

Ülesanne 5

Mis peab olema selle ülesande täitmise tulemusena tehtud?

- ✧ Olete välja mõelnud klassifikaatorite väärtused (v.a *Riik*, mille korral klassifikaatori väärtused on standardiga paika pandud ja mille enda andmebaasi laadimist puudutab järgmine punkt), dokumenteerinud need INSERT lausete abil ning need laused käivitanud.
 - Täitmiseks lugege palun selle ülesande jaotiseid *Taust* ja *Klassifikaatorid*.
- ✧ Olete täitnud ülesande, mille sisuks on JSON/XML formaadis lähteandmete laadimine enda andmebaasi tabelitesse. Selle abil saate andmed tabelitesse *Isik* ja *Riik*. Lähteandmed on JSON formaadis teise sama serveri PostgreSQL andmebaasi tabelis (PostgreSQL) ning XML formaadis teises skeemis asuvas XML tüüpi tabelis (Oracle).
 - Täitmiseks lugege palun selle ülesande jaotist *JSON/XML formaadis andmed*. Kui teete projekti PostgreSQLis, siis lugege palun ka Oracle osa ning vastupidi.
- ✧ Olete lisanud testandmeid ülejäänud tabelitesse. Hindamismudel nõuab, et kõikides projekti andmebaasi tabelites (ka nendes, mida Teie töökoht otseselt ei kasuta) peab olema vähemalt üks rida testandmeid.
 - Täitmiseks lugege palun selle ülesande jaotist *Taust*.
- ✧ Kui testandmete lisamise käigus selgus, et mõni kitsendus puudub või on valesti realiseeritud, siis olete teinud parandused andmebaasi disaini mudelis, genereeritud andmekirjelduskeele lausetes ja tegelikult andmebaasis.

Taust

Võimalikud ridade lisamise viisid (kuni Teil pole selle jaoks spetsiaalset andmebaasirakendust).

- ✧ Kirjutate INSERT laused ja käivitate need.
 - Erinevalt PostgreSQList ei tööta Oracle transaktsioonide automaatse kinnitamise (*autocommit*) režiimis. Seetõttu tuleb Oracles peale INSERT lausete käivitamist käivitada järgnev lause.

▪ COMMIT;
 - Sellega kinnitate transaktsiooni ja muudate lisatud andmed teistes sessioonides nähtavaks.
- ✧ Kasutate andmebaaside haldamise/arendamise programmide graafilist kasutajaliidest – PostgreSQL puhul näiteks *pgAdmin* ning Oracle puhul *SQL Developer*. Saamaks kätte INSERT laused eksportige lisatud andmed. Need laused on vaja panna projekti dokumenti ja neid läheb ka vajadusel tarvis andmebaasi taastamiseks.
- ✧ PostgreSQL puhul on võimalik linkida MS Accessi andmebaasi PostgreSQLi andmebaasi baastabelid ja kasutada andmete PostgreSQLis tabelitesse lisamiseks MS Accessi graafilist kasutajaliidest.
 - Vaadake videoid ODBC andmeühenduse spetsifikatsiooni loomise ja tabelite linkimise kohta kataloogist *Tarkvara saamine ja kasutamine/PostgreSQL + MS Access videod (MS Access 2013 baasil)*

Millele üldiselt INSERT lausete kirjutamisel/eksportimisel tähelepanu pöörata?

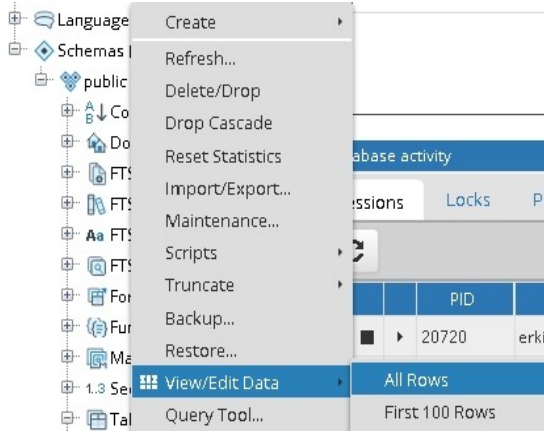
- ⤴ **Halvasti kirjutatud lause** – kui muutub tabeli veergude järjekord, siis lause ei tööta (halb) või töötab valesti (veel halvem, sest andmetesse tekib viga, mis avastatakse alles kunagi hiljem).
 - `INSERT INTO Amet_R VALUES (2,'dekanaadi sekretär');`
- ⤴ **Hästi kirjutatud lause.**
 - `INSERT INTO Amet (amet_kood, nimetus) VALUES (2,'dekanaadi sekretär');`
- ⤴ Ärge pange arvulisi väärtuseid ülakomade vahele (nii kirjutatakse tekstilisi väärtuseid), sest nii peab süsteem tegema asjatuid tüübiteisendusi.

