulee vielä laittaa valintamerkki ’First row has column names’ -laatikkoon Excel-tiedostoa importoidessa, jos taulukon ensimmäisellä rivillä on sarakkeiden nimet, kuten tavallisesti on (Kuva 15).



Kuva 15: Import and Export Wizard datalähteen valinta

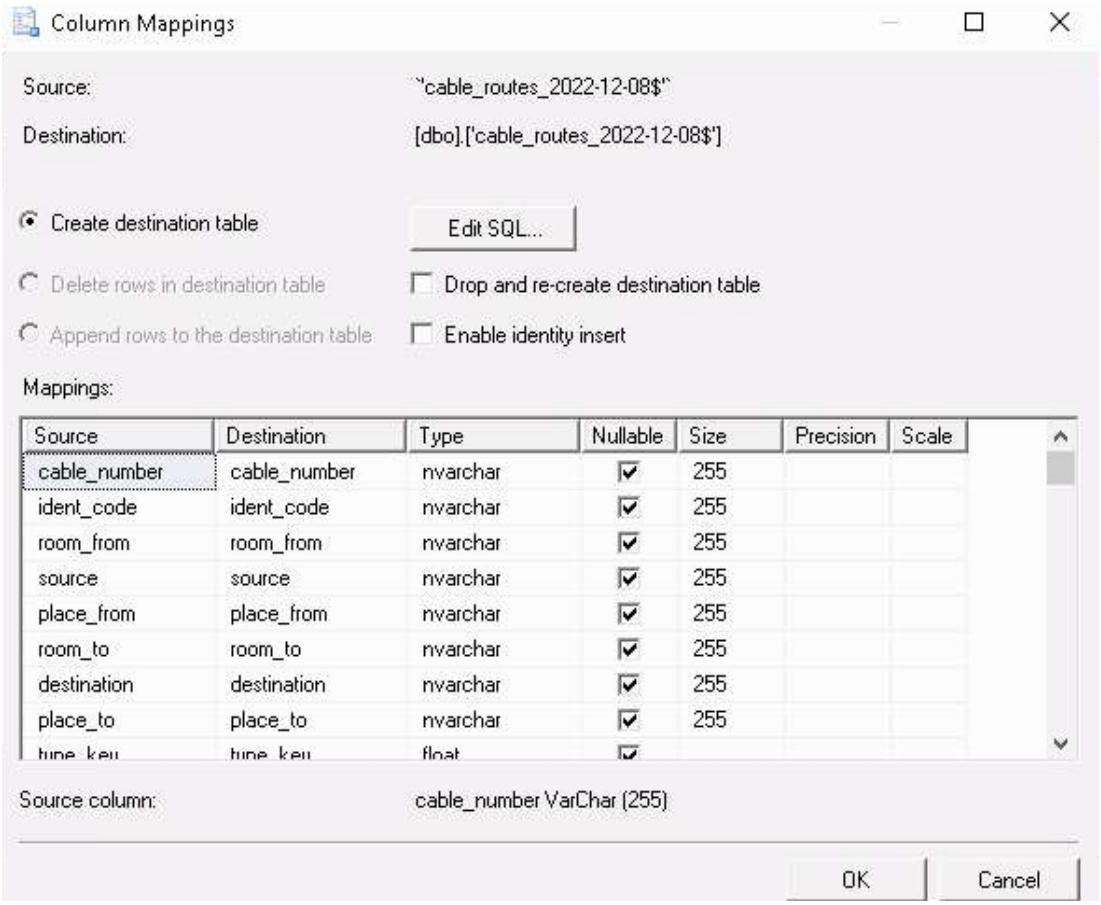
Datalähteen valinnan jälkeen seuraava vaihe on määritellä paikka, johon data on tarkoitus sijoittaa. Sijainti, SQL palvelimen nimi, ja aikaisemmin valittu tietokanta täyttyvät kuvan 16 mukaisesti itsestään, jos työkalun on avannut valitsemalla aluksi halutun tietokannan, kuten edellä tehtiin. Jos käytössä on SQL Server autentikointi Windows autentikoinnin sijaan, kuten tässä tapauksessa on, tulee syöttää SQL Serverin käyttäjätunnus ja salasana.



