OULUN AMMATTIKORKEAKOULU

Iida Stenius

SYNTEETTINEN DATA

SYNTEETTINEN DATA

Iida Stenius
Opinnäytetyö
Syksy 2023
Tradenomi (AMK), tietojenkäsittely
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tradenomi (AMK), tietojenkäsittely

Tekijä(t): Iida Stenius
Opinnäytetyön nimi: Synteettinen data
Työn ohjaaja(t): Perttu Hietala
Työn valmistumislukukausi ja -vuosi: Syksy 2023 Sivumäärä: 36

Opinnäytetyön projektin tavoitteena oli luoda menetelmä, jonka avulla voidaan siirtää mahdollisimman ketterästi staattista, keinotekoista dataa järjestelmän tietokannasta toiseen. Datan tulisi olla yhteensopivaa ja yleispätevää järjestelmän testaamista varten.

Jotta voidaan taata käyttäjien yksityisyys ja noudattaa tietosuoja-asetuksia, testitarkoituksiin ei voida käyttää olemassa olevaa asiakasdataa. Tämän vuoksi tarvitaan mahdollisimman kattava paketti testidataa, joka on helppo ladata esimerkiksi järjestelmän päivityksen jälkeen.

Opinnäytetyön teoriaosuus kattaa synteettisen datan ominaisuudet, kuten sen hyödyt ja heikkoudet sekä sen yleisimmät käyttökohteet. Painotetaan erityisesti sitä, minkälaista synteettinen data on tietosuojan kannalta ja mitä pitää ottaa tietoturvan näkökulmasta huomioon. Tutkitaan esimerkkien valossa, minkälaisissa tapauksissa synteettistä dataa voidaan hyödyntää ja mitä sen käyttöönotto mahdollistaa yritykselle. Lopuksi esitellään opinnäytetyön projekti, joka tehty toimeksiantajalle, johon viitataan nimellä Yritys Y.

Toteutusosiossa käydään tarkemmin läpi projektin vaiheet. Tähän sisältyy muun muassa vaatimusmäärittely, joka on räätälöity sopimaan yrityksen tarpeisiin. Käydään läpi toteutusmenetelmä, jossa perehdytään tarkemmin tietokantoihin ja PL/SQL-kyselykieleen, jolla projektin skriptit on toteutettu. Näiden skriptien avulla mahdollistetaan keinotekoisen, staattisen datan ketterä siirto tietokannasta toiseen. Osiossa käsitellään myös testaamista.

Opinnäytetyön lopussa reflektoidaan, kuinka menetelmää voisi vielä parantaa ja miten sitä voisi soveltaa muihin käyttötarkoituksiin. Samalla pohditaan myös synteettisen datan tulevaisuutta ja miltä sen kehitys näyttää.

Asiasanat: Synteettinen data, staattinen data, testidata.

ABSTRACT

Oulu University of Applied Sciences
Degree Programme in Business Information Systems

Author(s): Iida-Maria Regina Stenius
Title of thesis: Synthetic data
Supervisor(s): Perttu Hietala
Term and year when the thesis was submitted: Fall 2023
Number of pages: 36

The thesis analyzes the purpose and attributes of synthetic data, for example, its benefits and weaknesses. The main point of view is data protection and how a company can best utilize synthetic data. Synthetic data is very topical from this perspective because it could potentially solve many issues that have emerged with GDPR and the overall concern surrounding the matters of data protection. The instances described study mainly what kind of opportunities synthetic data will present and what kind of issues must be taken into consideration when the data is implemented. At best, synthetic data can save time and resources.

In the project described in the thesis, the idea was to create a method that could be utilized in implementing synthetic data from one environment to another. It was necessary that this method was quick, easy, and could be utilized especially in testing. The method was created at a database level and achieved via database scripts written in PL/SQL. The data in question had to essentially be selected, validated, and packed into a form, where it could be easily downloaded into any environment. The whole process had to be automatized, so it would be as agile as possible.

At the end of the thesis, the result of the project is described and analyzed by comparing the result to what has previously been stated about synthetic data. The future of synthetic data is also discussed.

Keywords: Synthetic data, Static data, test data

SISÄLLYS

1 JOHDANTO

Opinnäytetyö käsittelee synteettistä dataa ja sen lukuisia käyttömahdollisuuksia esimerkkiprojektin valossa. Työ on tehty yhteistyössä anonyymina pysyvän toimeksiantajan kanssa. Projektin ideana on mahdollistaa testaaminen staattisen, keinotekoisen datan avulla, jotta testaamisesta saadaan mahdollisimman nopea, resursseja säästävä ja helppokäyttöinen prosessi.

Opinnäytetyö on kirjoitettu pitkälti yrityksen näkökulmasta, ja painopiste on erityisesti synteettisen datan hyötyjen ja haasteiden kohdalla. Synteettinen data on uusi, edelleen kehittyvä ala, jonka vuoksi tutkitaan erityisesti sen potentiaalia. Projektin tavoite on helpottaa testaajien työtä ja mahdollistaa datan siirtäminen ympäristöstä toiseen ketterästi. Näin yritys säästää resursseja ja uusien ominaisuuksien testaaminen helpottuu merkittävästi. Synteettisen datan käytöstä voidaan tehdä rutiininomaista ja käytännöllistä.

Opinnäytetyön teoriaosuudessa käydään läpi synteettisen datan ominaisuuksia sekä sen hyötyjä ja heikkouksia. Erityistä huomiota keskitetään yksityisyydensuojan ajankohtaisiin kysymyksiin ja siihen, kuinka synteettinen data voi tarjota tietoturvaratkaisuja tällaisiin ongelmiin. Osiossa tarkastellaan, minkälaisia asioita yrityksen tulee ottaa huomioon, kun keinotekoista dataa halutaan implementoida.

Toteutusosiossa perehdytään tarkemmin opinnäytetyön projektin kulkuun, kehittämiseen ja lopussa tarkastellaan työn tulosta. Osiossa esitellään käytetyt työkalut ja mitä ohjelmointikieliä työssä on käytetty. Käydään läpi, kuinka metodi lopulta käytännössä toteutettiin ja minkälaista tietokanta-osaamista se vaati. Osiossa tutkitaan myös tietokantaskriptin ominaisuuksia ja kuinka testaaminen tapahtui. Projektin tulos esitellään tietokantaesimerkin kautta.

Koska synteettinen data on uusi, jatkuvasti kehittyvä aihe, on olennaista varautua muutoksiin. Opinnäytetyön lopussa analysoidaan ja pohditaan mahdollisia kehitysideoita projektin kannalta, mutta myös pohditaan synteettisen datan tulevaisuutta.

2 KEINOTEKOINEN DATA

Tietosuoja-asetukset rajoittavat nykypäivänä datan käyttöä ja jatkuvasti kiinnitetään enemmän huomiota siihen, kuinka arkaluontoista dataa voidaan suojella. Näin ollen usealla taholla on muodostunut kasvava tarve saada käyttöön tietosuoja-asetuksia noudattavaa dataa, jota kuitenkin voitaisiin hyödyntää samoihin käyttötarkoituksiin kuin oikeaakin dataa. Vaihtoehtona tähän on keinotekoinen data.

2.1 Synteettinen data

Tavallisesti data on lähtöisin autenttisesta lähteestä, kuten esimerkiksi potilastiedoista, jolloin data on kerätty talteen erilaisten prosessien avulla. Yksinkertaisimmillaan synteettinen data tarkoittaa keinotekoista dataa, joka ei sisällä vuorovaikutusta oikean lähteen kanssa, vaan se luodaan tietokonealgoritmin avulla. Kun synteettistä dataa luodaan, usein tavoitteena on generoida dataa, joka vastaa ominaisuuksiltaan ja suhteiltaan alkuperäistä niin hyvin, että sitä voidaan hyödyntää samaan käyttötarkoitukseen. Synteettisen datan avulla voidaan tutkia lähdedatan ominaisuuksia vaarantamatta tietosuojaa. Itse datan tuotto on Honkelan (2023) mukaan rekisteritietojen haltijoiden vastuulla.

Devaux (2022) jakaa synteettisen datan seuraavalla tavalla erilaisiin datatyyppeihin, joilla on eri käyttötarkoituksia. **Synteettinen teksti** on mallin tuottamaa keinotekoista tekstiä. Tekstin tuoton haasteellisuus riippuu täysin siitä, kuinka monimutkainen valittu kieli on. **Synteettiset kuvat ja videot** ovat keinotekoisesti luotuja mediatiedostoja, jotka voivat sisältää myös ääniraidan. Tällaista dataa voidaan käyttää korvaamaan alkuperäinen data. Keinotekoiset kuvat ja videot voivat olla hyödyllisiä esimerkiksi koulutustarkoituksissa, jos oikeiden mediatiedostojen käyttö ei ole vaihtoehto. **Taulukkomuotoinen synteettinen data** sen sijaan mallintaa reaalimaailman dataa tauluihin säilöttynä, esimerkiksi potilastietoja. Esimerkiksi Amazon käyttää synteettistä dataa virtuaaliavustaja Alexan kehitykseen.