Kui andmete eksportimise programm väljastab eelnimetatud puudustega laused, siis punktide mitte kaotamiseks, tuleb neid käsitsi parandada.

Kui teete tööd mitmekesi, siis registreerige andmeid paralleelselt. Oracle puhul ärge unustage, et käesolevas sessioonis lisatud andmete nähtavaks muutmiseks teistes sessioonides tuleb käesolevas sessioonis transaktsioon kinnitada.

pgAdmin:

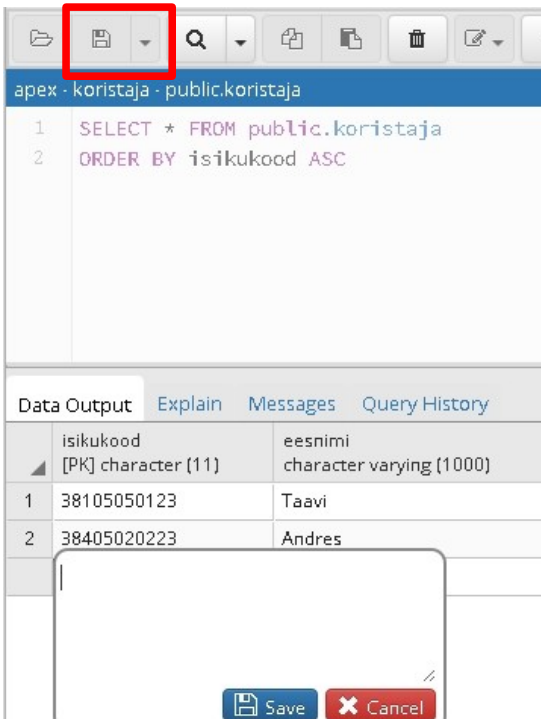
Valin tabeli ja valin selle andmete vaatamise



Kuidas sisestada kuupäeva või kellaaega?

Kasutatav formaat:
https://en.wikipedia.org/wiki/ISO_8601

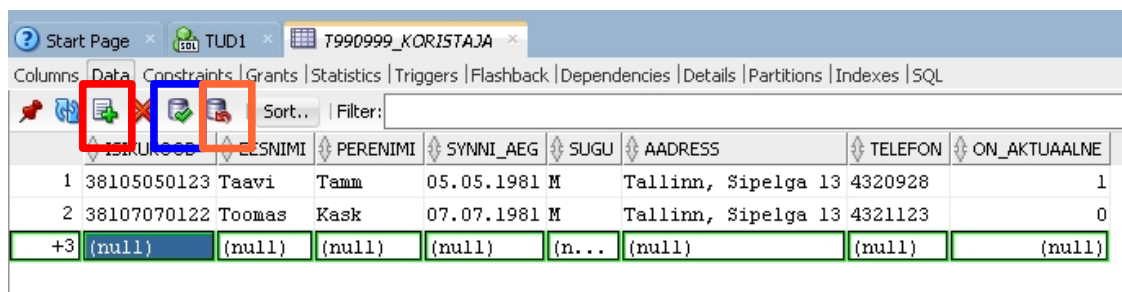
Sisestan rea ja salvestan



NB! pgAdmin 4 ver 2 põrkusin olukorraga, kus ühte tabelisse graafilise kasutajaliidese kaudu andmete lisamine andis CHECK kitsenduse vea, samas kui nende samade andmetega rea lisamine pgAdmin 3 kaudu või INSERT lausetega viga ei andnud. Sama probleemi on kohanud ka üliõpilased. Nagu ütleb tarkvara antimuster *Jalutades miiniväljal*, siis tarkvara vead pole midagi erakordset:

<https://sourcemaking.com/antipatterns/walking-through-minefield>

SQL Developer:



	ISKUKOOD	ELSNIMI	PERENIMI	SYNNI_AEG	SUGU	AADRESS	TELEFON	ON_AKTUAALNE
1	38105050123	Taavi	Tamm	05.05.1981	M	Tallinn, Sipelga 13	4320928	1
2	38107070122	Toomas	Kask	07.07.1981	M	Tallinn, Sipelga 13	4321123	0
+3	(null)	(null)	(null)	(null)	(n...	(null)	(null)	(null)

Punane kast – lisamise alustamine (kasutajaliideses tekivad uue rea jaoks sisestusväljad).

