TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Oskar Voldemar Lahesoo

Vertica andmebaasi struktuuri ja andmete kopeerimine

Bakalaureusetöö (9 EAP)

Juhendaja: Vambola Leping, MSc

Tartu 2020

Vertica andmebaasi struktuuri ja andmete kopeerimine

Lühikokkuvõte:

See töö kirjeldab loodud Pentaho Data Integrationi töödet Vertica andmebaasi skeemi objektide ja andmete kopeerimiseks failidesse ja salvestatud failidest teise samanimelisse skeemi.

Võtmesõnad:

Vertica, Pentaho Data Integration, andmebaas

CERCS: P175 Informaatika, süsteemiteooria

Copying Vertica database structure and data

Abstract:

This paper describes an original Pentaho Data Integration job for exporting Vertica database structure and data to files and using those files to recreate the schema in another Vertica instance.

Keywords:

Vertica, Pentaho Data Integration, database

CERCS: P175 Informatics, systems theory

Sisukord

1. Sissejuhatus

Tänapäeval on seoses kogutavate ja töödeldavate andmemahtude suurenemisega tekkinud vajadus uute spetsialiseeritud töövahendite jaoks, mis võimaldaksid korraga analüüsida senisest mitukümmend korda suuremaid andmehulki. Üheks selliseks töövahendiks on Vertica andmebaas, mille suhtelise uudsuse tõttu pole veel arendatud sellele kõiki igapäevatööks vajalikke vahendeid.

Käesoleva töö eesmärgiks on lihtsustada arendamist Vertica andmebaasi peal, luues selle jaoks vajaliku töövahendi. Selle töö käigus valminud vahend võimaldab Vertica objektide ja andmete failidesse salvestamist ning nende failide põhjal uuesti loomist. Antud vahendiga on võimalik mugavalt luua identseid andmebaase või pidada järge andmebaasi muudatustel.

Töö valmis firma jaoks, mis kasutab Verticat oma andmeaida haldamiseks ning valminud lahendust selleks, et kopeerida Vertica objekte uute arenduskeskkondade loomiseks.

Peatükis 3 kirjeldatakse täpsemalt Verticat ning Pentaho Data Integration tarkvara, mida kasutati tarkvaralahenduse loomiseks. Peatükis 4 kirjeldatakse probleeme, mille lahendamiseks tarkvara loodi, ning mainitakse alternatiivseid lahendusi. Peatükk 5 kirjeldab lahenduse võimalusi ja eripärasid. Peatükis 6 on kasutusjuhend ja konfiguratsioonifailide ülesehituse kirjeldus ning peatükis 7 on lõplikud tulemused ja testitud jõudlus.

2. Lühendid ja mõisted

SQL – Structured Query Language

PDI – Pentaho Data Integration ehk Kettle

JDBC – Java Database Connectivity

CSV – Comma Separated Value

UML – Unified Markup Language

DWH – Data Warehouse (andmeait)

tööde – arvuti poolt sooritatavate tööüksuste (programmide ja juhtimislausete) kirjeldus (inglise keeles *job*)

3. Kasutatud vahendid

Vahendite valimisel oli kõige olulisemaks ühilduvus nendega, mis olid juba kasutusel firmas, mille jaoks tarkvaralahendus loodi. PDI oli erinevate töödete jaoks kasutusel ning võimaldas raskusteta luua käesoleva lahenduse.

3.1. Vertica

Vertica Analytical Database (Vertica) on relatsioonilise andmebaasi haldussüsteem, mis keskendub suurte andmemahtude tõhusale töötlemisele ja salvestamisele [1]. Vertica salvestab andmeid veergude kaupa, mis võimaldab märkimisväärselt kiiremini sooritada analüütilisi töid nagu suure hulga kirjete summa või keskmise leidmine [1]. Verticat on võimalik tasuta katsetada kuni 1 TB andmetega[1]

3.2. Pentaho Data Integration

Pentaho Data Integration[2] (samuti tuntud kui Kettle) on vabavaraline töövahend andmete eraldamise, töötlemise, ja salvestamise töödete jaoks. Pentaho Data Integrationiga saab luua tööteid andmete liigutamiseks, võimaldades seejuures teavituste saatmist ja veahaldust. See on tasuta saadaval[3] GPLv2 litsentsiga.