Kuva 16: Siirrettävän tiedoston sijoituspaikan ja autentikointitavan valinta

Palvelinpohjaista autentikointia kannattaa käyttää esimerkiksi, jos käyttäjä ei ole Windows verkkotunnuksen jäsen, tai jos tietokantaan halutaan muodostaa yhteys laitteelta, joka ei ole Windows-pohjainen. Lisäksi se mahdollistaa sovellusten jakamisen käyttämällä monimutkaista, esiasetettuihin SQL Server-kirjautumisiin perustuvaa käyttöoikeushierarkiaa. (Microsoft, 2023c)

Seuraavassa ikkunassa valitaan taulut, jotka on tarkoitus importoida lähteestä. Tässä vaiheessa kannattaa tarkistaa ’Edit Mappings’ -painikkeesta, että data näyttää oikealta, ja että datatyypit sekä taulujen nimet ovat halutussa muodossa. Lisäksi määritetään taulut, joissa sallitaan tyhjäarvo Null (Kuva 17).



Kuva 17: Taulun sarakkeiden kartoitus (mapping)

4.2.1 Datan eheyden varmistaminen

Kuten teoriaosuudessa todettiin, datan eheys on tärkeää relaatiotietokannoissa monesta syystä. Tässä konfiguraatiossa datan siirtäminen SQL Serverille Access-tiedostoista SSMS Import and Export Wizardilla ei itse tietokantataulujen siirron yhteydessä siirtänyt samalla eheyssääntöjä, kuten avaimia. Selvittelyjen perusteella tämä on normaalia, koska wizard -työkalu on tarkoitettu pääasiassa datan siirtämiseen ilman tietokantaskeemaa. Eheyssääntöjen säilyttämiseksi siirron yhteydessä prosessissa voi vaihtoehtoisesti SQL Server Migration Assistant (SSMA) -työkalua, mutta tässä tapauksessa palvelimille ei haluttu asentaa ylimääräisiä ohjelmia, joten sitä ei käytetty.

Pelkkää dataa siirtäessä paras menetelmä eheyssääntöjen täytäntöönpanoon SQL Server-tietokannassa on luoda tauluille skeema SQL Serverissä, ja määrittää avaimet ja muut tarvittavat säännöt skeemalle ennen datan siirtoa. Näin toimien, ja samalla varmistaen, että siirtovaiheessa datatyyppien ja taulujen kartoitus tehdään oikein, eheyssääntöjen pitäisi olla ilman erillisiä virheilmoituksia kunnossa, kun skeemaan siirretään uutta dataa.

Tässä toteutuksessa suurin osa datasta siirrettiin SQL Serverille kuitenkin jo ennen skeeman luomista, jolloin eheyssääntöjä tuli määrittää ja ottaa käyttöön tauluihin siirron jälkeen. Menetelmässä on omat huonot puolensa, dataan saattaa esimerkiksi tulla duplikaattirivejä tai muita virheitä siirron yhteydessä. Tämä voi omalta osaltaan hidastaa prosessia ja aiheuttaa lisää työtä, kun virheelliset tiedot on löydettävä suurista tietokannoista jälkikäteen.

Eheyssääntöjen käyttöönotto SQL Serverillä datan siirron jälkeen aloitettiin tutkimalla alkuperäisten Access-tietokantojen skeemoja, huomioiden niihin määritellyt avaimet ja muut säännöt. Isoon osaan alkuperäisistä tietokannoista on piirretty ER-kaaviot, jotka kertovat visuaalisesti tietokantataulujen väliset suhteet ja avaimet.

Pääavain voidaan asettaa SSMS:ssä joko DDL-komennoilla ’ALTER TABLE’ ja ’ADD PRIMARY KEY’. Viiteavainta asettaessa DDL-komentoja käyttäen

tulee viitata siihen toisen taulun pääavaimeen, johon halutaan luoda yhteys lisäämällä komentoon ’REFERENCES’ -kohta ja taulun nimi. Avaimet voi asettaa myös SSMS:n graafisen käyttöliittymän avulla etsimällä haluttu taulu Object Explorerista ja valitsemalla hiiren oikealla klikkaamisen jälkeen avautuvasta valikosta ’Design’.

4.3 Tietokantalinkitys

Kun data on importoitu SQL Serverille, ja halutaan muodostaa tietokantayhteys Accessin ja SQL Serverin välille, aloitetaan linkitys Accessista valitsemalla ’External Data’ -välilehdeltä ’ODBC Database’ (Kuva 18).

Kuva 18: Tietokantalinkityksen aloittaminen Accessista

Seuraavassa ikkunassa valitaan, että halutaan importoinnin sijaan linkittää tauluja datalähteeseen. Työssä päädyttiin käyttämään ODBC:ta sen toimivuuden ja laajan saatavuuden vuoksi, joten seuraavaksi tulee määrittää DSN tiedosto, jota halutaan käyttää tietokantayhteyden muodostamiseksi. Jokaiselle SQL Serverillä olevalle tietokannalle on luotava erillinen tiedosto, jotta yhteys muodostuu oikeaan paikkaan.