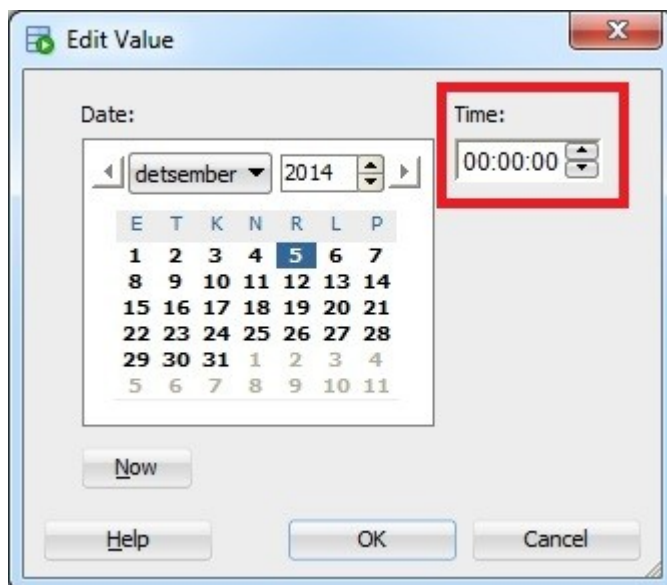
Sinine kast – lisamise kinnitamine (*commit*) ja sellega teistele kasutajatele teistes sessioonides nähtavaks tegemine.

Oranž kast – lisamise tühistamine (*rollback*) (saab teha enne seda kui olete lisamise kinnitanud).

Kui Teil on tabelil kitsendus, mis tagab, et DATE tüüpi väärtus registreeritakse kuupäeva täpsusega, siis peate SQL Developeri sisestusaknas tegema valiku, et kellaaeg on 00:00:00. Vastasel juhul lisate andmeid, mis ei ole selle kitsendusega kooskõlas.

Kitsenduse näide:

```
trunc(synni_kp)=synni_aeg
```



Hiljem saate loodud andmebaasi eksportida (teha sellest loogilise varukoopia) ning kasutada tulemusel olevaid INSERT lauseid, et täita dokumendi aljaotused "Klassifikaatorite väärtustamise SQL laused" ja "Täiendavate testandmete lisamine". Kui eksportimise programm genereerib ilma veergude nimedeta INSERT laused (vt eespool olevat juttu), siis tuleb need käsitsi lisada.

Loogilise varukoopia tegemiseks saate kasutada:

- ✦ PostgreSQL: pg_dump (programm serveris), pgAdmin
- ✦ Oracle: SQL Developer

Õppekeskkonnas on loogilise varukoopia tegemise kohta videod (kataloogides *PostgreSQL õppevideod (PostgreSQL 9.3 baasil)* ja *Oracle õppevideod (Oracle 12.1 baasil)*).

Vihje

Kui tegite kevadise projekti MS Accessis ja kasutasite SQL lausete käivitamiseks minu poolt pakutud abiprogrammiga faili, siis seal saate ekraanivormil *SQL skripti käivitamine* vajutada nupule, mis ekspordib MS Accessi andmebaasist CREATE TABLE, DROP TABLE ja INSERT laused. Võibolla on genereeritud INSERT lausetest kasu. Nendes INSERT lausetes puuduvad veergude nimed (tuleb ise juurde kirjutada). Kui peale faili avamist on sisu "välja lülitatud", siis tuleb eksportimise võimaldamiseks valida *Enable Content*.

Nõuded käivitavale failile.

- Võib sisaldada CREATE TABLE, CREATE INDEX, ALTER TABLE, DROP TABLE, DROP INDEX, INSERT, UPDATE ja DELETE lauseid ning kommentaare
- SQL lausete eraldaja peab olema ";"
- Kommentaarid peavad algama ja lõppema "/"
- Kui lause täitmine ei õnnestu (nt. vigane süntaks, ebasobivad nimed jne.), siis registreeritakse tekkinud veaolukord

Valige käivitav fail

Käivitage SQL laused

Vaadake SQL lausete täitmise tekkinud veaolukordade nimekirja

SQL lausete eksport

SQL lauseid sisaldavad failid lähevad kataloogi c:\temp
Kui sellist kataloogi ei ole, siis see luuakse programmi poolt! Loodud failide nimede eesliide on "esql". Lausetes kasutatakse PostgreSQL andmetüüpe ja seega neid lauseid uuesti MS Accessis ilma parandamata käivitada ei saa.