Kiinnostus synteettisen dataan on kasvanut merkittävästi kuluneiden vuosien aikana. ”Yksityisyyttä varjelevan, synteettisen datan luonti on aktiivinen tutkimuksen kohde ja hyvälaatuinen synteettinen

data voisi tulevaisuudessa mahdollistaa sellaisten arvokkaiden tietoaineistojen käytön, joita ei tällä hetkellä voida tietosuojariskien vuoksi käyttää" (Nieminen & Virkki 2022). Yksityishenkilöistä kerätään enemmän ja enemmän tietoja, ja näin ollen myös ajankohtaiset tietosuojakysymykset muuttuvat jatkuvasti. Synteettinen data pyrkii eliminoimaan nämä ongelmat ja mahdollistaa datan käyttämisen. Tämä ilmiö koskee niin tieteellistä näkökulmaa kuin liiketoiminnallista näkökulmaa. Esimerkiksi asiakaskäyttäytyminen ja potilastiedot ovat iso mielenkiinnon kohde.

Synteettinen data tarjoaa ratkaisuja yksityisyydensuojan ongelmiin, sillä datan **anynomisointi** eli henkilötietojen muokkaaminen tunnistamattomaan tilaan, herättää usein kysymyksiä ja kuluttaa resursseja. Tämä näkökulma on ajankohtainen erityisesti yrityksille, kun pohditaan, kuinka esimerkiksi asiakkaiden yksityistiedot saadaan suojattua. Puutteellinen anonymisointi voi pahimmillaan johtaa siihen, että yksityishenkilö voidaan edelleen tunnistaa datasta. "Monella alalla omaa dataa halutaan suojella viimeiseen saakka, etteivät yrityssalaisuudet ja keksinnöt paljastu kilpailijoille." (Tuomisto 2020)

2.2 Staattinen data

Techopedian (2018) määritelmän mukaan staattisella datalla tarkoitetaan dataa, joka on muuttumattomassa tilassa. Sen arvo ei muutu enää tallennuksen jälkeen. Staattista dataa voidaan käyttää järjestelmissä esimerkiksi referoimaan tiettyä, muuttumatonta arvoa. Vastakohtana staattiselle datalle on **dynaaminen data**, joka vaatii jatkuvaa päivittämistä ja ylläpitoa. Se on suorassa vuorovaikutuksessa järjestelmän kanssa. Ongelmanratkaisu ja datan käsittely on staattisen datan kohdalla haastavampaa ja hitaampi prosessi, kuin dynaamista dataa käsiteltäessä. Koska staattista dataa ei päivitetä ja ylläpidetä samalla tavalla kuin dynaamista dataa, on järjestelmän tietoturvan oltava ajan tasalla.

Staattinen data voi olla keinotekoista. Synteettisesti luodulle staattiselle datalle on käyttöä erityisesti testaustarkoituksissa. Se on hyvä testausmetodi tapauksissa, joissa dataa ei käyttäjän tarvitse itse muokata tai tallentaa. Tällaisissa tilanteissa testausprosessi ja sen vuorovaikutukset käsittelevät staattista dataa sen pysyvässä tilassa. Kun myöhemmin käydään läpi opinnäytetyön projektia, on otettava huomioon, että se on pitkälti toteutettu käyttäen staattista dataa.

2.3 Synteettisen datan hyödyt ja heikkoudet

Synteettisen datan huomattavimpiin etuihin kuuluu sen nopea tuottaminen. Sen generointi on merkittävästi nopeampaa kuin datan hankkiminen todellisten vuorovaikutusten kautta. Nopeus tarkoittaa, että resursseja kuluu vähemmän. ”Toinen synteettisen datan positiivinen ominaisuus on se, että varsinaiseen datan keräämiseen ei mene resursseja. Esimerkiksi kattavan puhelinhaastattelun järjestäminen voi helposti maksaa kymmeniä tuhansia euroja.” (Väisänen 2022)

Uusien menetelmien testaaminen tulee synteettisen datan avulla merkittävästi halvemmaksi ja tuotteliaammaksi, eikä riskinä ole testidatan loppuminen. Synteettisen datan luonti voidaan toistaa samalla prosessilla niin monta kertaa kuin on tarpeellista. Lamberti (2023) huomauttaa, että keskiarvoisesti datan keräämiseen ja käsittelyyn menee noin 60 % projektiin käytetystä ajasta, jolloin itse datan analysointiin jää vähemmän aikaa.

Data voidaan räätälöidä eri algoritmeilla sopimaan kohteen tarpeisiin, eikä näin ollen ole merkitystä sillä, onko kyseessä jokapäiväinen tilanne vai satunnainen tapahtuma, josta tarvitaan synteettistä dataa. Sen avulla voidaan varautua tilanteisiin, joista voisi normaalissa tapauksessa olla lähes mahdoton kerätä dataa. (Lamberti 2023)

Synteettistä dataa voidaan käyttää monella tavalla ja ala on jatkuvassa kehityksessä. Tieteen kontekstissa datan avulla voidaan testata eri teorioita ilman riskiä, että alkuperäinen data vahingoittuu tai katoaa. Tällaisissa tapauksissa dataa ja sen tuomia tuloksia on turvallista tutkia perusteellisesti ennen niiden soveltamista käytäntöön. Muissa tapauksissa kyseinen prosessi voi olla mahdoton, jos olemassa oleva data on puutteellista tai korvaamatonta. Voi myös olla, että oikean datan hankkimiseen ei yksinkertaisesti ole resursseja (Nieminen & Virkki 2022).

Kun synteettistä dataa ryhdytään generoimaan, on huomattava se, että vaikka generointi tapahtuu perustuen oikeaan dataan ja sen malleihin, ei keinotekoinen data tule olemaan täysin samanlaista (Lamberti 2023). Tarkkuudessa voi siis olla muutoksia ja se ei ole välttämättä yhtä eheää kuin alkuperäinen data. Tämä on tärkeä ottaa huomioon ennen kuin dataa lähdetään generoimaan, etenkin jos tarkoituksena on mallintaa dataa hyvin tarkasti. Muutoksia voi tapahtua ja niiden ennustaminen etukäteen on haastavaa.

Yksi synteettisen datan heikkouksista onkin sen vaihteleva laatu. Lamberti (2023) toteaa, että synteettinen data ei aina sovellu kuvaamaan reaalimaailmaa. Generoidun datan laatu riippuu aina lähdedatasta, jonka perusteella se on mallinnettu, ja tämän vuoksi sen validointi voi olla vaikeaa. Jos alkuperäisessä datassa on virheitä tai epäkohtia, ne tulevat esiintymään myös keinotekoisessa datassa. Pahimmassa tapauksessa epäkohdat aiheuttavat sen, että synteettinen data ei enää vastaa haluttua lopputulosta. Euroopan tietosuojavaltuutettu (EDPS) (2021) pitääkin synteettisen datan generoinnin tuloksia hyvin vaihtelevina.

Kun synteettinen data on generoitu, sitä on haastavaa sovittaa vaihteleviin olosuhteisiin. Jos lähdedata ei syystä tai toisesta vastaa reaalimaailman tapauksia, ei synteettinen data tule myöskään toimimaan halutulla tavalla reaalimaailman ympäristössä, eikä synteettinen data muutu reaalimaailman mukana (Lamberti 2023). Tällainen tilanne voi nousta erityisesti tapauksissa, joissa lähdedata on puutteellista tai ei tarjoa dataa juuri siihen ongelmaan, jota lähdetään ratkaisemaan.

Heikkouksistaan ja tilannekohtaisuudestaan huolimatta voidaan todeta, että synteettisen datan avaamat mahdollisuudet ja vahvuudet ovat merkittävä etu. Synteettinen data tarjoaa avainratkaisuja nykyhetken ongelmiin resursseja säästävällä tavalla. Tietosuojaa koskevat huolenaiheet ovat erityisen olennaisia, kun käsitellään kerättyä dataa. Arkaluontoista dataa, kuten esimerkiksi potilastietoja, ei voi käyttää turvallisesti, tietosuojasäädöksiä noudattaen. Synteettinen data tarjoaa ratkaisun useaan tietosuojakysymykseen, sillä se säilyttää oikean datan ominaisuudet.

2.4 Yksityisyydensuoja

Kaikissa EU-maissa otettiin käyttöön GDPR (General Data Protection Regulation) eli yleinen tietosuoja-asetus vuonna 2018. Tietosuojavaltuutetun toimiston (2023) mukaan tarkoituksena on parantaa henkilötietojen suojaa, vastata uusiin tietosuojakysymyksiin ja yhtenäistää tietosuojasäädöksiä EU-maiden sisällä. Näin ollen voidaan varmistaa, että yritykset käsittelevät henkilötietojen suojaamista laillisin perustein ja että jokaisella yksityishenkilöllä on oikeus tietosuojaan. Niin pitkään kuin tiedoista voi tunnistaa henkilön suoraan tai tiedot voidaan palauttaa tunnistettavaan muotoon, nähdään ne yleisen tietosuoja-asetuksen alla henkilötietoina.

Henkilötiedot voidaan käsitellä, jotta ne eivät enää ole henkilötietoja. Tiedot voidaan **anonymisoida** eli käsitellä muotoon, joista henkilöä ei enää tunnista. On huomioitava, että prosessi voi heikentyä ajan kuluessa. **Pseudonymisointi** sen sijaan varmistaa, että henkilötietoja ei voi enää yhdistää tiettyyn henkilöön, mutta ne nähdään edelleen henkilötietoina. Tätä menetelmää käytetään Tietosuojavaltuutetun toimiston (2023) mukaan esimerkiksi tilastoinnissa.