[1] https://www.vertica.com/try/ (külastatud 14.11.2019)

[2] Githubi repositoorium: https://github.com/pentaho/pentaho-kettle (külastatud 07.11.2019)

[3] https://sourceforge.net/projects/pentaho/ (külastatud 07.11.2019)

4. Probleem ja senised lahendused

Kuna programmide kirjutamisel tuleb ette nii vigu kui ka soovimatuid kõrvalnähte, on vaja kahjude vältimiseks programme enne nende kasutusele võtmist testida. Levinud praktika on eraldada töökeskkond, kus peal tehakse igapäevaselt firma jaoks olulisi töid, ning arenduskeskkond, kus arendatakse ning testitakse uusi lahendusi.

Arenduskeskkondi on võimalik üles seada mitmel viisil, käesoleva töö kontekstis on arenduskeskkonna all silmas peetud eraldiseisvat andmebaasi koopiat, millel muudatuste tegemine ei mõjuta teiste tööks kasutatavat andmebaasi.

Kui arenduskeskkond ning töökeskkond on identsed, on lihtne loodud programme peale nende valmimist töökeskkonda üle viia. Tarkvara valmimine seevastu võtab aega, mille jooksul võivad töökeskkonna andmebaasi struktuur või kasutusel olevad objektid muutuda. Seega on tõhusaks töötamiseks vaja lihtsat meetodit, millega viia arenduskeskkond samale kujule või luua uus, töökeskkonnaga identne arenduskeskkond.

4.1. Käsitsi loomine

Firmas, mille jaoks loodi käesoleva töö käigus tarkvaralahendus, oli senine lahendus uue arenduskeskkonna loomiseks teha arendajale uus skeem. Skeemi kopeeriti vajalikud andmebaasi objektid ning nende peal töötati. Arendustöö lõpuks valminud programmis asendati skeemide nimed töökeskkonna omadega. See meetod toimib hästi, kuni objektide arv on väike, ent nende hulga kasvamisel muutub ebarealistlikuks. Lisaks on objektide muutmisel vaja kõik arendusskeemid läbi käia ja neis muudatused sisse viia, mis on ajakulukas. Administratiivsete ülesannete peale vähema aja kulutamiseks on vaja arenduskeskkonna loomine automatiseerida.

4.2. Vertica *vbr*

Verticaga on kaasas töövahend *vbr*. Tegemist on käsurealt käivitatava tööriistaga, mis võimaldab teha tagavarakoopiaid andmebaasist või selle osadest [2]. Tegemist on sobiva töövahendiga täielike või osaliste tagavarakoopiate tegemiseks, ent see ei sobi arenduskeskkondade loomiseks. Sobimatuse põhjuseks on, et Vertica *vbr* võimaldab vaid terve andmebaasi struktuuri korraga üle kandmist, mis on ebapraktiline, kui on vaja ainult mõnda skeemi ning objekti. Eraldi võimaldab *vbr* ka üksikute objektide üle kandmist, kuid siis pole võimalik üle kanda kasutaja loodud funktsioone [3]. Siinses töös loodud lahendus võimaldab kopeerida üksikuid objekte, sealjuures ka kasutajate endi loodud funktsioone.