- ✦ Kui tabelis on mittekohustuslik veerg ja kõikides ridades selle väärtus puudub, siis see võib näidata, et sellist veergu polnud suure tõenäosusega tegelikult vaja. Järelikult on viga tehtud juba süsteemianalüüsi käigus.
- ✦ Kui tabelis on veerg *kirjeldus*, siis tuleb sinna lisada sisukad kirjeldused. Muidu jääb mulje, et kirjelduse veergu ei olegi midagi lisada ning tekib õigustatud küsimus, miks selline veerg siis tabelisse üldse pandi.
 - Näide: Ametikoha nimega "professor" kirjeldused.
 - *Sisutu*: Professor.
 - *Sisukam*: Professor on rahvusvahelisel tasemel teadus-, arendus- või muus loometegevuses aktiivselt osalev oma eriala juhtiv õppejõud, kes korraldab ja viib läbi oma ainevaldkonna õpet, juhiv teadus-, arendus- või muud loometegevust ning juhendab tulemuslikult nendesse tegevustesse kaasatud üliõpilasi, õppejõude ja teadustöötajaid. Professoril on doktorikraad või sellele vastav kvalifikatsioon.

Klassifikaatorid

- ✧ Klassifikaatorite väärtustamise laused tuleb projekti dokumendis paigutada alajaotusesse "Klassifikaatorite väärtustamise SQL laused".
- ✧ Võtke ametite registreerimisel aluseks pädevusalade nimekiri – igale pädevusalale vastab üks või mitu ametit.
- ✧ Seisundiklassifikaatorite tabelitesse leiate väärtused vastava põhiobjekti elutsükli kirjeldavalt seisundidiagrammilt. Eeldusaine projektis tuli seisundidiagramm luua ühe põhiobjekti kohta – ülejäänute puhul peate seda mõttes ette kujutama ja selle alusel seisundiklassifikaatori väärtused määrama.
 - Kui kõigis või enamikes sellistes tabelites on ainsad seisundid "aktiivne" ja "mitteaktiivne", siis tekitab see välises vaatelehtes küsimuse, kas süsteemianalüütik tegi ikka piisavalt korralikku tööd.
- ✧ Jälgige, et klassifikaatorite tabelitesse lisatavad koodid oleksid kooskõlas põhiobjektide tabelites kasutatavate vaikimisi väärtustega.
 - Näide: oletame, et tabelis *Kaup* on veerg:
 - `kauba_seisundi_liik_kood`
`SMALLINT NOT NULL DEFAULT 1`
 - ✧ 1 on seisundi "Ootel" kood.
 - ✧ Tabelis *Kauba_seisundi_liik* peab olema rida seisundi "Ootel" kohta, kus kood on 1.
- ✧ Pidage silmas, et klassifikaatorite koodid dokumenteeritakse ning registreeritakse süsteemis vastavate volitustega inimkasutaja poolt. Hiljem on võimalik neid väärtuseid lisada ja muuta (selleks on Teie süsteemi äriarhitektuuris *klassifikaatorite funktsionaalne allsüsteem*).
- ✧ Klassifikaatorite koodid ei pea tulema "järjest". Koodid ei pea olema 1, 2, 3, vaid võivad näiteks ka olla 10, 15, 20.

NB! 10, 15, 20 ei ole numbrid, vaid täisarvud, mis on kirja pandud kasutades kahte araabia numbrit (https://et.wikipedia.org/wiki/Araabia_numbrid)

- ✧ Kavandage andmebaas nii, et klassifikaatorite koodid oleksid võimalikult sisukad. Näiteks selle asemel, et kasutada riigi koode 1, 2, 3 ning mõõtühikute koode 21, 22, 23 peaksid riigi koodid olema EST, USA, FIN (http://kirste.userpage.fu-berlin.de/diverse/doc/ISO_3166.html) ning mõõtühikute koodid kg, l, m. Seega koodid ei pea olema täisarvud, vaid võivad olla ka näiteks tekstistringid. See nõuab loomulikult otsuseid andmebaasi disaini käigus ning vastava veeru deklareerimist CHAR(n) tüüpi (kui kõik koodid on ühesuguse pikkusega) või VARCHAR(n) tüüpi (kui koodide pikkus varieerub) + veerule kitsenduste deklareerimist (nt, et väärtus peab olema vähemalt kaks märki pikk ja tohib koosneda tähtedest ning numbritest). Info selliste koodide kasutamise kohta peab ka jõudma kontseptuaalses andmemudelisis olemitüübi *Klassifikaator* atribuudi *kood* definitsiooni.
 - Kui kasutate seisundiklassifikaatorite puhul arvulisi koode, siis arvestage võimaliku seisundite järjekorraga olemi elutsükli. Järgnev näide X_seisundi_liigi koodide kohta põhineb töövihikul.
 - **Halvem:** 2 – ootel; 1 – aktiivne; 4 – mitteaktiivne; 3 – lõpetatud
 - **Parem:** 1 – ootel; 2 – aktiivne; 3 – mitteaktiivne; 4 – lõpetatud.

