Pseudonymisoidun tuotantodatan hyödyntäminen ohjelmistotestausympäristössä

Jelena Laakkonen

Haaga-Helia ammattikorkeakoulu
Tradenomi
Amk-opinnäytetyö
2023

Tiivistelmä

Tekijä(t)
Jelena Laakkonen

Tutkinto
Tradenomi

Raportin/Opinnäytetyön nimi
Pseudonymisoidun tuotantodatan hyödyntäminen ohjelmistotestausympäristössä

Sivu- ja liitesivumäärä
41 + 3

Ohjelmistotestaus on välttämätön osa ohjelmistokehitystä ja järjestelmien ylläpitoa. Tietoturvan takaaminen on tärkeä osa ohjelmistotestausta, sillä testausprosessin aikana käsitellään usein suurta määrää henkilötietoja. Lisäksi yksityisyys ja tietosuoja ovat kasvavia huolenaihe maailmalla digitalisoitumisen yleistyessä. Tämän takia on erityisen tärkeää kunnioittaa käyttäjän yksityisyyttä ja tietoturvaa, eikä arkoja tunnistetietoja saa vuotaa vääriin käsiin ohjelmistotestauksen yhteydessä. Tietoturvan takaamiseksi tunnistetiedot voidaan esimerkiksi poistaa käsiteltävästä aineistosta.

Tämä opinnäytetyö käsittelee ohjelmiston tuotantodatan tunnistetietojen poistamista ja sen hyödyntämistä ohjelmistotestauksessa. Työ toteutettiin vuoden 2023 keväällä. Työn tavoitteena oli saada parempi ymmärrys tuotannonkaltaisen testausympäristön tarpeesta, hyödyistä ja ongelmista sekä lopulta rakentaa toimeksiantajayritykselle X Oy:lle tietoturvallista tuotantodataa hyödyntävä testausympäristö. Työn lopputulos on konseptitodistus, eli ohjelmiston kaikista tietokantatauluista ei poistettu tunnistetietoja, sillä tietokannan laajuuden vuoksi se olisi ollut työn puitteissa mahdotonta. Työ rajautuu datan henkilötietojen suojaamiseen ja sen hyödyntämiseen ohjelmistotestausympäristössä, eikä käsittele muita testausympäristön ongelmia, kuten ulkoisia integraatioita. Työn tulokset esitellään aina ilman oikeaa asiakasdataa ja paljastamatta toimeksiantajan ohjelmiston tietokantarakennetta.

Tietoperustassa käsitellään ensin henkilötietoihin liittyvää lainsäädäntöä, keskittyen pääasiassa yleiseen tietosuoja-asetukseen (GDPR). Tämän jälkeen tarkastellaan anonymisoinnin ja pseudonymisoinnin eroja. Sitten käsitellään ohjelmistotestausta: mitä se tarkoittaa ja miksi sen optimoiminen on tärkeää. Lopuksi nämä aiheet yhdistetään ja selvitetään, kuinka tunnistetiedot poistetaan tuotantodatasta ja siirretään testausympäristöön hyödynnettäväksi.

Opinnäytetyön empiirinen osuus rakennettiin neljän viikon aikana hyödyntäen ketterän kehityksen menetelmiä. Toisin sanoen, jokaisesta työvaiheesta oli mahdollista tarvittaessa siirtyä takaisin aiempaan vaiheeseen. Tässä osuudessa ensin kartoitetaan työn lähtötilanne ja avataan tarkemmin tavoitteita. Tämän jälkeen siirrytään työn etenemisen suunnitteluun ja lopulta toteutetaan suunnitelma vaihe vaiheelta. Kirjoittaja esittää, miten työ eteni, mitä mahdollisia ongelmia kohdattiin ja miten ne saatiin korjattua. Toteutuksen ensimmäisessä vaiheessa valittiin projektiin olennaiset tietokantataulut. Tauluista määritettiin tunnistetiedot, jonka jälkeen valittiin työkalu ja menetelmät tunnistetietojen peittämiseksi. Lopuksi tunnistetiedot poistettiin työkalun avulla ja testattiin ympäristön toimivuutta.

Opinnäytetyön lopussa pohditaan työn onnistumista, jatkokehitysmahdollisuuksia sekä kirjoittajan omaa oppimista.

Asiasanat
ohjelmistotestaus, tietoturva, henkilötiedot, pseudonymisointi, anonymisointi

Sisällys

1 Johdanto

Ohjelmistotestaus on välttämätön osa ohjelmistokehitystä ja järjestelmien ylläpitoa (Kasurinen 2013, alaluku Mitä on ”ohjelmistotestaus”). Yksityisyyden ja tietoturvan takaaminen on tärkeä osa ohjelmistotestausta, sillä testausprosessin aikana käsitellään usein suurta määrää henkilötietoja, kuten asiakkaiden henkilötunnuksia, pankkitietoja tai terveystietoja (Elfriede 2002, luku 11). Yksityisyys ja tietosuoja on kasvava huolenaihe maailmalla digitalisoitumisen yleistyessä. Meistä kerätään valtava määrä dataa päivittäin käyttäessämme erilaisia internetpalveluita. Erityisesti henkilötietojen kerääminen, tallentaminen ja käyttö ovat herättäneet huolta tietosuojan puolesta. (Goswami 14.12.2020.) Tämän takia on erityisen tärkeää, että käyttäjän yksityisyyttä ja tietoturvaa kunnioitetaan eikä arkoja tunnistetietoja pääsee vuotamaan vääriin käsiin ohjelmistotestauksen yhteydessä. Tietoturvan takaamiseksi tunnistetiedot voidaan esimerkiksi poistaa tai piilottaa käsiteltävästä aineistosta, jotta käyttäjää ei voida enää tunnistaa aineiston perusteella, vaikka se päätyisikin vääriin käsiin (Raghunathan 2013, 123).

Testausdatan puutteellisuus luo usein tarpeettomia ongelmia ohjelmistotestaamisessa. Ilman oikeanlaista dataa on hankala, ellei mahdoton, testata uutta tai vanhaa ominaisuutta oikeilla skenaarioilla. Lisäksi datan muokkaaminen tarvittavaan muotoon on aikaa vievää. Jos testausympäristön data olisi jo valmiiksi tuotantodatan kaltaista, ei olisi tarvetta käyttää aikaa datan manuaaliseen luomiseen ja muokkaamiseen. On myös mahdollista, ettei ohjelmointivirheitä pääsisi niin helposti tuotantoon, jos käytössä olisi monimuotoisempaa ja realistisempaa dataa. (Heusser & Kulkarni 2016, 113; Elfriede 2002, luku 11.)

Työn tavoite on saada parempi ymmärrys tuotannonkaltaisen testausympäristön tarpeesta, hyödyistä ja ongelmista sekä lopulta rakentaa tietoturvallista tuotantodataa hyödyntävä testausympäristö. Työssä siis rakennetaan tietotekninen ratkaisu olemassa olevan testausprosessin parantamiseksi. Toiminnallisen puolen lisäksi työ sisältää sekä tietoperustan, joka pohjautuu tekstilähteisiin, että pohdinnan työn tuloksista.

Työn toimeksiantajana toimii ohjelmistoalan yritys X Oy. X Oy tarjoaa asiakkailleen laskutus- ja kirjanpitopalveluja. Tuotantodatan tunnistetietojen poistaminen tehdään organisaation erään palvelun testausympäristöä varten sekä sen tuotantodataa hyödyntäen.

Työ tulee vastaamaan seuraaviin tutkimuskysymyksiin: Mitä on datan tunnistetietojen piilottaminen ja miten ohjelmiston tuotantodataa, josta on poistettu tunnistetiedot, voidaan hyödyntää käytännössä ohjelmistotestausympäristössä. Työ rajautuu datan henkilötietojen suojaamiseen ja sen hyödyntämiseen ohjelmistotestausympäristössä, eikä tule vastaamaan muihin testausympäristön ongelmiin, kuten ulkoisiin integraatioihin.

Yksi työn onnistumismittareista on luotettava, selkeä ja kattava vastaaminen ennalta-asetettuihin tutkimuskysymyksiin. Toinen onnistumismittari on se, että empiriassa on onnistuttu edistämään toimeksiantajan tavoitetta hyödyntää tietoturvallista tuotantodataa ohjelmistotestauksessa.

Keskeiset käsitteet

Anonymisoitu data	Data, josta on poistettu kaikki luonnollista henkilöä koskevat tunnistetiedot. Dataa ei kyetä kohtuullisin keinoin muuttamaan takaisin tunnistettavaksi. (Tietosuojavaltuutetun toimisto s.a. a.)
Data	Tieto, joka on tallennettu tai kerätty johonkin järjestelmään.
Dynaaminen poistaminen	Tunnistetietojen poistaminen, kun data on liikkeessä esimerkiksi tietokannasta testitietokantaan (Raghunathan 2013, 133).
EAL- ja ELA-malli	Extract-anonymize-load -malli ja extract-load-anonymize-malli ovat staattisen tunnistetietojen poistaminen malleja (Raghunathan 2013, 124-127).
Epäsuora tunnistetieto	Tieto, jota ei voi yksinään yhdistää henkilöön, mutta yhdistettynä muihin tietoihin se voi johtaa henkilön tunnistamiseen. Esimerkiksi henkilön sukupuoli tai ammatti. (Tietoarkisto s.a.)
Käyttöympäristö	Ympäristö, jossa ohjelmistoa voi käyttää. Ympäristöjä on erilaisia eri käyttötarkoituksiin, kuten testaus- ja tuotantoympäristö (Raghunathan 2013, 137).
Laravel Factory	Laravel-viitekehyksen ominaisuus, jonka avulla voidaan luoda keinotekoista dataa. Factory-ominaisuutta voidaan hyödyntää Seeder-luokissa. (Laravel s.a a.)
Laravel Seeder	Laravel-viitekehyksen ominaisuus, jonka avulla voidaan alustaa tietokanta keinotekoisella datalla. Dataa voidaan hyödyntää esimerkiksi ohjelmiston testauksessa. (Laravel s.a b.)

Ohjelmistotestaus	Prosessi, jolla testataan ohjelmiston toimivuutta ja suorituskykyä (Kasurinen 2013, alaluku Mitä on ”ohjelmistotestaus”).
On-the-fly-malli	Malli, jolla voidaan poistaa datasta tunnistetiedot ja tallentaa ne tietokantaan reaaliajassa (Cobb s.a.).
PHP Faker	PHP-kirjasto, jonka avulla voidaan luoda keinotekoista dataa, kuten nimiä ja osoitteita (FakerPHP s.a).
Pseudonymisoitu data	Data, josta ilman erillistä aineistoa ei saada pääteltyä luonnollista henkilöä koskevia tunnistetietoja (Tietosuojavaltuutetun toimisto s.a. a).
Konseptitodistus	(Proof of Concept) Pienimuotoinen selvitys, jonka avulla todennetaan, onko konsepti toimiva ja toteutettavissa (Monday.com 25.8.2022).
SQL-skripti	(Structured Query Language) Ohjelmointikielen muodossa oleva tiedosto, joka sisältää yhden tai useamman SQL-kyselyn eli tietokantakyselyn (Mooc 2021).
Staattinen poistaminen	Tunnistetietojen poistaminen, kun data on paikallaan esimerkiksi tietokannassa (Raghunathan 2013, 124).
Suora tunnistetieto	Tieto, jonka avulla henkilö on suoraan tunnistettavissa, kuten koko nimi ja henkilötunnus (Tietoarkisto s.a.).
Tietokanta	Tietojen järjestelmä, joka mahdollistaa tietojen tallentamisen, muokkaamisen ja hakemisen. Tietokanta koostuu tietokantatauluista. (Mooc 2021.)

2 Henkilötietojen käsittely lainsäädännön näkökulmasta

Tietosuojalaki määrittää myös yrityksille yhteiset pelisäännöt tunnistetietoja sisältävän datan käsittelyyn. Yritysten kannattaa ottaa henkilötietojen suojaaminen ja peittäminen vakavasti. Tietosuojavaltuutetun toimistolla on oikeus määrätä sanktioita yrityksille, jotka eivät käsittele henkilödataa lainmukaisesti. Näin kävi esimerkiksi vuonna 2020, kun potilasdatan kanssa toimiva Psykoterapiakeskus Vastaamo epäonnistui henkilödatan lainmukaisessa käsittelyssä ja arkaluonteisia asiakastietoja pääsi lopulta vuotamaan väärille tahoille. Vastaamo sai huomattavan kokoisen seuraamussakon ja ajautui lopulta konkurssiin. (Tietosuojavaltuutetun toimisto 2021.) Henkilötietoja käsittelevien organisaatioiden tulee hallita tunnistetietoja huolellisesti ja tarkoituksenmukaisesti. Vaikuttavien sakkojen lisäksi asiakkaan luottamus organisaation toimintaan loppuu taatusti, jos hänen henkilötietojaan käsitellään huolimattomasti tai niitä päätyy väärille tahoille.

GDPR (General Data Protection Regulation) eli yleinen tietosuoja-asetus on Euroopan unionin yhteinen laki, joka määrittää yhteiset säännöt, kuinka EU:ssa asuvien dataa tulee käsitellä ja mitä oikeuksia henkilöillä on omaan dataansa liittyen. Asetus otettiin käyttöön vuonna 2018. (Tietosuojavaltuutetun toimisto s.a. c.) Yleisen tietosuoja-asetuksen lisäksi EU-maat voivat asettaa omia tietojen käsittelyyn liittyviä paikallisia lakeja. Yleinen tietosuojalaki on asettanut seitsemän (7) perusperiaatetta henkilötietoja sisältävän datan käsittelyyn, joita noudattamatta jättäminen voi aiheuttaa seuraamussakkoja henkilötietoja sisältävän datan käsittelijälle. Niiden periaatteiden noudattaminen tai noudattamatta jättäminen arvioidaan tapaus- ja aineistokohtaisesti. (Clarin s.a.)

Ensimmäinen datan käsittelyn periaate koostuu laillisuudesta, oikeudenmukaisuudesta ja avoimuudesta (lawfulness, fairness and transparency). Tämä tarkoittaa sitä, että henkilön tulee saada tietää, mihin hänen dataansa käytetään ja miksi. Lisäksi hänellä on tiettyyn rajaan asti oikeus pyytää omien tietojensa poistamista. (Clarin s.a.) Kuitenkaan kaikkia tietoja henkilöistä ei voida välittömästi poistaa laillisista syistä, esimerkiksi kirjanpitolaki määrittää omat säilytysaikansa. (Tietosuojavaltuutetun toimisto s.a. b.) Informaatio datan käytöstä tulee antaa selkokielellä ja ilman aiheetonta viivästystä. Lisäksi henkilödatan keräämiseen ja käsittelyyn tarvitaan selkeä lupa henkilöltä, jolta dataa kerätään. (Clarin s.a.)

Toinen periaate on käyttötarkoituksen rajoittaminen (purpose limitation). Tämä tarkoittaa, että henkilödataa ei saa käyttää muuhun kuin alkuperäiseen ja dataa koskevan henkilön hyväksymään käyttötarkoitukseen tai uuteen siihen yhteensopivaan käyttötarkoitukseen. (Clarin s.a.) Tämä tarkoittaa, että jos organisaation on alun perin kerännyt asiakasdataa esimerkiksi tuotteiden kuljetusta varten, se ei voi enää käyttää alkuperäistä hyväksyntää mihinkään siihen liittymättömään asiaan. Dataa voidaan käyttää esimerkiksi kuljetuksen toimivuutta edistävään tutkimukseen, mutta se ei oikeuta lähettämään asiakkaalle markkinointiviestejä. Yleinen tietosuoja-asetus kuitenkin

määrittää, että muun muassa alun perin tieteelliseen tarkoitukseen kerätty data tulee aina olemaan jatkotutkimuksissa alkuperäiseen käyttötarkoitukseen sopiva, kunhan muita datan käsittelyn periaatteita noudatetaan. (Clarin s.a.) Tämä osaltaan helpottaa tieteellisten tutkimusten tekemistä kuitenkaan uhraamatta henkilön oikeuksia.

Tietojen minimointi (data minimisation) on tietosuoja-asetuksen periaate, jonka mukaan henkilötietoja ei saa kerätä, arkistoida tai käsitellä ilman asianmukaista ja pätevää tarkoitusta. Toisin sanoen, organisaation tulee kerätä vain vähimmäismäärä tietoja, jotka ovat välttämättömiä ennalta määrättyyn tarkoitukseen. Näin ollen, datan käsittelijän pitää myös pystyä perustelemaan datan tarpeellisuus kyseisessä kontekstissa. (Clarin s.a.)

Tarkkuus (accuracy) -periaatteen mukaan kerätyn datan tulee olla totuudenmukaista. Henkilöllä on myös oikeus pyytää korjausta virheellisiin tietoihin ja ne tulee korjata ilman tarpeetonta viivästystä. Lisäksi datan paikkansapitävyyttä tulee ylläpitää säännöllisesti datan käsittelijän puolesta. (Clarin s.a.) Virheelliset tiedot voivat aiheuttaa tarpeetonta haittaa henkilölle, sillä esimerkiksi väärin merkityt terveystiedot voivat olla vakava riski henkilön terveydelle.

Varastointirajoitus (storage limitation) -periaate edellyttää, ettei henkilön tietoja säilytetä pidempään kuin kyseinen käyttötarkoitus vaatii. Laillinen säilytysaika määräytyy tapauskohtaisesti, ja dataa saa tai pitää säilyttää kauemmin vain, jos se on perusteltua ja asianmukaista tiettyyn käyttötarkoitukseen nähden. Esimerkiksi tieteellisiin tarkoituksiin dataa saa yleensä säilyttää kauemmin. (Clarin s.a.)

Kuudes periaate, eli eheys ja luottamuksellisuus (integrity and confidentiality), määrittää, millainen suoja henkilön datalle tulee varmistaa. Organisaation on käytettävä asianmukaisia suojatoimia henkilötietojen suojelemiseksi, muun muassa organisaation ulkopuolisilta tahoilta, jotka saattavat käyttää dataa laittomiin ja henkilöä vahingoittaviin tarkoituksiin. Lisäksi suojatoimien tulee estää datan tahaton tuhoutuminen tai katoaminen. (Clarin s.a.)

Viimeinen ja uusin lisäys datan käsittelyn periaatteisiin on vastuullisuus (accountability), joka tarkoittaa sitä, että datan käsittelijällä on velvollisuus varmistaa, että dataa käsitellään edellä mainittujen periaatteiden mukaisesti sekä tarvittaessa todistaa, että niitä noudatetaan. (Clarin s.a.) Velvoittamalla datan käsittelijän noudattamaan näitä periaatteita, lisätään läpinäkyvyyttä ja luottamusta datan käsittelyyn. Tämä auttaa vähentämään mahdollisia riskejä ja vahinkoja, jota vääränlainen datan käsittely voi aiheuttaa.