Euroopan tietosuojavaltuutettu (EDPS) toteaa, että anonymisointi ei ole välttämättä lopullinen ratkaisu, sillä aina on olemassa riski, että data saadaan palautettua sen alkuperäiseen muotoon. Tätä kutsutaan termillä **deanonymisointi**. Tietoarkisto (2023) tiivistää, että tällaisessa tilanteessa datan anonymisointi on epäonnistunut, teknologia on kehittynyt, tai dataan on päästy käsiksi toisesta lähteestä. Anonymisoinnin osalta synteettisen datan kanssa on havaittu omat haasteensa. Honkelan (2023) mukaan synteettisen datan tulee muistuttaa riittävästi alkuperäistä dataa, jotta sillä voidaan korvata alkuperäisen datan käyttö. ”Miten yhdistetään alkuperäisen kaltaisuus ja anonymiteetti? Tähän tutkijat etsivät nyt ratkaisuja” (Honkela 2023).

Keinotekoista dataa on viime vuosina pyritty hyödyntämään yksityisyydensuojaan liittyvien kysymysten ratkaisussa. Euroopan Tietosuojavaltuutettu (2021) toteaa, että synteettinen data on aina syytä arvioida, ettei se sisällä henkilökohtaista dataa. Arvio paljastaa, voiko datan joukosta paljastua yksityishenkilöön viittaavia tekijöitä. Jos vertaillaan keinotekoista dataa ja datan anonymisointia, on todettu, että riskit keinotekoisen datan kohdalla ovat pienemmät (Bamford 2021).

3 SYNTEETTINEN DATA KÄYTÄNNÖSSÄ

3.1 Käyttökohteet

Kuten aikaisemmin on jo todettu, yksityisyydensuoja on ollut esillä viime vuosina ja luonnollisesti huomio kiinnittyy nimenomaan henkilötietoihin. Ne ovat arvokkaita monelle taholle. Vuonna 2018 Psykoterapiakeskus Vastaamo hakkeroitiin ja valtava määrä henkilötietoja pääsi vuotamaan (Rautio 2021). Vastaamo haettiin konkurssiin vuonna 2021, joten voidaan todeta, että vahingot olivat merkittävät. Datan keräyksessä on aina riskinsä, etenkin jos kyseessä oleva data sisältää arkaluontoista tietoa.

Synteettisen datan on ajateltu tarjoavan merkittäviä ratkaisuja tietoturvan alalla, mutta se on erityisen huomion kohteena myös tekoälyn kehityksessä. Generatiiviset tekoälymallit ovat herättäneet keskustelua. Niillä tarkoitetaan ohjelmistoja, jotka keskustelevat luonnollisella kielellä ja kykenevät tuottamaan uutta sisältöä, kuten esimerkiksi tekstiä (Heinäsenaho ym. 2023). Tällainen ohjelma voi esimerkiksi olla chattibotti, joka on vuorovaikutuksessa sovelluksen käyttäjän kanssa. Käyttäjä lähettää robotille viestin, ja robotti lähettää vastauksen takaisin. Vastaavia sovelluksia voidaan käyttää esimerkiksi asiakaspalvelussa.

Tekoälystä uutisoidaan jatkuvasti, ja yhä useampi sovellus on alkanut keräämään dataa tekoälyn opetukseen. Esimerkiksi sosiaalisen median sovellus X, entiseltä nimeltään Twitter, on päivittänyt tietosuojakäytäntönsä ja ilmoittanut, että julkista dataa voidaan käyttää tekoälyn opetukseen (Bonk 2023). Vastaavat muutokset herättävät keskustelua muidenkin sovellusten käyttäjien keskuudessa. Käyttäjiä kiinnostavat erityisesti eettiset kysymykset, joista päällimmäisenä se, onko heistä hyväksyttävää luovuttaa omia tietojaan tekoälyn kehitykseen. Synteettisen datan tapauksessa sama kysymys on paljon yksinkertaisempi.

”Data on tekoälyn elinehto. Oikeanlaisen datan löytäminen on kaikkein tärkein ja myös haastavin osuus, kun rakennetaan toimivaa tekoälyä” (Toews 2022). Kuten aikaisemmin todettu, synteettinen data on mahdollista generoida nopeasti ja edullisesti. Tämän vuoksi sitä on pidetty erinomaisena keinona opettaa tekoälyä ilman, että käytetään oikeaa henkilödataa. Martineau (2022) kertoo, että

synteettinen data on erinomainen työkalu tekoälyn opetuksessa ja se voi olla apuna luomassa tekoälystä täsmällistä ja luotettavaa. Synteettinen data on hyödyllistä myös tekoälyn testauksessa ja sen avulla saadaan selville tekoälyn virheitä ja epäkohtia.

Kun ajatellaan opinnäytetyön projektia, synteettistä dataa hyödynnetään testaustarkoituksiin. Jos testattavassa ympäristössä on ominaisuuksia, joiden käsittelyssä oleva data voi reaalimaailmassa olla arkaluontoista, niiden testaaminen on hoidettu generoimalla keinotekoista dataa vastaamaan järjestelmän tarpeita. Kuten Väisänen (2022) kertoo, synteettisen datan tapauksessa ei ole huolta datan tunnistettavuudesta.

On kuitenkin pidettävä mielessä, että synteettinen data ja erityisesti tekoälyn kehitys on vielä varsin uutta. ”Vaikka tällä hetkellä kehitys näyttää lupaavalta, edessä voi olla mahdollisesti hyvin haastavia teknisiä, lainsäädännöllisiä ja eettisiä ongelmia” (Heinäsenaho ym. 2023).

3.2 Toimeksiantajan esittely

Toimeksiantajan pyynnöstä yritys on anonymisoitu, ja tässä työssä yritykseen viitataan nimellä Yritys Y. Yrityksellä on erilaisia liiketoimia. Projektin kannalta on huomioitava, että liiketoimilla on erilaisia tarpeita ja palveluita on räätälöity vastaamaan näitä vaatimuksia. Painopisteenä on erityisesti asiakkaan tietojen suojaaminen. Ratkaisut ovat nykyaikaisia ja kehitys on jatkuvaa.

Järjestelmä, johon synteettinen data implementoidaan, on iso ja monimoduulinen tiedonhallintajärjestelmä. Järjestelmään on kerätty merkittävä määrä dataa eri moduulien välille ja relaatiotietokantojen rivimäärät ovat massiivisia. Järjestelmässä on ympäristöjä testaamista varten, joissa tarkistetaan uusien ominaisuuksien toiminta tai tutkitaan uusien ratkaisujen mahdollisuuksia. Testaaminen on jatkuvaa ja sitä tapahtuu monella saralla. Näin ollen saatavilla on oltava testaamisen soveltuvaa dataa.

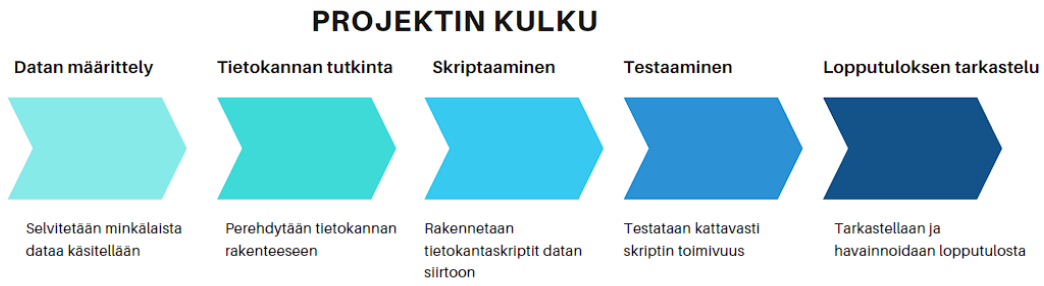
3.3 Projektin tutkimusongelman esittely

Tietosuojakysymysten valossa nousee esille testidataan liittyviä ongelmia. Mitä tehdään tapauksessa, jossa ei voida käyttää hyväksi olemassa olevaa dataa? Tällöin luonnollinen ratkaisu on käyttää synteettistä dataa, mutta kun ympäristöjä on useita ja testaamista tehdään usealla sektorilla, tarvitaan myös metodi, jolla data saadaan siirrettyä ympäristöstä toiseen. Kuinka generoitu data saadaan siirrettävään muotoon ja kuinka se saadaan implementoitua järjestelmään mahdollisimman ketterästi? Onko mahdollista räätälöidä metodia eri liiketoiminnoille?

Kuten aikaisemmin todettiin, staattinen data voi olla kiinteää synteettistä dataa, joka ei ole enää vuorovaikutuksessa ympäristön kanssa. Sitä ei tallenneta tai käsitellä järjestelmässä. Siinä missä nimi voidaan päivittää, staattinen komponentti on muuttumaton. Koska tällaisia staattisen datan elementtejä on ympäristössä paljon, haluttiin projektia varten luoda paketti dataa, joka soveltuisi mahdollisimman hyvin erilaisten ympäristöjen käyttöön. Pyrkimyksenä on testata ympäristön toimivuutta ja varmistaa, että prosessit suorittavat sen, mitä niiden täytyy.

4 PROJEKTIN TOTEUTUS

Ennen projektia oli jo tiedossa, että synteettinen data on käytössä ja sen hyödyt on tunnistettu. Data on ollut jo testikäytössä, mutta sen ketterä siirtäminen ympäristöstä toiseen vaati kehittämistä. Kuten aikaisemmin on todettu, synteettisen datan käyttöönotto säästää resursseja ja näin ollen sen nopea implementointi on myös olennaista. Kuviossa 1 on hahmoteltu projektin vaiheet.



