Tartu Ülikool

Arvutiteaduse instituut

Informaatika õppekava

Johannes Erik Laamann

Interaktiivse veebiõpiku loomine ainele "Sissejuhatus andmebaasidesse"

Bakalaureusetöö (9 EAP)

Juhendaja:

PhD Piret Luik

Tartu 2019

Resümee/Abstract

Interaktiivse veebiõpiku loomine ainele "Sissejuhatus andmebaasidesse"

Bakalaureusetöö eesmärgiks oli interaktiivse veebiõpiku arendamine ainele "Sissejuhatus andmebaasidesse", mis võimaldaks sessioonõppe õppuritel kui ka teistel andmebaaside huvilistel iseseisvalt teadmisi kinnistada. Töö käigus valmis veebirakendus, milles on võimalik lahendada ülesandeid vastu päris andmebaasi, ning õpikutekstid. Rakendus keskendub PostgreSQL-i õpetamisele. Töös on uuritud sagedasi SQL-i õppimisel esinevaid raskusi ja sarnaseid veebirakendusi ning kirjeldatakse valminud rakendust tehnilisest küljest.

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Märksõnad: SQL, andmebaasid, e-õpe

Interactive web textbook for course "Introduction to Databases"

The aim of this bachelor thesis was to create an interactive web textbook for course "Introduction to Database". The web application allows users to solve exercises in a real database. PostgreSQL was chosen as the SQL dialect. The author of this thesis also wrote textbook texts. An overview of similar existing web applications, common difficulties of learning SQL and a technical description of application's implementation is also presented.

CERCS: P170 Computer science, numerical analysis, systems, control

Keywords: SQL, databases, e-learning

Sisukord

Sissejuhatus

Andmebaaside ja SQL-i tundmine muutub aina olulisemaks tänapäevases suurandmete maailmas. Valdkonnad nagu andmeteadus ja masinõppe sõltuvad suurel määral andmetöötlusest ning nõuavad teadmisi SQL-ist. Andmebaasidega puutuvad aina enam kokku lisaks tarkvaraarendajatele ka teiste eluvaldkondade esindajad.

Tartu Ülikoolis on aineid, mis õpetavad algteadmisi andmebaasidest neile, kes ei õpi peaerialana informaatikat. Üheks selliseks aineks on "MTAT.03.105 Sissejuhatus andmebaasidesse". Kursuse teemade hulgas on tabelite ja vaadete loomine, andmete muutmine ning päringute kirjutamine.

Probleemi tutvustus

Töö vajadus tuleneb sellest, et hetkel ei eksisteeri sellist eestikeelset õpperakendust, kus õppurid saaksid katsetada SQL-päringuid vastu päris andmebaasi. SQL-i oskuste omandamiseks ei piisa vaid teooria õppimisest, mistõttu on oluline rohkem rõhku asetada praktiliste ülesannnete lahendamisele, millele saaksid õppurid ka vahetut tagasisidet.

Aine "Sissejuhatus andmebaasidesse" toimub lisaks päevasele õppele ka sessioonõppes, mille korral praktikumid toimuvad iga kahe nädala tagant ning õppejõududega kontakt on sellisel juhul väike. See nõuab suuremat individuaalset panust õppuritelt, kellele võivad antud töö käigus loodavad materjalid kasulikud olla. Lisaks saavad loodavaid õppematerjale kasutada kõik algajad andmebaaside ja SQLi huvilised. Ühe võimalusena näeb autor ka üldhariduskoolide informaatikatundides veebiõpiku kasutamist.

Töö eesmärk ja ülevaade

Töö tulemusena valmib vabalt kättesaadav interaktiivne veebiõpik. Õppematerjalide loomisel on rõhk asetatud näidete olemasolule ja ülesannetele. Rakenduses kasutatakse SQL-päringute kontrollimisel Dockeri konteinereid, mis pakuvad isoleeritud keskkonda päringute jooksutamiseks. Kuigi aine "Sissejuhatus andmebaasidesse" kasutab õpetatava andmebaasi juhtimissüsteemina SQL Anywhere 17, on autor otsustanud PostgreSQL-i kasuks, sest see on avatud lähtekoodiga, vabavaraline, laialdaselt levinud ja omab põhjalikku dokumentatsiooni [1]. Samuti plaanitakse hakata tulevikus aine raames õpetama SQL Anywhere 17 asemel PostgreSQL-i.

Käesoleva töö esimeses peatükis vaadeldakse erinevaid olemasolevaid SQL-i ja andmebaaside kursuseid, kirjeldatakse ainet "Sissejuhatus andmebaasidesse" ning analüüsitakse enamlevinuid raskuskohti nende õppimisel. Teises peatükis tehakse ülevaade praktilise

tööna valmivast interaktiivsest õpikust ning võrreldakse seda teiste sarnaste rakendustega. Kolmandas peatükis kirjeldatakse veebirakendust lähemalt tehnilisest küljest.

1 Andmebaaside ja SQL-i õpetamine

Käesolevas peatükis tehakse ülevaade ainest "Sissejuhatus andmebaasidesse", kirjeldatakse enamlevinud raskus- ja veakohti SQL-i õppimisel ning võrreldakse erinevaid SQL-i õppimisele suunatud veebirakendusi.

1.1 Aine "Sissejuhatus andmebaasidesse"

Aine "Sissejuhatus andmebaasidesse" on 3-ainepunktilise mahuga aine, mida õpetatakse lisaks päevaõppele ka sessioonõppes Tartu Ülikoolis [2]. Selle sihtrühma kuuluvad õppurid, kes ei õpi informaatikat peaerialana, kusjuures enamik neist õpivad rakenduskõrghariduse astmel infokorralduse erialal. Aines kasutatakse andmebaasi juhtimissüsteemi SQL Anywhere 17. Aine lõpus peavad õppurid, kas iseseisvalt või rühmatööna, esitama projekti, milleks on andmebaasi loomine koos andmete ja juurde käivate päringutega.

Tartu Ülikooli õppeinfosüsteemis on välja toodud järgnevad aine õpiväljundid, mida kursuse lõpetanu peaks valdama [2]:

- oskab koostada lihtsaid päringuid andmebaasidele kasutades SQL keeli;
- oskab eristada andmekogumeid, mida saab modelleerida andmebaasisüsteemi abil nendest, mille modellerimiseks tuleb kasutada teisi vahendeid;
- oskab koostada talle tuntud valdkondade lihtsaid andmemudeleid;
- oskab kirjeldada andmebaase SQL-keeles;

Sessioonõppe kursus on jagatud viieks teemaplokiks. Nendeks on:

1. Sissejuhatus andmebaasidesse. Tabeli(te) loomine, andmete sisestamine
2. Graafiline mudel. Seosed olemite vahel. Tabeli(te) muutmine, andmete importimine. Lihtsamad päringud.
3. Relatsiooniline mudel. Primaarvõti ja välisvõti. Tabelite seostamine. Päringud funktsioonidega.
4. Oma Projektide arutelu ja konsultatsioon. Keerulisemad päringud.
5. Andmebaasi administreerimine ja veebiteenused. Vaated.