Tietojenkäsittelyperiaatteet tarjoavat vahvan kehyksen henkilön tietosuojalle ja auttavat organisaatiota toimimaan eettisesti ja turvallisesti datan käsittelyn parissa.

3 Tunnistetietoja sisältävän datan käsittely

Kuten edellisessä luvussa kerrottiin, henkilön yksityisyyden suojaaminen on erittäin tärkeä tehtävä organisaatiossa, jos halutaan saavuttaa asiakkaan luottamus ja yrityksen maineen säilyminen. Organisaatioiden tulisi laatia suunnitelma asiakasdatan suojaamiseksi. Kuitenkin liiketoiminnan kehittämiseen ja palveluiden tarjoamiseen yleensä tarvitaan asiakasdataa. Yksi ratkaisu tähän haasteeseen on tunnistetietojen poistaminen tai piilottaminen asiakasdatasta, jotta vaikka ulkoiset osapuolet saisivat datan käsiinsä, tietoja ei enää voitaisi yhdistää tiettyyn henkilöön.

Tunnistetietojen poistamiseksi tai piilottamiseksi asiakasdata voidaan joko anonymisoida tai pseudonymisoida. Näitä kahta käsitettä ei tule sekoittaa toisiinsa, vaikka usein saatetaan viitata anonymisoimiseen, vaikka data olisi pseudonymisoitu. Dataa pseudonymisoidessa kuitenkin voidaan hyödyntää anonymisointitekniikoita, jotta data olisi tehokkaammin suojattu ulkoisilta osapuolilta. (Enisa 2018.) Seuraavissa kappaleissa käsitellään, kuinka tunnistetietoja sisältävää dataa voi osaltaan suojata anonymisoinnin ja pseudonymisoinnin avulla, miksi se on tärkeää ja mitä ongelmia siihen voi liittyä.

3.1 Datan anonymisointi

Dataa anonymisoidessa datasta poistetaan peruuttamattomasti kaikki tunnisteellinen tieto eli sellaiset tiedot, joilla yksittäinen henkilö tai ryhmä, kuten perhe, voidaan tunnistaa datasta. Jos anonymisointi on onnistunut, dataa ei kyetä enää muuttamaan takaisin tunnistettavaan muotoon dataa käsittelevän organisaation tai ulkoisen tahon toimesta. Jos näin pystytään tekemään datan anonymisoiminen on epäonnistunut. (Tietosuojavaltuutetun toimisto s.a. a.)

Asiakassuhteen päätyttyä asiakastietoja ei voida säilyttää ikuisesti, vaan ne tulee lopulta joko poistaa tai anonymisoida (Tietosuojavaltuutetun toimisto s.a. b). Onnistuneen anonymisoimisen jälkeen dataan ei päde enää yleinen tietosuojalaki, sillä se ei sisällä tunnistetietoja (Tietosuojavaltuutetun toimisto s.a. a). Anonymisoinnin avulla dataa ei tarvitse poistaa kokonaan, vaan sitä voidaan säilyttää tunnistetietoja sisältämättömässä muodossa niin kauan kuin sille on tarve.

Vaikka data anonymisoidaan, siitä ei voi kuitenkaan koskaan saada täysin anonyymia, ja jokaisessa tapauksessa on arvioitava erikseen, riittävätkö käytettävissä olevat teknologiat ja rajallinen ajankäyttö estämään datan takaisin muuttamisen tunnistettavaan muotoon (Tietoarkisto s.a.; Tietosuojavaltuutetun toimisto s.a. a). Mikäli datan takaisin muuttaminen tunnistettavaan muotoon vaatisi kohtuuttomasti resursseja, se voidaan katsoa riittävän anonyymiksi. On tärkeää ottaa huomioon, että teknologioiden kehittyessä kerran anonyymi data voi menettää anonymiteettinsa

tulevaisuudessa. Tästä syystä onkin tärkeää säännöllisesti tarkistaa ja ylläpitää datan anonymiteettia. (Tietosuojavaltuutetun toimisto s.a. a.)

Jos data anonymisoidaan harkitsemattomasti, on mahdollista menettää datasta tärkeitä arvoja, joita ilman dataa ei voida hyödyntää alkuperäiseen käyttötarkoitukseen. Anonymisointi voi siis vaikuttaa datan eheyteen. (Tietoarkisto s.a.)

Datan anonymisointimenetelmä tulee valita aineistokohtaisesti. Kaikki tavat eivät sovi kaikenlaiselle datalle ja kaikki menetelmät eivät suojaa dataa yhtä hyvin kuin toiset. Usein parhaaseen tulokseen päästään yhdistelemällä eri tekniikoita keskenään. Karkeasti menetelmät voidaan jakaa kahteen ryhmään: satunnaistamiseen ja yleistämiseen. (Tietoarkisto s.a.)

3.1.1 Satunnaistavat menetelmät

Satunnaistavilla menetelmillä yritetään poistaa tietojen ja henkilön välinen yhteys muuttamalla datan todenmukaisuutta. Satunnaistavia menetelmiä on kannattavampaa hyödyntää, jos datassa on vain vähän harvinaisia ilmentymiä. Näillä menetelmillä voi olla suurikin vaikutus aineiston käytettävyyteen, eikä ne yksinään takaa datan anonymiteettiä. (Tietoarkisto s.a.)

Kohinan lisääminen on yksi satunnaistamismenetelmistä, joka muuttaa aineiston arvojen tarkkuutta ennalta määrätyllä tavalla. Kohinan avulla pyritään johdonmukaisesti häivyttämään yhteyttä alkuperäisen datan ja henkilöiden välillä. Kohinaa voidaan lisätä esimerkiksi kertomalla yksilöivä arvo, kuten pituus, jollain satunnaisluvulla tai pyöristämällä se kymmenien tarkkuuteen. Myös arvojen lajitteleminen keskiarvoihin lisää kohinaa, kunhan muistetaan, että jokaisessa luokassa on tarpeeksi arvoja, jotta henkilöä ei voida tunnistaa luokan sisältä. (Tietoarkisto s.a.)

Toinen esimerkki satunnaistamismenetelmästä on permutaatio, jossa datan arvojen ja tunnistettavan henkilön välinen yhteys häivytetään satunnaistamalla arvot toiseen henkilöön. Permutaatiota kannattaa hyödyntää silloin, kun tutkittavien muuttujien välillä ei ole korrelaatiota eli yhteyttä. (Tietoarkisto s.a.) Esimerkiksi sukupuolen ja koulutustason välillä ei ole vahvaa korrelaatiota, toisin kuin tulojen ja varallisuuden välillä. Jos vain varallisuusarvo satunnaistetaan, jäljelle jääneestä tulot-arvosta voidaan helposti päätellä siirretty arvo. Satunnaistamisen tarkoituksena on säilyttää aineiston alkuperäinen jakauma. Jakauman säilyminen tarkoittaa, että vaikka arvojen paikka voi vaihtua, niiden suhteellinen määrä pysyy samana. Jotta tutkimuksen tulokset ovat luotettavia ja vertailukelpoisia, on tärkeää säilyttää alkuperäinen jakauma. (tietoarkisto s.a)

3.1.2 Yleistävät menetelmät

Yleistävien menetelmien tarkoitus on joko tunnistetietojen poistaminen tai niiden karkeistaminen yleiselle tasolle. Hyvin tehty yleistäminen säilyttää aineiston käytettävyyden. Yleistäviin menetelmiin kuuluvat muun muassa seuraavat: arvojen poistaminen datasta, luokittelu, karkeistaminen, sekä k-anonymiteetti ja l-diversiteetti. (Tietoarkisto s.a.)

Arvojen poistaminen tarkoittaa tunnistetietojen peruuttamatonta tuhoamista datasta. Suorien tunnisteiden, kuten käyttäjän henkilötunnus ja osoite, lisäksi datasta kannattaa poistaa epäsuorat tunnisteet, kuten IP-osoite ja postinumero, jos niiden yhdistäminen muihin tietoihin mahdollistaisi henkilön tunnistamisen. Yksittäisten avointen kysymysten vastauksia, kuten tutkittavan koulu, voidaan myös poistaa, jos kyseinen tieto on kategorisoitu omaan ryhmäänsä siten, ettei se enää paljasta yksilöllistä tietoa. Koulutiedot voidaan kategorisoida esimerkiksi koulutustason mukaan, jotta yksittäiset koulut eivät ole tunnistettavissa. Jos tarkka muuttuja on tarpeellista säilyttää, se on pidettävä erillään muusta datasta. Lisäksi on suositeltavaa, että aineistosta poistetaan erittäin harvinaiset arvot, kuten harvinaiset sairaudet, sillä ne saattavat paljastaa henkilön henkilöllisyyden. On myös mahdollista, että harvinaisuuden takia kokonainen havaintoyksikkö joudutaan poistamaan aineistosta, jos henkilön tietojen poistaminen on tarpeen muun datan anonymisoinnin varmistamiseksi. (Tietoarkisto s.a.)

Jos muuttujia ei ole pakko poistaa datasta, ne kannattaa luokitella. Muuttujat voidaan luokitella esimerkiksi ikäluokkiin, tuloluokkiin tai kaupunkeihin. Luokittelu voidaan tehdä myös ääriarvojen eli suurimpien tai pienimpien arvojen suhteen ja antaa muista muuttujista tarkkoja arvoja. On kuitenkin tärkeää huomioida, että jos yhdessä luokassa on vain muutama arvo, henkilön tunnistamisriski nousee. Tällöin kannattaa harkita luokituksen karkeistamista esimerkiksi kaupungeista maakuntiin. Karkeistamisen avulla vähennetään arvojen yksityiskohtaisuutta eli siis muutetaan arvojen mittakaavaa. Aineiston luokittelu tulee tehdä analyyttisesti, jotta tunnistettavat tiedot saadaan poistettua aineiston vaikuttamatta sen tilastolliseen hyödynnettävyyteen. Pelkän luokittelun käyttö on suhteellisen heikko anonymisointimenetelmä, sillä henkilön voi edelleen yhdistää tiettyyn aineistoluokkaan. Luokittelun kanssa olisi hyvä käyttää jotain toistakin anonymisointimenetelmää, kuten karkeistamista. (Tietoarkisto s.a.)

Aineiston anonymisoinnin onnistumista voidaan arvioida K-anonymiteetin ja l-diversiteetin avulla. Menetelmiä voidaan hyödyntää esimerkiksi silloin, kun aineistossa on epäsuoria muuttujia, joiden avulla henkilön henkilöllisyys voi paljastua. K-anonymiteetin avulla varmistetaan, että luokitellussa ryhmässä on tarpeeksi arvoja, jotta henkilöä ei voida tunnistaa. Arvojen määrä riippuu paljon aineistosta. K-anonymiteetin heikkous on se, ettei sen avulla voida tarkistaa päätyykö yhteen luokkaan monta harvinaista arvoa. Tämä voi mahdollisesti johtaa siihen, että arkaluontoinen tieto olisi

pääteltävissä. Ongelman ratkaisemiseksi on kehitetty l-diversiteetti, jonka tarkoitus on varmistaa, että jokaisessa K-anonyymissa luokassa on monimuotoisesti arvoja, jotta harvinaista arvoa ei voida tunnistaa. L-diversiteetti ei siis vaikeuta henkilön tunnistamista, vaan se tekee arkaluontoisen tiedon paljastumisesta hankalaa. Epäsuoria muuttujia voidaan myös karkeistaa, jos l-diversiteetti ei toteudu. (Tietoarkisto s.a.)

3.2 Datan pseudonymisointi

Data voidaan anonymisoimisen sijaan myös pseudonymisoida. Datan pseudonymisointi tarkoittaa tietoaineiston tunnistetietojen muuttamista tunnistemattomaan muotoon, mutta data voidaan kuitenkin muuttaa haluttaessa takaisin tunnistettavaksi erillisen tietolähteen avulla. Henkilötiedot voidaan esimerkiksi muuttaa epätodenmukaisiksi tai data voidaan salata erilaisilla koodeilla. Pseudonymisoinnin purkamisessa voidaan käyttää esimerkiksi koodiavainta, jonka avulla data voidaan taas yhdistää oikeaan henkilöön. Tietolähde, jota käytetään salauksen purkamiseen, tulee säilyttää erillään pseudonymisoidusta datasta. Sitä ei tule luovuttaa ulkopuolisille osapuolille, sillä pseudonymisoitu data tai erillinen tietolähde eivät ole yksinään hyödyllisiä pseudonymisoinnin purkamiseen, vaan niitä hyödynnetään yhdessä. Koska data voidaan manipuloida takaisin tunnistetietoja sisältäväksi, vaikkakin erillisen tietolähteen avulla, data kuuluu yleisen tietosuoja säädöksen piiriin. Pseudonymisoitu data voidaan muuttaa anonyymiksi tuhoamalla erillinen tietolähde, jolloin dataa ei voida muuttaa takaisin tunnistettavaksi. Tällöin tietosuoja-asetus ei enää päde dataan. Datan pseudonymisointi tulee toteuttaa huolellisesti ja analyyttisesti, jotta aineistosta saadaan poistettua kaikki, suorat ja epäsuorat, tunnistetiedot, ja sen purkaminen kohtuullisin keinoin ulkoisten osapuolten toimesta ei ole mahdollista. (Enisa 2018.)

Luonnollisesti hyvin toteutettu datan pseudonymisointi lisää datan suojaa ulkopuolisilta tahoilta ja näin ollen henkilön yksityisyydensuojaa ja luottamusta datan käsittelijän pätevyyteen. Ilman erillistä tietolähdettä, jolla pseudonymisointi voidaan purkaa, pseudonymisoidusta datasta ei voida tunnistaa henkilöitä. Kuitenkin tämä olettamus pitää paikkansa vain, jos erillinen tietolähde onnistutaan pitämään suojassa ja erillään pseudonymisoidusta datasta. Jos jokaisella henkilöllä on aina erilainen pseudonyymi eri tietoaineistoissa, eli pseudonyymit ovat kaikille uniikkeja, ulkopuolinen taho ei pysty yhdistämään tietoaineistoja keskenään. Tämä on hyödyllistä, sillä lisätietojen avulla henkilön tunnistaminen aineistosta on usein mahdollista. Pseudonymisoinnin avulla säilytetään myös datan eheys ja hyödynnettävyys, sillä se voidaan tarvittaessa aina purkaa erillisen tietolähteen avulla takaisin alkuperäiseen muotoonsa. Lisäksi hyvin toteutetulla pseudonymisoinnilla voidaan saavuttaa lain määräämät tietosuojavaatimukset henkilön tietosuojaan liittyen. (Enisa 2018.)

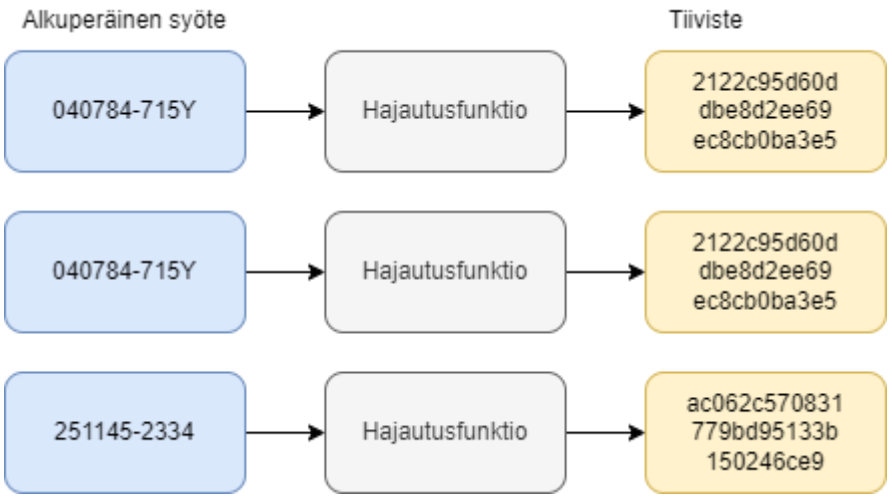
On monia menetelmiä, joita voidaan käyttää henkilön tunnistetietojen pseudonymisointiin. Seuraavissa kappaleissa esitellään muutamia vaihtoehtoja näistä menetelmistä.

3.2.1 Hajautus (Hashing)

Datan hajautus on menetelmä, jossa alun perin syötetty merkkijono, kuten henkilön henkilötunnus tai salasana, muutetaan tunnistamattomaksi tiivisteeksi eli hajautusarvoksi hajautusalgoritmin avulla. Tiiviste on satunnainen merkkijono. (Enisa 2018.) Esimerkiksi henkilön henkilötunnus voidaan hajauttaa 300397-699B-muodosta 215lldfv55-muotoon, jolloin alkuperäinen henkilötunnus on tunnistamaton ja näin suojassa ulkoisilta osapuolilta. Hajautuksen satunnaisuutta voidaan parantaa niin sanotun suola-arvon lisäämisen avulla, jossa alkuperäiseen syötteeseen lisätään suola-arvo eli satunnaismerkkejä ennen hajautusta. Hajautusalgoritmi tulee valita harkitusti ja tutustua eri vaihtoehtojen heikkouksiin ja vahvuuksiin. Algoritmiksi kannattaa valita nykyaikaista teknologiaa hyödyntävä vaihtoehto, jotta se on ajan tasalla. Hajautusta voidaan hyödyntää muun muassa autentikoinnissa sekä datan eheyden tarkistuksessa. Se ei ole kuitenkaan anonymisointimenetelmä, sillä se ei poista henkilötietoja kokonaan, vaan muuttaa ne pseudonyymeiksi, jotka voidaan lisätietojen avulla saattaa takaisin tunnistettavaan muotoon. Huonosti toteutetun hajautuksen mahdollinen ongelma on törmäys eli tilanne, jossa hajautus tuottaa kahdesta eri arvosta saman tiivisteen. (Enisa 2018.) Hajautusalgoritmin tulee siis olla tarpeeksi monimutkainen, jotta törmäystä ei tapahdu, mutta samalla tarpeeksi yksinkertainen, jotta hajautus suoritetaan riittävän nopeasti eikä se vaadi liikaa laskentatehoja (GeeksforGeeks 2022). Hajautusprosessi voidaan suorittaa salaisen hajautusavaimen kanssa tai ilman avainta (Enisa 2018).