KUVIO 1. Projektin kulku

4.1 Vaatimusmäärittely

Projektin työstäminen aloitettiin luomalla suunnitelma ja vaatimusmäärittely. Tärkein tavoite oli luoda metodi, jonka avulla saadaan siirrettyä staattinen data ympäristöstä toiseen mahdollisimman ketterästi. Ideana oli luoda paketti testaamiseen soveltuvaa dataa, joka voidaan ladata uuteen ympäristöön tarpeen mukaan. Dataa on voitava käyttää järjestelmän toimivuuden ja uusien ominaisuuksien testaamiseen. Prosessien testaamisen tulee myös onnistua ja käyttöliittymän tulee vastata lähdedatan näkymää.

Projektin alussa oli useita avoimia kysymyksiä. Saadaanko data pakattua helposti ladattavaan muotoon? Voidaanko se räätälöidä eri liiketoiminnoille? Kuinka helposti sitä saadaan kehitettyä eteenpäin? Oli kehitettävä metodi, joka on helppokäyttöinen ja nopea.

Testaamista tapahtuu useamman osapuolen toimesta ja monella eri sektorilla, joten testidatan tarve on jatkuva ja vaihteleva. Metodin pitää olla helppokäyttöinen, jotta testaaja voi ohjeita seuraamalla ladata paketin dataa olematta keinotekoisen datan asiantuntija.

Ongelmaksi koituivat erityisesti pienemmät liiketoimet, jotka tarvitsivat spesifimpää dataa kuin jo käytössä olevat ympäristöt. Näitä varten täytyy luoda ratkaisu, joka mahdollistaa ympäristön ominaisuuksien käytön. Täytyy löytää kompromissi datan laadun välillä. Sen on oltava tarpeeksi skaalautuvaa ympäristöjen välillä, mutta ei liian spesifiä, jotta paketteja ei jouduta tekemään useita. Todettiin parhaimmaksi ratkaisuksi luoda yksi paketti tiettyä liiketoimintoa kohti.

Kehitettävä metodi on myös pystyttävä automatisoimaan kokonaan, jotta sen käyttö on mahdollisimman nopeaa ja helppoa. Projektia tehdessä on otettava tämä huomioon erityisesti työkaluja valittaessa. Metodi tulee voida ajaa suoraan komentoriviltä ja tietokannan kautta.

Pakatussa datassa käytetään staattista dataa. Näin ollen data on suoraan ladattaessa jo testaukseen sopiva, eikä siihen tarvitse tehdä mitään muutoksia. Staattinen data on tässä tapauksessa esimerkiksi tuote, jonka arvo on 1, ja tämän tuotteen ominaisuuksia halutaan testata uudessa ympäristössä.

4.2 Datan määrittely

Yksi projektin olennaisimmista kysymyksistä oli se, minkälaista dataa halutaan ladata. Koska järjestelmässä on useita eri liiketoimintoja ja eri ympäristöillä on eri tarpeet, staattinen data vaatii tutkimista ja määrittelyä. On selvitettävä, minkälaisia arvoja ympäristöön tuodaan, jotta liiketoiminnolle saadaan käyttöön parhaiten soveltuvaa dataa.

Jos esimerkiksi liiketoiminnon X kannan taulussa Y on tuote Z, voidaan tätä tuotetta käyttää myös muissa saman liiketoiminnon X kaltaisissa ympäristöissä. Tuotteelle ei tapahdu mitään tallennusta vaativia muutoksia, vaan sen tila on staattinen. Vaatimuksena on vain se, että toisessa ympäristössä on vastaava taulu Y, johon tuote voidaan siirtää. Käyttöliittymä tulee näyttämään samalta, jotta testaaminen onnistuu.

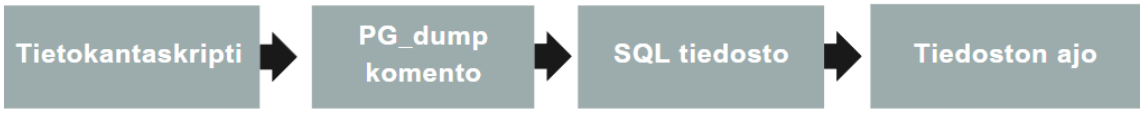
Datan määrittelyä tehdessä on selvitettävä, mitkä tiedot ovat tarpeellisia testaamisen kannalta, ja mitä voidaan jättää paketin ulkopuolelle. Jos liiketoiminto X ei käytä tuotetta Q, ei sitä myöskään tarvitse ladata toiseen liiketoiminto X:n kaltaiseen ympäristöön. Kun validoidaan data huolellisesti, voidaan säästää aikaa myöhemmissä projektin vaiheissa. Jos päätetään ladata tuote Z testattavaksi toiseen ympäristöön, on ymmärrettävä mitä kaikkea dataa tuote tarvitsee toimiakseen kunnolla. Datan määrittely vaatii perehtymistä tietokantaan ja ymmärrystä datan ominaisuuksista.

4.3 Työkalut

Projektissa käytettiin kahta työkalua. **DBeaver** on ilmainen SQL-asiakasohjelmistosovellus ja tietokannan hallintatyökalu. Skriptit on kirjoitettu, ajettu ja testattu käyttäen tätä työkalua. Työkalu mahdollistaa skriptien editoinnin ja **debuggaamisen** eli ongelmien paikantamisen ja korjaamisen. Dbeaver tukee useita tietokantoja ja sen käyttöliittymää on helppo muokata tukemaan käyttäjän tarpeita. Sovellusta voi käyttää myös tietokannan mallintamiseen diagrammien avulla.

DBeaverin lisäksi työssä on käytetty **PG dump** -aputyökalua, jonka avulla voidaan luoda isoja skriptikokonaisuuksia. Aputyökalu on ison järjestelmän kanssa merkittävä ajansäästäjä, sillä se eliminoi kokonaan manuaalisen taulujen ja lisäyslauseiden luonnin. Aputyökalu on asennettava ominaisuus, joka lisää DBeaveriin komentoja. Aputyökalu asennetaan työasemalle, valitaan halutut käyttöasetukset ja luodaan erikseen oma kansio, johon luodut tiedostot ohjataan. Lisäkomennot mahdollistavat skriptien muodostamisen SQL-tiedostoiksi, jotka voidaan ajaa suoraan kantaan. Tiedostojen siirtäminen ja dokumentointi on projektin kannalta erityisen tärkeää.

Jos DBeaverissä muodostaa skriptin tai kyselyn, voidaan se komentoriviltä muuttaa suoraan SQL-tiedostoksi. Aputyökalu tallentaa sen automaattisesti komennossa ilmoitettavalle polulle. Tiedosto voidaan jatkossa ajaa komentoriviltä tai tuoda tietokantaan. Aputyökalun avulla voidaan noutaa kyselyitä myös suoraan DBeaveristä perustuen jo olemassa oleviin tietokantarakenteisiin tai funktioihin. Aputyökalu avaa käyttöliittymään uuden ikkunan, josta voi valita esimerkiksi koko tietokannan luontilauseet. Kuviosta 2 ilmenee aputyökalun yksinkertaistettu toimintaperiaate.



KUVIO 2. PG dump

5 PROJEKTIN TIETOKANNAT

Projektissa kehitettävän metodin oli syytä olla helppokäyttöinen ja nopea. Parhaiten tällainen ratkaisu onnistuu sen ympäristön tietokannassa, johon keinotekoinen data halutaan ladata. Näin ollen projektia varten päädyttiin rakentamaan tietokantaskripti. Skriptin tehtävä on koota ladattava data, jalostaa se ja siirtää se uuteen ympäristöön. Projekti toteutettiin käyttäen SQL- ja PL/SQL-kieliä.

5.1 Tietokannat

"Tietokanta ei ole muuta kuin joukko toisiinsa yhteydessä olevaa tietoa. Puhelinluettelo, esimerkiksi, on tietokanta ihmisten nimistä, puhelinnumeroista ja osoitteista, jotka asuvat tietyllä alueella" (Beaulieu 2005, 1).

Vuonna 1970 julkaistussa artikkelissa, *A Relational Model of Data for Large Shared Data Banks* (E. F. Codd), ehdotetaan, että tietokannat esitettäisiin nuolien sijaan tauluissa. Näin syntyi alkuperäinen käsitys **relaatiotietokannasta**. Relaatiotietokannalla tarkoitetaan tietokantaa, jossa data näytetään taulukoissa. Taulukoiden, **taulujen**, välillä on yhteyksiä. Nämä yhteydet auttavat tietojen haussa ja niitä kutsutaan avaimiksi. Käytännössä tämä tarkoittaa sitä, että kahden tai useamman taulun oleva avain esiintyy kaikissa näissä tauluissa. Tauluja tietokannassa voi olla useita satoja. Yhden tai useamman taulun kokonaisuus muodostaa kaavion eli **skeeman**.

Taulussa data esitetään sarakkeilla ja riveillä. Näillä kaikilla on omat nimensä tietokannassa, joita hyödynnetään skriptejä ja kyselyitä luodessa. Esimerkiksi skeemassa nimeltä Tietokanta X on taulu Z, johon säilötään esimerkiksi tuotetietoja. Taulussa Z on sarakkeet X ja Y, joista kummallakin on esimerkiksi kahdeksan riviä dataa. Jos arvo on tyhjä, se esitetään tyypillisesti arvolla NULL. Pakolliset kentät eivät kuitenkaan saa olla tyhjiä. Tällaisissa tapauksessa tietokanta palauttaa virheen.