1.2 Raskused ja enamlevinud vead SQL-i õppimisel

SQL-i õppimise teeb raskeks selle erinevus enamlevinud imperatiivsetest või objekt-orienteeritud programmeerimiskeeltest, millega tudengid on enamasti enne sissejuhatavat andmebaaside ainet kokku puutunud [3]. Seetõttu ei ole varasemad programmeerisalased oskused otseselt rakendatavad ning tuleb omandada uued lähenemisviisid probleemide lahendamisele.

Andmebaaside ainete õppuritele valmistavad suurimat raskust süntaksivead. Need sageli takistavad ülesannete edasist lahendamist erinevalt semantilistest vigadest. Seejuures on süntaktiline viga defineeritud kui PostgreSQL-i andmebaasi juhtimissüsteemi tagastatav veakood ning semantiline viga kui vale tulemust tagastav, ent süntaktiliselt korrektne päring. Ühes uurimustöös vaadeldi õpilaste andmebaaside kursuse käigus lahendusena esitatud päringuid ning leiti, et 54% nendest sisaldas süntaktilisi vigu, 40% semantilisi vigu ning ainult 6% olid veatud. Kõige sagedasemateks vigadeks olid süntaksivead SELECT-lausete erinevates klauslites, defineerimata veerud ja grupeerimisvead [4].

Jyväskylä ülikoolis läbi viidud uurimuses uuriti, millised vead on päringute koostamisel kõige sagedasemad. Leiti, et nendeks on vigane grupeerimine, süntaksivead ning loogilised ja semantilised vead avaldiste moodustamisel ja tabelite seostamisel [5].

1.3 SQL-i õpetamise viisidest

SQL-i õpetamisele on erinevaid lähenemisviise. Selles osas kirjeldatakse neist lähemalt mõnda.

Manchester Metropolitan ülikoolis välja arendatud õpirakendus SQL Tester võimaldab õpilastel viiekümne minuti jooksul lahendada kümmet ülesannet. Rakenduses koosneb ülesandevaade neljast komponendist - ülesandepüstitusest, andmebaasi skeemist, õppuri sisestatud päringu tulemustest ning õige päringu tulemustest. Andmebaasi skeemi kuvamine ülesande kõrval vähendab õppuril kognitiivset koormust ning defineerimata veergude vigu päringutes. Samuti leiti küsitluse teel SQL Testerit kasutanud tudengitelt, et ülesannete lahendamisele aitas kaasa nii sisestatud kui ka õige päringu tulemuste nägemine [6].

SQL Testeri kasutuselevõtuga leiti, et tudengid hakkasid ülesandeid aktiivsemalt lahendama. 90% tudengitest nõustus väitega, et õpperakenduse kasutamisega soovisid rohkem aega pühendada õppimisele; 94% väitega, et soovisid lahendada ülesandeid senikaua, kuni said kätte õige vastuse [6].

Võimalik on rakendada ka mängulist lähenemist õpetamisele. Olemas on õpirakendus, milles saab õppida SQL-i kolmemõõtmelise graafilise rollimänguna. Õppurid paigutavad end mängus detektiivi kingadesse ning peavad kasutama SELECT päringuid kurjategijate vahistamisel, seejuures on võimalik valida erinevate õpistiilide vahel (vastavalt eelistusele kuvatakse selgitusi kas teksti kujul või pildina/diagrammina). Uurimuse käigus leiti, et õppurite õpistiilidele kohandumine ning mänguline lähenemisviis omasid positiivset mõju õppetulemustele. Lisaks sellele oli ka õpistiilide järgi kohanduvat rakendust kasutaval kontrollgrupil kiirem ülesannete lahendamisaeg, võrreldes paberkujul õppematerjale ka-

sutava kontrollgrupiga [7].

Välja on töötatud viis, mis aitab õppuritel hõlpsalt ülesandepüstitus tõlgendada ümber SQL-päringuks. Selle nimeks on *jaga-ja-valitse*. Meetodis on defineeritud termin *päringuühik* kui üks *SELECT ... FROM* päring koos valikulise arvu teiste klauslitega. Ülesandepüstituse põhjal on päringuid võimalik jagada kas ühe- või kaheühikulisteks päringuteks, millest viimased sisaldavad lõpp-päringus hulgateoreetilisi operaatoried (nt *INTERESECT*, *UNION* või *MINUS*) [3].

Kaheühikulisi päringuid on võimalik sageli kirjutada ümber üheühikulisena ehk ilma hulgateoreetiliste operaatoriteta, mistõttu järgnevalt kirjeldatakse vaid üheühikulise päringu lahendamise meetodit.

Üheühikulise päringu lahendamisviisiks on järgmised sammud:

1. Vaadata üle päringu tingimused ja määrata tabelid, kust oleks võimalik andmeid pärida.
2. Juhul, kui tabeleid on mitu, siis tuleb need siduda JOIN lausega FROM-klauslis.
3. Lisada WHERE-klausel, kui andmeid on vaja filtreerida.
4. Kui vaja on ainult kindlaid veerge, siis lisada veerud SELECT-võtmesõna järele.

Näitena, kui olemas on tabel *laul*, mis sisaldab välju *id* (täisarv), *pealkiri* (sõne), *autor* (sõne), *album* (sõne), *pikkus* (täisarv), ning sõnastatud on järgnev ülesanne: "Leia kõikide laulude pealkirjad, mille autor on David Bowie ja pikkus väiksem kui 240 sekundit". Sellisel juhul oleksid ülesande lahendamise etapid järgnevad:

1. Kuna ülesande lahendamisele eelnevalt on teada juba andmebaasi struktuur, siis teame, et vajaminevad andmed peaksid olema kättesaadavad tabelist *laul*.
2. Leidsime eelmises punktis vajaliku tabeli ning saame päringu esialgseks kujuks *SELECT * FROM laul;*
3. Ülesande sõnastusest leiame kaks tingimust - laulu autor on David Bowie ning laulu pikkus on väiksem kui 240 sekundit. Lisame päringule WHERE-klausli ning saame tulemuseks *SELECT * FROM laul WHERE autor = 'David Bowie' AND pikkus < 240;*
4. Leiame ülesande tekstist päringu tulemusena nõutud veeru, milleks on väli *pealkiri*. Saame tulemuseks *SELECT pealkiri FROM laul WHERE autor = 'David Bowie' AND pikkus < 240;*

