**Kirjetöötluse rakendus - andmebaaside– tunniülesanne**

***Tunnitöö eesmärk on tutvuda andmebaaside, SQL päringukeele ja andmebaasihalduri liidesega. Tööülesannete jaoks on olemas järgnevate ühendusparameetritega PostgreSQL andmebaas: Host: ekleer.pld.ttu.ee Andmebaas: ewis Kasutaja: read\_ewis Parool: RO-A11ik45-2023***

*Tunniülesande lahendamiseks on vajalik tabel nimega ’joe\_andmed’, mis sisaldab endas Eesti jõgesid ja tabel nimega ’seire\_jaamad’. Tabeli ’joe\_andmed’ väljadest on olulised jõe pikkus (pikkus\_km) ning jõe algus- ja lõpp-punkti geograafilised koordinaadid ( lon1, lat1 – algus; lon2, lat2 – lõpp).Igal jõel ja jões toimuvate muutuste jälgimiseks mõeldud seirejaamal on oma identifikaator. Tabelid ’joe\_andmed’ ja ’seire\_jaamad’ on omavahel seotud identifikaatori ’id\_jogi’ abil – st iga seirejaam on seotud ühe kindla jõega, mille ääres ta paikneb.*

**Ülesanded 1.** Kirjuta programm, mis looks ühenduse eeltoodud andmebaasiga ja väljastaks 5 pikimat jõge.

**Ülesanne2**. Pärida jõed, mis on peajõed ja pikemad kui 70 km. Peajõe kindlaks tegemiseks saab kasutada id\_peajogi välja.

**Ülesanne 3.** Leida eelmises punktis leitud jõgedel asetsevad seirejaamad. • Võtmesõnadeks ’inner join’ ja ’subqueries’. • Alustuseks võib ülesannet lihtsustada ja teha päring ilma eelmist päringut arvestamata

**Ülesanne 4.** Pärida andmebaasist jõe algus- ning lõpp-punkti koordinaadid ja leida nende punktide vaheline kaugus linnulennult. Tulemuse võib limiteerida näiteks 10 esimese kirje peale. • Kahe punkti vahelise kauguse leidmiseks kasutada järgnevat valemit: d = R\*acos(sin(lat1)\*sin(lat2) + cos(lat1)\*cos(lat2)\*cos(lon1-lon2)); • R - maa raadius. Ühik vastavalt sellele, mis ühikutes soovid vastust väljastada. • Tabelitest päritavad väärtused on kraadides, kuid C math.h trigonomeetrilised funktsioonid vajavad argumente radiaanides. • PQgetvalue funktsioon tagastab string tüüpi tulemuse. Seega tuleks see ümber konverteerida ujupunktarvuks, et temaga oleks võimalik arvutusi teostada.

***Edasijõudnutele***

***1. Kasutada lisaks tabelit seire\_jogi\_hydrol, mis sisaldab endas jõe vooluhulga mõõtetulemusi ühikuga m3 /s (väli vaartus). Leida 5 kõige suurema keskmise vooluhulgaga Eesti jõge.***

***2. Leia 5 pikima jõe aastased vooluhulga keskmised ajaperioodil 1990-2000 (kokku 50 rida).***

***3. Kas erinevate jõgede aastased vooluhulgad on korrelatsioonis? Leia Pärnu ja Võhande jõgede aastase keskmise vooluhulga korrelatsiooni koefitsient eelmise päringu põhjal kasutades CORR(X, Y) funktsiooni.***

***4. Kasutades väljasid id\_jogi, id\_suubla, id\_peajogi väljasta jõed võttesse arvesse nende omavahelist hierarhiat. Väljundi limiteerimiseks küsida kasuajalt, millist jõge soovitakse väljastada. Näiteks Choose river: Pärnu -Pärnu +Navesti -Sauga +Hirve -Are -Kõrisoo +Saki -Tõntsu -Marjassaa +Elbu***

**Abistavad materjalid • Postgre dokumentatsioon - http://www.postgresql.org/docs/9.1/interactive/index.html • Postgre libpq C teegi kasutamisest - http://www.postgresql.org/docs/9.1/interactive/libpq.html • SQL lausete testimiseks võib kasutada mõnda klienti, nt** http://ewis.pld.ttu.ee/mywebsql/

gcc psql.c –lpq

• Kompileerimisel on soovitatav on kasutada ICT-501 arvuteid, kuna PostgreSQL teegid on sinna eelnevalt paigaldatud.

**• Postgre libpq teegi lisamiseks #include + kompileerimisel käsk -lpq**

• Math.h teegi lisamine: #include + kompileerimisel käsk –lm

***• Põhilised libpq funktsioonid:***

***//Ühenduse loomine PGconn \*conn; conn = PQconnectdb("dbname= host= user= password=");***

***//Kontrollime ühendust if (PQstatus(conn) == CONNECTION\_BAD***) { //Kui ühendus ebaõnnestus /\*Error handling\*/ }

***//Päringu teostamine, nt pärime 10 esimest kirjet tabelist***

***res = PQexec(conn, "SELECT \* FROM <table > LIMIT 10);***