Ilman avainta suoritetussa hajautuksessa hajautusalgoritmi tuottaa aina identtisen tiivisteen, jos kaksi alkuperäistä syötettä ovat identtisen keskenään (Enisa 2018). Hajautus on yksisuuntainen prosessi eli tiivistettä ei saada muutettua enää takaisin tunnistettavaan muotoon (Enisa 2022). Hajautuksen avulla hajautetun syötteen tiivistettä voidaan vertailla alkuperäiseen tiivisteeseen. Esimerkiksi käyttäjän kirjautuessa palveluun salasana muutetaan tiivisteeksi alkuperäisen hajautusalgoritmin avulla, jonka jälkeen salasanatiivistettä verrataan tietokannassa olevaan alkuperäiseen tiivisteeseen. Jos tiivisteet ovat identtisiä, salasana on oikea ja käyttäjä pääsee kirjautumaan sillä sisään palveluun. Heikot salasanat ja heikosti toteutettu hajautus mahdollistavat esimerkiksi väsytyshyökkäyksen, jolloin ulkopuolinen taho, eli palveluun hyökkääjä, pyrkii pääsemään järjestelmään sisään hyödyntämällä yleisesti käytettyjä salasanoja ja hajautuskeinoja. Tällöin hajautettuja salasanavaihtoehtoja syötetään, kunnes joku niistä osuu oikeaan. (Gibbs 15.12.2016.) Hajautusta ilman salausavainta tai suola-arvoa ei suositella tunnistetietojen pseudonymisointimenetelmäksi. Tämä johtuu siitä, että kaikki tahot, joilla on mahdollisuus hyödyntää alkuperäistä hajautusmenetelmää, saattavat muodostaa alkuperäisen tiivisteen esimerkiksi aiemmin mainitun väsytyshyökkäyksen avulla. Hajautus on kuitenkin hyödyllinen, esimerkiksi silloin kun halutaan varmistaa datan eheys. Toisin sanoen, hajautuksen hyödyntäminen mahdollistaa syötteen tarkistamisen sen varalta, että sen siirron aikana on tapahtunut muutoksia. Jos annettu syöte ei vastaa alkuperäistä tiivistettä, on

todennäköistä, että joku ulkoinen taho on manipuloinut syötettä sen siirron aikana. On tärkeää tunnistaa olemassa olevat riskit, jos ilman avainta tapahtuvaa hajautusta hyödynnetään tunnistetietojen turvaamiseen ja pseudonymisointiin. (Enisa 2018.)

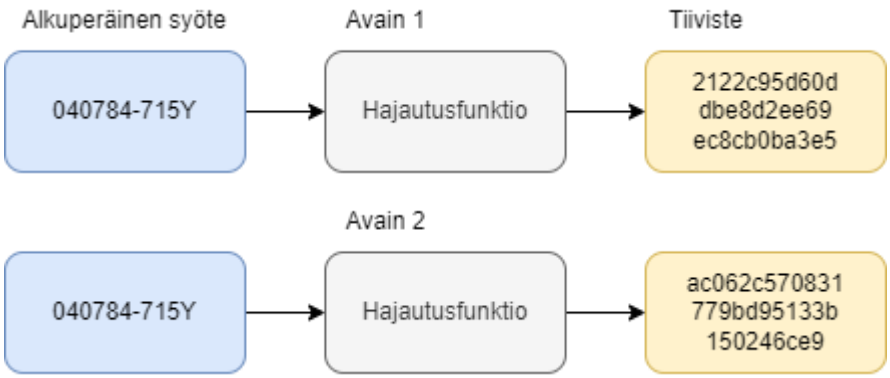


Kuva 1. Hajautus ilman avainta (mukaillen Enisa 2018)

Ilman avainta tehdyn hajautuksen tietoturvaa voidaan lisätä, jos syötteeseen lisätään suola-arvo ennen hajautusta. Tämä johtuu siitä, että suola-arvojen ollessa erilaisia identtisetkin syötteet tuottavat erilaisen tiivisteen. Kuitenkin suola-arvon lisäämiseen liittyy tietoturvariski, jos ulkopuolinen taho saa tietoonsa alkuperäisen suola-arvon. Tällöin ulkopuolinen taho voi asettaa suola-arvon syötteeseen ennen hajautusta ja saa näin sen tulokseksi oikean tiivisteen. Näin ollen suolamerkkijonot tulee säilyttää erillään hajautetusta datasta. Hajautus on vaikeampi murtaa, jos syötteen hajautus ja suola-arvon lisääminen suoritetaan useita kertoja peräkkäin. Se, että suola-arvo lisätään syötteeseen, ei usein takaa riittävää pseudonymisointia, sillä heikon suola-arvon käyttäminen helpottaa kyseisen arvon ja tiivisteen arvattavuutta. Lisäksi suola-arvot usein säilytetään yhdessä tiivisteiden kanssa, huolimatta suosituksista. (Enisa 2018.)

Luotettava tapa luoda pseudonyymejä on hajauttaa syöte salausavaimen avulla. Salausavain on satunnaisesti generoitu merkkijono. Salausavain eroaa suola-arvosta siten, että avain on huomattavan pitkä verrattuna suola-arvoon ja se säilytettään usein erillään muusta aineistosta. Ilman salausavainta ulkoinen osapuoli ei pysty muuttamaan syötettä alkuperäisen tiivisteen kanssa yhteensopivaksi. Hajautuksen tulos riippuu siis sekä alkuperäisestä syötteestä että käytetystä avaimesta. Erilaisia avaimia käyttämällä voidaan hajauttaa myös samanlaiset syötteet aina erilaisiksi tiivisteiksi, jolloin identtisiä tiivisteitä ei pääse syntymään. Tämän vuoksi hajautuksen purkaminen on paljon vaikeampaa salausavaimen kanssa. Tarpeen mukaan on kuitenkin myös mahdollista hajauttaa samanlainen syöte yhteisellä avaimella, jotta hajautuksen tiivisteet ovat identtisiä. Salausavaimen tulee olla riittävän pitkä ja monimutkainen, jotta pseudonymisointia ei voida purkaa ulkoisten

osapuolien toimesta. Salausavain on pidettävä erillään muusta hajautetusta aineistosta, jotta hajautuksen tehokkuus pseudonymisointimenetelmänä ei heikkene. Lisäksi yleisen tietosuoja-asetuksen mukaan salausavain lasketaan lisätiedoksi, jonka avulla henkilö voidaan mahdollisesti tunnistaa, joten myös tästä syystä se tulee pitää erillään pseudonymisoidusta datasta. Jos avain päätyy ulkoisen osapuolen tietoon, avaimella toteutettu hajautus muuttuu tavalliseksi hajautukseksi ilman avainta. Avain voidaan tarvittaessa tuhota, jolloin pseudonymisoitu data muuttuu anonyymiksi dataksi, sillä dataa on mahdoton muuttaa takaisin tunnistettavaan muotoon olettaen, että hajautus on riittävän tehokas. (Enisa 2018.)

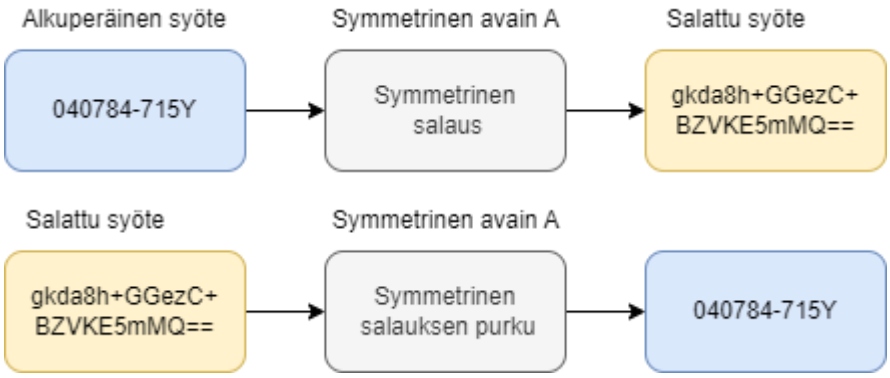


Kuva 2. Hajautus salausavaimen avulla (mukaillen Enisa 2018)

3.2.2 Salaus (Encryption)

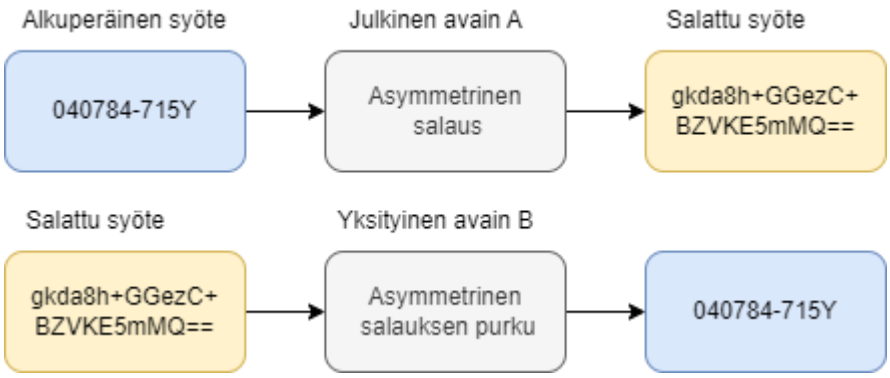
Datan salauksessa salausalgoritmille syötetään selkotekstinen merkkijono, jonka se muuttaa uudeksi, tunnistamattomaksi merkkijonoksi salausavaimen avulla. Toisin kuin hajautus, salaus on kaksisuuntainen prosessi. Tämä tarkoittaa sitä, että salattu syöte voidaan purkaa takaisin selkotekstiseen muotoon salausavaimen avulla valtuutetun henkilön toimesta. Kuten hajautuksessa, salausavain tulee säilyttää erillään hajautetusta aineistosta. Lisäksi sen tulee olla riittävän pitkä ja monimutkainen, jotta salausta ei voida purkaa ulkoisten osapuolten toimesta. Salauksessa tulee käyttää nykyaikaisia algoritmeja. Salausmenetelmät voidaan jakaa kahteen päätyyppiin: symmetriseen ja epäsymmetriseen. (Enisa 2018.)

Symmetriset salausalgoritmit käyttävät yhtä avainta sekä syötteen salaukseen että sen purkamiseen. Symmetrinen salaus on nopeampi suorittaa kuin asymmetrinen salaus. Samalla tavalla kuin hajautuksessa avaimen avulla, symmetrinen salaus mahdollistaa sen, että identtiset syötteet saavat identtiset tiivisteet, mikäli niiden salauksessa käytetään samaa avainta. Lisäksi, jos avain tuhotaan, salauksella pseudonymisoidusta aineistosta poistuu yhteys henkilöihin, ja se muuttuu täysin anonyymiksi. (Enisa 2018.)



Kuva 3. Symmetrinen salaus ja sen purku (mukaillen Enisa 2018)

Asymmetrisessä eli julkisen avaimen salauksessa käytetään kahta avainta: julkista ja yksityistä salausavainta. Yksityistä avainta käytetään salatun syötteen purkamiseen. Yksityistä avainta käyttää vain se henkilö, jolla on oikeus purkaa salaus. Julkinen avain on kaikkien saatavilla ja sen avulla selkokielinen syöte voidaan muuttaa salatuksi. Julkisesta avaimesta ei kuitenkaan voi päätellä yksityistä avainta, vaikka ne ovatkin vastinpareja, joten ulkoiset tahot eivät hyödy julkisen avaimen haltuun saamisesta. Jos identtiset syötteet muutetaan salaiseksi useamman kerran samalla avaimella, molemmat syötteet saavat uniikin salauksen. Erilaiset salaukset voidaan kuitenkin edelleen purkaa yhteisen yksityisen avaimen avulla. Yksi asymmetrisen menetelmän hyödyistä on, ettei yksityistä avainta tarvitse jakaa kaikille osapuolille. Sen sijaan, vain julkinen avain voidaan turvallisesti välittää sitä tarvitseville tahoille. Julkisen avaimen -menetelmä vaatii usein todella suuria avaimia, jotta riittävä turva tiedoille on taattu. Tämä voi puolestaan hankaloittaa menetelmän käyttöönottoa, koska suurien avaimien luominen vie paljon laskentatehoja ja aikaa. Lisäksi nykyaikaisilla asymmetrisen salauksen algoritmeilla ei vielä riittävästi tehoa turvatakseen tietoja kvanttitietokoneiden laskentatehoja vastaan, toisin kuin symmetrisen salauksen algoritmeilla tai salaista avainta käyttävillä hajautusalgoritmeilla. (Enisa 2018.)



Kuva 4. Asymmetrinen salaus ja sen purku (mukaillen Enisa 2018)

3.2.3 Tietojen häivytys (Data masking)

Tietojen häivytys -menetelmässä osa selkokielisen syötteen merkeistä korvataan toisella merkillä esimerkiksi X-kirjaimella, eli henkilötunnus voisi muuttua muotoon 2306XX-XXXX. Menetelmän haittapuoli on, jos kaksi eri arvoa saavat saman häivytetyn arvon, tämä voi johtaa datan törmäykseen. (Enisa 2018.) Lisäksi, jos tietojen häivyttäminen on toteutettu huolimattomasti, se voi johtaa henkilön tunnistamiseen. Tämä voi tapahtua, jos tunnistetiedoista häivytetään väärä osa, kuten henkilön henkilötunnuksesta peitetään vain viimeiset neljä merkkiä. Henkilön syntymäpäivä on vahva tunnistetieto. (Tietoarkisto s.a) Tietojen häivyttämistä ei yleisesti pidetä tehokkaana pseudonymisointimenetelmänä (Enisa 2018).

3.3 Tietosuojasuunnitelma

Arkaluontoisen datan suojeleminen tulee ottaa vakavasti. Parhaiten datan suojeleminen onnistuu, kun se otetaan huomioon jo projektin suunnitteluvaiheessa. On tärkeää arvioida datan arkaluonteisuus sekä mahdolliset tietoturvariskit. Ei voida myöskään luottaa siihen, että edellisessä projektissa toiminut tunnistetietojen suojausmenetelmä toimisi uudessa projektissa tehokkaasti. Projektin suunnitteluvaiheessa tulee tunnistaa projektille tärkeät ja tarvittavat tiedot. Henkilön dataa ei tule kerätä ilman asianmukaista ja pätevää syytä. Tämän avulla ei vain toteuteta yleisen tietosuoja-asetuksen tietojen minimointi -periaatetta, vaan se myös helpottaa tietoturvan takaamista, sillä ei ole turhaa tietoa suojattavana. Tietosuojasuunnitelman tulee ottaa huomioon muutkin yleisen tietosuoja-asetuksen periaatteet ja niiden lisäksi maakohtaiset säännökset datan käsittelylle. (Enisa 2018.)

Suunnittelussa tulee aina ottaa huomioon anonymisoinnin tai pseudonymisoinnin implementoinnin laajuus, datan luonne ja käyttötarkoitus, sekä tietenkin henkilön yksityisyydensuojan varmistaminen. Yksilön yksityisyydensuojan kannalta on myös tärkeää päättää, kenellä on oikeus käsitellä dataa. Jos data on luonteeltaan arkaluontoista, kuten terveystiedot, tulee datan käsittelyssä noudattaa erityistä varoivaisuutta. (Euroopan komissio 2014.)

Päättäminen datan anonymisoinnin ja pseudonymisoinnin välillä riippuu paljolti datan käyttötarkoituksesta. Luonnollisesti, jos kerätystä datasta tarvitaan tunnistetietoja yksittäisistä henkilöistä, valitaan pseudonymisointi tunnistetietojen poistamiseksi. Datan anonymisointi on usein haastavampi prosessi kuin pseudonymisointi, sillä riittävän anonymiteettitason saavuttaminen voi olla hankalaa. Usein voidaan erehtyä luulemaan, että data on anonymisoitu, vaikka se onkin vain pseudonymisoitu. (Enisa 2018.)

Kun on päätetty, onko data anonymisoitava vai pseudonymisoitava, on aika valita menetelmä sen toteutukseen. Menetelmävalinnan tulisi perustua huolelliseen harkintaan, sillä jotkut menetelmät ovat kattavampia kuin toiset ja soveltuvat paremmin tietyntyyppiselle datalle. Voi olla myös hyödyllistä, että datan suojaamiseksi implementoidaan monia eri menetelmiä. Pseudonymisointimenetelmien kanssa voi hyödyntää myös anonymisointimenetelmiä. (Enisa 2018.)

Lisäksi suunnittelussa tulee tunnistaa mahdolliset haasteet tunnistetietojen poistamisessa tai peittämisessä. Ennen datan muuttamista sellaiseen muotoon, ettei sitä voida yhdistää tiettyyn henkilöön, pitää tunnistetiedot ensin tunnistaa datasta. Suorat tunnistetiedot, kuten nimi ja henkilötunnus, on usein helppo tunnistaa datasta, mutta ne eivät kuitenkaan aina ole tunnistetietoja aineistosta riippuen. Henkilötunnus on aina tunnistetieto, koska se on jokaiselle henkilölle uniikki tieto, mutta pelkkä yleinen nimi ei usein riitä tunnistamaan henkilöä esimerkiksi silloin kun aineisto on suuri. Kuitenkin, jos nimi yhdistetään esimerkiksi henkilön työpaikkaan, nimi muuttuu tunnistetiedoksi. Siksi on erityisen tärkeää analysoida kerättyä aineistoa ja tutkia arvoja sekä yksittäisinä tapauksina että osana kokonaisuutta. Epäsuorien tunnistetietojen paikantaminen ei usein ole yhtä suoraviivaista. Epäsuorat tunnistetiedot muodostuvat yleensä aineistokohtaisesti, kuten silloin kun arvon ilmentyminen on ainoa laatuaan tai erittäin harvinainen, kuten harvinainen sairaus. Datan lisääntyessä voi olla myös mahdollista, että aineistoon tulee uusia tunnistetietoja, vaikka ne olisi jo aiemmin poistettu datasta. Jos aineistoon tulee jatkuvasti uutta dataa, tunnistetietojen peittämistä tai poistamista tulee ylläpitää jatkuvasti. (Enisa 2018.)