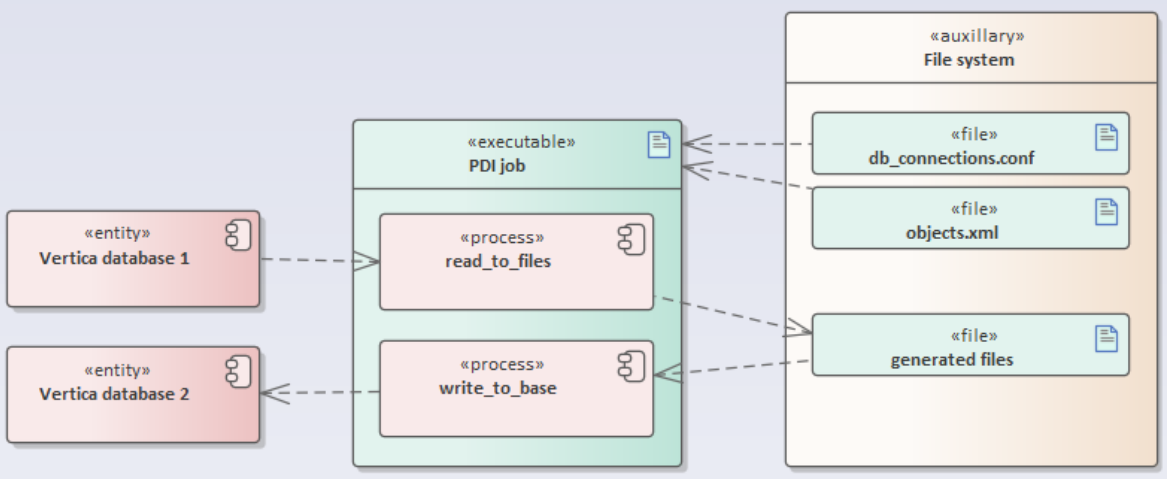
5. Valminud lahendus

Käesoleva töö käigus valminud tarkvara võimaldab ühest andmebaasist teise kanda üle kasutaja määratud jadad, tabelid, vaated ja funktsioonid, samuti määrata uue andmebaasi rollidele samasugused õigused objektide üle ning kopeerida valitud tabelite sisu uude andmebaasi. Tarkvara salvestab koopiad failidesse, millest saab luua mitu uut samasugust keskkonda ka ilma ligipääsuta originaalbaasile.

Käesolev peatükk kirjeldab loodud tarkvara tööpõhimõtteid. Kasutamiseks vajalik info on kirjas peatükis 6.

5.1. Tarkvara ülesehitus

Graafiline kujutis lahenduse kasutamisest on toodud Joonis 1. Kasutaja loob enne programmi käivitamist *objects.xml* faili, millesse ta kirjutab soovitud skeemide ja nendes olevate objektide nimed. Kuna kasutaja, kes kopeerib objekte, ei pruugi olla sama, kes nendel arendab, saab määrata ka rolle, millele antakse sihtandmebaasis samad õigused kui originaalis. Samasse faili kirjutatakse ka tabelite nimed, mille andmeid soovitakse kopeerida. Asukohad failide salvestamiseks ning vajalik konfiguratsioon, et luua ühendused andmebaasidega, võetakse *db_connections.conf* failist.



Joonis 1: UML diagramm lahenduse töövoost ja vajalikest osadest

5.2. Struktuuri lugemine andmebaasist failidesse

Konfiguratsioonis määratud *DWH_SRC_* parameetrite järgi luuakse JDBC ühendus andmebaasi ning sealt loetakse *objects.xml* failiga määratud jadad, tabelid, vaated ja funktsioonid. Jadade ning õiguste lugemisel kasutatakse süsteemitabeleid, mille lugemiseks on kasutajal vaja *SYSMONITOR* rolli.

5.2.1. Vertica *EXPORT_OBJECTS*

Suurem osa objektide kopeerimisest toimub kasutades Vertica *EXPORT_OBJECTS* funktsiooni. See funktsioon tagastab argumendiks antud objekti loomiseks vajaliku SQL käsu, mis kopeeritakse faili ning käivitatakse hiljem teises andmebaasis.

5.2.2. Jadade lugemine

Jadade lugemisel andmebaasist on oluline saada esialgse definitsiooni asemel jada praegune seis, et vältida konflikte, kui jada tabeli unikaalse võtmena kasutatakse. Selle jaoks kasutatakse *EXPORT_OBJECTS* funktsiooni asemel päringut Vertica süsteemitabelile *v_catalog.sequences*, kus on salvestatud jadade hetkeseis. Saadud päringu põhjal koostatakse SQL käsk, mis salvestatakse faili.

5.2.3. Õiguste lugemine

Vertica õigused pole andmebaasi objektid, mistõttu pole neid võimalik saada *EXPORT_OBJECTS* funktsiooni kasutades. Õiguste loomiseks vajalik info loetakse Vertica süsteemitabelist *v_catalog.grants* ja loetu põhjal koostatakse vajalikud SQL käsud.

5.3. Struktuuri kirjutamine failidest andmebaasi

Struktuuri lugemisel pole objektide järjekord oluline, ent objektid võivad sõltuda teistest objektidest ja objekti loomine ebaõnnestub, kui puuduvad tema eelduseks olevad objektid. Seepärast luuakse objektid kindlas järjestuses: jadad, funktsioonid, tabelid ja vaated. Sama tüüpi objektid luuakse samas järjekorras, millega nad on *objects.xml* failis, mis on oluline näiteks välisvõtmetega tabelite kirjutamisel.

Varem tehtud failid on salvestatud SQL käskude jadana, mida nüüd kasutatakse, et *DWH_DST_* parameetritega määratud andmebaasis vastavad objektid luua.