1.4 Olemasolevad veebipõhised keskkonnad

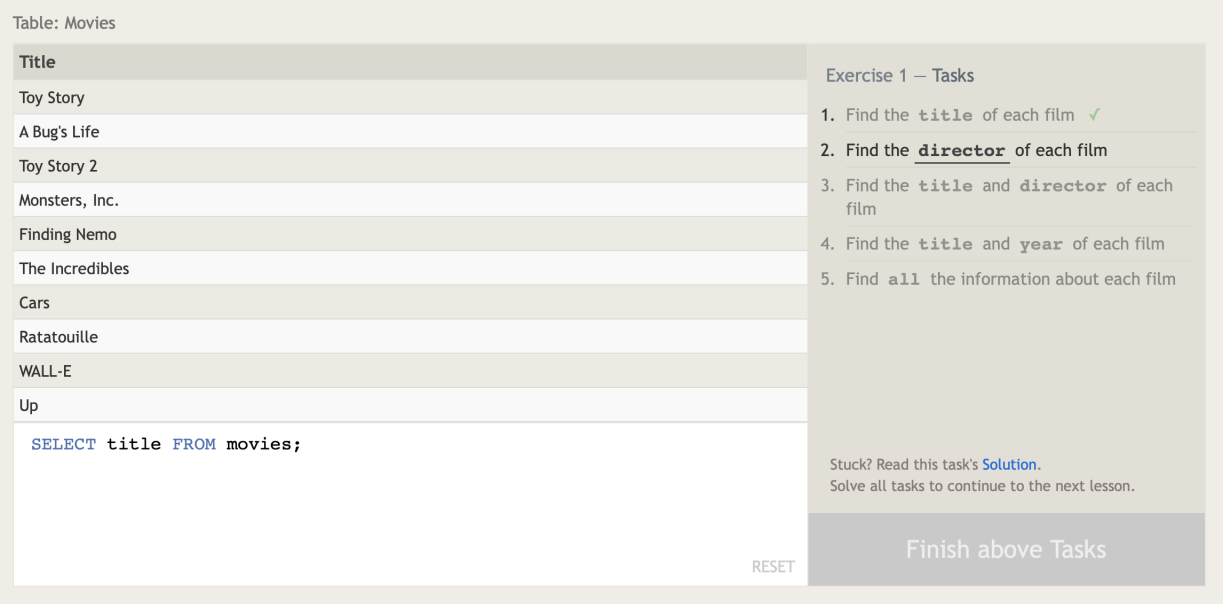
Internetis leidub mitmesuguseid interaktiivseid SQLi õppimisele suunatud lehekülgi. Selles sektsioonis vaatlen lähemalt kolme neist.

1.4.1 SQL Bolt

Üheks interaktiivseks kursuseks on SQLBolt [8]. Selles käsitletavate teemade hulgas on päringud, JOIN-laused, agregeerivad funktsioonid, andmete ja tabelite lisamine/muutmine/kustutamine – kokku 18 peatükki. Iga peatükk koosneb lühikesest tutvustusest ja sellega seonduvast näitest. Peatüki lõpus on interaktiivne sektsioon, kus külastaja saab näidisandmete peal lahendada ülesandeid.

Kursus on tasuta ligipääsetav. Kasutajaliides on erinevatel ekraanisuurustel skaleeruv ning info on visuaalselt selgelt esitatud. Kuna SQL-päringu käivitamisel ei tehta ühtegi päringut üle võrgu, siis tundub, et näidisandmed ei tule füüsilisest andmebaasist, vaid kasutatakse Javascripti. SQL Bolt ei keskendu ühegi konkreetse SQL-dialekti õpetamisele, mistõttu ei pöörata tähelepanu nendevahelistele erinevustele.

Ülesande kuva (vt joonist 1.1) koosneb kasutaja sisestatud päringust, selle tagastavatest tulemustest kui ka ülesannetest. Tagasiside on vahetu, sest päring ei jookse füüsilise andmebaasi peal. Vastust on võimalik näha lingile *Solution* vajutades.

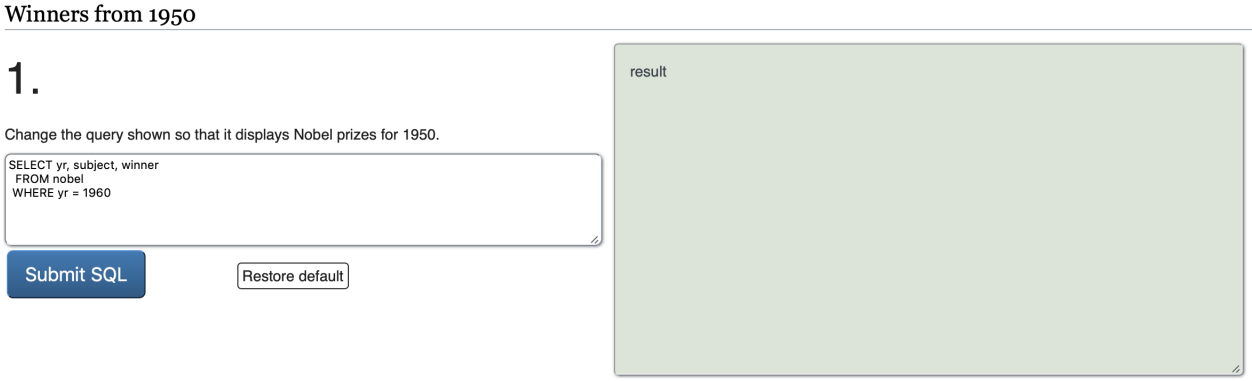


Joonis 1.1: SQL Bolt ülesande kuva

1.4.2 SQL Zoo

Teine sarnane kursus on SQL Zoo [9]. Teemavalik sarnaneb suures osas eelmises alampeatükis kirjeldatud SQL Bolt-ile. Märkimisväärseim erinevus on see, et kasutaja saab valida andmebaasi juhtimissüsteemi, kus sisestatud päring käivitatakse. Valida saab *MySQL*, *Oracle*, *SQL Server* ja *MySQL at Amazon* vahel.

Osasid õppepeatükke toetab ka näitlik video. Ülesannete lehel on alguses kuvatud andmebaasi skeem. Paraku ei ole võimalik osade ülesannete puhul vastuseid näha. Kasutaja sisestatavad päringud jooksevad füüsilise andmebaasi peal, millele on antud piiratud ligipääsuõigus. Ühte ülesannet on näha joonisel 1.2.