Lisäksi huonosti suunniteltu ja implementoitu anonymisointi tai pseudonymisointi aiheuttaa usein ongelmia datan suojaamisessa. Markkinoilla on myös todella paljon erilaisia algoritmeja tunnistetietojen poistamiseen, joista osa ei nykypäivänä enää takaa riittävää tietoturvaa. Tämän takia on tärkeää tutkia eri vaihtoehtojen heikkouksia ja vahvuuksia sekä käyttää mahdollisimman nykyaikaisia ratkaisuja. (Enisa 2018.) Teknologia kehittyy jatkuvasti, joten säännöllinen tutkimus parhaista vaihtoehdoista ja vanhojen menetelmien ylläpidosta on tärkeää (Euroopan komissio 2014). Varsinkin kvanttitietokoneiden aikakautena jotkut algoritmit ja menetelmät ovat liian heikkoja vaihtoehtoja (Euroopan komissio 2014).

4 Ohjelmistotestaus

Ohjelmistotestaus on tärkeä osa ohjelmiston elinkaarta, erityisesti nykypäivänä, sillä ohjelmistot ovat usein monimutkaisia kokonaisuuksia. Testauksen tarkoitus on varmistaa, että ohjelmisto tai siihen kehitetty uusi ominaisuus toimii vaaditulla tavalla ilman virheitä. Virheettömyys testataan vertailemalla kehitettyjä ominaisuuksia ennalta määrättyihin vaatimuksiin. Ohjelmistotestaajalla on monipuolinen työnkuva, joka voi vaihdella paljon eri organisaatioiden välillä. Testaajan työnkuvaan voi kuulua ohjelmiston testauksen lisäksi muun muassa vaatimusten selvittelyä, dokumentoinnin kirjoittamista, testauksen tulosten raportointia ja mahdollisten parantamisehdotusten tarjoamista testaustulosten perusteella. (Kasurinen 2013, alaluku Mitä on ”ohjelmistotestaus”.)

Ohjelmisto tai sen ominaisuus on valmis käyttöönotettavaksi, kun testausvaiheessa ei löydetä merkittäviä virheitä. Merkittävällä virheillä tarkoitetaan tässä virheitä, jotka haittaavat ominaisuuden toimintaa. Usein ohjelmisto ei ole koskaan täysin virheetön, ja tuotteesta saattaa löytyä harvinaisia virheitä, joita ei ilmennyt testauksen aikana tai joiden korjaamista ei koeta tarpeelliseksi tuotteen toimivuuden kannalta. (Kasurinen 2013, alaluku Mitä testaustyö tarkoittaa.)

4.1 Ohjelmistotestauksen tärkeys

Organisaatiot sijoittavat huomattavasti resursseja ohjelmistotestaukseen, sillä tuotantoon päässeet ohjelmointivirheet voivat aiheuttaa merkittäviä vahinkoja yritykselle. Ohjelmistotestaus on tärkeä työvaihe organisaation kannattavuuden kannalta, sillä toimivat palvelut ovat tärkeitä asiakkaiden luottamuksen saavuttamiseksi ja varmistavat, että asiakkaat kokevat saavansa rahalleen vastinetta. Lisäksi ohjelmointivirheiden korjaaminen jälkikäteen maksaa organisaatiolle enemmän kuin niiden havaitseminen ja korjaaminen kehitysvaiheessa. (Kasurinen 2013, alaluku Testaus lukuina ja faktoina.)

4.2 Lyhyesti ohjelmistotestausmenetelmistä

Ohjelmistotestaus on monivaiheinen prosessi, johon liittyy erilaisia testausmenetelmiä. Jokaisella testausmenetelmillä on oma käyttötarkoituksena. Yksi esimerkki ohjelmistotestausmenetelmistä on yksikkötestaus, jonka tarkoituksena on testata yksittäisen komponentin tai ominaisuuden toimivuutta, kuten sitä, että käyttäjä pystyy syöttämään oikeat arvot tekstikenttään. Usein tämän menetelmän suorittaa ohjelmoija, joka on kirjoittanut koodin. Tässä vaiheessa ohjelmoijan on helppo suorittaa yksikkötestaus ja korjata mahdolliset virheet. (Kasurinen 2013, alaluku Yksikkötestaus.)

Toinen esimerkki ohjelmistotestausmenetelmistä on integraatiotestaus. Tämän menetelmän avulla testataan, toimivatko yksittäiset ominaisuudet ja komponentit keskenään. Menetelmässä voidaan esimerkiksi testata, toimiiko ominaisuuksien välinen viestintä. Integraatiotestauksessa saatetaan

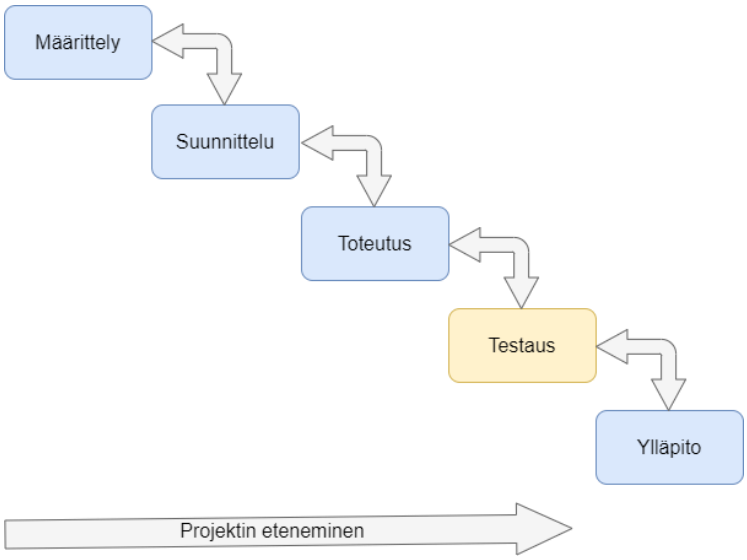
vielä tarvittaessa käyttää keinotekoista dataa ja apukomponentteja. (Kasurinen 2013, alaluku Integrointitestaus.)

Yksikkö- ja integraatiotestauksen jälkeen voidaan siirtyä järjestelmätestaukseen, jonka tarkoitus on testata koko järjestelmän toimivuutta yhtenäisenä kokonaisuutena. Järjestelmätestaus suoritetaan omassa testausympäristössä, joka simuloi lopullista käyttöympäristöä. Järjestelmätestauksen tulokset mahdollistavat tarvittaessa nopeat muutokset järjestelmään. (Kasurinen 2013, alaluku Järjestelmätestaus.)

Viimeiseksi suoritettava testausmenetelmä on hyväksymistestaus, jolloin testattava ominaisuus tai kokonaisuus testataan tuotantoympäristössä. Tämän menetelmän tarkoitus on testata, että uusi ominaisuus on korkealaatuinen ja täyttää sille ennalta-asetetut vaatimukset. Yleensä asiakkaan tulee hyväksyä saavutetut tulokset ja ottaa tuotokset käyttöönsä tässä menetelmän vaiheessa. (Kasurinen 2013, alaluku Hyväksymistestaus.)

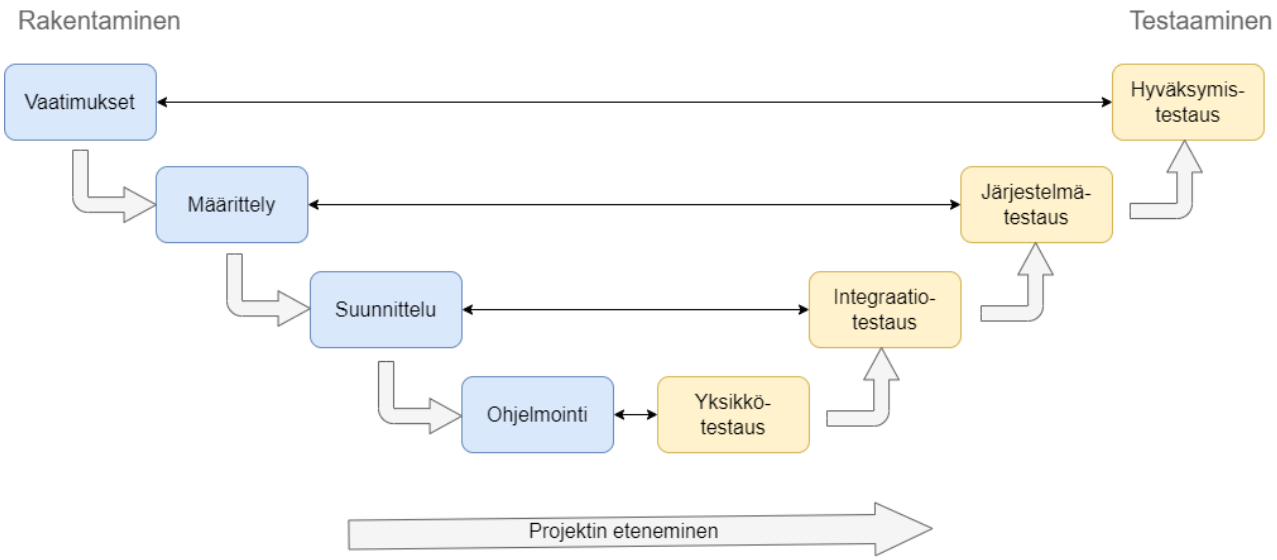
4.3 Ohjelmistotestausmalleja

Perinteisen vesiputousmallin mukaan ohjelmiston elinkaari alkaa vaatimusten määrittelystä ja ohjelmistotestaus tapahtuu ohjelmistotuotannon loppupäässä, tarkemmin sanottuna ohjelmiston toteutuksen ja sen ylläpidon välissä. Jos testauksessa havaitaan ohjelmointivirheitä, ohjelmisto voidaan palauttaa takaisin toteutusvaiheeseen, josta se palaa korjausten jälkeen jälleen testaukseen. Nykyään vesiputousmalli on hyvin vanhentunut tapa toteuttaa projekti, sillä usein ominaisuuksien vaatimusten kerääminen ja määrittely tapahtuu vasta ohjelmointityön aikana. Lisäksi ohjelmisto testataan vasta ohjelmointityön päätyttyä, jolloin virheiden löytyessä on hankalaa ja hidasta palata takaisin vaatimusten uudelleenmäärittelyyn ja niiden implementointiin. (Kasurinen 2013, alaluku Mitä testaustyö tarkoittaa.)



Kuva 5. Vesiputousmalli (Mukaillen Kasurinen 2013, alaluku Mitä testaustyö tarkoittaa)

Parempi vaihtoehto vesiputousmallin sijaan on esimerkiksi V-malli, jossa ohjelmistotestaus on dynaaminen prosessi, joka on mukana ohjelmiston elinkaaren kaikissa vaiheissa, alusta loppuun. V-malli koostuu useasta testausvaiheesta: hyväksymistestauksesta, järjestelmätestauksesta, integraatiotestauksesta ja yksikkötestauksesta. Jokainen testausvaihe testaa eri asioita ja eri tavoin. Ohjelmiston tulee läpäistä kaikki testausvaiheet ennen kuin se voidaan viedä tuotantoon. Mallin avulla testausprosessit pystyvät vaikuttamaan helpommin ohjelmiston toimintaan jo sen alkuvaiheista lähtien. (Kasurinen 2013, alaluku Mitä testaustyö tarkoittaa.)



Kuva 6. V-malli (Mukaillen Kasurinen 2013, alaluku Mitä testaustyö tarkoittaa)

On mahdollista ja suositeltavaa, että ohjelmiston testaus on osa koko tuotteen elinkaarta eikä vain erillinen työvaihe projektin lopussa. Koska ohjelmistotestaus on integraali osa ohjelmointityötä, on tärkeää optimoida testausprosessia ja poistaa siihen liittyvät mahdolliset ongelmat ja hidasteet.

4.4 Testausympäristö

Yksi ohjelmistotestauksen haasteista on luoda testausympäristö, jossa voi luoda todenmukaisia testaustilanteita. Testauksen aloittaminen edellyttää toimivaa testausympäristöä. (Heusser & Kulkarni 2016, 109) Usein ohjelmiston testausympäristönä käytetään osittaista kopiota tuotantoympäristöstä, jotta testaus voidaan suorittaa mahdollisimman realistisessa ympäristössä. Täydellisen tuotantokopion luominen voi kuitenkin viedä liikaa resursseja. (Elfriede 2002, luku 11.)

Ohjelmistotestaus voidaan suorittaa esimerkiksi erillisessä testausympäristössä tai integraatiotestausympäristössä. Erillisessä testausympäristössä voidaan testata ohjelmiston yksittäisiä ominaisuuksia, jotka eivät edellytä ulkoisten integraatioiden käyttöä. Integraatiotestausympäristössä puolestaan voidaan testata monimutkaisia ominaisuuksia, jotka liittyvät ulkoisiin integraatioihin. (Raghunathan 2013, 139-140.) Tämä tutkimus keskittyy erillisessä testausympäristössä suoritettavaan testaukseen, joten integraatiotestausympäristö voidaan sivuuttaa. On kuitenkin tärkeää huomata, että integraatiotestausympäristö on merkittävä osa testausprosessia, koska se tarjoaa totuudenmukaisemman kuvan tuotantoympäristöstä (Raghunathan 2013, 144). On perusteltua tehdä lisätutkimusta integraatiotestausympäristön toiminnasta, implementoinnista ja hyödyistä.

Jotta ohjelmistotestaus olisi mahdollisimman nopeaa ja vastaisi todenmukaisia tilanteita, testidatan tulisi olla mahdollisimman samankaltaista tuotantodatan kanssa (Heusser & Kulkarni 2016, 113). Parhaimmassa tapauksessa testausympäristössä hyödynnetään suoraan tuotantodataa, sillä se usein sisältää monipuolisempaa dataa (Elfriede 2002, luku 11). Testausympäristön datan tulee siis olla samanlaista kuin tuotannon, mutta yleinen tietosuoja-asetus asettaa tälle omat rajoitteensa. Asiakkaan tunnistetietoja ei tulisi käsitellä turhaan eikä turvattomasti (Clarin s.a).

5 Tuotantodatan turvallinen hyödyntäminen testausympäristössä

Yhä useammat organisaatiot hyödyntävät tunnistetietojen poistamista tai piilottamista datasta, sillä se ei vain noudata yleisiä tietosuojalakeja, vaan se myös vähentää kuluja, joita tietovuodot voivat aiheuttaa organisaatiolle. Lisäksi tunnistetietojen piilottaminen datasta suojelee dataa tietovuodon sattuessa. (Raghunathan 2013, 1.)

Ohjelmiston tuotantodatan tunnistetietojen poistamiseksi tai piilottamiseksi sekä sen hyödyntämiseksi testausympäristössä on kehitetty monenlaisia menetelmiä ja työkaluja. Menetelmien avulla pyritään luomaan tuotantodatan kaltaista dataa, mutta piilottamaan siitä tunnistetiedot. Palvelun testattavuuden kannalta on kuitenkin tärkeää, että testauksen kannalta oleellisia tietoja ei poisteta kokonaan. Oleelliset tiedot voidaan muuttaa epätosiksi arvoiksi, mutta niissä tulee säilyttää alkuperäisen datan luonne ja olla yhdenmukaisia sen kanssa. Esimerkiksi asiakkaan nimi ’Matti Meikäläinen’ voitaisiin muuttaa ’Pekka Pouta’ -pseudonyymiksi. Tietoja voidaan myös esimerkiksi salata (encryption) tai häivyttää (data masking). Haasteena voikin olla tasapainoilu testidatan eheyden ja henkilön tunnistettavuuden välillä. (Raghunathan 2013, 123.)

Seuraavissa kappaleissa käsitellään tärkeitä näkökohtia, jotka on otettava huomioon tunnistetietoja piilottaessa, sekä annetaan käytännön vinkkejä siitä, miten se voidaan toteuttaa tehokkaasti ja turvallisesti.

5.1 Tunnistetietojen piilottaminen tuotantodatasta

On tärkeää tarkkailla kolmea eri osa-aluetta, jotta voidaan valita käyttötarkoitukseen sopiva vaihtoehto tuotantodatan tunnistetietojen piilottamiseksi. Nämä ovat datan tila (data state), datan käyttöympäristö (environment) sekä tietovirta (data flow), joka kuvaa, mistä data saapuu ja minne se suuntautuu.

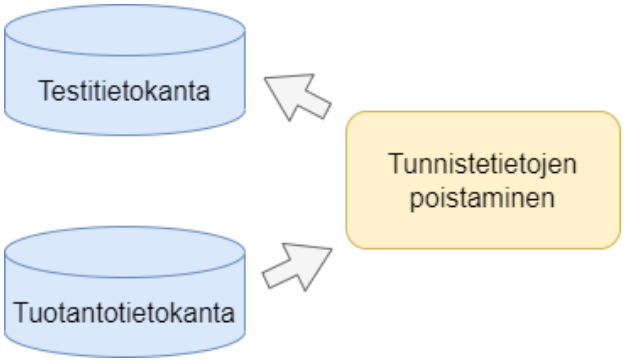
5.1.1 Tunnistetietoja sisältävän datan tila

Tuotantodatan tunnistetiedot voidaan poistaa, kun data on joko paikallaan (staattisena) tai liikkeessä (dynaamisena) (Raghunathan 2013, 123). Tämän tutkimuksen tarkoituksena on keskittyä tuotantodatan hyödyntämiseen testausympäristössä, joten tunnistetietojen poistamista dynaamisesta datasta käsitellään kappaleessa vain lyhyesti.

Tuotantodatan tunnistetiedot voidaan piilottaa staattisesti, kun data on paikallaan (staattisena), esimerkiksi tuotantotietokannassa. Staattista datan piilottamista käytetään yleisesti muissa kuin tuotantoympäristöissä, kuten testaus- ja ohjelmointiympäristöissä. Staattisen tietojen piilottamisen voi toteuttaa joko EAL (extract-anonymize-load) -mallin tai ELA (extract-load-anonymize) -mallin

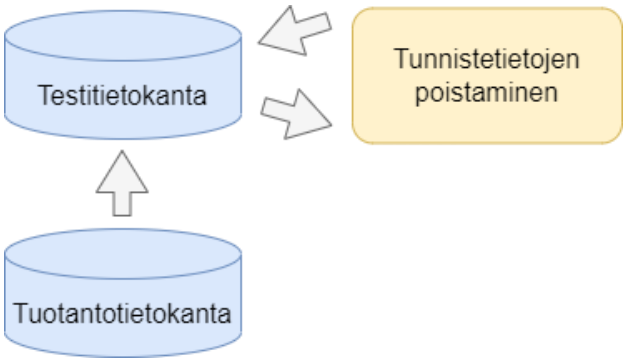
avulla. Vaikka mallit ovat keskenään hyvin samankaltaisia, niiden toiminta tapahtuu eri järjestyksessä. (Raghunathan 2013, 124-127.)