Jokaisella kentällä on tietokantaa luodessa määritetty **tietotyyppi**. Tietotyypit määritellään tietokannan luontilauseiden yhteydessä. Luonnin jälkeen kenttään ei voi tallentaa toisen tietotyypin arvoja, mutta ne voidaan määritellä tarvittaessa uudestaan. Beaulieu (2005, 18–24) jakaa tietotyypit eri luokkiin seuraavalla tavalla.

Integer eli kokonaisluku säilyttää negatiivisia ja positiivisia numeroarvoja. Tällä tietotyypillä kuvataan lukuja tai määriä, kuten esimerkiksi numeroarvot 2 ja -2.

Timestamp eli aikaleima tarkoittaa päivämäärää muodossa YYYY-MM-DD-hh-mm:ss. Ensin määritetään vuosi, sitten kuukausi, päivä, tunti, minuutti ja sekunti.

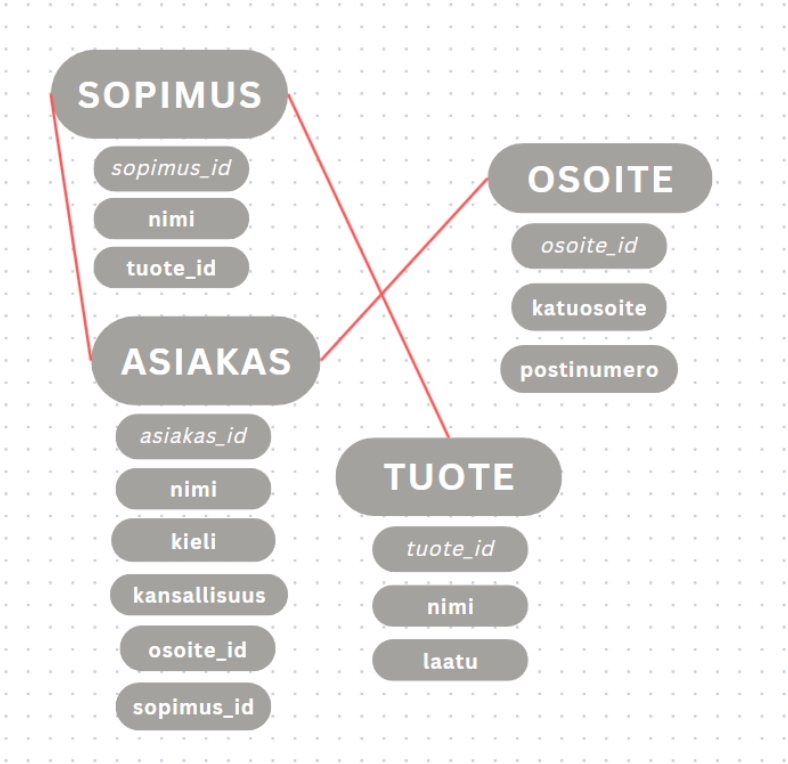
Varchar eli muuttuva merkkikenttä säilyttää tietyn määrän merkkejä, jotka voivat olla numeroita tai kirjaimia. Maksimimäärä on tavallisesti 255, mutta tälle tietotyypille voidaan määritellä tietokantaan myös esimerkiksi varchar(50), jolloin maksimimäärä on 50 merkkiä.

Text eli teksti on tietotyyppinä soveltuva sellaisiin tapauksiin, joissa merkkimäärä voi ylittää aikaisemmin määritellyn varchar-kentän merkkilukeman. Jos kenttään tallennetaan hyvin vaihteleva määrä tekstiä, on tämä tietotyyppi silloin parempi ratkaisu.

Boolean tarkoittaa totuusarvoa. Tässä tapauksessa kentän arvo on tosi (true) tai epätosi (false). Vaihtoehtoisesti, jos kentän arvo on tuntematon, se merkitään arvolla NULL.

Projektin kannalta on myös otettava huomioon tietotyyppi **Uuid**, joka säilyttää uniikin identifikaattorin. Nämä generoidaan tietokantaan erillisellä komennolla, kun tauluun lisätään rivi.

Kuviossa 3 esitetty neljä tietokannan taulua, jotka tallentavat asiakkaaseen liittyviä tietoja. Tallennettu on esimerkiksi asiakkaan nimi ja osoite, asiakkaan sopimus ja sopimuksella oleva tuote. Taulujen yhteydet on kuvattu punaisella, ja ensimmäisellä rivillä oleva sarake kuvaa jokaisella taululla olevaa identifikaattoria. Vastaava kenttä löytyy jostain toisesta taulusta. Tässä tapauksessa Tuotetaulun tuote_id-kenttä löytyy myös Sopimus-taulusta. Taulujen yhteyttä kutsutaan myös **viiteavaimeksi**.



KUVIO 3. Yksinkertainen tietokannan rakenne

Projekti vaatii tietokantaosaamista ja järjestelmän tuntemista, joten erityisesti taulujen väliset yhteydet on oltava tiedossa. Kun projektia varten tarvittavia skriptejä lähdetään kehittämään, on taulujen väliset yhteydet otettava huomioon. Ilman niiden käyttöä kehitettävä metodi ei tule onnistumaan vaatimusmäärittelyn mukaisesti, sillä datan yhtenäisyys ei olisi tällaisessa tilanteessa enää taattua.

Taulujen sisältämässä datassa on eroja ympäristöjen välillä, ne eivät siis ole vakioita. Datalle tapahtuu muutoksia niin tietokannan kuin käyttöjärjestelmän kautta ja näin ollen se voi olla eri muodoissa, jos ympäristöjä verrataan keskenään. Jos tietokantaan yhdistetyllä käyttöliittymällä tehdään muutoksia, esimerkiksi tallennetaan kenttään uusi arvo, tulee se näkymään myös tietokannassa. Tietokannan ja käyttöliittymän välillä on jatkuva vuorovaikutus. Tässä on tiettyjä rajoitteita, jotka pätevät kaikkien ympäristöjen välillä.

Kaikkia muutoksia ei voi tehdä käyttöliittymän kautta, vaan ne on tehtävä tietokannassa. Tällainen rajoite ilmenee esimerkiksi tuotteen X ominaisuuksissa. Tuotteelle voidaan vaihtaa nimi käyttöliittymällä, mutta tuotteen tyyppi on staattinen tieto, eikä sitä voida vaihtaa käyttöjärjestelmän kautta. Tämä voidaan kuitenkin muuttaa tietokannan kautta päivityslauseella.

Tietokannan data näkyy suoraan käyttöliittymällä. Jos data ei ole oikeassa muodossa tai data on puutteellista, se ei tule näkyviin oikein. Kun keinotekoinen data ladataan toiseen ympäristöön, on sen oltava samanlainen niin käyttöliittymällä kuin alkuperäisessäkin ympäristössä. Muussa tapauksessa testaaminen ei tule myöhemmin onnistumaan halutulla tavalla. Jos data ei ole yhtenäistä tai siinä on virheitä, tulee se ilmenemään käyttöliittymällä esimerkiksi virheilmoituksena.

5.2 Tietokantaskripti

Kuten aikaisemmin todettiin, parhaiten datan siirto ympäristöstä toiseen onnistuu, kun luodaan tietokantaskripti. Oracle (2023) kuvailee skrptin olevan joukko komentoja, jotka kertovat tietokannalle, kuinka toimitaan. Skripti voi sisältää esimerkiksi lauseita, jotka hakevat ja päivittävät taulujen dataa.

Projektin kannalta skriptin tärkeimmät tehtävät ovat testidatan kokoaminen, testidatan jalostaminen, jotta se sopii parhaiten kohdejärjestelmän tarpeisiin, ja testidatan lataaminen kyseiseen järjestelmään. Skriptin on täytettävä usean eri tehtävän vaatimukset. Itsessään tietokantaskripti on helppokäyttöinen ja nopea ajaa, eikä se vaadi merkittävästi muokkaamista latauksien välissä. Yksi vaatimus on, että skripti voidaan automatisoida eli se ei tarvitse merkittävästi käyttäjän tekemiä valintoja, vaan se voidaan ajaa suojaan komentoliittymältä.

Tietokannan käsittelyyn käytetään projektissa SQL-kyselykieltä ja skriptaamiseen PL/SQL-kyselykielen laajennusta. Projektin kannalta on ymmärrettävä näiden kielien ero ja käyttötarkoitukset. Sen lisäksi, on oltava vankka tietämys erityisesti PL/SQL syntaksista. Vaikka kielet voivat näyttää syntaksiltaan hyvin samanlaisilta ja PL/SQL voikin rakenteeltaan muistuttaa paljon tuttuja ohjelmointikieliä, käydään silti läpi muutama kielen ominaisuus.

5.2.1 SQL

SQL (Structured Query Language) on tietokantojen hallinnointiin käytettävä **kyselykieli**. Techopedian (2016) mukaan kyselykielellä tarkoitetaan ohjelmointikieltä, joka pyytää ja hakee tietokannasta kyselyssä määriteltyjä tuloksia. Tämän vuoksi SQL-lausetta kutsutaan usein **kyselyksi**. Kielen avulla tietokannan hoitaja voi esimerkiksi muokata tietokannan rakennetta, ajaa kyselyitä, tehdä

datapäivityksiä tai muuttaa tietokannan asetuksia. Tietokannan rakennetta voidaan muokata merkittävästi. Koko tietokanta voidaan esimerkiksi poistaa tai skeemaan voidaan lisätä kokonaan uusia tauluja.