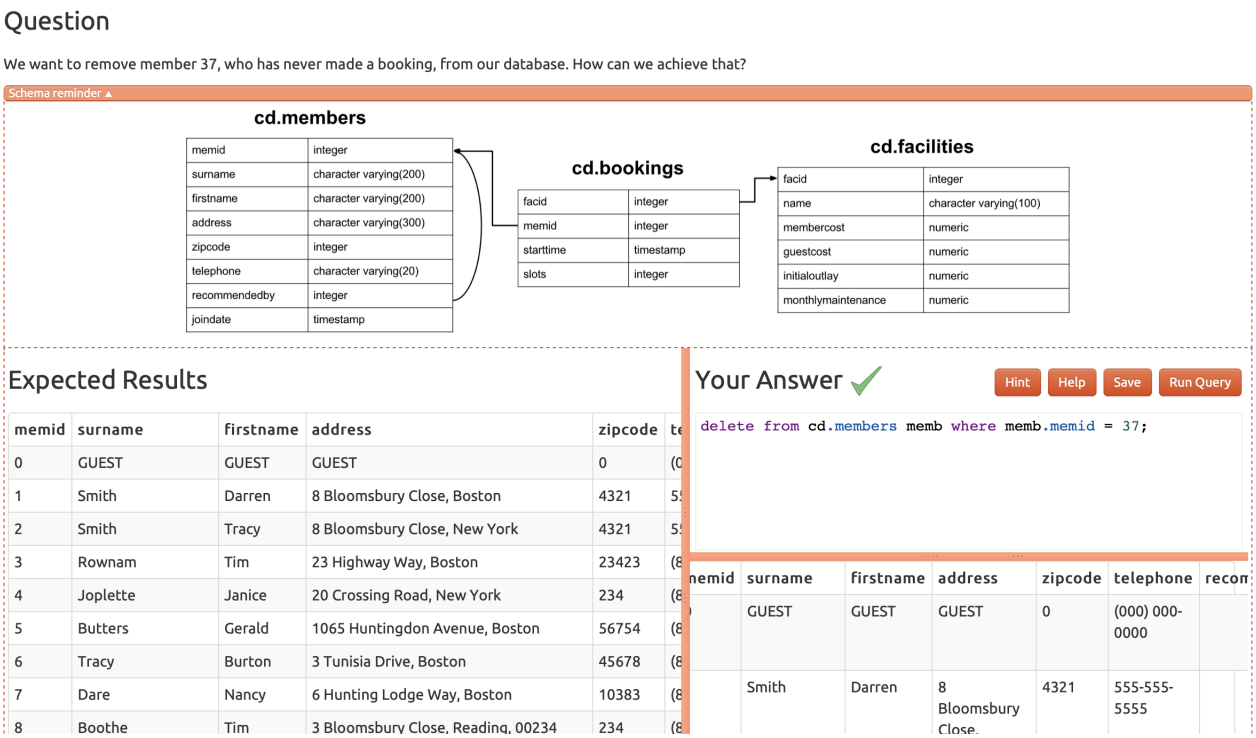


Joonis 1.2: SQL Zoo ülesande kuva

1.4.3 PostgreSQL Exercises

PostgreSQL Exercise keskendub vaid ühe andmebaasi juhtimissüsteemi õpetamisele - PostgreSQL [10]. Teemade valik on sarnane eelmistes alampeatükkides kirjeldatud rakendustele, ent samas põhjalikum, sisaldades ka ülesandeid sõnetöötluse kui ka rekursiivsete päringute kohta.

Ülesande kuva (vt joonist 1.3) sisaldab ülesande küsimust, andmebaasi skeemi, oodatavat tulemust, päringu sisestamiseks mõeldud kastikest kui ka tabelit, mis näitab andmebaasi seisu pärast päringu käivitamist. Samuti on võimalik näha ka vastust koos põhjaliku kirjeldusega. See teeb ülesande lahendamise väga mugavaks, sest kogu vajaminev informatsioon on kompaktselt esitatud.



Joonis 1.3: PostgreSQL Exercises ülesande kuva

2 Veebirakenduse ülevaade

Käesolevas peatükis tehakse ülevaade töö käigus loodud veebirakendusest. Esimeses alampeatükis kirjeldatakse veebirakenduse eesmärke, teises alampeatükis kirjeldatakse veebiõpiku peatükke ja ülesandeid. Kolmandas alampeatükis kirjeldatakse rakenduse kuvaelemente. Rakendus on ligipääsetav veebiaadressil https://andmebaas.dev.

2.1 Eesmärk

Veebirakenduse sihtrühmaks on sessioonõppes või iseseisvalt õppivad inimesed. Oluline on seejuures see, et õppurid saaksid mugavalt lahendada ülesandeid, mis kinnistaksid teadmisi. Rakenduse kaudu ülesannete lahendamine väldib probleemi, et õppurid peaksid ise hakkama testandmeid enda andmebaasi lisama. Selle asemel on võimalik õppida ja harjutada ülesannete lahendamist ühes kohas.

2.2 Peatükkide ja ülesannete ülevaade

Peatükkide valimisel ning nende järjestuse määramisel võttis autor eeskuju mitmetest olemasolevatest interaktiivsetest veebiõpikutest. Peatükkide järjekord ei vasta täpselt aine "Sissejuhatus andmebaasidesse" praktikumide teemade järjekorrale. Näiteks kui sessioonõppe praktikumides õpetatakse tabelite loomist ning andmete sisestamist enne lihtsamaid päringuid, siis antud rakenduses tehakse vastupidi. Puudu on rakendusest ka 5. praktikumi üks teemadest, andmebaasi administreerimine ja veebiteenused. Peatükkide ja järjekorra määramisel on autor lähtunud ka osaliselt isiklikest tarkvaraarendaja töökogemustest, asetades ettepoole teemad, mis on olemuselt fundamentaalsemad ning igapäevases töös sagedasemad.

Rakendus sisaldab järgnevaid peatükke:

1. SELECT, FROM ja WHERE
2. AND, OR ja NOT
3. LIKE ja IN
4. ORDER BY
5. CASE
6. Tabelite loomine
7. INSERT UPDATE ja DELETE

8. Tabelite seostamine
9. Vaated
10. Agregeerivad funktsioonid
11. GROUP BY

Iga peatüki lõpus on ülesanded, mis aitavad õppuritel kinnistada teadmisi. Peatükkide tekstide sees on tähelepanu asetatud ka näidete olemasolule, mis peaksid ülesannete lahendamisele kaasa aitama. Iga ülesande juures on võimalik ka näha vastust, ent õppuri sisestatud päring ei pea olema sama, kusjuures kontrollitakse vaid seda, kas sisestatud SQL-i tagastatavad tulemused või andmebaasi seis on ekvivalentsed vastusena kuvatava SQL-i käivitamisega.

Näiteks peatükis *Tabelite loomine* on sõnastatud järgnev ülesanne, mida on võimalik õppuril lahendada:

> Meil on olemas tabel RAAMAT väljadega id (INTEGER), pealkiri (TEXT), autor (TEXT) ja pikkus (INTEGER). Sisesta tabelisse kirje, kus raamatu pealkirjaks on "Sõda ja rahu", autoriks Lev Tolstoi ning pikkuseks 1225.