EAL-mallissa tuotantotietokannasta kopioidaan data, jonka tunnistetiedot piilotetaan ennen sen siirtämistä testitietokantaan. EAL-mallin hyöty on, että koko tuotantotietokanta anonymisoidaan ja pseudonymisoidaan vain ensimmäisellä kerralla. Tämän jälkeen vain tuotantotietokantaan lisätystä uudesta datasta poistetaan tunnistetiedot, jonka jälkeen se lisätään osaksi testitietokantaa. EAL-mallin haasteena on kuitenkin usein se, että ulkoisen sovelluksen ei ole sallittua käyttää suoraan tuotantotietokantaa. Tällöin käytetään usein muunneltua EAL-mallia, jossa data poimitaan tuotantotietokannasta tiedostoon, jonka jälkeen tunnistetiedot poistetaan ja data siirretään testitietokantaan. (Raghunathan 2013, 124-126.)



Kuva 7. EAL-malli (mukaillen Raghunathan 2013, 125)

ELA-mallin mukaan tuotantotietokannasta otetaan kopio suoraan testitietokantaan, jonka jälkeen siitä poistetaan tunnistetiedot. Tämä ratkaisee ongelman, jossa ulkopuolinen sovellus ei saa ottaa suoraan yhteyttä tuotantotietokantaan, koska vain kopioon tehdään muutoksia. ELA-malli on helpompi toteuttaa kuin EAL-malli. Kuitenkin tulee varmistaa, että tuotantotietokantakopiota suojellaan tarkkaan ja että siitä ei pääse dataa ulos ennen tunnistetietojen poistamista. (Raghunathan 2013, 124-126.)



Kuva 8. ELA-malli (mukaillen Raghunathan 2013, 126)

Staattisen tunnistetietojen poistamisen tukemiseksi voidaan käyttää myös osittaista menetelmää. Tässä menetelmässä tuotantotietokannasta otetaan vain ne tiedot, jotka ovat testauksen kannalta oleellisia. Tämän tavan etu on, että sitä on helpompi ylläpitää kuin koko tuotantotietokannan kopioimista. Erityisesti jos tuotantotietokannassa on paljon dataa ja testausympäristöjä on useita. Lisäksi tämä tapa on usein halvempi vaihtoehto, koska datan varastoinnista ei tarvitse maksaa yhtä paljon. (Raghunathan 2013, 126-127.)

Tunnistetiedot voidaan myös poistaa tuotantodatasta on-the-fly-mallin avulla. Tämän mallin avulla voidaan jatkuvasti poimia dataa tuotantotietokannasta testitietokantaan. Datan tunnistetiedot poistetaan datan siirron aikana. Tämä on erityisen hyödyllistä, jos ohjelmistoa päivitetään jatkuvasti tai kun halutaan poimia vain osa tuotantodatasta. On-the-fly-malli eroaa EAL-mallista siten, että dataa ei tarvitse kopioida manuaalisesti tuotantotietokannasta, vaan data haetaan ja muokataan automaattisesti aina kun sitä tarvitaan. (Cobb s.a.)

Dynaaminen tunnistetietojen piilottaminen tarkoittaa, että tunnistetiedot piilotetaan liikkeessä, eli kun data siirtyy tuotantotietokannasta käyttäjän nähtäväksi. Toisin kuin on-the-fly-mallissa, dynaamisessa mallissa tuotantodataa ei tallenneta toiseen tietokantaan. (Cobb s.a.) Tätä menetelmää käytetään usein tuotannossa, jotta arkaluontoinen data ei olisi näkyvissä kaikille osapuolille. Tällä tavalla varmistetaan, että alkuperäinen data näkyy vain valtuutetuille henkilöille, mutta muut datan parissa työskentelevät henkilöt voivat silti käsitellä sitä. (Raghunathan 2013, 128.) Dynaamista datan piilottamista ei kuitenkaan yleensä käytetä testausympäristöissä. Tämä johtuu siitä, että testausympäristön on oltava mahdollisimman samankaltainen tuotantoympäristön kanssa, jotta testit ovat mahdollisimman luotettavia. Dynaaminen tunnistetietojen poistaminen saattaa nimittäin piilottaa joitakin ohjelmiston toiminnallisuuksia ja dataa. (Raghunathan 2013, 133.)

5.1.2 Datan käyttöympäristö

Erilaisilla käyttöympäristöillä ja niiden käyttäjillä on erilaisia tarpeita ja käyttötarkoituksia datalle. Siksi on ensiarvoisen tärkeää määrittää nämä tarpeet ja tarkoitukset ennen kuin päätetään, miten tunnistetietoja käsitellään eri käyttöympäristöissä. Lisäksi datan käytön rajoittamiselle ja suojelemiselle on asetettu erilaisia vaatimuksia eri käyttöympäristöissä. Esimerkiksi tuotantoympäristöissä saattaa olla tarpeetonta vastaanottaa dataa ilman tunnistetietoja, kun taas testausympäristöissä olisi hyvä käyttää dataa, josta on piilotettu tunnistetiedot, jotta ne eivät vahingossa päädy ulkopuolisille tahoille. (Raghunathan 2013, 137.)

Erillisissä testausympäristöissä, joissa ei ole ulkoisia integraatioita, tunnistetietojen piilottamiseen tuotantotietokannasta käytetään yleisesti yksinkertaisia staattisia EAL- ja ELA-malleja. Integraatiotestausympäristöissä tarvitaan monimutkaisempia EAL- ja ELA-malleja tunnistetietojen

piilottamiseen, sillä testattava ohjelmisto voi sekä lähettää dataa ulkoiselle ohjelmistolle että vastaanottaa sitä. (Raghunathan 2013, 140-141.) Tämän tutkimuksen puitteissa ei kuitenkaan käsitellä ulkoisia integraatioita, joten niitä ei käsitellä tarkemmin.

5.1.3 Tietovirta

Tunnistetietojen poistamisen suunnittelussa on tärkeää miettiä, miten data kulkeutuu tuotantotietokannasta testaustietokantaan niin, että data säilyy suojattuna. Datan tunnistetietojen poistamisen työtehtävät tulee suorittaa tietovirran oikeassa kohdassa, jotta tunnistetietoja ei pääse vuotamaan testausympäristöihin ja sen implementoiminen on mahdollisimman selkeää. Tietovirran suunnittelulla päätetään, mitä askeleita on suoritettava ja milloin tuotantotietokannan ja testitietokannan välillä. Lisäksi on hyvä miettiä, kuinka usein testitietokannan data olisi hyvä päivittää tuoreimpaan versioon tuotantotietokannasta. Testidataa ei ole välttämättä järkevää kopioida uudelleen, jos testaus on pahasti kesken ja data on jo muunneltu tuotantotietokannan versiosta testeille sopivaksi. (Raghunathan 2013, 153-154.)

On tärkeää pohtia, missä ympäristöissä dataa ilman tunnistetietoja halutaan hyödyntää. Tämä auttaa suunnittelemaan tietovirran etenemistä. Suunnittelua helpottaa myös, jos ensin määritellään, mitä testejä suoritetaan testausympäristössä ja onko testausympäristöjä erilaisia. Kuten aiemmin mainittiin, eri ympäristöillä ja niiden käyttäjillä on erilaisia tarpeita ja oikeuksia datan suhteen. Tuotantodatan hyödyntäminen implementoidaan usein eri aikaan eri ympäristöissä. Tämä helpottaa työn suunnittelua ja vähentää virheiden mahdollisuutta. (Raghunathan 2013, 160-161.)

5.2 Tunnistetietojen poistamistyökaluja

Datan tunnistetiedot voidaan poistaa SQL-skripteillä. Tämä vaatii kuitenkin, että tietyt ehdot täyttyvät. Ensinnäkin ohjelmiston on tallennettava data sisäisesti laitteeseen tai palvelimeen, jossa ohjelmisto on käynnissä. Toiseksi ohjelmiston ei tule jakaa tai vastaanottaa dataa ulkoisten ohjelmistojen kanssa. Kolmanneksi tunnistetietoja sisältäviä arvoja tulee olla vähän, eikä niitä verrata muihin tietokantoihin tallennettuihin arvoihin. Edellä mainitut tilanteet ovat kuitenkin suhteellisen harvinaisia. (Raghunathan 2013, 89.)

Monet yritykset yrittävät ensin poistaa datan tunnistetiedot omilla SQL-skripteillään, mutta pian huomaavat tämän tavan olevan hyvin rajallinen ja aikaa vievä. Syynä tähän on, että SQL-skriptejä täytyy päivittää manuaalisesti aina, kun tietokantarakenne muuttuu uusien ominaisuuksien tai datan vuoksi. Lisäksi SQL-skriptit voivat aiheuttaa ongelmia erityisesti integraatiotesteissä, sillä jos sama tunnistetieto saa eri arvon eri tietokannoissa, datan vertailu eri tietokantojen välillä ei ole mahdollista. (Raghunathan 2013, 61-62.)

Tunnistetietojen poistamiseen kannattaa hyödyntää jo olemassa olevia valmiita työkaluja, koska ensimmäisessä kappaleessa mainitut ehdot eivät ole välttämättömiä. Työkalujen avulla voidaan tuottaa yhtenäistä dataa, ja monet niistä toimivat myös monimutkaisen datan kanssa. Valmiita työkaluja voidaan usein muokata omaan käyttötarkoitukseen sopivaksi, ja niiden avulla voidaan hyödyntää monia eri anonymisointi- tai pseudonymisointimenetelmiä. (Raghunathan 2013, 62.) Työkalujen implementointi erilaisille alustoille ei ole yhtä aikaa vievää kuin oman ratkaisun keksiminen ja implementointi (Raghunathan 2013, 90).

Työkalujen valinta tulee tehdä harkiten ja valita organisaatiolle ja ohjelmiston datalle sopiva sekä luotettava vaihtoehto. Kaikki työkalut eivät sovellu kaikkiin tarpeisiin, eikä niitä voi aina muokata tai yhdistää omiin menetelmiin. Lisäksi on huomioitava työkalun hinta ja vertailtava eri vaihtoehtoja. Valittu työkalu voidaan ottaa käyttöön vähitellen ja testata sen toimivuutta pienellä määrällä dataa. (Raghunathan 2013, 62-64.) Organisaation omien ohjelmistojen tekniikoita kannattaa vertailla työkalun kanssa ja määrittää, ovatko ne yhteensopivia toistensa kanssa ja onko työkalussa turhia ominaisuuksia. Lisäksi tulee varmistaa, että työkalu tukee käytössä olevaa tietokantatyyppiä, kuten MySQL tai PostgreSQL. (Raghunathan 2013, 96-99.)

Kun valitaan työkalua, tulee tarkastella datan käyttötarkoitusta, luonnetta ja määrää. Joitakin työkaluja ei voida käyttää monimutkaiselle datalle, kuten kuville tai tiedostoille. Jos organisaation data on erittäin herkkäluontoista, esimerkiksi terveystiedot, on erityisen tärkeää käsitellä sitä huolellisesti. (Raghunathan 2013, 62-64.) Työkalun tulee täyttää tietosuojalainsäädännön määrittelemät ehdot datan käsittelylle ja peittää kaikki tunnistetiedot riittävän voimakkaasti. On tärkeää pohtia, tarvitaanko dataa muuttaa takaisin tunnistettavaan muotoon ja tukeeko valittu työkalu tätä. Lisäksi on syytä varmistaa, pystyykö valittu työkalu luomaan realistisen näköistä dataa tai käsittelemään osittaista dataa sen sijaan, että käsiteltäisiin koko tietokanta kerralla. Usein organisaation on hyödyllistä päivittää dataa säännöllisesti ja automaattisesti, joten on tärkeää selvittää, tukeeko työkalu tätä joustavasti ja riittävän nopeasti. (Raghunathan 2013, 91-99.)

5.3 Tunnistetietojen poistaminen vaiheet

Tunnistetietojen poistaminen tai piilottaminen tuotantodatasta toteutetaan vaiheittain. Ensimmäisessä vaiheessa tarkastellaan dataan liittyviä lakeja, määritellään toimenpiteet datan turvaamiseksi ja määrätään vastuuhenkilöt edellä mainittuihin tehtäviin. Lisäksi tunnistetiedot kategorisoidaan esimerkiksi terveys-, talous- tai henkilötietoihin. Ensimmäinen vaihe on suoritettava ennen kuin prosessi tunnistetietojen piilottamiseksi voidaan aloittaa. (Raghunathan 2013, 197-200.)

Tunnistetietojen poistamisprosessin seuraavat vaiheet on tehtävä ensimmäisessä kohdassa määritettyjen raamien sisällä. Toisessa vaiheessa (Application Architecture Analysis) tunnistetaan

lähteet, joista tunnistetiedot tulee peittää, kuten tuotantotietokannat. Seuraavassa vaiheessa (Sensitivity Analysis Phase) etsitään arvot, jotka sisältävät tunnistetietoja. Löydettyjen arvojen tärkeysjärjestys määritellään. Lisäksi tietokantarakennetta tutkitaan ja löydetään arvojen yhteydet toisiinsa sekä niiden nimet tietokannassa. Tämän jälkeen siirrytään vaiheeseen (Anonymization Design Phase), jossa valitaan tarvittavat menetelmät datan tunnistetietojen poistamiseksi tai piilottamiseksi. Toiseksi viimeisessä vaiheessa (Anonymization Implementation, Testing, and Rollout Phase) toteutetaan edeltävässä vaiheessa päätetyt menetelmät, ja testataan menetelmien implementoinnin onnistumista. Viimeisessä vaiheessa (Operations phase) tunnistetietojen piilottaminen sisällytetään osaksi ohjelmiston päivitysprosessia, eli se tulee huomioida aina ohjelmiston päivittämisen yhteydessä. (Raghunathan 2013, 197-201.)

6 Tietokantadatan tunnistetietojen poistamisen toteutus

Tässä luvussa käsitellään toimeksiantajayritykselle toteutettavan projektin elinkaarta lähtötilanteesta lopullisiin tuloksiin. Projekti keskittyy tietokantadatan tunnistetietojen poistamiseen ja siihen liittyviin ongelmiin ja ratkaisuihin. Luvun valinnat pohjautuvat vahvasti tietoperustassa kerättyihin tietoihin, joita hyödynnetään kehityksen jokaisessa vaiheessa.

6.1 Lähtötilanteen kartoitus

Projektin toimeksiantajana toimii ohjelmistoalan yritys X Oy, joka tarjoaa laskutus- ja kirjanpitopalveluja asiakkailleen. Tunnistetietojen poistaminen tuotantodatasta tehdään organisaation erään palvelun testausympäristöä varten sekä palvelun tuotantodataa hyödyntäen. Keskeiset tiedot palvelussa sisältävät palvelun yritykset sekä niiden laskut, menot ja asiakkaat.

Tuotteen testaus toteutetaan osittain vesiputousmallilla ja osittain ketterän kehityksen menetelmillä. Tavoitteena on siirtyä vähitellen enemmän ketterän kehityksen pariin, jotta testaus saadaan paremmin integroitua kehityksen jokaiseen vaiheeseen. Testausympäristö vaihtelee testaajien välillä: jotkut testaavat omassa paikallisessa ympäristössään, kun taas toiset käyttävät yhteisestä testausympäristöä, jossa on yhteinen tietokanta. Tuotteen testaus keskittyy lähinnä järjestelmä- ja hyväksymistestaamiseen. Yhtenäisen testausympäristön puuttuminen aiheuttaa usein ongelmia, sillä eri ympäristöissä on erilaisia puutteita. Yhteistä testausympäristöä ei ylläpidetä säännöllisesti. Paikallisen testausympäristön pystyttäminen on puolestaan hidasta ja aikaa vievää. Paikallisen ympäristön vahvuus on kuitenkin se, että eri ominaisuuksien testausten välillä on helppo siirtyä.

Toimeksiantajayrityksen salassapitovelvollisuuden vuoksi työn tuloksia, ohjelmiston tietokantarakennetta sekä sen sisältöä esitellään yleistäen. Esimerkiksi tietokantataulujen nimet ja arvot saattavat poiketa todellisuudesta. Lisäksi työssä ei suoraan esitellä SQL-lausekkeita, sillä ne paljastavat osaltaan tietokannan rakenteen ja sisällön. Projektista otetut kuvankaappaukset eivät sisällä oikeaa asiakasdataa asiakkaiden tietosuojan kunnioittamiseksi.

Projektin lopputulokset on tarkoitettu toimeksiantajayrityksen ohjelmistotestaajille ja ohjelmoijille, sekä muille henkilöille ja organisaatioille, jotka haluavat paremmin ymmärtää, kuinka tuotantodataa voidaan hyödyntää turvallisesti ja luottamuksellisesti ohjelmistotestausympäristössä.

6.2 Projektin tarkoitus ja tavoite

Tämän projektin tavoitteena on kehittää tietotekninen ratkaisu olemassa olevan testausprosessin parantamiseksi. Toimeksiantajan tuotteiden testausympäristöissä käytetään tällä hetkellä Laravel-viitekehyksen Seeder- ja Factory-ominaisuuksia sekä PHP Faker -kirjastoa keinotekoisen datan

luomiseen. Vaikka näillä työkaluilla usein saadaan luotua tarpeeksi aidon näköistä dataa, se on usein melko yksinkertaista. Keinotekoisen asiakkaan tiedoissa voi esimerkiksi olla vain muutama tänä vuonna luotu lasku tai meno. Asiakkaan datasta puuttuu kuitenkin monipuolisuus, kuten erilaisia laskuja ja pidempi laskutushistoria. Valmiiden työkalujen käyttö edellyttää myös jatkuvaa ylläpitoa ja manuaalista päivittämistä ohjelmoijan toimesta. Lisäksi keinotekoisten asiakkaiden välillä ei usein ole paljon eroavaisuuksia, ja jotkut asiakkaat koostuvat jopa puutteellisesta datasta. Heiltä saattaa esimerkiksi puuttua yrityksen osoite, joka täytyy asettaa palvelun käyttöönoton yhteydessä. Osoitteen puuttuminen voi aiheuttaa odottamatonta toimintaa palvelussa.

Nykytilanteessa ohjelmistotestaaja joutuu usein muokkaamaan manuaalisesti testitietokannan dataa. Tämä ei kuitenkaan luo usein todenmukaisia tilanteita, sillä yksi tapahtuma palvelussa voi johtaa useiden tietokantakenttien muuttumiseen. Jos testaaja ei tiedä kaikkia muokkautuvia kenttiä, testausvaihe voi hidastua tai jopa epäonnistua. Lisäksi, jos testaaja tekee virheitä datan manuaalisessa muokkaamisessa paikallisessa ympäristössä, koko tietokanta voidaan joutua tyhjentämään.