- ⤴ Kui klassifikaatorite koodid on tekstilises veerus, siis ärge kasutage seal sisuliselt arvulisi väärtuseid. Näiteks isiku seisundi liigi koodidel: ISL-001 ja ISL-002 ühelt poolt ning 1 ja 2 teiselt poolt ei ole lõppkasutajale arusaadavuse ning meeldejäätavuse seisukohalt sisulist vahet. Koodid 1 ja 2 on antud juhul isegi paremad, sest andmemahut on väiksem. Kui klassifikaatorite koodid on tekstilised väärtused, siis koodide ISL-001 ja ISL-002 asemel oleksid paremad koodid, mis jäävad inimestele paremini meelde – näiteks EL (nagu "elus") ja SU (nagu "surnud").
- ⤴ Üritage võimalikult palju kasutada riiklikke/rahvusvahelisi standardeid, mis kirjeldavad ära mingit tüüpi klassifikaatorite väärtuseid. Kui avastate alles nüüd, et mõne klassifikaatori väärtused on standardis ära kirjeldatud, siis võib olla vaja muuta vastava klassifikaatori tabeli disaini (siis tuleb muuta nii mudelit, lähtekoodi kui ka tegelikku andmebaasi). Näited:
 - ⤴ Riigid ja territooriumid: https://en.wikipedia.org/wiki/ISO_3166-1
 - ⤴ Kaupade kategooriad: *Global Product Classification* – <https://www.gs1.org/gpc>
 - ⤴ Raamatute kategooriad: *Book Industry Communication Standard Subject Categories* – <http://www.bic.org.uk/7/bic-standard-subject-categories/>

JSON/XML formaadis andmed

Sageli vahetavad infosüsteemid andmeid XML või JSON formaadis dokumentidena. Siin on näide teenusest, mis pakub kasutajale XML formaadis andmeid: <https://www.ilmateenistus.ee/teenused/ilmainfo/eesti-prognoos-xml/>

Veel näiteid XML formaadis jagatavatest andmehulkadest – antud juhul on need seotud Euroopa Liidu ja selle institutsioonidega: https://data.europa.eu/euodp/data/dataset?q=xml&ext_boolean=all&sort=

Hindamismudeli alusel annab selle ülesande mittetäitmine miinuspunkte. Kui Teie projektis pole tabeleid, kuhu selles ülesandes ettenähtud andmeid laadida (võimalik omaloomingu projekti korral), siis võtke palun ühendust õppejõuga. Ülesanne loetakse täidetuks, kui Teie andmebaasis on nõutud välised tabelid (PostgreSQL), Teie andmebaasi tabelitesse on laaditud andmed ning vastavad laused on esitatud projekti dokumendis.

Eesmärgiks on laadida enda andmebaasi tabelitesse *Riik* ja *Isik* andmed väljastpoolt enda andmebaasi. Tabelisse *Riik* laaditakse andmed, mida kirjeldab ISO 3166 standard. See on standard, "mis reguleerib riike, maid, sõltuvaid territooriume, geograafilisi piirkondi ja haldusüksusi tähistavaid koode."

- http://metaweb.stat.ee/view_xml_multi_code.htm?id=3477719&siteLanguage=ee
- https://et.wikipedia.org/wiki/ISO_3166

Lugege järgnev läbi, käivitage oma andmebaasi põhjal laused ja pange need ka oma projektis jaotisesse "JSON formaadis läheandmete laadimine" (kui teete projekti PostgreSQLis) või "XML formaadis läheandmete laadimine" (kui teete projekti Oracles). **t990999** tuleb asenda enda *apex.ttu.ee* kasutajanimelga, samuti tuleb kasutada enda *apex.ttu.ee* **parooli**. Kui Teie andmebaasi tabelite või veergude nimed erinevad näites toodust, siis tuleb käivitavas koodis ka nimesid muuta. Kui vaja, siis tuleb teha ka muid muudatusi – näiteks, kui tabelis *Isik* on kohustuslikke veerge, millele vastavaid

väärtuseid ei saa välisest tabelist, siis tuleb väärtused nende veergude jaoks käigult (INSERT lauses) ise konstrueerida. Kui isiku seisundi liikide koodid on Teie andmebaasis tekstilised, siis tuleb INSERT lause SELECT klauslis asendada *isiku_seisundi_liik_kood* sobiva tekstilise konstandiga.

Tabelisse *Isik* laadite parooli avatekstina – selle räsiväärtusega asendamise küsimustega tegeleb ülesanne 10.

