TARTU ÜLIKOOL

Arvutiteaduse instituut

Informaatika õppekava

Martti Kakk

Automaattestid andmebaaside ainele

Bakalaureusetöö (9 EAP)

Juhendaja: Vambola Leping, MSc

Tartu 2020

Automaattestid andmebaaside ainele

Lühikokkuvõte:

Käesoleva bakalaureusetöö raames valmisid automaattestid ainele „Andmebaasid“. Töö eesmärk oli kiirendada ja ühtlustada kodutöö ülesannete kontrollimist ja hindamist. Bakalaureusetöö koosneb autori kirjutatud skriptist ja analüüsist, kus võrreldakse vana ja uut hindamisviisi, antakse ülevaade skripti ülesehitusest ja töötamisest ning kirjeldatakse ka edasiarendamisvõimalusi.

Võtmesõnad:

Andmebaas, automaattestimine, SQL

CERCS: P175 Informaatika, süsteemiteooria

Automated tests for Databases course

Abstract:

This Bachelor’s thesis is about developing automated tests for the course “Databases”. The aim of the thesis was to fasten and to equalize the evaluation and grading of student homework. The thesis contains the script and the analysis in which the author compares the old and the new assessment method, gives an overview of the script and how it works and describes how it could be further developed in the future.

Keywords:

Database, automated tests, SQL

CERCS: P175 Informatics, systems theory

Sisukord

Sissejuhatus

Bakalaureusetöö eesmärk on automatiseerida kodutööde kontrollimist aines „Andmebaasid“ (LTAT.03.004). Aine „Andmebaasid“ on oluline, sest annab põhitõed ja baasteadmised andmebaasidest. Kuna viimastel aastatel on ainest huvitatute arv kasvanud, on sellega tõusnud ka õppejõudude koormus kodutööde kontrollimisel, sest siiani on kodutööde kontrollimine toimunud käsitsi. Seetõttu on mõistlik luua uusi vahendeid ülesannete lahenduste automaatseks kontrolliks.

Bakalaureusetöö koosneb autori kirjutatud skriptist ja analüüsist, mis on jaotatud kaheksaks sisupeatükiks. Esimeses peatükis defineeritakse töös esinevad mõisted ja terminid. Teises peatükis seletatakse, mis on automaatkontroll ja tutvustatakse lühidalt varasemaid sarnaseid lõputöid. Kolmandas peatükis kirjeldatakse, millised on aines „Andmebaasid“ antavad kodused tööd.

Neljandas peatükis kirjeldatakse kodutöö lahenduse hindamise protsessi nii vana kui ka uue hindamisviisi järgi. Viiendas peatükis on kirjas süsteeminõuded, mida on vaja, et automaatkontrolli saaks käivitada. Kuuendas peatükis kirjeldatakse skripti ülesehitust. Seitsmendas peatükis kirjeldatakse põhiliste kontrollprotseduuride töötamist. Kaheksandas peatükis kirjeldatakse loodud automaatkontrolli edasiarendusvõimalusi.

1. Mõisted ja terminid

Andmebaas (ingl *database*) on süstemaatiline andmete kogum, mille abil saab andmeid koguda ja kasutada. Andmebaasid teevad andmete töötlemise lihtsamaks. (Guru99)

Funktsioon (ingl *function*) on spetsiifilist toimingut sooritav tarkvaramoodul, mis aktiveerub ta nime ilmumisel avaldises, saab vastu võtta sisendväärtusi ja väljastab üheainsa väärtuse (Cybernetica, 2020).

Indeks (ingl *index*) on järjestuse määraja, mille abil on tabelitest infot kiirem leida, eriti kui andmete hulk on mahukas (SAP, 2016).

Kitsendus (ingl *constraint*) on andmetüübi mugandus, mis kitsendab selle vahemikku või tehteid (IT terministandardi..., 2001).

Pakkfail (ingl *batch file*) sisaldab pakina töödeldavaid üksusi (Hanson, Tavast, 2005). Iga üksus on käsu rida, mis käivitatakse järjest pakkfaili käivitamisel. (Laurie)

Protseduur (ingl *procedure*) on alamprogramm, mis koosneb SQL päringutest (Guru99), mida saab korduvalt käivitada ning protseduur võib tagastada tulemuse, aga ei pea (W3schools).

SQL skriptifail (ingl *SQL script file*) on tekstifail, mille sees on SQL laused ja see on kasulik kui soovitakse käivitada samasid SQL lauseid mitmel korral (Sybase, 2005).

Struktureeritud päringukeel (ingl SQL - *structured query language*) on poolstandardne päringukeel. Päringukeel on komplekt reegleid, mille alusel konstrueeritakse päringuid andmebaasist andmete otsimiseks. (Liikane, Kesa, 2006)

Süsteemne triger (ingl *system trigger*) on triger, mis luuakse koos välisvõtmega, kui välisvõtmes on juures tingimused kustutamiseks või muuks operatsiooniks (Sybase, 2005).

Tabel (ingl *table*) on andmete selline korraldus, mille puhul iga andmeüksust saab identifitseerida argumentide või võtmete abil ja informatsioon on paigutatad ridadesse ja tulpadesse (Liikane, Kesa, 2006).

Triger (ingl *trigger*) on protseduur, mis jälgib teatud tabelis muutuseid, kui tabelis toimub sündmus, siis triger käivitub (W3resource, 2020).

Vaatel on read ja veerud nagu päris tabelil. Vaate informatsioon on võetud ühest rohkemast andmebaasi tabelist. (W3schools)

Veerg (ingl *column*) on üks kahest või mitmest leheküljel või ekraanil kõrvuti paigutatud püstisest reakogumist (Liikane, Kesa, 2006).