Puutteellinen tai virheellinen data saattaa johtaa testaajan virheellisesti oletukseen, että palvelussa tai sen jossain toiminnallisuudessa on ohjelmointivirhe, vaikka todellisuudessa ongelma johtuisi datan puutteellisuudesta. Tämä hidastaa tarpeettomasti ohjelmistotestausprosessia ja saattaa jopa viivästyttää koko toiminnallisuuden kehitystä, jos ominaisuus palautetaan turhaan takaisin ohjelmoijien työlistalle. Jos ohjelmistotestaaja haluaa olla varma, ettei keinotekoinen data vaikuta testaustulokseen, yksinkertaisin tapa on luoda uusi asiakastili. Tämä voi kuitenkin olla erittäin hidasta, jos testiasiakas tarvitsee paljon dataa. Lisäksi uuden asiakkaan data ei ole yhtä monimuotoista kuin pitkään asiakkaana olleen oikean yrityksen data.

Datan monimuotoisuuden puutetta ja virheellisyyttä voidaan mahdollisesti ehkäistä hyödyntämällä tuotantodataa testausympäristössä. Tällä tavoin saadaan aitoa ja monimuotoista dataa testauskäyttöön, mikä voi nopeuttaa testaus- ja ohjelmointiprosesseja sekä vähentää ohjelmointivirheiden mahdollisuutta tuotantoympäristössä. Projektissa tutkitaan, kuinka tietokantadatasta voidaan piilottaa asiakkaiden tunnistetiedot, jotta dataa voidaan hyödyntää turvallisesti.

6.3 Projektin rajaukset

Projekti rajautuu henkilötietojen suojaamiseen tietokannasta ja sen hyödyntämiseen ohjelmistotestausympäristössä. Tutkimuksen aikataulun puitteissa tutkimus ei koske testausympäristössä toimivia ulkoisia integraatioita. Ulkoisten integraatioiden huomioiminen on kuitenkin tärkeää mahdollisten tulevien iteraatioiden kohdalla.

Produkti on konseptitodistus (Proof of Concept), sillä projektin aikataulu on rajallinen ja tuotantotietokannan koon vuoksi on mahdotonta poistaa tunnistetiedot tuotteen tuotantotietokannan

jokaisesta tietokantataulusta. Koko tuotantotietokannan käsitteleminen on myös tarpeetonta, sillä tuotoksen tarpeellisuus, hyödyllisyys ja heikkoudet voidaan selvittää pienemmän kokonaisuuden avulla.

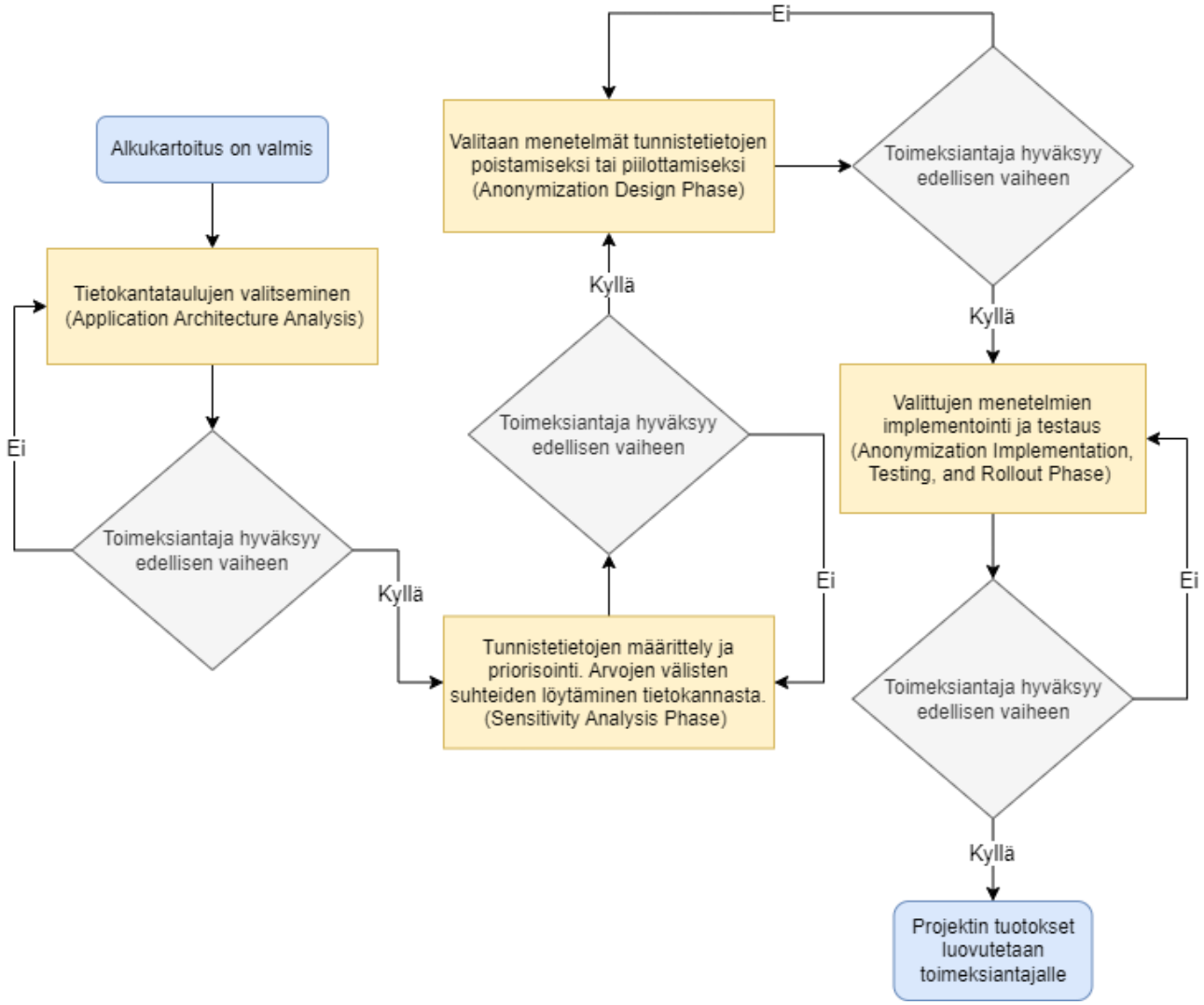
6.4 Onnistumisen mittarit

Yksi työn onnistumismittareista on luotettava, selkeä ja kattava vastaaminen johdannossa esitettyihin tutkimuskysymyksiin. Toinen onnistumismittari on, että projektissa on onnistuttu edistämään toimeksiantajayrityksen tavoitetta hyödyntää tietoturvallista tuotantodataa ohjelmistotestauksessa. Lisäksi voidaan todeta, että lopputulos on hyvä, kun tunnistetietojen poistamisen jälkeen yksittäisiä henkilöitä, eli asiakkaita, ei voida tunnistaa datasta kohtuullisin keinoin. Datan tulee kuitenkin edelleen olla käyttökelpoista, koska muuten sitä ei voida hyödyntää testausympäristössä.

6.5 Projektin prosessikuvaus

Projekti toteutetaan noudattaen ketterän kehittämisen periaatteita. Tämä tarkoittaa, että jokaisesta päävaiheesta (kuva 9) voidaan tarvittaessa palata takaisin aiempaan vaiheeseen. Näin mahdolliset muutokset projektin aikana voidaan helposti sisällyttää osaksi projektia.

Kun alkukartoitus on saatu valmiiksi, voidaan aloittaa projektin toiminnallinen osuus. Projekti etenee Raghunathanin (2013) teoksessa mainittujen vaiheiden mukaisesti, pois lukien ensimmäinen ja viimeinen vaihe. Ensimmäistä vaihetta ei toteuteta projektissa, sillä se koskee yrityksen sisäisiä prosesseja, kuten toimeksiantajayrityksen tietosuojavastuuhenkilöiden valintaa ja yrityksen omia datan käsittelykäytäntöjä. Lisäksi on jo tiedossa, että dataan liittyvät lait määräytyvät GDPR:n ja Suomen paikallisten tietosuojalakien perusteella. Viimeinen vaihe rajataan projektin ulkopuolelle, sillä se käsittelee tunnistetietojen huomiointia tuotteen tulevissa iteraatioissa, kun taas projekti keskittyy vain tuotteen nykytilaan ja dataan. Prosessikuvaksessa (kuva 9) esitetyt hyväksymisvaiheet mainitaan tekstissä vain silloin, jos toimeksiantaja ei hyväksy vaiheen tuloksia. Tällöin kerrotaan syyt hylkäämiseen ja tehdään tarvittavat korjaukset kyseiseen vaiheeseen.



Kuva 9. Projektin prosessikuvaus (mukaillen Raghunathan 2013, 197-201)

6.6 Tietokantataulujen valitseminen

Aloitin projektin paikallisessa ympäristössä, sillä testaaminen suoraan tuotantotietokannassa olisi riskialtista. Paikallinen tietokanta on kuitenkin rakenteeltaan samanlainen kuin tuotantotietokanta, eli molemmissa on samat taulut ja sarakkeet. Paikallisen tietokannan data on kuitenkin alustettu keinotekoisella datalla.

Kuten lähtötilanteen kartoituksessa mainittiin, toimeksiantajayrityksen toiminta keskittyy kirjanpito-palveluiden tarjoamiseen asiakkailleen. Tästä syystä tärkeimmiksi tietokantatauluiksi valikoituivat ne, jotka liittyvät yrityksen perustietoihin, laskuihin, menoihin, ja asiakkaisiin. Toteutuksessa pyrittiin käyttämään mahdollisimman monipuolista dataa. Useissa tietokantataulussa oli samoja tietoja, joten vain yksi niistä valittiin työhön. Tämä johtui siitä, että konseptitodistuksen kannalta ei ollut tarpeellista puhdistaa tunnistetietoja kaikista tauluista. Tietokantataulujen karsiminen voi kuitenkin

aiheuttaa ongelmia. Jos kaikkia tunnistetietoja ei poisteta kaikista tietokantataulusta, työn testaaminen oikealla tuotantodatalla voi olla mahdotonta tietoturvariskien vuoksi. Lisäksi asiakkaan dataa ei tule käsitellä tarpeettomasti. Karsittuja tietokantatauluja ei myöskään voida yksinkertaisesti tyhjentää, sillä se voi aiheuttaa ongelmia testiympäristössä. Kaikkia tietokantatauluja ei ole mahdollista käsitellä työn aikana, joten työ etenee muutamilla tietokantataululla. Mahdolliset ongelmat ratkaistaan työkalun valinnan ja implementoinnin aikana.

Ensimmäiseksi valitsin sopivimmat tietokantataulut projektissa käytettäviksi. Kävin tietokannan läpi taulu kerrallaan ja kirjasin Excel-taulukkolaskentaohjelmaan ylös ne tietokantataulut, jotka liittyivät edellisessä kappaleessa mainittuihin yrityksen tietoihin. Sen jälkeen karsin pois tietokantataulut, jotka eivät sisältäneen tunnistetietoja. Jäljelle jääneistä tietokantatauluista valitsin lopulta ne, joissa oli monipuolisesti dataa. Lisäksi, jos taulukossa oli kaksi eri tietokantataulua, jotka sisälsivät paljon samoja arvoja, poistin toisen niistä. Valintani perustui siihen, kummassa tietokantataulussa oli enemmän tunnistetietoja. Säilytin kuitenkin muutaman samanlaisen tietokantataulun, jotta voisin myöhemmin kokeilla, voiko valitsemallani työkalulla muokata helposti samoja arvoja eri tietokantatauluista. Lisäksi valitsin yhden tietokantataulun, jossa kaksi saraketta muodostivat avain-arvo-pareja. Lopulta projektiin valittiin yhteensä yhdeksän tietokantataulua. (Liite 1.)

6.7 Tunnistetietojen määrittely

Jatkoin projektia kirjaamalla taulukkoon valittujen tietokantataulujen sarakkeet, jotka kuvaavat tunnistetietoja. Merkitsin taulukkoon kunkin sarakkeen nimen ja sen datatyypin, kuten totuusarvon (boolean), kokonaisluvun (int) tai merkkijonon (string). Lisäksi määritin jokaiselle sarakkeelle arkaluontoisuutta kuvaamaan, onko se suora vai epäsuora tunniste. Mikäli taulun sarake kuvaa arvoa, joka voi olla kontekstista riippuen sekä suora että epäsuora tunniste, tein siitä erikseen maininnan. Jos sarake kuvasi jotain ulkoisen integraation arvoa, merkitsin sarakkeen INTEGRAATIO-sarakkeeksi. (Liite 1.) Päätin olla laskematta lukuja, kuten laskujen summia tai yrityksen tulosta, tunnistetiedoiksi, sillä niiden odotetaan muuttuvan samassa suhteessa. On kuitenkin tärkeää muistaa, että varsinkin isot luvut voivat olla tunnistetietoja, esimerkiksi jos yrityksen tuotto on suuri.

Kun olin saanut tehtyä alustavan version valituista sarakkeista, kävin ne läpi toimeksiantajan yhteyshenkilön kanssa, minkä seurauksena taulukkoon lisättiin muutamia uusia sarakkeita. (Liite 1.) Alla on esimerkki (taulukko 1) siitä, kuinka yritykset-tietokantataulu käsiteltiin.

Taulukko 1. Yritykset-tietokantataulun käsittelyä

Sarake	**Datatyyppi**	**Arkaluontoisuus**
alv-numero	VARCHAR	Suora tunniste
INTEGRAATIOT	-	-

Sarake	Datatyyppi	Arkaluontoisuus
nettisivusto	VARCHAR	Suora/Epäsuora tunniste
nimi	VARCHAR	Suora tunniste
puhelinnumero	VARCHAR	Suora/Epäsuora tunniste
sähköposti	VARCHAR	Suora/Epäsuora tunniste
y-tunnus	VARCHAR	Suora tunniste

6.8 Työkalun ja menetelmien päättäminen

Valitsin muunnellun EAL-mallin tunnistetietojen poistamiseen. Valitsin mallin, koska en tarvinnut erityisiä lisäoikeuksia tuotantotietokannan kopioimiseen, eikä minun tarvinnut kytkeä työkalua toimimaan AWS-pilvipalvelun kanssa, toisin kuin normaalin EAL-mallin kanssa. En valinnut ELA-mallia, sillä se on riskialtis tapa poistaa tunnistetiedot. Tämä johtuu siitä, että tuotantodata kopioidaan suoraan testitietokantaan, jonka jälkeen datasta vasta poistetaan tunnistetiedot.

Tutkin erilaisia työkaluja, joita voidaan käyttää tunnistetietojen poistamiseen tietokannoista. Valitsin työkaluja, jotka toimivat MySQL-tietokannan kanssa ja joita voi hyödyntää muunnellussa EAL-mallissa. Työkalun tulee pystyä poistamaan erilaisia tunnistetietoja, kuten nimiä, henkilötunnuksia ja tilinumeroita sekä muokkaamaan tietokantatauluja niin, että data pysyy yhdenmukaisena ja realistisen näköisenä. Lisäksi työkalun tulee olla helppokäyttöinen, tehokas ja edullinen. Mikäli tunnistetietojen poistaminen halutaan tulevaisuudessa automatisoida, työkalun tulee toimia myös komentoliittymän kautta. Näiden kriteerien perusteella valitsin Dataveil-työkalun tunnistetietojen poistamiseen. Muut tutkimani työkalut olivat joko liian kalliita, vaikeakäyttöisiä tai ilmaisversioita tai kokeiluja ei ollut saatavilla.

Selvitin seuraavaksi, miten tunnistetiedot tulee poistaa jokaisesta valitsemastani tietokantataulusta. Pohdin, mitkä arvot ovat tarpeellisia testauksen kannalta. Esimerkiksi, jos asiakkaalla on kaksi osoitetta, toista niistä tuskin tarvitaan ja se voidaan poistaa. Olin tarkempi suorien tunnistetietojen kanssa kuin epäsuorien. Datan tulisi saada eri arvo jokaisella pseudonymisointikerralla. Jos esimerkiksi yrityksille annetaan aina sama nimi, datan tietoturva heikkenee, joten tätä ei pitäisi tehdä ilman hyvää syytä. Kokonaisuudessaan pseudonymisoin tietokantataulut, koska alkuperäiset arvot voidaan aina palauttaa tarvittaessa tuotantotietokannan avulla, sillä pseudonymisoidun datan ID:t ovat edelleen samat kuin tuotannossa. Käytin kuitenkin pseudonymisoinnin lisäksi myös anonymisointitekniikoita, esimerkiksi silloin, kun poistin kokonaan arvoja. (Liite 1.)

Yleensä toimeksiantajan testausympäristössä keskitytään järjestelmätestaukseen, joten pyrin välttämään menetelmiä, jotka rajoittavat datan monipuolisuutta. Joihinkin sarakkeisiin jouduin kuitenkin

asettamaan vakioarvon suoraan testidatasta kaikille yrityksille. Keinotekoisten vakioarvojen käyttämisellä on huono puoli: mitä enemmän niitä käytetään, sitä enemmän testidata poikkeaa tuotantodatasta ja sen monimuotoisuus vähenee. Vakioarvojen käyttö oli välttämätöntä silloin, kun tunnistetieto, kuten y-tunnus tai henkilötunnus, validoidaan ohjelmiston ympäristössä, eikä työkalussa ollut keinoa niiden generointiin. Lisäksi asetin ulkoiset integraatiot vakioarvoksi, sillä niiden puuttuminen voi vaikuttaa palvelun toimintaan. Tietokantakenttiin, jotka on salattu Laravel-viitekehyksen avulla, oli myös pakko käyttää vakioarvoja. Vaikka Dataveil tarjoaa arvojen salausta, se ei ole yhteensopiva Laravelin oman salaustavan kanssa, joten palvelu ei tunnista työkalun käyttämää salaustapaa. (Liite 1.) Alla on esimerkki (taulukko 2) siitä, kuinka yritykset-tietokantataulun käsittelyä jatkettiin lisäämällä sarakkeen käsittelytapa taulukkoon.