PostgreSQL

```
CREATE EXTENSION IF NOT EXISTS postgres_fdw;
/*Lisan andmebaasi välise andmete pakendamise laienduse, mis võimaldab PostgreSQL andmebaasis lugeda andmeid teisest PostgreSQL andmebaasist (võib olla samas serveris või teises serveris) ja neid seal ka muuta (seda viimast pole antud ülesandes vaja teha).
https://robots.thoughtbot.com/postgres-foreign-data-wrapper
PostgreSQL välise tabelite mehhanism võimaldab juurdepääsu erinevat tüüpi failide sisule, aga ka juurdepääsu andmetele, mis on teistes andmebaasides (võimalik, et teine andmebaasisüsteem/andmemudel), veebis, infokataloogides jne. Vaadake PostgreSQL erinevaid välise andmete pakendajaid:
https://wiki.postgresql.org/wiki/Foreign\_data\_wrappers */

CREATE SERVER minu_testandmete_server_apex FOREIGN DATA WRAPPER
postgres_fdw OPTIONS (host 'apex.ttu.ee', dbname 'testandmed',
port '5432');
/*Testandmed on apex.ttu.ee serveris andmebaasis testandmed. Viite loomine välise andmete asukohale. Seda lauset pole vaja muuta!*/

CREATE USER MAPPING FOR t990999 SERVER
minu_testandmete_server_apex OPTIONS (user 't990999', password
'secret');
/*Vastavuse defineerimine kohaliku andmebaasi kasutaja ning selles käskude käivitaja ja välise andmebaasi kasutaja vahel. Kui teete rühmatööd, siis võite luua ühe sellise vastavuse iga rühma liikme kohta. Kohaliku ja välise kasutaja nimi langevad praegu kokku, sest andmebaasid on samas serveris. Siin lauses kasutage oma apex.ttu.ee kasutajanime ja parooli!*/

CREATE FOREIGN TABLE Riik_jsonb (
riik JSONB )
SERVER minu_testandmete_server_apex;
/*Loon välise tabeli, mis viitab teises andmebaasis olevale tabelile, kus riikide andmed on JSON formaadis. Lähteandmed pärinevad: https://gist.github.com/jeremybuis/4997305
psqlis on välise tabelite nimekirja nägemiseks käsk \det */

SELECT * FROM Riik_jsonb;

INSERT INTO Riik (riik_kood, nimetus)
SELECT riik->>'Alpha-3 code' AS riik_kood,
riik->>'English short name lower case' AS nimetus
FROM Riik_jsonb;
```

```

/*Loen välisest tabelist JSON formaadis andmed, teisendan need
sobivale kujule ja laadin enda andmebaasi tabelisse Riik.

Koodinäiteid ja viiteid PostgreSQLis JSON formaadis andmetega
töötamise kohta vaadake slaidikomplektist "Andmebaasisüsteemide
Oracle ja PostgreSQL kasutamine" - otsige sealt sõna JSON.*/
SELECT * FROM Riik;
/*Veendun, et andmed on lisatud.*/

CREATE FOREIGN TABLE Isik_jsonb (
isik JSONB )
SERVER minu_testandmete_server_apex;

SELECT * FROM Isik_jsonb;
/*Sellesse tabelisse andmete genereerimiseks kasutasin:
https://www.json-generator.com/ */

INSERT INTO Isik(riik_kood, isikukood, eesnimi, perenimi,
e_mail, synni_kp, isiku_seisundi_liik_kood, parool, elukoht)
SELECT riik_kood, isikukood, eesnimi, perenimi, e_mail,
synni_kp::date, isiku_seisundi_liik_kood::smallint, parool,
elukoht
FROM (SELECT isik->>'riik' AS riik_kood,
jsonb_array_elements(isik->'isikud')->>'isikukood' AS isikukood,
jsonb_array_elements(isik->'isikud')->>'eesnimi' AS eesnimi,
jsonb_array_elements(isik->'isikud')->>'perekonnanimi' AS
perenimi,
jsonb_array_elements(isik->'isikud')->>'email' AS e_mail,
jsonb_array_elements(isik->'isikud')->>'synni_aeg' AS synni_kp,
jsonb_array_elements(isik->'isikud')->>'seisund' AS
isiku_seisundi_liik_kood,
jsonb_array_elements(isik->'isikud')->>'parool' AS parool,
jsonb_array_elements(isik->'isikud')->>'aadress' AS elukoht
FROM isik_jsonb) AS lahteandmed
WHERE isiku_seisundi_liik_kood::smallint=1;
/*Loen välisest tabelist JSON formaadis andmed, teisendan need
sobivale kujule ja laadin enda andmebaasi tabelisse Isik. Loen
andmeid ainult isikute kohta, kes on seisundis koodiga 1 e
elus.*/

SELECT * FROM Isik; /*Veendun, et andmed on lisatud.*/

```

Teie andmebaasi tabelitesse *Isik* ja *Riik* andmete lisamisel tuleb andmete struktuuriga (skeemiga) arvestada kohe – peate konstrueerima sobiva struktuuriga ja kitsendustele vastavate andmetega rea, muidu lisamine ei õnnestu. Skeem on jõustatud andmebaasi tasemel ja andmebaasisüsteem kontrollib andmete lisamisel andmete sellele vastavust. Selle kohta öeldakse, et andmetel on kirjutamise skeem (*schema-on-write*).