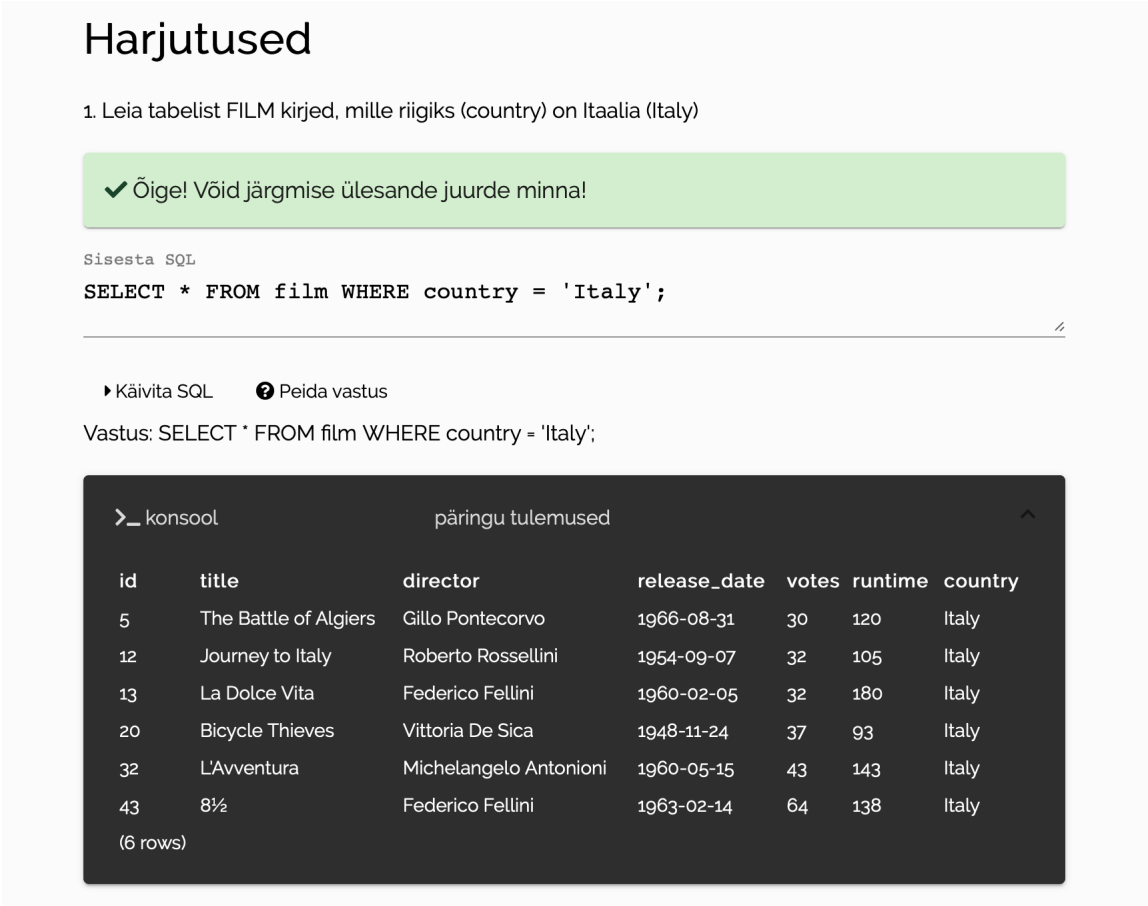
Vastuseks sobivad sellise ülesande puhul nii "INSERT INTO raamat (pealkiri, autor, pikkus) VALUES ('Sõda ja rahu', 'Lev Tolstoi', 1225)" kui ka "INSERT INTO raamat (pikkus, autor, pealkiri) VALUES (1225, 'Lev Tolstoi', 'Sõda ja rahu')", sest automaatkontroll kontrollib ainult tabeli seisu pärast SQL-i käivitamist ning veergude järjekord SQL-i INSERT lauses pole oluline.

2.3 Veebirakenduse kuvad

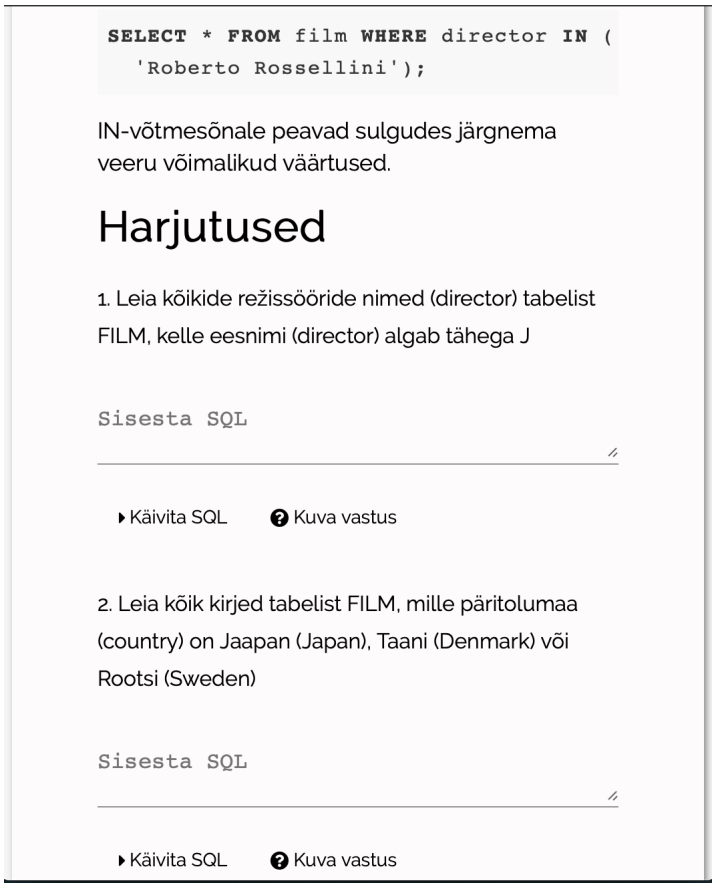
Rakenduse loomisel lähtus autor sellest, et kõik komponendid oleksid ühelt lehelt kättesaadavad. Iga peatüki leht algab õppetekstiga (vt joonist 2.1), mille all on harjutuste sektsioon (vt joonist 2.2). Harjutuste lahendamiseks ei pea õppur navigeerima eemale õpikutekstist, sest need paiknevad lehe allosas. Õppur saab vajadusel lugeda lehe ülaosas olevaid näiteid, mis peaksid aitama ülesannete lahendamisele kaasa, sest ülesannete lahendused peaksid olema tuletatavad näidetest. Samuti kuvatakse päringute puhul ka ülesande all päringutulemusi, mis annab vajalikku tagasisidet selles osas, mida õppuri sisestatud päring tegelikult tegi. Juhul, kui ülesanne osutub liiga keeruliseks, siis saab nupu "Kuva vastus" vajutamisega näha ka ühte võimalikku õiget SQL-lauset.

Tähelepanu on pööratud ka sellele, et lehekülg oleks mugavalt loetav ka mobiiliekraanidel. Joonisel 2.3 on näha, milline näeb rakendus välja väiksemal ekraanisuurusel.