Jos lähdetiedosto on luotu, valitaan se listalta ja syötetään SQL Server instanssin kirjautumistiedot. Seuraavaksi pitäisi aueta näkymä, josta voi valita tietokannasta linkitettävät taulut. Taulujen valinnan jälkeen ohjelma kysyy, mitkä sarakkeet asetetaan taulujen yksilöllisiksi tunnisteiksi datan eheyden varmistamiseksi. Tunnisteiksi asetettiin ne sarakkeet, jotka olivat alkuperäisessä Access tietokannassa taulun avaimina, jotta makrojen ja moduulien suorittamisessa tulisi mahdollisimman vähän häiriöitä.

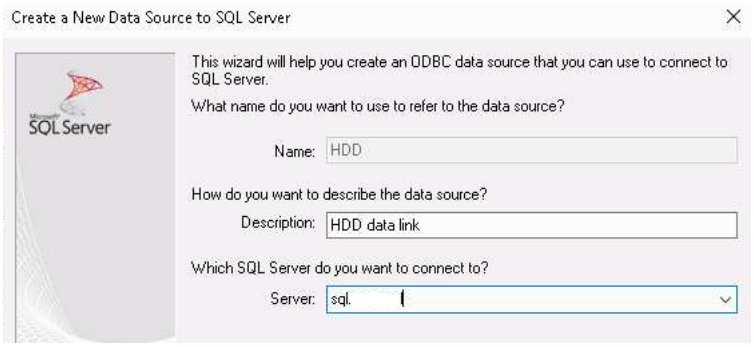
Kun kaikkien taulujen tunnistesarakkeet on määritetty, linkitys on valmis. Onnistuneesti linkitetty taulu on merkitty Accessin Object Explorerissa pallon ja nuolen kuvalla. Access nimeää linkitetyt taulu skeeman mukaisesti esimerkiksi ’dbo_’-alkuisesti, jolloin Accessiin tehty koodi ei toimi nimien muuttumisen johdosta. Koska linkitettäviä tauluja on niin paljon, taulujen nimien manuaalinen korjaaminen vie paljon aikaa. Tämä on kuitenkin mahdollista tehdä automaattisesti kaikille tauluille kuvan 19 mukaisella yksinkertaisella Visual Basic-koodilla, vaihtamalla dbo:n tilalle oikea skeeman nimi.

```
Sub UpdateTableNames()
Dim tbl As TableDef
For Each tbl In CurrentDb.TableDefs
    If Left(tbl.Name, 4) = "dbo_" Then
        tbl.Name = Mid(tbl.Name, 5)
    End If
Next tbl
End Sub
```

Kuva 19: VBA-koodi, joka poistaa kaikista tauluista etuliitteen 'dbo_'

4.3.1 Lähdetiedoston luominen

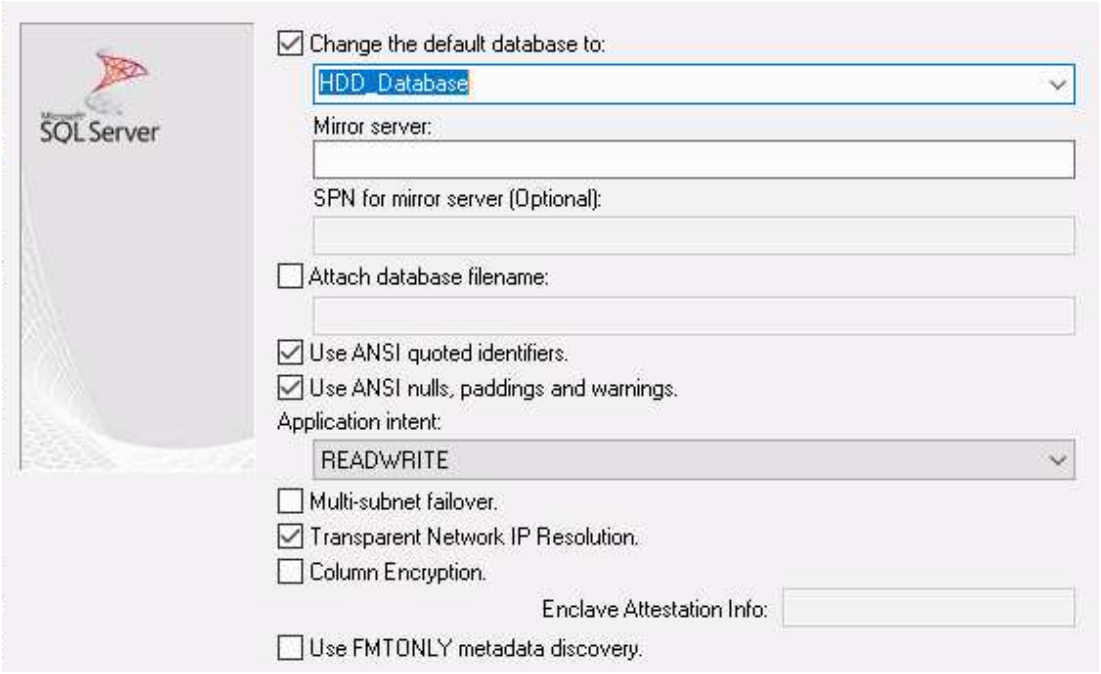
Uuden tiedoston luomista aloittaessa valitaan ’New…’. SQL Serverille ja työasemalle, jolla Access käyttöliittymiä ajetaan, tulee molemmille olla asennettu ’ODBC Driver 17 for SQL Server’ -ajuri, joka valitaan käytettäväksi linkityksessä seuraavasta näkymästä. DSN tiedostoa luodessa sille kannattaa antaa tulevien linkitysten helpottamiseksi kuvaava nimi, esimerkiksi tietokannan nimi. Lisäksi nimitysikkunassa tulee määrittää SQL Server instanssin nimi (Kuva 20).