Techopedia (2021) jakaa SQL-komennot eri luokkiin seuraavasti.

Data Definition Language (DDL) sisältää taulujen rakennetta koskevat komennot, kuten esimerkiksi create, alter ja drop. Näiden avulla luodaan, muutetaan tai poistetaan tauluja. Kun luodaan ympäristölle tietokanta, taulujen luonti on tehtävä alussa.

Data Manipulation Language (DML) tarkoittaa taulujen sisäisen datan manipulointia eli datan muokkaamista. Tähän sisältyy esimerkiksi komennot select, insert, update, delete. Näiden komentojen avulla dataa voidaan hakea, lisätä, päivittää tai poistaa.

Data Control Language (DCL) sisältää komennot grant ja revoke, joiden avulla säädellään tietokannan oikeuksia.

Tästä näkökulmasta kyselykieli on taipuisa, mutta projektia varten SQL ei kuitenkaan tarjoa tavanomaiselle ohjelmointikielelle tyypillisiä ominaisuuksia, kuten esimerkiksi ehtolauseiden luomista ja ehtojen ketjuttamista. Lauseet ajetaan tietokantaan yksi kerrallaan, joka on toisinaan hidasta ja lisää tietokannan kuormaa. Lauseet eivät myöskään palauta epäonnistuessaan virhettä. Näin ollen **debuggaaminen** on haastavaa.

Kuviossa 4 on kuvattuna yksinkertainen SQL-kysely, joka hakee tietokannasta taulun X kaikki tuotteet, joiden laatu on Y. Kysely palauttaa tuotteen identifikaattorit ja nimet.

```
select product_id, product_name
from table X
where product_quality = quality Y
```

KUVIO 4. SQL-kysely

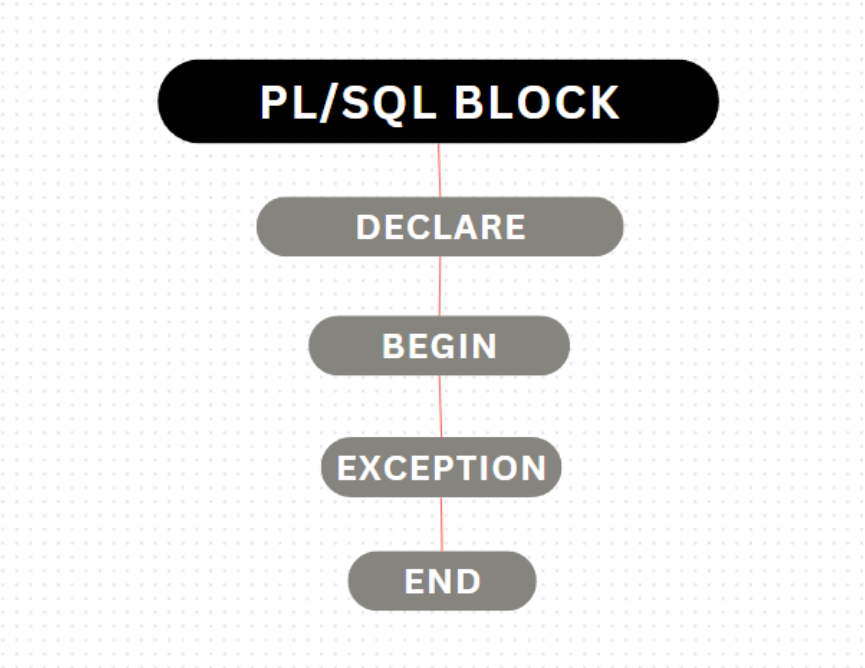
5.2.2 PL/SQL

PL/SQL (Procedural Language/Structured Query Language) on kyselykielen laajennus, joka mahdollistaa edellä määriteltyjen heikkouksien paikkaamisen. ”PL/SQL on erittäin jäsennelty, sujuva ja helppokäyttöinen kieli” (Feuerstein & Pribyl 2014).

Lauseet ovat PL/SQL-kielellä mahdollista ajaa koodina, eikä yksi kerrallaan. Siinä missä SQL pyrkii muokkaamaan olemassa olevaa rakennetta tai dataa, PL/SQL on yleisemmin käytetty sovelluksen luonnissa. PL/SQL mahdollistaa esimerkiksi ehtolauseen luomisen, niiden nopean ajamisen ja debuggaamisen. Näin ei myöskään rasiteta tietokantaa niin merkittävästi kuin tavallisen SQL-lauseen ajossa. ”PL/SQL tarjoaa laajan valikoiman lauseita, jotka mahdollistavat ohjelman osien ajamisen tarkan valvonnan” (Feuerstein & Pribyl 2014).

PL/SQL-koodilohkoa voidaan kutsua **skriptiksi**. Tällä tarkoitetaan kirjoitettua ohjelmaa, joka ajetaan tietokantaan. Projektin kannalta ehtolauseiden rakentaminen on erityisen tärkeää. Datan latauksessa on otettava huomioon useita järjestelmän eri tilanteita, ja näin ollen skriptaaminen on paras ratkaisu. Ehtolauseiden rakentaminen ja ketjuttaminen on etualalla. Syntaksiltaan PL/SQL muistuttaa SQL-kieltä, ja useissa lauseissa sisältö onkin luettavissa täysin samalla tavalla kuin SQL-kielessä. Jos SQL osaaminen on vahva, ymmärtää silloin valtaosan myös PL/SQL-skriptin sisällöstä.

Kuviossa 5 on hahmoteltuna tyypillisen koodilohkon rakenne. **Declare**-komento aloittaa listaamaan muuttujat, jotka ovat lohkossa käytettäviä arvoja. Arvoja voidaan näin ollen kutsua lohkon muissa osissa, jos niille tehdään toimenpiteitä. **Begin**-komento aloittaa ajon. Sen jälkeen voidaan kirjoittaa itse ohjelma, esimerkiksi ehtolause, joka käyttää esimerkiksi aikaisemmin määriteltyjä muuttujia. Tätä kohtaa on kuviossa kuvattu nimellä **exception**. Lopuksi skripti päätetään komennolla **end**.



KUVIO 5. PL/SQL rakenne

5.2.3 Skriptin kirjoittaminen

Metodin skriptaaminen aloitettiin listasta funktioita ja ominaisuuksia, jotka skriptillä täytyy pystyä toteuttamaan. Tällaisia olivat esimerkiksi ehtolause, joka tarkistaa, onko kohdetaulussa duplikaatteja ja funktio, joka tarvittaessa yhdistää ominaisuuden Y tuotteeseen X. Skripti säilöö muuttujat ladattavasta datasta ja kohdedatasta ja vertailee niitä keskenään ehtolauseiden avulla. Näin skripti saa selville, mitkä osuudet datasta ladataan ja mitkä ei.

Kuviossa 6 on määriteltynä muutama esimerkkimuuttuja. Ensimmäisenä voidaan esimerkiksi määritellä tietokannat, joista dataa siirretään. Näihin muuttujiin viitataan skriptissä aina, kun käsitellään ohjelman niitä koskevia toimenpiteitä.

```
do $$
declare dataFrom text = ‘database_name’
declare dataTo text = ‘database_name’
declare selected_data text = ‘table_name’
```

KUVIO 6. Esimerkkimuuttujat

Skriptin pääasiallinen tarkoitus on jalostaa dataa mahdollisimman sopivaan muotoon, jotta se soveltuu kohdeympäristön testidataksi. Skripti päivittää esimerkiksi päivämääriin uudet aikaleimat ja poistaa mahdolliset päivittäjien nimet kannasta. Datassa on oltava kenttä, joka jäljittää, mitkä rivit on tuotu synteettisen datan skriptillä. Kuviossa 7 on esitetty yleispiirteinen esimerkki tällaisen kentän hausta.

```
(select table_schema, table_name, column_name
from information_schema.columns
where table_schema = 'synthetic_data' and table_name LIKE '%'
and column_name LIKE '%created')
```

KUVIO 7. Datan luonnin kenttä

Skriptissä vertaillaan ladattavaa dataa ja ympäristössä jo olevaa dataa. Skriptin olennaisin tehtävä tässä vaiheessa on tarkastella, onko ympäristössä olemassa jo esimerkiksi tuote X, jota skripti on lataamassa ympäristöön. Jos tuote löytyy, voidaan se poistaa latauksesta kokonaan. Kaikki taulut käydään järjestyksessä läpi tauluyhteyksiä noudattaen ja jokainen sarake käydään yksitellen läpi.