Joonis 2.1: Peatüki algus koos menüüga



Joonis 2.2: Harjutus koos sisestatud päringu tulemusi kuvava konsooliga



Joonis 2.3: Mobiilivaade

3 Veebirakenduse arhitektuur

Peatükk annab ülevaate veebirakenduses kasutatavatest tehnoloogiatest. Lisaks on ka alampeatükk edasiarendamise võimaluste kohta. Kood on kättesaadav GitHubi koodihoidlates github.com/jlaamann/smart-sql-be ja github.com/jlaamann/smart-sql-web.

3.1 Docker

Antud rakenduse puhul oli vajalik, et igakordsel päringu käivitamisel oleks andmebaasis alati sama seis. Näiteks, kui õppur kustutab kõik tabelikirjed ära, siis järgmise päringu käivitamisel peavad olema need alles. Selleks, et seda saavutada, kasutas autor tarkvara Docker.

Docker on käsureaprogramm ja taustal töötav deemon (ingl daemon), mis lihtsustab tarkvaraarenduse protsessi. Selle peamiseks eesmärgiks on võimaldada rakendustel töötada operatsioonisüsteemist isoleeritud keskkonnas, mida nimetatakse konteineriks. Dockeri konteinerid ehitatakse üles Dockeri pildist, mis sisaldab viiteid kõikidele konteineri jooksutamiseks vajalikele failidele ning muudele konfiguratsioonireeglitele [11].

Erinevate Dockeri piltide jaoks on olemas ka avalik register, mida haldab ettevõte Docker Inc. Näiteks on seal kättesaadav Dockeri pilt *postgres*, mis sisaldab PostgreSQL andmebaasi [12]. Sellel pildil põhinevat konteinerit saab käivitada järgneva käsuga:

```
docker run --name konteineri-nimi -e POSTGRES_PASSWORD=parool
-d postgres
```

Autor vajas, et Dockeri konteineris töötav andmebaas sisaldaks testandmeid, mille peal õppurid saaksid ülesandeid lahendada. Selle jaoks ei piisanud ainult avalikust registrist pärit Dockeri pildi käivitamisest, vaid tuli defineerida eraldiseisev Dockerfile. Selle sisu on järgnev:

```
FROM library/postgres
COPY ./docker-entrypoint-initdb.d/ /docker-entrypoint-initdb.d/
```

Võtmesõna FROM järel määratakse, milline Docker pilt võetakse aluseks. COPY järel olev esimene kaust peaks vastama failiga Dockerfile samas kaustas olevale kaustale *docker-entrypoint-initdb.d*. Kausta sisu kopeeritakse samanimelisse kausta konteineris, kus käivitakse selle sees olevad .sql ja .sh failid.

Autor lisas eelmainitud kausta Bash skripti ja hetktõmmise lokaalses andmebaasis olevast tabelist, millele on üles ehitatud õppurite lahendatavad ülesanded. Skript sisestab hetktõmmise Dockeri konteineris töötavasse andmebaasi.

Veebirakenduses, kui õppur sisestab SQL-päringu ülesande vastusena, siis taustal käivitatakse Dockeri konteiner. Selle sees omakorda käivitatakse Bash kest käsuga *docker exec -it konteineri_nimi /bin/bash*. Kestas on kättesaadav PostgreSQL-i interaktiivne terminal *psql*, millega on võimalik andmebaasis SQL-lauseid käivitada.

3.2 Kasutatud veebiraamistikud

Järgnevalt vaadeldakse rakenduse loomisel kasutatud tehnoloogiaid lähemalt. Front-end rakendus arendati veebiraamistikus Angular 2 ning serveripoolne rakendus raamistikus Spring Boot.

3.2.1 Angular 2

Angular 2 on ettevõtte Google arendatav front-end veebiraamistik. Kuigi selle nimes esineb arv 2, on nüüdseks väljas juba 7. versioon, mida autor ka kasutas [13]. Angular võimaldab hõlpsalt jagada rakenduse erineva funktsionaalsuse eraldi komponentidesse. Samuti võimaldab veebiraamistik mugavalt hallata süsteemi olekut.

Angulari programmikood kirjutatakse keeles TypeScript, mida arendab ettevõte Microsoft [14]. TypeScript põhineb keelel JavaScript, lisades sellele juurde funktsionaalsust, mille seas on ka staatiline tüübisüsteem.

Angulari rakenduses on programmikood jaotatud erinevatesse kihtidesse. Nendest kõige kasutajapoolsem kiht on komponent. Sellele vastavad tavaliselt TypeScript-fail, mis sisaldab dekoraatorit *@Component*, HTML-fail ja CSS-fail. HTML-fail ei pea olema eraldiseisev ning selle saab ka lisada dekoraatorisse *@Component* välja *template* väärtusena. Komponendid suhtlevad teenuse kihiga (ingl service), mis omakorda suhtlevad hoidla kihiga (ingl repository), kus saadetakse REST-päringuid serveripoolsele rakendusele. Teenuse kihis töödeldakse saadetavaid või päritud andmeid. Selline jaotusviis muudab programmikoodi paremini hallatavaks ning võimaldub teatud funktsionaalsuse teha taaskasutavaks.

Osad veebirakenduse kuvaelemendid (näiteks nupud ja tekstiväljad) pärinevad disainiraamistikust Angular Material [15]. Angular Material toetab kahte kõige viimast versiooni kõige populaarsematest veebibrauseritest (Chrome, Firefox, Safari, IE11/Edge) [16].

Dockeri konteineri käivitamiseks ning selle sees päringu jooksutamiseks kulub ligikaudu 5 sekundit, mistõttu on vaja anda kasutajale visuaalset tagasisidet, et süsteem reageeris nupu "Käivita SQL" vajutamisele. Autor otsustas siinkohal kasutada Angulari teeki ngx-progressbar, mille ülesseadmine on väga lihtne [17]. Nupu vajutamisel tekib veebilehe päise vasakusse nurka punane riba, mis hakkab terve lehe laiuse ulatuses vasakult paremale liikuma.

3.2.2 Spring Boot

Spring Boot on Java-põhine serveripoolne veebiraamistik. Hetkel on väljas selle 2. versioon, mida autor ka rakenduse kirjutamisel kasutas [18].

Serveripoolse rakenduse loomisel lähtuti sellest, et see oleks olekuvaba (ingl stateless). See tähendab seda, et server ei hoiaks mälus, millisel lehel kasutaja parasjagu viibib, vaid selle asemel käitub õhukese vahelülina front-end rakenduse, andmebaasi ja Dockeri konteineri vahel. Rakendus võtab vaid vastu kliendipoolselt rakenduselt tulevaid REST-päringuid.