Kuva 20: Datalähteen nimitys ja yhdistettävän SQL Serverin määrittäminen

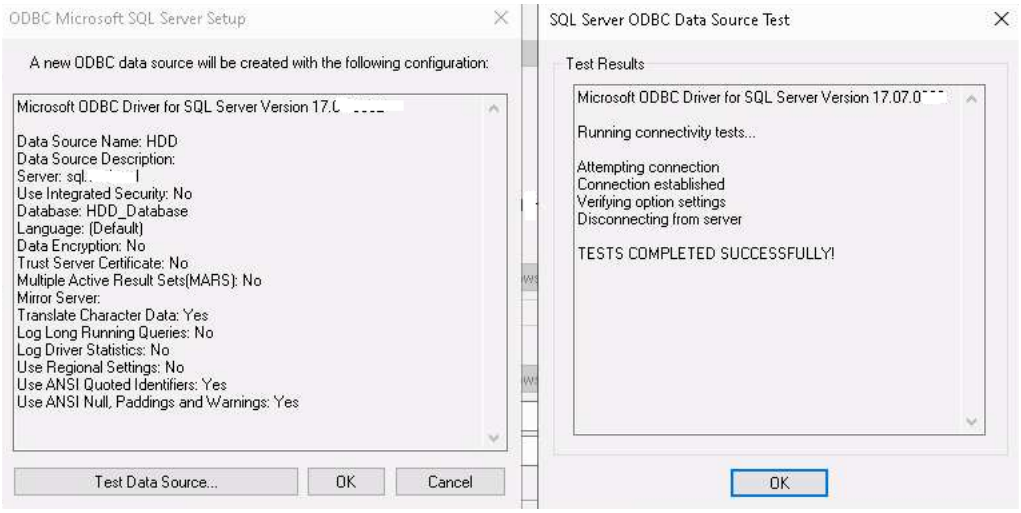
Seuraavassa ikkunassa valitaan käytössä oleva autentikointitapa, tässä tapauksessa SQL Server autentikointi, ja syötetään SQL Serverin käyttäjätunnus ja salasana, joita ajuri käyttää tietokantayhteyden luomiseen samalla tavalla kuin importointivaiheessa.

Autentikoinnin määrittämisen jälkeisessä ikkunassa on muutamia tärkeitä kohtia, jotka tulee huomioida. Ensimmäisessä ruudussa laitetaan merkki kohtaan ’Change default database name to:’, jolloin tekstilaatikkoa klikkaamalla pitäisi aueta alasvetovalikko, josta valitaan linkitettävä tietokanta. Jos valikon sijasta tulee virheilmoitus, tämä tarkoittaa, että joko SQL Server instanssin nimi ja/tai autentikointitiedot ovat väärin, tai valittu ajuri ei ole yhteensopiva/asennettuna. Toisessa laatikossa ’Mirror Server’ voi määrittää toisiopalvelimen, jos sellainen on käytössä. Toisiopalvelimella voidaan esimerkiksi lyhentää latausaikoja ja tasata kuormitusta, mutta tässä työssä sitä ei käytetty. ’Set ANSI quoted identifiers’ -laatikkoon rastin laittamalla SQL Server noudattaa ANSI-sääntöjä merkeissä, eli lainausmerkkejä voidaan käyttää ainoastaan tunnisteissa, kuten sarakkeiden ja taulujen nimissä, kun taas merkkijonot tulee olla heittomerkeissä. ’ Use ANSI nulls, paddings, and warnings’ -kohdan rastittamalla voi ottaa lisää ANSI syntaksin mukaisia sääntöjä ja virhevaroituksia käyttöön. ’Application intent’ -kohtaan tulee valinta sen mukaan, halutaanko linkitettyjä tauluja ainoastaan lukea vai myös muokata. Muissa kohdissa voi määrittää vikasietoisuutta, sarakkeiden salausta ja väliaikaistaulujen ominaisuuksia (Kuva 21).