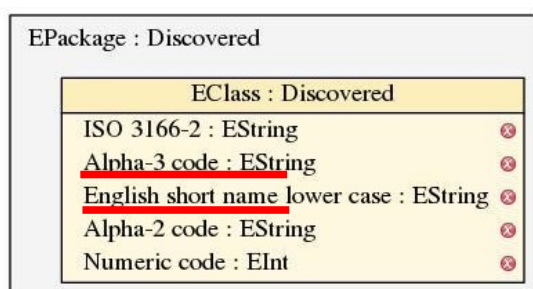
Tabelisse *Riik_jsonb* ja *Isik_jsonb* võib panna igasuguse struktuuriga JSON dokumente. Nende struktuuri ei kirjeldata ega jõustata andmebaasi tasemel. Kuid skeemi on vaja teada andmete lugemisel, muidu ei oleks võimalik andmetest aru saada ja lugejale oleksid need lihtsalt müra. Selle kohta öeldakse, et andmetel on lugemise skeem (*schema-on-read*). Seda tegelikult tähendabki NoSQL süsteemidest tuntud andmete "skeemituse" (*schemaless*)

põhimõte: <https://martinfowler.com/articles/schemaless/> (lugemisel navigeerimiseks nooled lehekülje ülaservas).

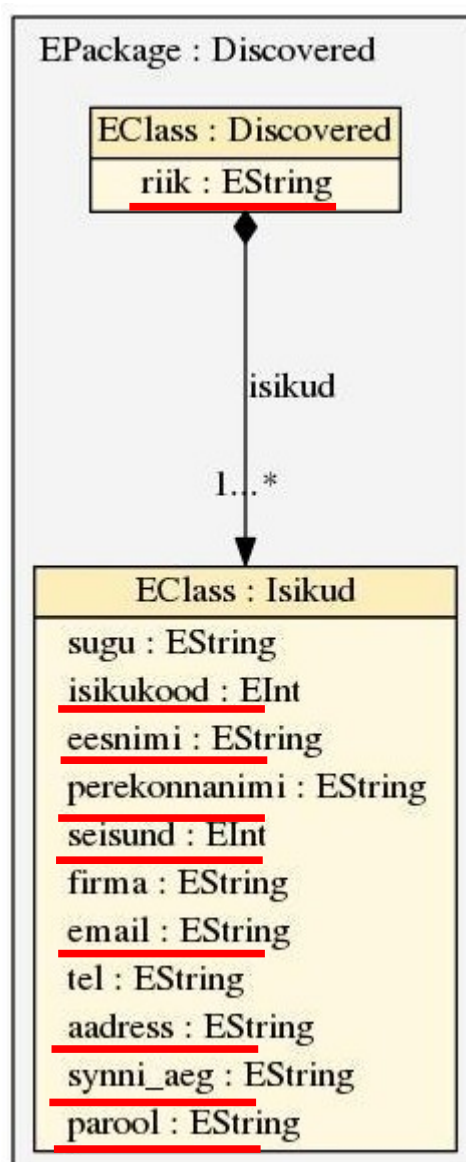
Järgnev illustratsioon kirjeldab välistest tabelitest loetud riikide ja isikute andmete struktuuri e skeemi. Skeemi leidmiseks kasutati vahendit JSON Discoverer, mis oskab JSON formaadis andmetest tuletada nende andmete struktuuri kirjelduse: <http://som-research.uoc.edu/tools/jsonDiscoverer/#/>

Riikide andmed ei ole hierarhilised, st kõikide riikide andmed on samal tasemel. Isikute andmed on esitatud kahetasemelise hierarhiana – esimesel tasemel isikukoodi riik ja teisel tasemel sellise riigiga isikute ülejäänud andmed. Punase alakriipsuga on tähistatud andmed, mida Te enda andmebaasi laadisite.