5.4. Andmete lugemine ja kirjutamine

Lisaks skeemi objektidele, mida soovitakse kopeerida, saab määrata ka tabelid, mille andmed üle kanda. Nende tabelite andmed salvestatakse Vertica binaarformaadis failidesse ja kirjutatakse peale struktuuri loomist andmebaasi. Andmed kirjutatakse selles järjekorras, millega need on *objects-xml* failis, mis võib olla erinev järjekorrast, millega kirjutatakse tabelid.

6. Tarkvara kasutamine

PDI tööte põhisammud on lisas 1. Sammud *read_sequences_to_file* kuni *read_data_to_file* tegelevad andmebaasi struktuuri ja andmete failidesse lugemisega ning sammud *write_sequences_to_base* kuni *write_data_to_base* struktuuri ja andmete kirjutamisega failidest teise baasi. Kumbagi osa saab jooksutada eraldiseisvalt ilma teiseta, näiteks tagavarakoopia tegemiseks ja sellelt taastamiseks.

Vigade korral, näiteks ebakorrektse sisendi korral või eelduseks olevate objektide puudumisel, katkestatakse töö, logitakse erind ja e-maili seadete olemasolu korral saadetakse vigast sammu kirjeldav kiri määratud aadressile.

6.1. Eeldused tarkvara kasutamiseks

Kasutajal peab olema *db_connections.conf* fail, mille asukoht tuleb kirjutada PDI tööte esimesse sammu *get_parameters*. Seal määratakse andmebaasi ühendamiseks vajalikud parameetrid, *objects.xml* faili asukoht ning kaustad soovitud objektide kirjutamiseks ja lugemiseks.

6.2. *db_connections.conf* fail

Iga faili rida on kujul “MUUTUJA_NIMI=muutuja_väärtus”. Näide *db_connections.conf* failist on toodud lisas 2.

Vajalikud read koos seletustega:

`COPY_CONFIG_FILE` – *objects.xml* faili absoluutne asukoht
`GRANTS_COPY_FILE` – õiguste faili absoluutne asukoht
`OBJECTS_COPY_PATH` – kaust, kuhu kirjutada objektide failid
`DATA_COPY_PATH` – kaust, kuhu kirjutada tabelite andmed

`DWH_SRC_HOST_NAME`,
`DWH_SRC_DATABASE_NAME`,
`DWH_SRC_PORT`,
`DWH_SRC_USERNAME`,
`DWH_SRC_PASSWORD` – koopia tegemiseks vajaliku (lähteandmebaasi) JDBC ühenduse andmed

DWH_DST_HOST_NAME,
DWH_DST_DATABASE_NAME,
DWH_DST_PORT,
DWH_DST_USERNAME,
DWH_DST_PASSWORD – andmebaasi kirjutamiseks vajaliku (sihtandmebaasi) JDBC ühenduse andmed

Soovituslikud read vigade korral teavituse saamiseks:

DWH_ADMIN_EMAIL – e-maili aadress, millele saata teavitused
EMAIL_SENDER_NAME,
EMAIL_SMTP_SERVER,
EMAIL_SMTP_SERVER_PORT – e-maili serveri seaded
EMAIL_SUBJECT_ERROR,
EMAIL_SUBJECT_WARNING,
EMAIL_SUBJECT_DONE – e-mailide pealkirjad vastavalt sellele, kuidas tööde õnnestus

6.3. *objects.xml* fail

Faili nimi ei pea olema täpselt *objects.xml*, kuid see peab olema XML fail ning selle nimi koos asukohaga peab olema kirjas *db_connections.conf* failis. Faili tuleb kirjutada kõik objektid, skeemid, kus onjektid asuvad, ning rollid, mille õiguseid soovitakse üle kanda.

Faili juureks on silt <*object*>, mille alla käivad sildid <*role*> ja <*schema*>. Iga silt <*role*> tähistab üht rolli, mille õiguseid soovitakse kopeerida, sildis on tekstina kirjas rolli nimi. Sildi <*schema*> vajalik osa on atribuut *name*, mis tähistab vastava skeemi nime. <*schema*> sees on sildid <*sequence*>, <*function*>, <*table*>, <*view*> ja <*data*>. Esimese nelja sildi tekstiks on vastava jada, funktsiooni, tabeli ja vaate nimi, mille struktuuri soovitakse salvestada või andmebaasi kirjutada. Sildi <*data*> sisuks on tabeli nimi, mille andmeid soovitakse kopeerida. Näide *objects.xml* failist on toodud lisas 3.