Kuva 21: Linkityksen ominaisuuksien määrittely

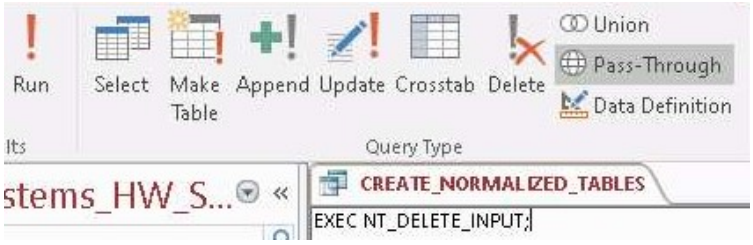
Seuraavassa ikkunassa määritetään SQL Server viestien kieli, käännetäänkö ANSI merkkijonot Unicode merkistöstandardin mukaan, aluekohtaisten asetusten käyttö ja lokitiedostojen tallennus. Painamalla ’Finish’ seuraa yhteenveto valinnoista, ja yhteyden testaus (Kuva 22).



Kuva 22: Linkityksen yhteenveto ja validointi

4.3.2 Läpivientikyselyn teko

Tietokannoista osassa on paljon prosessointia vaativia kyselyitä, joissa niiden ohjelmallinen suorittaminen SQL Serverin puolella on kannattavaa tehokkuuden vuoksi. Esimerkiksi I&C Application Function -tietokannassa luodaan useita normalisoituja tauluja Visual Basic koodiin upotetuilla Access SQL lauseilla. Prosessoinnin suorittamiseksi SQL Serverillä lauseet oli ensin muunnettava syntaksiltaan standardi SQL:n mukaiseksi. Muunnetusta koodista tehtiin proseduuri, joka voidaan kutsua läpivientikyselynä Accessista EXEC-komennolla. Kuvan 23 proseduurissa käännettiin erään tietokannan datan normalisoivat lauseet Access SQL:stä standardi SQL:ään, luotiin proseduuri SSMS:ssä. Proseduurin voi kutsua sen tietokannan Access-käyttöliittymän Object Explorerista, jolloin proseduuri normalisoi taulut SQL Serverissä.



Kuva 23: Proseduurin suoritus läpivientinä Accessista

Läpivientikyselyn teko aloitetaan valitsemalla ’Create’-välilehdeltä ’Query Design’, jolloin aukeavaan kenttään kirjoitetaan standardi SQL:n mukainen koodi, joka halutaan lähettää SQL Serverille prosessoitavaksi. Lisäksi tulee valita ’Property Sheet’ -välilehdeltä käytettävä datanlähde, tässä tapauksessa halutuun tietokantaan yhdistävä DSN-tiedosto. Tämän jälkeen painamalla ’Run’ läpivientikyselyn suoritus alkaa, ja ilman virheilmoituksia kysely tuli suoritetuksi onnistuneesti SQL Serverillä.

4.4 SQL Server muutosten seuranta

Toiveena oli, että SQL Serverillä olisi tietokannoissa mahdollisuus muutosten seurantaan tietokantojen helpompaan ja tehokkaampaan hallinnointiin. Change Data Capture saatiin toimimaan palvelimella, mutta koska dataa on

niin paljon, CDC:n laajempi käyttö voisi hidastaa sen toimintaa huomattavasti. Tästä johtuen päädyttiin ottamaan laajemmin käyttöön Change Tracking.