Kuviossa 8 on esitettynä yleispiirteinen esimerkki taulujen vertailusta. Jos dataa ei löydy kohdeympäristöstä, ajaa skripti tarkistuksen jälkeen datan latauksen kohdetauluun. Raise info -komennon avulla voidaan seurata skriptin kulkua komentoliittymältä. Kyseinen komento tulee hyödylliseksi erityisesti, kun skriptiä testataan.

```
if exists (select * from synthetic_data.table_name where data not in
(select data from schema_name.table_name)
RAISE info 'data does not exist'
```

KUVIO 8. Taulujen vertailu

Skriptissä käytetään hyödyksi aputauluja, jotka helpottavat datan vertailua taulujen välillä. Aputauluista tiedot eivät siirry käyttöliittymälle samalla tavalla kuin virallisista tauluista, mutta niiden rakenne ja luonti toimii samalla tavalla. Aputauluja voidaan poistaa ja luoda ilman merkittävää kuormitusta ja niitä voi olla lukuisia kappaleita yhtä aikaa. Näin ollen aputauluihin voidaan siirtää dataa väliaikaisesti varastoon, tai dataan voidaan tehdä päivityksiä aputaulun sisällä. Data voidaan siirtää

aputaulusta eteenpäin. Kuviossa 9 aputaulua on hyödynnetty, jotta voidaan säilöä tarvittaessa ladattava data. Muussa tapauksessa data poistetaan latauksesta, joten jälkikäteen voidaan silti tarkastella, minkälaista dataa jäi latauksen ulkopuolelle.

```
create table if not exists synthetic_data.table_name_backup
as (select * from synthetic_data.table_name);
delete from synthetic_data.table_name
where data in (select data from schema_name.table_name
```

KUVIO 9. Aputaulu

Dataa on syytä jalostaa, jos testidataan halutaan esimerkiksi uudet identifikaattorit. Jos data ladataan useaan ympäristöön, on laadun ja turvallisuuden kannalta parempi, kun viittaukset edelliseen ympäristöön on poistettu. Skriptin on aina generoitava ajon yhteydessä uudet identifikaattorit. Kuviossa 10 haetaan ohjelmointiin edellinen datarivi ja generoidaan sen tilalle uusi arvo. Ketjun sisällä tapahtuisi uuden arvon päivitys taulun kenttiin, mutta tässä esimerkissä se on jätetty tyhjäksi.

```
for data in select old_data, generate_new as new_data
from synthetic_data.table_name where value = old_data
loop

end loop;
```

KUVIO 10. Datan jalostus

Skriptiä luodessa käytettiin apuna PG dump -aputyökalua, jotta skriptistä saadaan luotua helppokäyttöiset tiedostot. Nämä voidaan ajaa komentoriviltä tai tuoda suoraan kantaan. Skriptit jaettiin kahteen osaan: datan lataukseen ja datan jalostukseen. Datan latauksessa tapahtuu taulujen vertailu ja datan jalostuksessa käydään läpi esimerkiksi aiemmin mainitut aikaleimojen päivitykset.

5.3 Testaaminen

Ennen skriptin varsinaisen testaamisen aloittamista luotiin lista tilanteista, joita ympäristössä voi esiintyä. Ympäristön datassa voi olla useita eroavaisuuksia ja skriptin täytyy osata reagoida oikein, kun testidataa lähdetään lataamaan. Ympäristö voi olla kokonaan tyhjä, jolloin skriptin ei tarvitse tehdä muuta kuin ladata data normaalisti. Jos ympäristössä on kuitenkin valmista dataa, on skriptin käytävä kaikki tietokannan taulut läpi oikeassa järjestyksessä, jotta datan eheys säilyy. Jos esimerkiksi ympäristössä on jo olemassa tuote X, ei siitä saa muodostua duplikaattia. Tällaiset tapaukset ilmenevät ohjelmistovirheenä käyttöliittymällä.

Jokainen testauskerta tapahtuu askeleittain samalla tavalla: ensin ajetaan skripti onnistuneesti tietokantaan ja sitten tarkistetaan, miten data näkyy käyttöliittymällä, jos mitään virheilmoituksia ei ilmaantunut skriptin ajon aikana tai jälkeen. Kun testikierros on valmis, poistetaan kaikki data, jonka skripti on tuonut. Tämän jälkeen voidaan aloittaa uuden tilanteen testaaminen. Tämän vuoksi testaaminen oli hidasta ja tarkkaa työtä. Useassa tapauksessa ilmeni, että skripti ei toimi halutulla tavalla, tai esiin nousi uusia tekijöitä, joita ei ollut huomattu vaatimusmäärittelyä tehdessä. Oli systemaattisesi käytävä läpi, kuinka data realistisesti toimisi käytössä.

Tapaukset, joissa ympäristössä on jo valmista dataa, olivat erityisen haastavia. Aina, kun skriptiin lisättiin uusi ehtolause, oli sen toiminta testattava koko prosessin yhteydessä. Jos olemassa olevalla tuotteella X on sama ominaisuus, kuin tuotteella Y, on skriptin osattava käsitellä tämä yhtenäisyys. Muussa tapauksessa virhe ilmenee käyttöliittymällä.

Ongelmanratkaisu ja debuggaaminen ovat testaamisen aikana olennaisia. Kun skriptin testaamisessa tulee vastaan ongelma, on se pystyttävä ratkaisemaan järjestelmän antaman virheen perusteella. Aikaisemmin mainittu Raise info -komento on erityisen hyödyllinen, jos halutaan seurata muuttujia ja testata missä kohti skriptiä ongelma on. Testaamisen aikana skriptiin saatiin korjattua useita heikkouksia, kuten esimerkiksi se, missä järjestyksessä skripti käy läpi skeeman taulut. Data on ladattavassa tarkassa järjestyksessä, jotta taulujen väliset yhteydet säilyvät eheinä.

6 PROJEKTIN TULOS

Projektin aikana tapahtui muutamia muutoksia vaatimusmäärittelyyn ja sisäisiä askeleita jouduttiin priorisoimaan resurssien puutteen vuoksi. Useat ominaisuudet havaittiin vasta projektin aloitettua ja erityisesti skriptiin tehtiin useita muutoksia. Kun jälkikäteen ajattelee, niin useat tällaiset ominaisuudet olisi ollut lähes mahdoton havaita ennen projektin työstämistä. Datan sisältö yllätti erityisesti ja se olikin projektin haastavin vaihe, kun lähdettiin luomaan skriptiä vastaamaan ympäristön erilaisiin datamääriin ja datan vaihtelevaan laatuun.

Projektin päämääränä oli siis saavuttaa mahdollisimman helppokäyttöinen ja nopea metodi, jotta testidatan käyttö järjestelmässä trivialisoituisi. Kehitetty metodi on mahdollista automatisoida osittain helpottamaan käytettävyyttä. Oletuksena on, että saatavilla on generoitua dataa, joka vastaa laadultaan järjestelmän lähdedataa. Kyseinen data on staattista ja näin ollen soveltuu tärkeimpien ominaisuuksien testaamiseen.

Skripti jaettiin useaan osaan. Data asetetaan ensimmäisellä skriptillä tietokannan aputauluun, jossa toinen skripti käsittelee ja jalostaa datan uuteen ympäristöön sopivaksi. Tässä vaiheessa skripti poistaa esimerkiksi lähdedatan aikaleimat, tekijöiden nimet tai muut dataa yksilöivät piirteet. Skripti generoi jokaiselle yksilöivälle riville myös uudet identifikaattorit. Dataan päivitetään synteettisen datan leima, jotta sitä voidaan helposti jäljittää.

Käydään läpi esimerkki, jossa tuote X halutaan siirtää ympäristöstä A ympäristöön B. Kuviossa 11 on kuvattuna yksinkertaistetut askeleet.

KUVIO 11. Datan siirto

Ympäristöt ovat siis esimerkissä jo tiedossa. Datan valinnassa skripti valitsee tuotteen X, johon esimerkiksi sisältyy kaksi ominaisuutta: laatu ja hinta. Koska jokaisella rivillä on oma identifikaattori, generoi skripti dataa jalostaessa laadulle ja hinnalle uudet identifikaattorit. Aikaleimaan päivittyy esimerkiksi datan siirtopäivä ja tekijäksi skriptin nimi. Kun jalostettu data lähdetään siirtämään, tutkii skripti ensin kohdeympäristöä ja tarkastelee erityisesti, onko siellä jo olemassa tuote X. Skripti etsii myös tuotteen X ominaisuuksia ja vertailee, voiko ympäristössä olla päällekkäin samanlainen tuote. Jos ei, skripti pysähtyy. Jos kuitenkin tuotteen siirtäminen on mahdollista, asettaa skripti sen tietokantaan.

7 POHDINTA

Projekti oli hyvin haastava ja erityisesti skriptaamisen taito kasvoi projektin aikana merkittävästi. Koska osa olennaisista ominaisuuksista havaittiin vasta testaamisen aikana, muuttui tehtävänanto useaan kertaan ja työ vaati joustamista. Aihe oli uusi ja teoriaosuudessa oli paljon tutkittavaa. Itse projektin toteutus ei vaatinut pelkästään hyvää PL/SQL-osaamista, vaan järjestelmään ja ympäristöihin täytyi perehtyä hyvin. Oli ymmärrettävä, minkälaiseen käyttötarkoitukseen data menee ja kuinka metodista saisi mahdollisimman helppokäyttöisen.

7.1 Projektin kehitysideat

Yksi projektin isoimmista käytännön haasteista oli saada tietokantaskripti toimimaan mahdollisimman monissa tilanteissa. Jos tietokannassa on jo reilusti dataa ja siihen lisätään sen kanssa ristiriidassa olevaa dataa, syntyy ongelmia. Jos esimerkiksi on olemassa tuote X, joka jakaa komponentin A toisen tuotteen kanssa, on skriptin tiedettävä, mitä tehdä, jos ympäristöön tuodaan lisää komponentin jakavia tapauksia. Ongelmana on, että näitä tilanteita on haastava havaita etukäteen ilman, että kannan tilanteeseen ja ympäristöön perehdytään ensin. Senkin jälkeen skriptiin olisi tehtävä muutoksia. Tällaisia tapauksia voi ilmetä useassa tilanteessa, joka merkittävästi rajoittaa skriptin käyttöä. On arvioitava, minkälaisten muutosten tekeminen on mahdollista ja erityisesti se, kuinka paljon skriptiä voisi vielä yksilöidä.