Andmebaasi muudatuste tegemiseks kasutati teeki Liquibase, mis konfigureeriti rakenduse käivitamisel konfiguratsioonifailis kindlaksmääratud kausta lisanduvaid või muutuvaid SQL-faile jooksutama. Näiteks, kui on vaja andmebaasi sisetada tabel *isik*, siis tuleks lisada eelmainitud kausta järgnev SQL-fail:

```
--liquibase formatted sql

--changeset autor:000-create-table-isik
CREATE TABLE isik (
        id SERIAL,
        eesnimi TEXT,
        perenimi TEXT,
        synnipaev DATE
);
```

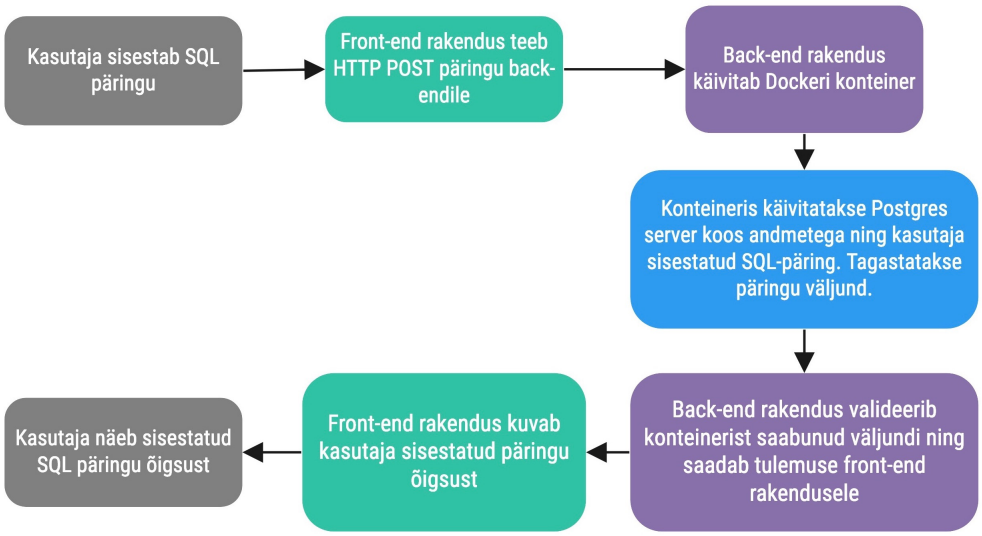
Liquibase loob andmebaasi tabeli *databasechangelog*, kuhu skripti käivitamisel sisestatakse autor, failinimi, silt (eelmise näite puhul *000-create-table-isik*), MD5 kontrollsumma faili sisust ning veel mõned väljad. Juhul, kui Liquibase on juba faili töödelnud, aga faili sisu muutub, siis rakenduse taaskäivitamisel tuleb veateade. Sellisel juhul tuleks muuta faili silti või lisada uus fail muudetud sisuga.

Lokaalsest andmebaasist andmete pärimiseks kasutati teeki JpaRepository. Selle kasutamisel pole vaja SQL-päringuid kirjutada - piisab vaid liidesest, mis omakorda pärib kas liidest *CrudRepository<T,ID>* või *JpaRepository<T,ID>*. Liideses tuleks defineerida meetodid, mille nime ja tagastustüübi põhjal genereeritakse implementatsioon [19]. Näiteks, kui soovime eelmainitud tabelist *isik* pärida kõiki kirjeid sünnikuupäeva järgi kahanevalt, siis tuleks defineerida järgnev liides ja meetod:

```
public interface IsikRepository extends JpaRepository<Isik, Long> {
    List<Isik> findAllByOrderBySynnipaevDesc();
}
```

3.3 Andmebaas

Lokaalseks andmebaasiks serveripoolsel rakendusel valis autor PostgreSQL 11, sest see on vabavaraline ja põhjaliku dokumentatsiooniga [1]. Andmebaas on üsna väike, sisaldades tabelit *exercise*, mis sisaldab ülesande tekste koos vastustega. Samuti ka ülesannete aluseks olevaid tabeleid, mis kopeeritakse Dockeri konteineris olevasse andmebaasi, kuid rakenduses neid lokaalsest andmebaasist ei pärita. Võimalik oleks ka lisada peatükkide tekstid andmebaasi ning pärida neid REST-päringute kaudu front-end rakenduses, ent arenduse lihtsustamiseks jättis autor need front-end rakenduse failikogusse.



Joonis 3.1: SQL-päringu hindamise etapid rakenduses

3.4 Päringute hindamine

Joonisel 3.1 on näha, mis toimub siis, kui kasutaja vajutab nuppu "Käivita SQL".

Üheks alternatiiviks Dockeri konteineri asemel oleks olnud käivitada päringuid piiratud õigustega lokaalse andmebaasi kasutajana, ent sellega oleks päringute automaatne kontrollimine piirdunud vaid SELECT-lausetega. Dockeri konteinerid tagavad selle, et alati oleks igal ülesande kontrollil andmebaasis sama seis.

SELECT-päringute automaatkontroll võrdleb, kas õige päringu tagastatav andmehulk on sama, mille tagastab õppuri sisestatud päring. Selleks leitakse järgnev ühend: (õige päring - kontrollitav päring) + (kontrollitav päring - õige päring). Juhul, kui eelnevas ühendis pole ühtegi kirjet, siis on mõlemad päringud võrdsed.

ORDER BY päringutega tehakse esmalt sama kontroll, mis eelmises lõigus SELECT päringutega. Seejärel kontrollitakse, kas kasutaja sisestatud päringu tagastatud ridade järjekord on sama, mis õigel päringul. Kui need kaks tingimust on täidetud, siis on sisestatud päring korrektne.

Tabeli loomise kontrollimisel käivitatakse esmalt õppuri sisestatud SQL-lause. Seejärel üritatakse sisestada INSERT lausega sinna andmeid. Juhul, kui see õnnestub, siis loetakse sisestatud SQL-lause õigeks.

INSERT lausete ülesannetel on aluseks kindel tabel, kuhu õppur peab õiged andmed sisestama. Sisestatud INSERT lauset kontrollitakse nii, et automaatkontroll teeb SELECT päringu, mille tingimustes on samad väärtused, mis INSERT lauses. Juhul, kui SELECT päring tagastab sisestatud rea, siis loetakse sisestatud SQL-lause õigeks. UPDATE lauseid kontrollitakse sarnaselt INSERT lausetele.

DELETE lausete ülesannete puhul tuleb õppuril kustutada mõni olemasolev kirje andmebaasi tabelist. Automaatkontroll kontrollibki, kas antud kirje kustus.

JOIN lauseid kontrollitakse sarnaselt SELECT lausetega. Erinevuseks on see, et ülesanded põhinevad rohkem kui ühel tabelil, mida on vaja õppuril õigete seoste kaudu päringus ühendada. Kuna rakendus kontrollib vaid seda, kas tagastatavad andmed on õiged, siis loetakse õigeks ka sellised päringud, kus JOIN lausete asemel on kasutatud näiteks alampäringuid.