Taulukko 2. Yritykset-tietokantataulun käsittelyä

Sarake	**Datatyyppi**	**Arkaluontoisuus**	**Käsittelytapa**
alv-numero	VARCHAR	Suora tunniste	Aseta vakioarvo
INTEGRAATIOT	-	-	Aseta vakioarvo
nettisivusto	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
nimi	VARCHAR	Suora tunniste	Satunnaista
puhelinnumero	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
sähköposti	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
y-tunnus	VARCHAR	Suora tunniste	Aseta vakioarvo

6.9 Työkalun implementointi

Aloitin toteutuksen paikallisessa ympäristössä ja tietokannassa. Päätin luoda uuden tietokannan, johon pseudonymisoitava tietokanta voidaan kopioida. Käynnistin WSL-alijärjestelmän, joka mahdollistaa Linux-komentoriviympäristön käytön Windows-käyttöjärjestelmässä. Sieltä käynnistin MySQL-komentorivikäyttöliittymän, jonka avulla loin uuden tietokannan. Tämän jälkeen loin kopion pseudonymisoitavasta tietokannasta käyttäen mysqldump-työkalua. Työkalu luo kopion sql-tiedostona, joka koostuu SQL-komennoista, joiden avulla tietokanta voidaan tarvittaessa luoda uudelleen. Tietokantakopio tulee vielä asettaa aiemmin luotuun tietokantaan. Käyttäjällä tulee olla lukuoikeus tietokantaan kopion luomiseksi, ja kun kopio asetetaan tietokantaan, käyttäjällä tulee olla luomisoikeus.

Komennot tietokannan luomiseen:

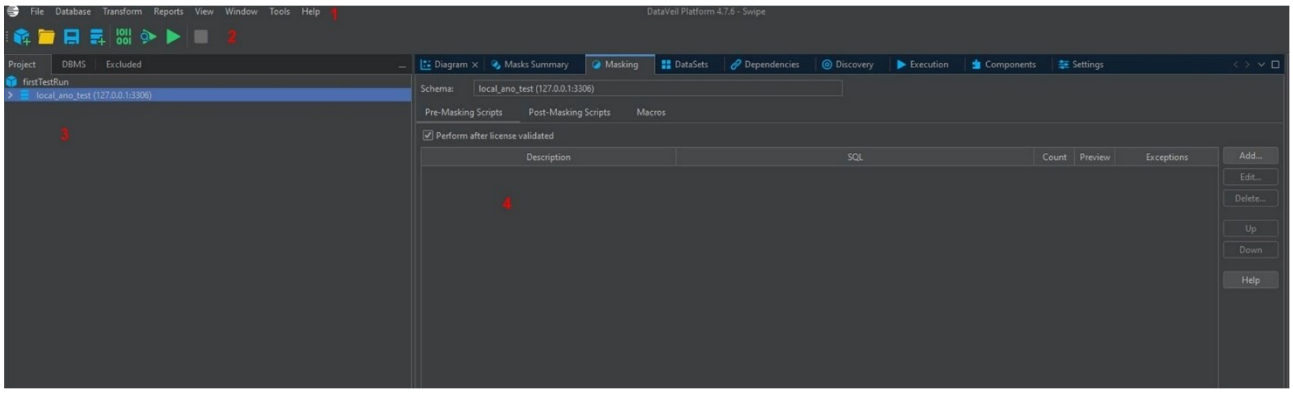
```
1. $ sudo mysql
2. mysql> CREATE DATABASE local_ano_test;
3. mysql> exit;
```

Komento tietokannan kopiointiin (<host> on oletuksena ’localhost’):

```
1. $ mysqldump -h<host> -u<username> -p<password> <database> > db_copy.sql
```

Komento SQL-tiedoston asettamiseksi tietokantaan:

```
1. $ mysql -h<host> -u<username> -p<password> <database> < db_copy.sql
```

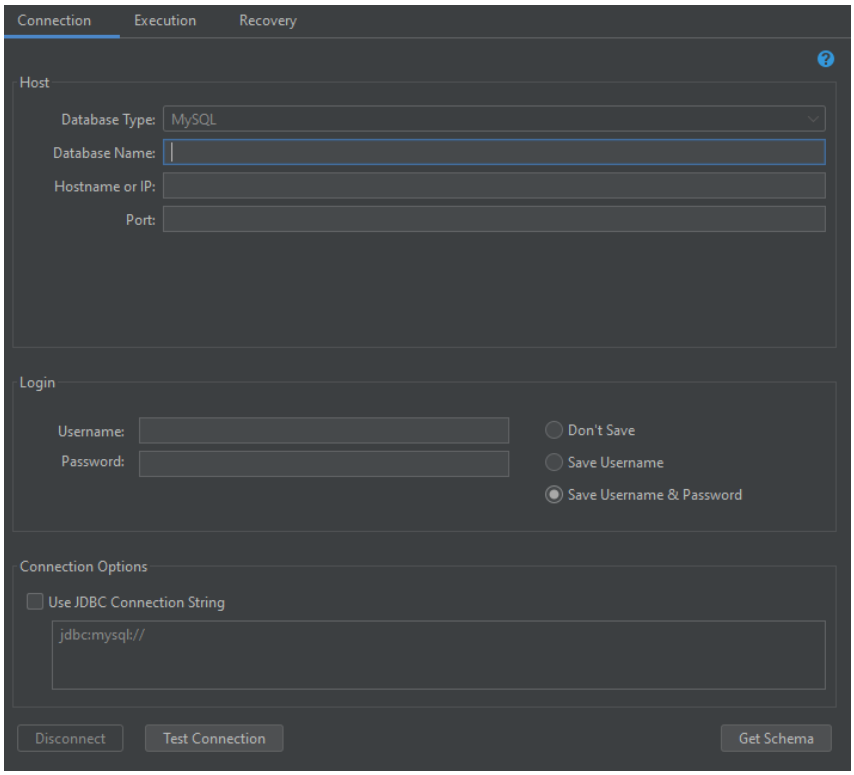


Kuva 10. Dataveil-työkalun käyttöliittymä

Kuvassa 10 nähdään:

1. Ylävalikko, josta löytyy kaikki työkalun yleisvalikot.
2. Pikakomennot yleisimmistä toiminnoista, kuten tallennus ja projektin suorittaminen.
3. Tietokantaikkuna, jossa nähdään projektiin liittyvät tietokannat ja niiden tietokantataulut.
4. Toimintonäkymä, josta voi tarkastella muun muassa yksittäisen sarakkeen käsittelytapaa, ohjelman suorituksen raportointia ja tietokantataulujen välisiä yhteyksiä.

Kun kopio oli asetettu uuteen tietokantaan, oli aika aloittaa pseudonymisointi Dataveil-työkalun avulla. Aloitin yhdistämällä aiemmin luodun tietokannan työkaluun. Yhdistämiseen vaadittiin tietokannan nimi, palvelin ja portti sekä kirjautumistiedot tietokantaan. (Kuva 11.) Yhdistämisen jälkeen toin tietokannan työkaluun, ja se ilmestyi tietokantaikkunaan (kuva 10). Siirryin DBMS (Database Management System) -välilehdelle, josta pääsin tarkastelemaan yhdistetyn tietokannan kaikkia tietokantatauluja. Poimin projektiin valitut yhdeksän tietokantataulua (liite 1) ja siirsin ne Project-välilehdelle, jotta ne voidaan pseudonymisoida.



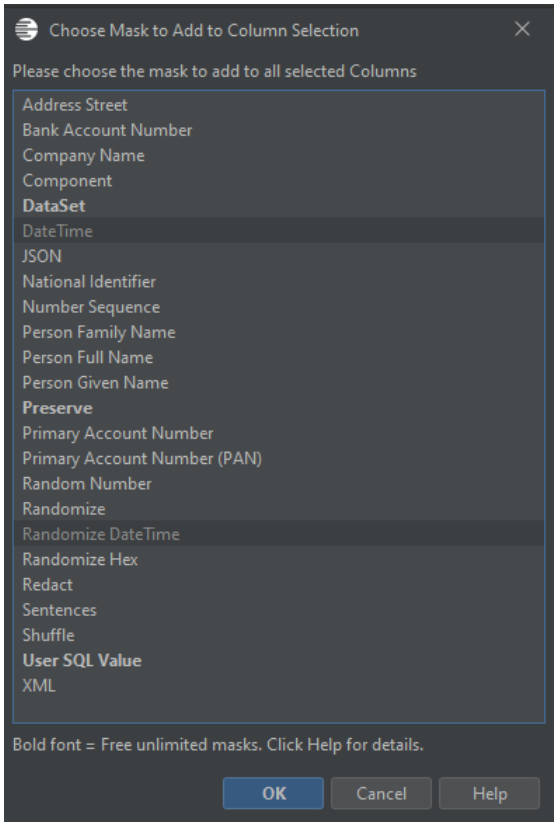
Kuva 11. Tietokannan yhdistäminen työkaluun

Ennen kuin ryhdyin manuaalisesti valitsemaan pseudonymisoitavat tietokantasarakkeet ja miettimään, miten tunnistetiedot voidaan poistaa, kokeilin Dataveil-työkalun toimintonäkymän (kuva 10) Discovery-toimintoa. Discovery-toiminnon avulla voi etsiä sarakkeita, jotka saattavat sisältää tunnistetietoja, ja se asettaa automaattisesti pseudonymisointimenetelmät löydetyille sarakkeille. Tunnistetut sarakkeet voidaan halutessa ottaa mukaan projektiin. Valitettavasti toiminto ei tuottanut erityisen hyvää tulosta. Se löysi joitakin oikeita tunnistetietoja sisältäviä sarakkeita, mutta sivuutti niitä myös paljon. Lisäksi se merkitsi paljon tietokantasarakkeita, kuten aikaleimat, virheellisesti tunnistetiedoiksi. On mahdollista, että huono tulos johtui vain siitä, että työkalun ilmaisversio merkitsi vain 50 ensimmäistä arkaluontoista saraketta. Hyväksyin oikein löydetyt tietokantasarakkeet ja jätin muut sarakkeet huomiotta.

Siirryin toimintonäkymän (kuva 10) Masking-välilehdelle, josta voi asettaa tietokantasarakkeen tunnistetiedoksi ja määrittää sille pseudonymisointimenetelmän. Kävin läpi kaikki tietokantataulut, merkitsin tarvittavat sarakkeet (liite 1) tunnistetiedoiksi ja määritin niille pseudonymisointimenetelmän. Lisäksi poistin projektista ne sarakkeet, jotka eivät olleet arkaluontoisia.

Pseudonymisointi oli suoraviivaista tietokantasarakkeiden osalta, joihin työkalu tarjosi valmiita tapoja tunnistetietojen poistamiseen (kuva 12), kuten yrityksen nimi ja osoite. Työkalussa ei kuitenkaan ollut valmista keinoa kaikkien tunnistetietojen poistamiseksi, joten minun täytyi kirjoittaa omia

SQL-lausekkeita (User SQL Value) niiden poistamiseksi. Tällaisiin tapauksiin kuuluivat esimerkiksi y-tunnus ja henkilötunnus, sekä muut arvot, joiden oikeellisuuden ohjelmisto tarkistaa tai jotka ovat salattuja.



Kuva 12. Työkalun tarjoamat pseudonymisointimenetelmät

Työkalu tarjosi tavan luoda omia pseudonymisointikomponentteja, joiden avulla voi muun muassa tallentaa omia SQL-lausekkeita uudelleenkäytettäväksi. Loin esimerkiksi puhelinnumerolle oman komponentin, sillä myös ne tarkistetaan ohjelmistossa. Työkalulla oli myös mahdollista määrittämään tietokantasuhteita, jotta tietokantataulut voivat periä arvoja toisiltaan. Tämä oli erityisen kätevää silloin, kun kahdessa eri taulussa oli samoja arvoja samalta yritykseltä. Näin arvot pysyivät yhdenmukaisina. Jos tietokantataulun sarakkeet riippuivat toisistaan, kuten y-tunnus ja ALV-numero, pystyin yhdistämään sarakkeiden arvot pseudonymisointia varten. Lisäksi oli hyödyllistä, että tyhjät (NULL) arvot säilytettiin tarvittaessa automaattisesti.

Kun olin määritellyt pseudonymisointimenetelmän kaikille tietokannan arkaluontoisille sarakkeille, siirryin toimintonäkymän (kuva 10) Execution-välilehdelle ja aloitin ohjelman ajamisen esikatseluajolla. Ajon aikana ilmeni kuitenkin muutamia virheitä. Ajoraportista selvisi, että virheet johtuivat viallisista SQL-lausekkeista. Korjasin virheet ja suoritin ohjelman uudestaan, tällä kertaa onnistuneesti. Seuraavaksi käynnistin ohjelman, joka pseudonymisoi peruuttamattomasti työkaluun

yhdistetyn tietokannan sarakkeet. Tässä vaiheessa ilmeni kuitenkin muutamia uusia virheitä, jotka eivät esiintyneet esikatseluajossa. Ohjelma palautti tietokannan alkuperäiseen muotoonsa virheiden tapahtuessa. Raportista selvisi, että yhdessä sarakkeessa oli duplikaatti arvoja, vaikka kyseisen sarakkeen ehtona oli, että kaikkien arvojen tulee olla uniikkeja. Tämä johtui siitä, että olin vahingossa jättänyt vakioarvon sähköpostiosoitteen sarakkeeseen. Kun korjasin ongelman, pseudonymisointi onnistui virheettömästi (kuva 13).

Vaster Row Groceries	OFR Violet Glove Clothing	1980349-5	1640581-1	FI19803495	FI16405811	VasterRowGroceries@testi.fi	OFRVioletGloveClothing@testi.fi
VHT Shorter Abroad Sports	Discreeter Performance Centre	1100450-9	1640581-1	FI11004509	FI16405811	VHTShorterAbroadSports@testi.fi	DiscreeterPerformanceCentre@testi.fi
IEG Brass Television	Initial Kind Services	1343121-3	1640581-1	FI13431213	FI16405811	IEGBrassTelevision@testi.fi	InitialKindServices@testi.fi
VZL Separate Solution Consultancy	South Estates International	1526255-6	1640581-1	FI15262556	FI16405811	VZLSeparateSolutionConsultancy@testi.fi	SouthEstatesInternational@testi.fi
Splendid Substance Co	IGX Thinner Leader Supermarket	1467596-5	1640581-1	FI14675965	FI16405811	SplendidSubstanceCo@testi.fi	IGXThinnerLeaderSupermarket@testi.fi
Ready Road Savings	NHH Short Week Networks	1097709-3	1640581-1	FI10977093	FI16405811	ReadyRoadSavings@testi.fi	NHHShortWeekNetworks@testi.fi
Cousin Loans	Treat Studio	1742402-9	1640581-1	FI17424029	FI16405811	CousinLoans@testi.fi	TreatStudio@testi.fi
Most Platform Adventure	Conscious Fortune Medicine	1781278-5	1640581-1	FI17812785	FI16405811	MostPlatformAdventure@testi.fi	ConsciousFortuneMedicine@testi.fi
GTI Pleasant Car Studio	Royal Worlds Enterprise	1056585-7	1640581-1	FI10565857	FI16405811	GTIPleasantCarStudio@testi.fi	RoyalWorldsEnterprise@testi.fi
Weekly Monitor Oils	Pearl Coast Center	1780996-5	1640581-1	FI17809965	FI16405811	WeeklyMonitorOils@testi.fi	PearlCoastCenter@testi.fi
Wild Beaches Industries	Perception Foundation	1154749-0	1640581-1	FI11547490	FI16405811	WildBeachesIndustries@testi.fi	PerceptionFoundation@testi.fi
Day Cars	State Distribution	1558092-5	1640581-1	FI15580925	FI16405811	DayCars@testi.fi	StateDistribution@testi.fi
BAT Aware Garden Therapies	YCS Economical Switch Constructions	1884351-6	1640581-1	FI18843516	FI16405811	BATAwareGardenTherapies@testi.fi	YCSEconomicalSwitchConstructions@testi.fi

Kuva 13. Yritys-tietokantataulu ennen pseudonymisointia ja sen jälkeen

Pseudonymisoinnin jälkeen loin uuden kopion pseudonymisoidusta tietokannasta käyttäen mysqldump-komentoa. Siirsin kopion paikalliseen tietokantaan, jonka jälkeen varmistin, että pseudonymisointi oli onnistunut, paikallinen ympäristö toimi edelleen ja palveluun pääsi kirjautumaan sisään.

Seuraavaksi toistin prosessin, mutta tällä kertaa kopioin testitietokannan. Tein tämän ennen tuotantokannan kopioimista siistä syystä, että se oli turvallisempi vaihtoehto virheiden välttämiseksi, ja testitietokanta sijaitsee myös pilvipalvelussa, kuten tuotantotietokantakin. Lisäksi testitietokanta on huomattavasti pienempi, mikä mahdollistaa nopeammat testiajot. Ajoin mysqldump-komennon tällä kertaa siten, että historiataulut eivät sisälly kopioon. Näitä tauluja ei tarvita testauksen kannalta, ja ne ovat erittäin suuria, joten niiden pois jättäminen nopeuttaa merkittävästi komennon suoritusta. Kirjoitin poissuljettujen tietokantataulujen nimet tekstitiedostoon, jotta voisin kätevästi asettaa ne osaksi komentoa. Siksi lisäsin komentoon “`cat ignored_tables.txt | xargs printf -- '--ignore-table=$DATABASE.%s '`”.

Pilvipalvelussa sijaitsevan tietokannan kopiointi ei kuitenkaan ollut yhtä suoraviivaista kuin paikallisen tietokannan kopiointi. Minulla ei ollut kaikkia tarvittavia oikeuksia tietokannan kopioimiseen. Yksi vaihtoehto olisi myöntää minulle laajemmat käyttöoikeudet, tai vaihtoehtoisesti voisin kopioida tietokannan ilman sen tallennustilaan liittyviä tietoja. Kuitenkin tietoturvallisuussyistä paras vaihtoehto oli ajaa mysqldump-komento --no-tablespaces-option kanssa, jolloin lisäoikeuksia ei tarvitse antaa. Toinen ongelma uudelleen ajossa oli, että pilvitietokannan MySQL-versio oli vanhempi kuin paikallinen versio. Vanhemmassa versiossa ei ole tapaa kerätä tilastotietoja, jotka paikallinen mysqldump-ajo asetti oletuksena. Siksi

tilastotietojen keruu piti estää käyttämällä mysqldump-komennon --column-statistics=0-optiota. Tämän jälkeen jatkoin samalla tavalla kuin pseudonymisoidessani paikallista tietokantaa: asetin kopioidun tietokannan pseudonymisoitavaksi, jonka jälkeen kopion sen ja asetin testitietokantaan.