Kun ajatellaan ympäristön eri tilanteita ja datan määrää, skriptissä voidaan aina varautua paremmin erilaisiin tapauksiin, kun niitä testatessa tai muuten vain havainnoidessa nousee esille. Hyvä puoli on se, että skripti ei merkittävästi kuormitu, jos ehtolauseiden määrää lisää. Skripti voi periaatteessa varautua useaan tilanteeseen, vaikka niiden ehdot eivät dataa ladatessa toteutuisikaan. Tällaisten toteuttaminen kuitenkin vie aikaa ja skripti on aina testattava kokonaisuudessaan. Myös automatisointia voi kehittää vielä pidemmälle.

Samaan aiheeseen liittyen nousee esille jo aikaisemmin havaittu ongelma ympäristöjen eri luonteiden välillä. Todettiin, että on hankalaa luoda yleispiirteistä metodia, joka sopisi eri liiketoimien ympäristöihin. On myös mahdollista, että toisinaan järjestelmään tehdään päivityksiä, jotka muuttavat kannan rakennetta. Jos näin tapahtuu ja skriptin käsitteleviin tauluhiin tulee muutoksia, ei skripti

enää toimi ollenkaan. On siis syytä ajatella, että ylläpidolle voi olla tarvetta. Skripti on pidettävä ajan tasalla.

On hyvin mahdollista, että synteettisen datan kehittyessä luonnollisesti nousee esille uusia ideoita ja toteutustapoja. Kyseessä on kuitenkin vielä varsin uusi aihe, ja vaikka datan generointi on jo monella saralla aktiivisessa testauksessa, voidaan sen hyötyä edelleen kasvattaa. Näin ollen myös tietokantapuolella voi syntyä uusia tarpeita.

7.2 Johtopäätökset

Synteettinen data herättää paljon keskustelua erityisesti yksityisyydensuojan yhteydessä. Keinotekoinen data kuulostaa erinomaiselta keinolta päästä samaan lähtöpisteeseen, jossa oltaisiin oikean datan kanssa, mutta ilman tietosuojariskejä. Yksi ongelma kuitenkin nousee esille, kun ajatellaan synteettisen datan uutuutta. Useat lähteet spekuloivat synteettisen datan mahdollisuuksia, mutta testejä ja tuloksia erityisesti yksityisyydensuojan parissa on tehty loppupeleissä varsin vähän. Usein ajatellaan, että tietoturvamurto on vain ajan kysymys, ja erityisesti synteettisen datan tapauksessa aikaa ei ole kulunut paljoa. Uusia havaintoja tehdään koko ajan lisää. Synteettiseen dataan voi liittyä haasteita, joita ei ole vielä edes havaittu. Tutkimus on kuitenkin aktiivista.

Voi olla haastavaa pohtia synteettisen datan hyötyjä ja heikkouksia erityisesti yrityksen näkökulmasta. Synteettiseen dataan liittyy paljon avoimia kysymyksiä. Mistä tietää, milloin data on tarpeeksi anonyymiä ja kuinka luotettavia testit ovat, kun dataa validoidaan? Voidaanko synteettinen data murtaa? Onko synteettisen datan generointi mahdollista peruuttaa ja näin ollen palauttaa alkuperäinen data? Vastuu on pitkälti tekijällä ja esimerkiksi synteettistä dataa generoidessa, riippuu tulos lähdedatan määrästä ja laadusta. Tulokset voi kokea vaihteleviksi, erityisesti jos metodit ovat vielä kehitysvaiheessa.

Kun ajatellaan opinnäytetyön projektia ja synteettisen datan hyödyntämistä vastaavissa tapauksissa, on potentiaali suuri ja ehdottomasti hyödynnettävä. Synteettiselle datalle on useita käyttökohteita ja sen maksimaalista hyötyä ei ole vielä välttämättä edes havaittu. Synteettinen data avaa uusia ovia erityisesti tekoälylle, tutkimukselle, data-analyysille ja erilaisten tuotteiden ja palveluiden

testaamiselle. Kyseessä on erityisen ajankohtainen aihe, jota varmasti hyödynnetään tulevaisuudessa usealla saralla.

LÄHTEET

Bamford, Steven 2021. Synthetic Data and Privacy: Experiences Implementing Data Synthesis in a Global Life Sciences Company. Euroopan Tietosuojavaltuutettu. Hakupäivä 19.6.2023. https://edps.europa.eu/system/files/2021-06/01_stephen_bamford_en_0.pdf

Beaulieu, Alan 2005. Learning SQL. O'Reilly Media, Inc. Hakupäivä 17.6.2023. Vaatii käyttöoikeuden. https://www.oreilly.com/library/view/learning-sql/0596007272/

Bonk, Lawrence 2023. Elon Musk's X will use public data to train AI models. Endgadget. Hakupäivä 10.9.2023. https://www.engadget.com/elon-musks-x-will-use-public-data-to-train-ai-models-184924197.html

Codd, Edgar F. 1970. A Relational Model of Data for Large Shared Data Banks. University of Pennsylvania. Hakupäivä 17.6.2023. https://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf

Feuerstein, Steven & Pribyl, Bill 2014. Oracle PL/SQL Programming, Sixth Edition. O'Reilly Media, Inc. Hakupäivä 17.6.2023. Vaatii käyttöoikeuden. https://www.oreilly.com/library/view/oracle-plsql-programming/9781449324445/

Devaux, Elise 2022. Types of synthetic data and 4 examples of real-life applications. Statice. Hakupäivä 11.7.2023. https://www.statice.ai/post/types-synthetic-data-examples-real-life-examples

Tietoarkisto 2023. Tunnisteellisuus ja anonymisointi. Hakupäivä 20.9.2023. https://www.fsd.tuni.fi/en/services/data-management-guidelines/anonymisation-and-identifiers/

Tietosuojavaltuutetun toimisto 2023. EU:n tietosuoja-asetus. Hakupäivä 20.9.2023. https://tietosuoja.fi/gdpr

Heinäsenaho, Markku, Äyräs-Blumberg, Outi & Lähesmaa, Jukka 2023. Tekoäly mullistaa terveydenhuoltoa – mahdollisuudet hyödynnettävä viipymättä. Sosiaali- ja terveysministeriö. Hakupäivä 22.9.2023.
https://valtioneuvosto.fi/-//1271139/tekoaly-mullistaa-terveydenhuoltoa-mahdollisuudet-hyodynnettava-viipymatta

Honkela, Antti 2023. Synteettinen data voi suojata yksityisyyttä. Helsingin yliopisto. Hakupäivä 5.10.2023.
https://www.helsinki.fi/fi/matemaattis-luonnontieteellinen-tiedekunta/ajankohtaista/synteettinen-data-voi-suojata-yksityisyytta

Lamberti, Aldo 2023. The benefits and limitations of generating synthetic data. Syntheticus. Hakupäivä 17.6.2023.
https://syntheticus.ai/blog/the-benefits-and-limitations-of-generating-synthetic-data

Martineau, Kim 2023. What is synthetic data? IBM Research. Hakupäivä 17.6.2023.
https://research.ibm.com/blog/what-is-synthetic-data

Nieminen, Valtteri & Virkki, Arho 2022. Aitoa hyötyä keinotekoisesta datasta. Terveyskampus Turku. Hakupäivä 17.6.2023.
https://www.healthcampusturku.fi/blogs/aitoa-hyotya-keinotekoisesta-datasta-2/

Oracle 2015. Application docs, Release 3.2. Hakupäivä 20.8.2023.
https://docs.oracle.com/cd/E14373_01/user.32/e13370/sql_rep.htm#AEUTL190

Rautio, Marjatta 2021. Poliisi alkaa pian kuulla Vastaamo-tietomurron tuhansia uhreja – ”Ratkaiseva tekijä voi olla jo nurkan takana”, sanoo tutkinnanjohtaja. Yle. Hakupäivä 19.9.2023.
https://yle.fi/a/3-12152682

Riemann, Robert 2023. Synthetic data. Euroopan tietosuojavaltuutettu. Hakupäivä 20.9.2023.
https://edps.europa.eu/press-publications/publications/techsonar/synthetic-data_en

Rouse, Margaret 2016. Query Language. Techopedia. Hakupäivä 19.6.2023.
https://www.techopedia.com/definition/3948/query-language

Rouse, Margaret 2018. Static data. Techopedia. Hakupäivä 19.6.2023. https://www.techopedia.com/definition/31590/static-data

Rouse, Margaret 2021. Structured Query Language. Techopedia. Hakupäivä 19.6.2023. https://www.techopedia.com/definition/1245/structured-query-language-sql

Toews, Rob 2022. Synthetic data is about to transform artificial intelligence. Forbes. Hakupäivä 20.9.2023. https://www.forbes.com/sites/robtoews/2022/06/12/synthetic-data-is-about-to-transform-artificial-intelligence/?sh=2fca652c7523

Tuomisto, Emmi 2021. Synteettinen data ratkaisee yksityisyydensuojan ongelmia – näin sitä syntyy suomalaissovelluksella. Tivi. Hakupäivä 19.6.2023. https://www.tivi.fi/uutiset/synteettinen-data-ratkaisee-yksityisyydensuojan-ongelmia-nain-sita-syntyy-suomalaissovelluksella/325bfeee-0818-44ee-aa33-304eb018b8f6