3.5 Rakenduse veebimajutus

Rakenduse majutamiseks kasutati teenusepakkujat Digital Ocean [20]. Serveri operatsioonisüsteemiks on 64-bitine Ubuntu 18.04. Masinal on 1 CPU, 2GB muutmälu ja 50GB SSD kettamaht.

3.6 Kasutusjuhend

Rakendus töötab UNIX-il põhinevatel süsteemidel, nende seas Linuxil ja MacOS-il. Kuna osa funktsionaalsusest on kirjutatud Bash-skriptidena, siis tuleks Windowsi operatsioonisüsteemil kasutada Command Prompti asemel teist käsurea kesta, mis võimaldab käivitada UNIX-i käske. Üheks selliseks Windowsi rakenduseks on Cygwin [21].

Front-end rakenduse käivitamiseks tuleb kasutada järgnevaid käske käsureal:

```
npm install;
npm start;
```

Käsk *npm install* laeb alla kõik failis *package.json* defineeritud moodulid ja *npm start* käivitab serveri koos rakendusega.

Serveripoolse rakenduse käivitamiseks on vaja kõigepealt konfigureerida konfiguratsioonifail *application.properties*, mille jaoks on koodihoidlas olemas ka näidisfail *sample.application.properties*. Tuleb ka luua konfiguratsioonifailis määratud PostgreSQL-i andmebaas ning seejärel käivitada käsureal järgnev käsk:

```
gradle bootrun
```

Samuti tuleb ka luua Dockeri pilt serveripoolse rakenduse kaustas *etc/* oleva faili *Dockerfile* põhjal.

```
docker build -t pg-docker Dockerfile_asukoht
```

Linuxi operatsioonisüsteemil tuleks käivitada ka järgnev käsk, et *Dockeri* käivitamiseks ei oleks vaja administraatori õiguseid:

```
sudo usermod -aG docker ${USER}
```

Kaustas *etc/* ja selle alamkaustades olevatele .sh skriptidele tuleb anda ka käivitamisõigus:

```
chmod +x etc/*.sh;
```

4 Kokkuvõte

Käesoleva töö käigus valmis interaktiivne veebiõpik, kus õppurid saavad PostgreSQL-i baasil õppida SQL-i põhialuseid. Rakendus on suunatud aine "Sissejuhatus andmebaasidesse" õppuritele kui ka teistele andmebaaside algteadmiste huvilistele. Rakendus on ligipääsetav veebiaadressil https://andmebaas.dev.

Veebiõpik koosneb üheteistkümnest peatükist. Iga peatüki lõpus on ka mitu ülesannet, mis võimaldavad õppuritel teadmisi kinnistada.

Rakenduse loomisel kasutati Dockerit, et käivitada SQL-i käske isoleeritud andmebaasis. Serveripoolne rakendus implementeeriti Java raamistikus Spring Boot ning kliendipoolne rakendus raamistikus Angular 2.

Hetkel on peatükkide tekstid ja ülesanded ehitatud üles PostgreSQL-i peale. Edasiarendamise võimalusena saaks ka lisada teisi andmebaasi juhtimissüsteeme, mis võimaldaks kohandada õppuril õppekeskkonda enda vajadustele vastavaks.

Viited

[1] M.Asay infoworld.com, 07.12.2017, https://www.infoworld.com/article/3240064/why-old-school-postgresql-is-so-hip-again.html (25.04.2019)

[2] ois2.ut.ee, Aine "Sissejuhatus andmebaasidesse" üldinfo, https://ois2.ut.ee/#/courses/MTAT.03.105/details (05.04.2019)

[3] Qian. G Teaching SQL: A Divide-And-Conquer Method For Writing Queries, *Journal of Circuits, Systems and Computers (JCSC)*, 2018

[4] Ahadi A., Behbood V., Vihavainen A., Prior J., Lister R. Students' Syntactic Mistakes in Writing Seven Different Types of SQL Queries and its Application to Predicting Students' Success, *Special Interest Group on Computer Science Education (SIGCSE)*, 2016

[5] Taipalus T., Perälä P. What to Expect and What to Focus on in SQL Query Teaching, *Special Interest Group on Computer Science Education (SIGCSE)*, 2019

[6] Kleerekoper A., Schofield A. SQL Tester: An Online SQL Assessment Tool and Its Impact, *Innovation and Technology in Computer Science Education (ITiCSE'18)*, 2018

[7] Soflano M., Connolly T., Hainey T. An application of adaptive games-based learning based on learning style to teach SQL, *Computers & Education 86*, 2015

[8] sqlbolt.com, SQL Bolt, https://sqlbolt.com (14.04.2019)

[9] sqlzoo.net, SQL Zoo, https://sqlzoo.net (14.04.2019)

[10] pgexercises.com, PostgreSQL Exercises, https://pgexercises.com (14.04.2019)

[11] Nickoloff J. *Docker in Action*, Manning Publications, 2016

[12] hub.docker.com, PostgreSQL-i Dockeri pildi dokumentatsioon, 2019, https://hub.docker.com/_/postgres (23.03.2019)

[13] angular.io, Angular 2 ametlik dokumentatsioon, 2019, https://angular.io/docs (21.03.2019)

[14] typescriptlang.org, TypeScripti dokumentatsioon, 2019
https://www.typescriptlang.org/docs/home.html (21.03.2019)

[15] https://material.angular.io/, Angular Material dokumentatsioon, 2019, https://material.angular.io/ (21.03.2019)

[16] https://github.com/, Angular Material GitHub koodihoidla, 2019, https://github.com/angular/material2 (02.04.2019)

[17] npmjs.com, @ngx-progressbar/core, 2019, https://www.npmjs.com/package/@ngx-progressbar/core (21.03.2019)

[18] docs.spring.io, Spring Boot dokumentatsioon, 2018, https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/ (21.03.2019)

[19] spring.io, Getting started with Spring Data JPA, 2011, https://spring.io/blog/2011/02/10/getting-started-with-spring-data-jpa/ (21.03.2019)

[20] digitalocean.com, Digital Ocean 2019, https://www.digitalocean.com/ (21.04.2019)

[21] cygwin.com, 2019, https://www.cygwin.com/ (21.03.2019)

Lisad

4.1 Lähtekood

Programmi koodihoidlad asetsevad järgnevatel aadressidel:

1. https://github.com/jlaamann/smart-sql-be (serveripoolne Spring Boot rakendus)
2. https://github.com/jlaamann/smart-sql-web (kliendipoolne Angular 2 rakendus)

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, Johannes Erik Laamann

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

   Interaktiivse veebiõpiku loomine ainele "Sissejuhatus andmebaasidesse",

   mille juhendaja on Piret Luik,

   reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja 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 punktis 1 nimetatud õigused jäävad alles ka autorile.

4. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Johannes Erik Laamann
10.05.2019