Riikide andmeid esitavate JSON dokumentide stuktur



Isikute andmeid esitavate JSON dokumentide stuktur



Dokumenti tuleb panna ka laused loodud andmebaasiobjektide kustutamiseks, kuid andmebaasis jätke need objektid alles (ärge lauseid käivitage).

```
DROP FOREIGN TABLE IF EXISTS Riik_jsonb CASCADE;
DROP FOREIGN TABLE IF EXISTS Isik_jsonb CASCADE;
DROP USER MAPPING FOR t990999 SERVER
minu_testandmete_server_apex;
DROP SERVER IF EXISTS minu_testandmete_server_apex
CASCADE;
DROP EXTENSION IF EXISTS postgres_fdw CASCADE;
```

Oracle

```
SELECT * FROM c##naited.Riik_xml;
/*Riikide andmed on sama andmebaasi skeemis c##naited XML tüüpi
tabelis Riik_xml.*/

INSERT INTO t990999_Riik (riik_kood, nimetus)
SELECT x.riik_kood, x.nimetus
FROM c##naited.Riik_xml r,
xmltable('//countries/country' PASSING r.SYS_NC_ROWINFO$
COLUMNS riik_kood VARCHAR2(3) PATH 'Alpha-3_code',
nimetus VARCHAR2(100) PATH 'English_short_name_lower_case') x;
/*Laadin andmed tabelisse Riik. XML formaadis andmete tabeli
kujule teisendamiseks kasutatakse XMLTABLE funktsiooni. See
funktsioon on osa XQuery päringukeelest, mis on mõeldud XML
formaadis andmete käitlemiseks.*/

COMMIT;
/*Kinnitan lisamise.*/

SELECT * FROM t990999_Riik;
/*Veendun, et andmed on lisatud.*/

SELECT * FROM c##naited.Isik_xml;
/*Isikute andmed on sama andmebaasi skeemis c##naited XML tüüpi
tabelis Isik_xml*/
```

```

INSERT INTO t990999_Isik(riik_kood, isikukood, eesnimi,
perenimi, e_mail, synni_kp, isiku_seisundi_liik_kood, sool,
parool, elukoht)
SELECT riigid.riigi_kood, isikud.isikukood, isikud.eesnimi,
isikud.perenimi, isikud.e_mail, isikud.synni_kp,
isikud.isiku_seisundi_liik_kood, 'puudub', isikud.parool,
isikud.elukoht
FROM c##naited.Isik_xml,
xmltable ('*/riik'
    PASSING Isik_xml.SYS_NC_ROWINFO$
    COLUMNS riigi_kood CHAR(3) PATH 'riigi_kood',
             isikud xmltype PATH 'isikud'
) riigid,
xmltable ('*/isik'
    PASSING riigid.isikud
    COLUMNS eesnimi VARCHAR2(1000) PATH 'eesnimi',
             perenimi VARCHAR2(1000) PATH 'perekonnanimi',
             elukoht VARCHAR2(1000) PATH 'aadress',
             e_mail VARCHAR2(254) PATH 'email',
             isikukood VARCHAR2(100) PATH 'isikukood',
             synni_kp DATE PATH 'synni_aeg',
             parool VARCHAR2(100) PATH 'parool',
             isiku_seisundi_liik_kood NUMBER PATH 'seisund'
) isikud
WHERE isiku_seisundi_liik_kood=1;
/*Laadin andmed tabelisse Isik. Loen andmeid ainult isikute
kohta, kes on seisundis koodiga 1 e elus. Lause kirjutamisel
kasutati: https://technology.amis.nl/2006/10/31/extracting-master-detail-data-from-an-xmltype-with-a-single-sql-statement/*/

COMMIT;
/*Kinnitan lisamise.*/

SELECT * FROM t990999_Isik;
/*Veendun, et andmed on lisatud.*/

```

Taustaks olgu öeldud, et Oracle kasutajate poolt laaditavad andmed on samad, mis PostgreSQLis. Selleks teisendasin kõigepealt JSON dokumendid XML formaati: <https://www.freeformatter.com/json-to-xml-converter.html> Enne teisenduse tegemist tuli muuta JSON dokumentides märgendite nimesid, sest need võivad sisaldada tühikuid, kuid XML elementide nimed ei tohi sisaldada tühikuid: <https://stackoverflow.com/questions/14028211/json-xml-conversion-issue-whitespace> Järgnevalt esitatakse laused, mida kasutasin tabeli *c##naited.Riik* loomiseks ja mida Teil pole vaja uuesti käivitada.

```

CREATE TABLE Riik_xml OF XMLTYPE;
GRANT SELECT ON Isik_xml TO PUBLIC;

CREATE OR REPLACE DIRECTORY data_dir as '/tmp/';
GRANT READ, WRITE ON DIRECTORY data_dir TO C##Yliopilane;

INSERT INTO Riik_xml VALUES (XMLType(bfilename('DATA_DIR',
'Riigid.xml'), nls_charset_id('AL32UTF8')));

```