CT:n käyttöönotossa on ensin otettava se käyttöön tietokantatasolla, ja tämän jälkeen halutussa taulussa. Tämän jälkeen luotiin konfiguraatiotaulu, joka tallentaa viimeisen synkronoinnin version, jotta muutosten seuranta on järkevämpää ja tehokkaampaa. Tauluun lisättiin rivi, johon tulee alkuperäinen synkronoinnin versio tietystä taulusta (Kuva 24).

```
--CT kayttoon tietokannassa
ALTER DATABASE <tracked_database_name>
SET CHANGE_TRACKING = ON
(CHANGE_RETENTION = 2 DAYS, AUTO_CLEANUP = ON);
--CT kayttoon taulussa
ALTER TABLE <tracked_table_name>
ENABLE CHANGE_TRACKING
WITH (TRACK_COLUMNS_UPDATED = ON);
--taulu joka tallentaa CT sync version
CREATE TABLE dbo.ChangeTrackingConfig
(
    TableName NVARCHAR(128) PRIMARY KEY,
    LastSyncVersion BIGINT
);
--lisaa tauluun sync versio
INSERT INTO dbo.ChangeTrackingConfig (TableName, LastSyncVersion)
VALUES (N'<tracked_table_name>', NULL);
```

Kuva 24: CT:n alustus

CT:n alustustoimenpiteiden jälkeen luotiin proseduuri, jolle voidaan sitä kutsuessa antaa parametrina taulun nimi, josta muutoksia halutaan selville (Kuva 25).

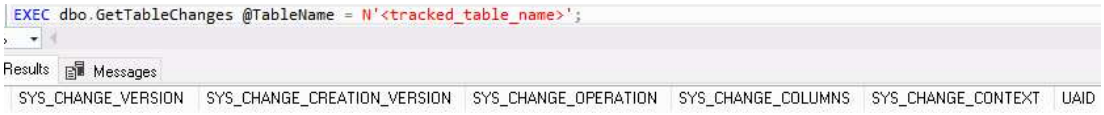
```sql
CREATE PROCEDURE dbo.GetTableChanges
    @TableName NVARCHAR(128)
AS
BEGIN
    -- Aseta nykyinen minimi CT versio
    DECLARE @current_version BIGINT;
    DECLARE @min_valid_version BIGINT;
    SET @current_version = CHANGE_TRACKING_CURRENT_VERSION();
    SET @min_valid_version = CHANGE_TRACKING_MIN_VALID_VERSION(OBJECT_ID(@TableName));

    -- Hae viimeinen tallennettu sync versio
    DECLARE @last_synchronization_version BIGINT;
    SELECT @last_synchronization_version = LastSyncVersion
    FROM dbo.ChangeTrackingConfig
    WHERE TableName = @TableName;

    -- Jos viimeinen sync versio on NULL tai pienempi kuin nykyinen min, kayta minimi
    IF (@last_synchronization_version IS NULL) OR (@last_synchronization_version < @min_valid_version)
    BEGIN
        SET @last_synchronization_version = @min_valid_version;
    END

    -- Hae viimeisen sync version jalkeiset muutokset
    SELECT *
    FROM CHANGETABLE(CHANGES <tracked_table_name>, @last_synchronization_version) AS CT;

    -- Paitivta sync versio
    UPDATE dbo.ChangeTrackingConfig
    SET LastSyncVersion = @current_version
    WHERE TableName = @TableName;
END;
```

Kuva 25: CT-proseduuri, jolle annetaan parametrina taulun nimi

Proseduurissa alustetaan kaksi muuttujaa, ’@current_version’ ja ’@min_valid_version’ ja alustetaan niiden arvot ’@current_version’ ja ’@min_valid_version’ -funktioilla. ’@current_version’ tallentaa nykyisen CT-version, kun taas ’@min_valid_version’ tallentaa voimassa olevan minimiversion, jota voidaan käyttää muutosten seurantaan. Tämän jälkeen alustetaan muuttuja ’@last_synchronization_version’ tallentamaan viimeisin synkronointiversio ’Chan-geTrackingConfig’ -taulusta, ja haetaan viimeisin versio. Seuraavaksi määritetään, että jos ’@last_synchronization_version’ on NULL tai pienempi kuin ’@min_valid_version’, asetetaan ’@last_synchronization_version’ arvoksi ’@min_valid_version’. Tämä varmistaa, että prosessoitavaksi tulee kelvollisia muutoksia. Viimeiseksi proseduuri hakee muutokset, jotka tauluun on tullut viimeisimmän synkronointiversion jälkeen, ja päivittää ’ChangeTrackingConfig’ -taulun viimeisimmän synkronoinnin arvoon varmistaen, että seuraavalla kerralla proseduuri näyttää viimeiset tauluun tulleet muutokset.