Erinevat tüüpi objektid kirjutatakse järjekorras, mis ei põhjusta konflikte, ent sama tüüpi objektid kirjutatakse järjekorras, millega nad ilmuvad *objects.xml* failis. Teineteisest sõltuvate objektide (nagu välisvõtmetega tabelite) kirjutamisel on oluline, et teistest sõltuvad objektid oleksid peale objekte, millest need sõltuvad.

7. Tulemused

Valminud PDI tööde suudab probleemideta salvestada andmebaasi struktuuri ning salvestuse põhjal seda taasluua. See täidab arendajate vajadusi nii arenduskeskkondade loomiseks kui ka andmebaasi hetkestruktuuri salvestamiseks. Loodud PDI tööde võimaldab andmete salvestamist ning ülekandmist, seejuures ka tagavarakoopiate tegemist, kuid ainult tagavarakoopiate jaoks sobivad paremini Vertica sisseehitatud vahendid.

Lahendust testiti andmeaida dimensioonide skeemi peal, kandes üle 81 tabelit koos andmetega, 3 vaadet, 3 jada ja 2 funktsiooni. Kogu protsess sujus tõrgeteta ning võttis aega 1 minuti ja 27 sekundit. Testimise eesmärgiks oli veenduda tööte toimimises, kuid selle käigus mõõdeti ka jõudlust, kopeerides tabeleid nii koos andmetega kui ilma. Tulemused on toodud tabelites 1 ja 2.

Tabel 1: Kulunud aeg vastavalt kopeeritud tabelite arvule (ilma andmeteta)

Tabelite arv	**Aeg (s)**
10	10
30	20
50	29
81	57

Tabel 2: Kulunud aeg vastavalt kopeeritud tabelite arvule (koos andmetega)

Tabelite arv	**Andmete kogumaht (MB)**	**Aeg (s)**
10	1,57	17
30	1,97	38
50	2,01	46
81	2,40	78

8. Kokkuvõte

Käesoleva töö käigus valmis originaalne Pentaho Data Integrationi tööde Vertica andmebaasi objektide ning tabelite sisu kopeerimiseks failide ja andmebaasi vahel. Loodud tööde võimaldab kasutaja määratud jadade, tabelite, vaadete, ja funktsioonide salvestamist failidesse ning nende failide põhjal objektide loomist skeemidesse.

Loodud töödet kasutatakse Verticaga hallatava andmeaida struktuuri salvestamiseks ning arenduskeskkondade loomiseks.

Edaspidiseks arenduseks tuleks esmajärjekorras lisada tugi ka teistele andmebaasimootoritele peale Vertica. Veel tasuks uurida paremaid meetodeid andmete kopeerimiseks baaside vahel, mis ei sisaldaks ajakulukat failidesse lugemist.

9. Viidatud kirjandus

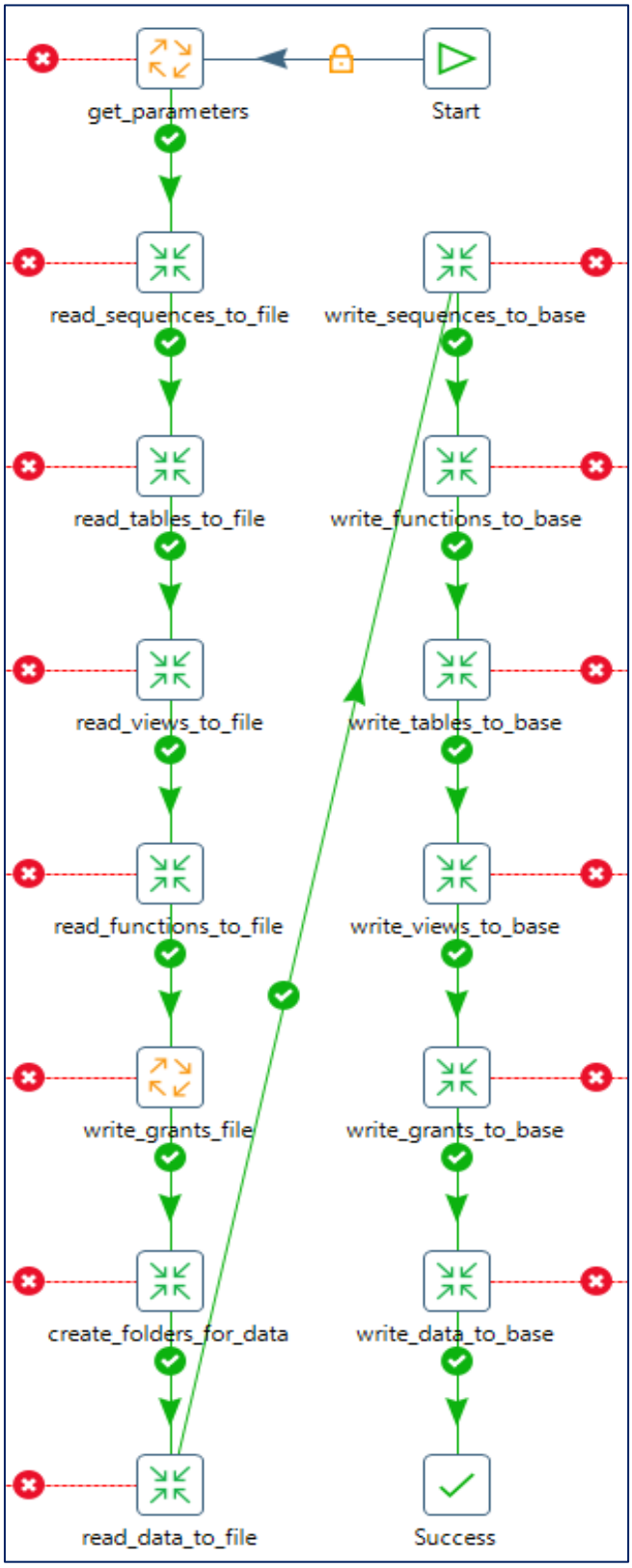
[1] A. Lamb, M. Fuller, R. Varadarajan, N. Tran, B. Vandiver, L. Doshi ja C. Bear, „The vertica analytic database: C-store 7 years later,“ *Proceedings of the VLDB Endowment,* kd. 5, nr 12, pp. 1790-1801, August 2012.