Lopulta mysqldump-komennot näyttivät seuraavanlaisilta:

```
1. $ mysqldump -u<user> -p<password> --host=<host> --no-tablespaces --
   column-statistics=0 --no-data <database> > db_copy.sql
2. $ mysqldump -u<user> -p<password> --host=<host> --no-tablespaces --
   column-statistics=0 --no-create-info `cat ignored_tables.txt | xargs
   printf -- '--ignore-table=<database>.%s '` <database> >> db_copy.sql
```

Ensimmäinen komento kopioi tietokannan rakenteen tiedostoon ilman dataa, kun taas toinen komento lisää tietokannan datan tiedostoon, jättäen pois ei-toivotut taulut. Nämä komennot on suoritettava tässä järjestyksessä, jotta kaikki tietokantataulut saadaan mukaan projektiin, mutta data voidaan jättää pois ei-halutuista tauluista. Pelkästään käytettäessä toista komentoa ilman --no-create-info-optiota voi aiheuttaa mahdollisia ristiriitoja taulujen välillä, mikä saattaa johtaa tietokanta-ajon epäonnistumiseen, kun pseudonymisoitu tietokanta asetetaan uuteen tietokantaan.

Viimeinen vaihe työssä oli toistaa pseudonymisointiaskeleet tuotantotietokannalle. Koska tuotannon kopioiminen tapahtuu samalla tavalla kuin testitietokannan kopiointi, en käy näitä vaiheita uudelleen läpi. Sen sijaan selvitin, kuinka kauan prosessi kesti ja miten tehokas Dataveil-työkalu oli suurten tietokantojen käsittelyssä.

Kun kopioin tuotantokannan tiedostoon, sen ajamiseen kului noin 20 minuuttia. Lyhyt kesto johtui pääosin siitä, että jätin historiataulujen datan pois kopiosta. Tiedoston siirtäminen paikalliseen tietokantaan vei noin tunnin, ja sen pseudonymisointi kesti arviolta toisen tunnin. Dataveil-työkalu tarjoaa kuitenkin vain rajatun määrän maksullisia pseudonymisointimenetelmiä, joten jouduin korvaamaan osan niistä omilla SQL-lausekkeilla. En voi varmuudella sanoa, miten tämä vaikutti pseudonymisoinnin kestoon. Viimeinen tunti kului pseudonymisoidun tietokannan siirtämiseen paikalliseen tietokantaan. Tämä oli välttämätöntä, koska en ollut pseudonymisoinut koko tuotantokantaa, ja sen asettaminen testiympäristöön oli aiheuttanut tietoturvariskin. On myös äärimmäisen tärkeää muistaa, että paikalliset tunnistetietoja sisältävät tietokannat ja tiedostot tulee viipymättä tuhota. Asiakkaan dataa ei saa säilyttää tai käsitellä tarpeettomasti. Vaikka en voinut käyttää osittain pseudonymisoitua tuotantokantaa testiympäristössä, aiemmin testasin ympäristön toimintaa pseudonymisoidulla datalla ilman suurempia ongelmia. Kaiken kaikkiaan tuotantokannan käsittely ei ollut erityisen pitkäkestoinen prosessi, vaan kesti noin 3,5 tuntia.

7 Pohdinta

Tämän opinnäytetyön tavoitteena oli saada parempi ymmärrys tuotannonkaltaisen testausympäristön tarpeesta, hyödyistä ja haasteista sekä rakentaa pseudonymisoitua tuotantodataa hyödyntävä testausympäristö. Koen, että opinnäytetyön tavoite saavutettiin pääosin onnistuneesti. Työssä onnistuttiin kehittämään menetelmä toimeksiantajan tuotantodatan pseudonymisointiin ja lisättiin ymmärrystä pseudonymisointiprosessin vaiheista ja huomioon otettavista seikoista. Olisin kuitenkin toivonut, että pseudonymisoinnin lopputuloksena olisi syntynyt monimuotoisempaa dataa, mutta vakioarvojen käyttäminen oli usein välttämätöntä.

Työ tulokset ovat hyödyllisiä ja tärkeitä, sillä testausprosessin optimointi nopeuttaa ohjelmiston tai sen uuden ominaisuuden saamista tuotantoon ja vähentää mahdollisia ohjelmointivirheitä. Erityisesti projektin aikana saavutettu ymmärrys pseudonymisoinnin prosesseista ja käytännön toteutuksesta on arvokasta tietoa, joka voi olla hyödyllistä mahdollisissa tulevaisuuden pseudonymisointiprojekteissa. Lisäksi asiakkaan datan turvaaminen on erittäin tärkeä osa yrityksen toimintaa, jotta asiakkaan luottamus yrityksen toimintaan säilyy ja arkaluontoiset tiedot pysyvät suojattuina ulkoisilta osapuolilta.

Koen kuitenkin, että pseudonymisoitu tuotantodata on hyödyllisintä silloin, kun testataan jotain uutta ja monimutkaista ominaisuutta, jolloin tarve monipuoliselle datalle on suurempi. Pienempien ominaisuuksien yhteydessä voitaisiin hyödyntää keinotekoista dataa, joka luodaan esimerkiksi käyttäen Laravel-viitekehyksen ominaisuuksia ja PHP Faker -kirjastoa. Tällöinkin olisi tärkeää optimoida keinotekoisen datan luominen, esimerkiksi korjaamalla virheelliset yritykset, monipuolistamalla dataa ja mahdollisesti luomalla komento, jonka avulla testaaja voi itse luoda dataa. Lisäksi, jos testausympäristöä käytettäisiin kaikkeen testaamiseen, dataa kertyisi yhteiseen tietokantaan, joka monipuolistuisi ajan myötä.

Jatkokehitysmahdollisuuksia ovat pseudonymisoinnin laajentaminen kaikkiin tietokantatauluihin, joissa on tunnistetietoja. Lisäksi on mahdollista automatisoida pseudonymisointiprosessi tai luoda komentosarja, jolla tietokanta voidaan puhdistaa ja täyttää uudella, pseudonymisoidulla tuotantodatalla. Tämä on luultavasti suhteellisen suoraviivaista, sillä monia pseudonymisointityökaluja voidaan käyttää komentoliittymän kautta. Tulevaisuudessa voitaisiin myös keksiä keinoja vähentää vakioarvojen käyttöä datassa, jotta testidatasta saadaan vielä monimuotoisempaa. Tämä voidaan toteuttaa esimerkiksi SQL-lausekkeilla tai hyödyntämällä Laravel Factory -ominaisuutta keinotekoisen datan luomiseen komentosarjassa.

Opinnäytetyön aikana huomasin, kuinka vaikeaa on löytää tietoa tietokantadatan tunnistetietojen poistamisesta. Usein käytettävissä olevat tietolähteet olivat liian vanhoja tai kaupallisia sivustoja,

jotka myyvät valmiita pseudonymisointiratkaisuja. Vaikeuksia oli myös siinä, etten ollut koskaan aiemmin pseudonymisoinut tietokantadataa enkä tiennyt siitä juuri mitään. Jos voisin aloittaa työn alusta, käyttäisin todennäköisesti työn rakenteeseen vetoketjumallia, koska uskon, että näin olisin saanut heti alussa paremman käsityksen tietokantadatan tunnistetietojen poistamisesta käytännössä. Huomasin myös, että minulla oli välillä vaikeuksia tunnistaa, mikä arvo luokitellaan tunnistetiedoksi. Siksi ensimmäinen vaihe on erittäin tärkeä: yrityksen tulee käydä läpi tietosuojavastaavan kanssa, mitkä arvot lasketaan tunnistetiedoiksi. Jos kaikkia tuotteen ominaisuuksia tai tietokantasarakkeiden merkityksiä ei tunneta, on vaikeaa tai jopa mahdotonta löytää kaikkia tunnistetietoja tietokannasta.

Työn aikana opin luonnollisesti, kuinka tietokantadatasta poistetaan tunnistetietoja ja sain paremman ymmärryksen tietokannan käsittelystä ja niiden rakenteista. Tämän lisäksi opin testaamisesta, sen hyödyistä ja ongelmista sekä siitä, miksi sen optimointi on tärkeää. Huomasin myös, kuinka tärkeää on hahmottaa kokonaiskuva ennen projektin aloittamista. Usein uutta ohjelmointitehtävää aloittaessani ryhdyn suoraan ohjelmoimaan, enkä suunnittele tai tee prosessikuvausta kokonaisuudesta. Jatkossa uskon tekeväni enemmän kaavioita ohjelmiston olemassa olevista ja uusista ominaisuuksista, mikä helpottaa työn tekemistä. Jos joudun joskus palaamaan samaan työhön, kaavio on jo olemassa ja hyödynnettävissä. Kaiken kaikkiaan opinnäytetyö oli erittäin opettavainen kokemus. Toivon, että tämä opinnäytetyö voi toimia oppaana muille, jotka haluavat syventää tietämystään aiheesta.

Lähteet

Clarin s.a. Principles of Data Processing. Luettavissa: https://www.clarin.eu/content/principles-data-processing. Luettu: 27.2.2023.

Cobb, M. s.a. data masking. TechTarget. Luettavissa: https://www.techtarget.com/searchsecurity/definition/data-masking. Luettu: 24.3.2023.

Elfriede, D. 2002. Effective Software Testing: 50 Specific Ways to Improve Your Testing. Addison-Wesley Professional. E-kirja. Luettu: 17.3.2023.

Enisa 2018. Recommendations on shaping technology according to GDPR provisions - An overview on data pseudonymization. Enisa. Luettavissa: https://www.enisa.europa.eu/publications/recommendations-on-shaping-technology-according-to-gdpr-provisions. Luettu: 26.2.2023.

Enisa 2022. DEPLOYING PSEUDONYMISATION TECHNIQUES. Enisa. Luettavissa: https://www.enisa.europa.eu/publications/deploying-pseudonymisation-techniques/@@download/fullReport. Luettu: 26.2.2023.

Euroopan komissio 2014. Opinion 05/2014 on Anonymisation Techniques. Euroopan komissio. Luettavissa: https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf. Luettu: 26.2.2023.

FakerPHP s.a. Faker. Luettavissa: https://fakerphp.github.io/. Luettu: 24.4.2023.

GeeksforGeeks 2022. What is Hashing?. Luettavissa: https://www.geeksforgeeks.org/what-is-hashing/. Luettu: 9.3.2023.

Gibbs, S. 15.12.2016. Passwords and hacking: the jargon of hashing, salting and SHA-2 explained. The Guardian. Luettavissa: https://www.theguardian.com/technology/2016/dec/15/passwords-hacking-hashing-salting-sha-2. Luettu: 9.3.2023.

Goswami, S. 14.12.2020. The Rising Concern Around Consumer Data And Privacy. Forbes. Luettavissa: https://www.forbes.com/sites/forbestechcouncil/2020/12/14/the-rising-concern-around-consumer-data-and-privacy/?sh=598ef009487e. Luettu: 24.3.2023.

Heusser, M. & Kulkarni, G. 2016. How to Reduce the Cost of Software Testing. Auerbach Publications. E-kirja. Luettu: 15.3.2023.

Kasurinen, J. P. 2013. Ohjelmistotestauksen käsikirja. Docendo. Jyväskylä. E-kirja. Luettu: 12.3.2023.

Laravel s.a a. Eloquent: Factories. Luettavissa: https://laravel.com/docs/10.x/eloquent-factories. Luettu: 24.4.2023.

Laravel s.a b. Database: Seeding. Luettavissa: https://laravel.com/docs/10.x/seeding. Luettu: 24.4.2023.

Monday.com 25.8.2022. A Step-By-Step Guide to Writing a Proof of Concept. Mondayblog:n blogi. Luettavissa: https://monday.com/blog/project-management/proof-of-concept/. Luettu: 24.4.2023.

Mooc 2021. Tietokantojen perusteet kevät 2021. Luettavissa: https://tikape.mooc.fi/kevat-2021/content/osa-1/index.html. Luettu: 12.2.2023.

Raghunathan, B. 2013. The Complete Book of Data Anonymization. Auerbach Publications. E-kirja. Luettu: 21.3.2023.

Tietoarkisto s.a. Tunnisteellisuus ja anonymisointi. Luettavissa: https://www.fsd.tuni.fi/fi/palvelut/aineistonhallinta/tunnisteellisuus-ja-anonymisointi/. Luettu: 22.2.2023.

Tietosuojavaltuutetun toimisto 2021. Psykoterapiakeskus Vastaamolle seuraamusmaksu tietosuojarikkomuksista. Luettavissa: https://tietosuoja.fi/-/psykoterapiakeskus-vastaamolle-seuraamusmaksu-tietosuojarikkomuksista. Luettu: 24.2.2023.

Tietosuojavaltuutetun toimisto s.a. a. Pseudonymisoidut ja anonymisoidut tiedot. Luettavissa: https://tietosuoja.fi/pseudonymisointi-anonymisointi. Luettu: 25.2.2023.

Tietosuojavaltuutetun toimisto s.a. b. Säilytyksen rajoittaminen. Luettavissa: https://tietosuoja.fi/sailytyksen-rajoittaminen. Luettu: 23.2.2023.

Tietosuojavaltuutetun toimisto s.a. c. Usein kysyttyä EU:n tietosuoja-asetuksesta. Luettavissa: https://tietosuoja.fi/gdpr. Luettu: 25.2.2023.

Liitteet

Liite 1. Työhön valitut tietokantataulut ja niiden käsittelymenetelmät yksinkertaistettuna

Taulukko 3. Yritykset-tietokantataulu

Sarake	Datatyyppi	Arkaluontoisuus	Käsittelytapa
alv-numero	VARCHAR	Suora tunniste	Aseta vakioarvo
INTEGRAATIOT	-	-	Aseta vakioarvo
nettisivusto	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
nimi	VARCHAR	Suora tunniste	Satunnaista
puhelinnumero	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
sähköposti	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
y-tunnus	VARCHAR	Suora tunniste	Aseta vakioarvo

Taulukko 4. Laskut-tietokantataulu

Sarake	Datatyyppi	Arkaluontoisuus	Käsittelytapa
asiakas_viitenumero	VARCHAR	Epäsuora tunniste	Tyhjennä
kuljetus_nimi	VARCHAR	Suora/Epäsuora tunniste	Tyhjennä
tiedosto	VARCHAR	Suora tunniste	Aseta vakioarvo
viesti	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
viitenumero	VARCHAR	Epäsuora tunniste	Satunnaista
viivakoodi	VARCHAR	Suora tunniste	Aseta vakioarvo
yritys_viitenumero	VARCHAR	Epäsuora tunniste	Tyhjennä

Taulukko 5. Asiakkaat-tietokantataulu

Sarake	Datatyyppi	Arkaluontoisuus	Käsittelytapa
alv-numero	VARCHAR	Suora tunniste	Aseta vakioarvo
etunimi	VARCHAR	Suora tunniste	Satunnaista
faksi	VARCHAR	Epäsuora tunniste	Satunnaista
INTEGRAATIOT	-	-	Aseta vakioarvo
lisätietoja	TEXT	Suora/Epäsuora tunniste	Satunnaista
nimi	VARCHAR	Suora tunniste	Satunnaista
puhelinnumero 1	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
puhelinnumero 2	VARCHAR	Suora/Epäsuora tunniste	Tyhjennä
sukunimi	VARCHAR	Suora tunniste	Satunnaista
sähköposti	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
yhteyshenkilö	VARCHAR	Suora tunniste	Satunnaista
y-tunnus	VARCHAR	Suora tunniste	Aseta vakioarvo

Taulukko 6. Maksut-tietokantataulu

Sarake	Datatyyppi	Arkaluontoisuus	Käsittelytapa
maksaja	VARCHAR	Suora tunniste	Satunnaista
maksajan_bic	VARCHAR	Epäsuora tunniste	Aseta vakioarvo
maksajan_tilinumero	VARCHAR	Suora/Epäsuora tunniste	Aseta vakioarvo
vastaanottaja	VARCHAR	Suora tunniste	Satunnaista
vastaanottajan_bic	VARCHAR	Epäsuora tunniste	Aseta vakioarvo
vastaanottajan_tilinumero	VARCHAR	Suora/Epäsuora tunniste	Aseta vakioarvo
viesti	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
viite	VARCHAR	Epäsuora tunniste	Satunnaista

Taulukko 7. Asetukset-tietokantataulu (ei esitellä erityisemmin taulun poikkeavuuden takia)

Sarake			
avain			
arvo			

Taulukko 8. Yritystilit-tietokantataulu

Sarake	Datatyyppi	Arkaluontoisuus	Käsittelytapa
bic	VARCHAR	Epäsuora tunniste	Aseta vakioarvo
osoite	TEXT	Suora tunniste	Satunnaista
pankki	LONGTEXT	Epäsuora tunniste	Satunnaista
tilinumero	VARCHAR	Suora/Epäsuora tunniste	Aseta vakioarvo

Taulukko 9. Yrityskäyttäjät-tietokantataulu

Sarake	Datatyyppi	Arkaluontoisuus	Käsittelytapa
bic	VARCHAR	Epäsuora tunniste	Aseta vakioarvo
etunimi	VARCHAR	Suora tunniste	Satunnaista
henkilötunnus	VARCHAR	Suora tunniste	Aseta vakioarvo
nettisivusto	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
puhelinnumero	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
sukunimi	VARCHAR	Suora tunniste	Satunnaista
tilinumero	VARCHAR	Suora/Epäsuora tunniste	Aseta vakioarvo

Taulukko 10. Laskun_yritystiedot-tietokantataulu

Sarake	Datatyyppi	Arkaluontoisuus	Käsittelytapa
alv-numero	VARCHAR	Suora tunniste	Aseta vakioarvo
INTEGRAATIOT	-	-	Aseta vakioarvo
kaupunki	VARCHAR	Epäsuora tunniste	Satunnaista
kotisivu	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
nimi	VARCHAR	Suora tunniste	Satunnaista
osoite 1	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
osoite 2	VARCHAR	Suora/Epäsuora tunniste	Tyhjennä

Sarake	Datatyyppi	Arkaluontoisuus	Käsittelytapa
postinumero	VARCHAR	Epäsuora tunniste	Aseta vakioarvo
postitoimipaikka	VARCHAR	Epäsuora tunniste	Aseta vakioarvo
puhelinnumero	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
sähköposti	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
y-tunnus	VARCHAR	Suora tunniste	Aseta vakioarvo

Taulukko 11. Käyttäjät-tietokantataulu

Sarake	Datatyyppi	Arkaluontoisuus	Käsittelytapa
nimi	VARCHAR	Suora tunniste	Satunnaista
puhelinnumero	VARCHAR	Suora/Epäsuora tunniste	Satunnaista
salasana	VARCHAR	Epäsuora tunniste	Aseta vakioarvo
sähköposti	VARCHAR	Suora tunniste	Satunnaista
valtuus	VARCHAR	Epäsuora tunniste	Tyhjennä