Proseduuria suoritettaessa annetaan parametrina ’@TableName’ eli taulun nimi, josta muutoksia halutaan hakea, ja tulokseksi tulee Kuvan 26 mukaiset

sarakkeet, joissa on muutoksista tietoa. Proseduuria voi hyödyntää kaikkien tietokantojen kanssa, kunhan CT on ensin alustettu kuvan 26 mukaisesti.



Kuva 26: GetTableChanges-proseduurin suoritus ja tulos

5 TULOKSET/YHTEENVETO

Ensisijainen tavoite oli saada Olkiluoto 3 -ydinvoimalaitoksen turvallisuusautomaation tietokannat siirrettyä Accessista SQL Serverille, ja tässä onnistuttiin. Kaikki tietokannat tulivat siirrettyä kappaleessa 4.2 kuvatulla menetelmällä, ja ne ovat hallinnoitavissa SSMS:llä. Lisäksi data on SQL Serverille siirtämisen jälkeen yhtä eheää kuin ennen migraatiota Access-tietokannoissa; taulujen rivimäärät on tarkistettu, ja kaikki täsmää niiden osalta aikaisempaan migraation jälkeen. Myös pää- ja viiteavaimet määritettiin datalle alkuperäisten Access-tiedostojen ER-kaavioiden ja taulurakenteiden mukaan. Tietokantoja, tauluja ja dataa on kuitenkin niin suuria määriä, että datan eheyttä tulee tarkkailla ja tarvittaessa jatkokehittää.

Lisäksi oli tarkoituksena selvittää, onko valmiita Access-pohjaisia työkaluja mahdollista saada toimimaan datan SQL Serverille siirtämisen jälkeen tietokantayhteydellä, ja toteuttaa yhteys. Tämä onnistui joidenkin tietokantojen osalta kokonaan taulujen linkitysten ja läpivientikyselyjen avulla, mutta joidenkin kanssa ainoastaan osittain. Haasteiksi osoittautui muun muassa se, että kaikissa Access-työkaluissa pääsy muokkaamaan ja tarkastelemaan VBA-koodia, jossa datalähde on määritetty ei ole mahdollista. Joidenkin vanhojen Access-pohjaisten käyttöliittymien käyttöä on siis jatkettava ilman yhteyttä SQL Server dataan siihen asti, kun kehitetään kokonaan uusia työkaluja, jos työkalujen kaikkia ominaisuuksia halutaan käyttää. Näiden tietokantojen osalta toinen vaihtoehto on kirjoittaa SQL-kyselyitä, joiden avulla on mahdollista

korvata osa käyttöliittymien toiminnallisuutta. Kyselyjen kirjoittaminen SSMS:ssä on kuitenkin hyödyllisyydeltään rajoitettu esimerkiksi siitä syystä, että jotkut Access-työkalut kykenevät ottamaan suoraan yhteyden Visio-tiedostoihin tietokantojen pohjalta. Vähän käytettyjen ja SQL Server -datan kanssa toimimattomien työkalujen osalta väliaikainen ratkaisu voisi olla datan säilyttäminen SQL:ssä, ja tarvittavan datan kopioiminen Access-työkalun tiedostoon työkalun käyttämiseksi. Ratkaisu on ylimääräistä aikaa vievä, mutta siirto on yksisuuntainen, joten tämä ei aiheuttaisi datan eheyden suhteen vaaraa varsinaiselle tietokannalle SQL Serverin puolella, ja toimisi yksittäisissä käyttötapauksissa.

Opinnäytetyön pohjalta voidaan todeta, että varsinkin suuren yrityksen mittakaavassa, jolla on paljon dataa ja erilaisia työkaluja sen käyttämiseen, datan migraatio on laaja ja melko monimutkainen prosessi, jossa on mietittävä monia eri vaihtoehtoja ja varattava migraatioon tarpeeksi aikaa ja resursseja. Oikean migraatiomenetelmän valinta on ensisijaisen tärkeää ja se olisi hyvä olla tiedossa ennen, kuin datan siirto aloitetaan. Joissain tapauksissa voidaan todeta, ettei migraatiota kannata suorittaa lainkaan, kun taas muissa tapauksissa on kannattavinta suorittaa datalle sen siirron yhteydessä työkalujen täydellinen uudelleentoteutus.

Tässä tapauksessa uudelleentoteutus onkin pitkällä aikavälillä todennäköisesti paras ratkaisu, jota kannattaa harkita tulevaisuudessa. Siihen asti kuitenkin työssä toteutettu välimuoto eli SQL Server -tietokannan ja valmiiden Access-työkalujen yhdistäminen linkitysten avulla on toimiva ratkaisu.