[2] Vertica, „Backing Up and Restoring the Database,“ [Võrgumaterjal]. Available: https://www.vertica.com/docs/9.2.x/HTML/Content/Authoring/AdministratorsGuide/BackupRestore/BackingUpAndRestoringTheDatabase.htm. [Kasutatud 7.11.2019].

[3] Vertica, „Copying Data Between Similar Vertica Clusters,“ [Võrgumaterjal]. Available: https://www.vertica.com/kb/Copying-Data-Between-Similar-Vertica-Clusters/Content/BestPractices/Copying-Data-Between-Similar-Vertica-Clusters.htm. [Kasutatud 7.11.2019].

Lisad

I. Tööte põhisammud



Pildilt on välja jäetud veahaldus, kuna see on primitiivne, kuid võtab väga palju ruumi.

II. Näide failist *db_connections.conf*

```
COPY_CONFIG_FILE=C:/database_structure/conf_files/objects.xml
GRANTS_COPY_FILE=C:/database_structure/grants/grants.sql
OBJECTS_COPY_PATH=C:/database_structure/objects
DATA_COPY_PATH=C:/database_structure/data

DWH_SRC_HOST_NAME=source.base.address
DWH_SRC_DATABASE_NAME=DWH
DWH_SRC_PORT=5433
DWH_SRC_USERNAME=kasutajanimi
DWH_SRC_PASSWORD=parool

DWH_DST_HOST_NAME=destination.base.address
DWH_DST_DATABASE_NAME=DWH
DWH_DST_PORT=5433
DWH_DST_USERNAME=kasutajanimi
DWH_DST_PASSWORD=parool
```

III. Näide failist *objects.xml*

```
<objects>
      <role>reporting_read_only</role>
      <role>full_read_only</role>
      <role>full_admin</role>
      <role>dwh_load</role>
      <schema name="test_schema_copy_01">
            <table>dim_country</table>
            <table>dim_date</table>
            <table>dim_language</table>
            <table>dim_company</table>
            <table>dim_inventory</table>
            <data>dim_country</data>
            <data>dim_date</data>
            <data>dim_language</data>
            <data>dim_company</data>
            <data>dim_inventory</data>
            <view>country_vw</view>
            <view>currency_vw</view>
            <view>domain_vw</view>
            <sequence>seq_dim_inventory_row_id</sequence>
            <function>get_gender</function>
      </schema>
      <schema name = "test_schema_copy_02">
            <table>strtest</table>
            <sequence>strtest_seq</sequence>
            <data>strtest</data>
      </schema>
</objects>
```

IV. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

1. Mina, Oskar Voldemar Lahesoo, annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose „Vertica andmebaasi struktuuri ja andmete kopeerimine“, mille juhendaja on Vambola Leping, reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.
2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.
3. Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.
4. Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Oskar Voldemar Lahesoo

09.01.2020