LÄHTEET

Amazon Web Services. (2023). What is SQL? Viitattu https://aws.amazon.com/what-is/sql/

Clark, D. (2021). SQL vs T-SQL: Understanding the Differences. Dataquest. Viitattu 9.11.2023. https://www.dataquest.io/blog/sql-vs-t-sql/

Devart. (n.d.). OLEDB vs ODBC: Which Driver to Choose? Viitattu 9.1.2023. https://blog.devart.com/oledb-vs-odbc-which-driver-to-choose.html

Doorn, J. & Rivero, L. (2001). Database Integrity: Challenges and Solutions. IGI Global.

FMS. (n.d). When and How to Upsize Microsoft Access Databases to SQL Server. Viitattu 11.1.2023. https://www.fmsinc.com/MicrosoftAccess/SQLServerUpsizing/how/index.htm

GeeksforGeeks. (2019). SQL | DDL, DQL, DML, DCL and TCL Commands. Viitattu 13.1.2023. https://www.geeksforgeeks.org/sql-ddl-dql-dml-dcl-tcl-commands/

Peterson, R. (n.d.). What is SQL Server? Introduction, History, Types, Versions. Guru99. Viitattu 25.10. 2023. https://www.guru99.com/sql-server-introduction.html

IBM. (2021). What is a database schema? IBM. Viitattu 2.2.2023. https://www.ibm.com/topics/database-schema#:~:text=the%20next%20step-,What%20is%20a%20database%20schema%3F,the%20relationships%20between%20these%20entities.

Microsoft. (n.d.a). Introduction to Forms. Viitattu 11.1.2023. https://support.microsoft.com/en-us/office/introduction-to-forms-e8d47343-c937-44e8-a80f-b6a83a1fa3ae#:~:text=A%20form%20in%20Access%20is,data%20from%20that%20data%20source.

Microsoft. (n.d.b). Connect Access to SQL Server. Viitattu 1.12.2022. https://support.microsoft.com/en-us/office/connect-access-to-sql-server-050d88f3-b2d6-4e76-b6f9-f3c556f139ea

Microsoft. (n.d.c). What is a DSN? Viitattu 14.2.2023. https://support.microsoft.com/en-us/topic/what-is-a-dsn-data-source-name-ae9a0c76-22fc-8a30-606e-2436fe26e89f

Microsoft. (n.d.d). Import or link to data in an SQL Server database. Viitattu 2.11.2022. https://support.microsoft.com/en-us/office/import-or-link-to-data-in-an-sql-server-database-a5a3b4eb-57b9-45a0-b732-77bc6089b84e

Microsoft. (n.d.e). Create a pass-through query. Viitattu 16.11.2022. https://support.microsoft.com/en-us/office/create-a-pass-through-query-b775ac23-8a6b-49b2-82e2-6dac62532a42

Microsoft. (2023a). What is SQL Server Management Studio (SSMS)? Viitattu 11.11.2023. https://learn.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16

Microsoft. (2023b). Track data changes (SQL Server). Viitattu 11.1.2023. https://learn.microsoft.com/en-us/sql/relational-databases/track-changes/track-data-changes-sql-server?view=sql-server-ver16

Microsoft. (2023c). Choose an Authentication Mode. Viitattu 15.2.2023. https://learn.microsoft.com/en-us/sql/relational-databases/security/choose-an-authentication-mode?view=sql-server-ver16

Ranwa, S. (2022). Data Integrity in SQL Server. C#Corner. Viitattu 15.2.2023. https://www.c-sharpcorner.com/blogs/data-integrityin-sql-server

Sarja, J. (2006). Relaatiotietokanta. Jarin Nexus. https://verkkopedagogi.net/vanhat/fi/sisalto/materiaalit/access2003/luku0375c6.html?C:D=419702&selres=419702

Terra, J. (2023). What is Microsoft Access? An Introductory Guide. Simplilearn. Viitattu 6.2.2023. https://www.simplilearn.com/what-is-microsoft-access-article

TVO. (2022a). Yhtiö. Viitattu 26.9.2022 osoitteesta https://www.tvo.fi/yhtio.html

TVO. (2022b). Ydinvoimalaitosyksikkö Olkiluoto 3. Viitattu 26.10.2022. https://www.tvo.fi/material/collections/20220825132746/7bmHsNHjV/ydinvoimalaitosyksikko_ol3_fin.pdf

What is Database Management System? (n.d.). PrepInsta. https://prepinsta.com/dbms/what-is-database-management-system/

Wiech, A. (2020). What Is a DBMS? LearnSQL. https://learnsql.com/blog/what-is-dbms/