Välisvõti (ingl *foreign key*) on võti, millega ühendatakse kahte tabelit. Välisvõti seob ühte või mitut veergu alamtabelis vanemtabeli primaarvõtmega. (W3schools)

2. Automaatne lahenduste kontroll ja selle kasutamise võimalused

Automaatkontroll ehk automaattestimine on ühe programmi testimine teise tarkvara abil ning tegeliku saadud tulemuse kontrollimine ennustatava tulemusega (Siiber, 2015). Testide automatiseerimine on vajalik, kui on programmikoode, mida tuleb korduvalt kontrollida, ja koode, mis on keerulised ja mahukad. Veel on automaattestide eelisteks kiirus ja inimlike vigade puudumine (Sepp, 2018).

Automaatkontrolli puudusteks on lähtekoodi pidev muutumine ja see, kui testimisel pole arvestatud kõigi testimisvõimalustega, mistõttu tekib vale arusaam, et testid läbitakse edukalt (Sepp, 2018). Käesolev lõputöö ei ole aga automaattestimine, vaid automaatne lahenduste testimine ja hindamine. Tegelikult testitakse ja hinnatakse koodi asemel andmebaasi vastavust ülesannete kaudu esitatud funktsionaalsusele.

Tartu Ülikoolis on mitmetele ainetele – näiteks „Programmeerimine“ ja „Automaadid, keeled ja translaatorid – tehtud automaattestid, mis kontrollivad tudengite kodutöid. Nendes ainetes on automaatkontrollid Moodle'is, kuhu tudeng esitab oma kodutöö ning automaatkontroll annab esitatud tööle tagasiside.

2.1. Varasemad lõputööd ja automaattestid ainele „Andmebaasid“

Varasemalt on selle aine raames teinud 2015. aastal bakalaurusetöö Marten Siiber pealkirjaga „Aine „Andmebaasid“ automaattestimise vahend“ ja 2018. aastal Robert Sepp pealkirjaga „Automaattestide loomine andmebaasile Edu“. Kuna Marten Siiberi bakalaurusetöö automaattestid tehti viis aastat tagasi, on need aegunud ja tema loodud programm ei leidnud ka kuigi laialdast kasutust. Siiberi loodud automaattestide inspiratsioonil hakati aga looma käesoleva bakalaurusetöö automaatteste. Robert Sepa automaattestid on loodud „Edu“ andmebaasile, käesolev töö aga keskendub „Ope“ andmebaasidele, seega pole Sepa töö otseselt käesolevaga seotud.

Ka ainele „Sissejuhatus andmebaasidesse“ loodi 2019. aastal automaattestid (Mikk Õunamaa bakalaureusetöö „Automaattestide loomine sessioonõppe ainele „Sissejuhatus andmebaasidesse“), aga neid ei võetud kunagi kasutusele. Käesoleva bakalaureusetöö automaattestidega saab kontrollida ka aine „Sissejuhatus andmebaasidesse“ kodutöid, kuid

kuna kodused ülesanded on ainetes pisut erinevad, on automaatkontroll üles ehitatud nii, et selle abil saab kontrollida ükskõik kumma õppeaine koduste tööde lahendusi.

3. Kodutööd ja nende kontrollimine

Õppeaines „Andmebaasid“ hakkavad automaattestid kontrollima nelja ülesannete komplekti (edaspidi kodutööd): kolmas, viies, kuues ja seitsmes. Põhjuseks on see, et esimene ja teine kodutöö tehakse praktikumis ning neljas sisaldab vaid päringute kirjapanekut, seepärast neid automaatkontrolliga ei kontrollita.

Kolmas kodutöö on esimene, mida tudengid peavad iseseisvalt kodus tegema. Enne automaatkontrolli loomist toimus selle kodutöö kontrollimine praktikumis, kus õppejõud käis kõik tudengid ükshaaval läbi ja vaatas, kas neil on kõik õigesti tehtud ja andmed korrektselt sisestatud. Tänu loodud programmile saavad tudengid nüüd selle kodutöö esitada Moodle’is, mistõttu on õppejõu töökoormus kodutöö kontrollimisel kordades väiksem ning praktikumis saab kohe asuda praktikumiülesannete juurde. Teisi koduseid töid kontrollis õppejõud küll praktikumivälisel ajal, aga ka nende kontrollimisele kuluv aeg on automaatkontrolli kasutades kordades väiksem.

3.1. Ülesanded kodutöödes ehk funktsionaalsuse nõuded andmebaasile

Kolmanda kodutöö rõhuasetus on tabelite ja välisvõtmete loomisel ning andmete sisestamisel tabelitesse. Tabelid on „Isikud“, „Klubid“, „Turniirid“ ja „Partiid“. Kahe tabeli vahelised välisvõtmed on tabelite „Isikud“ ja „Klubid“, „Isikud“ ja „Partiid“ ning „Turniirid“ ja „Partiid“ vahel. Andmete sisestamiskorrektsust kontrollitakse kõigis tabelites.

Viiendas kodutöös luuakse vaateid. Vaated on „Edetabelid“, „Kolmik“, „Klubipartiid“, „Klubipartiikogused 1“, „Klubipartiikogused 2“, „Mängijad“, „Partiid“ ja „Punkt“. Kui skripti alguses on muutuja „aine“ väärtuseks „S“, siis on vaadete „Klubipartiikogused 1“ ja „Klubipartiikogused 2“ kontrollide asemel ainult vaate „Klubipartiikogused“ kontroll. Seetõttu muutuvad eri kohtades ka ülesannete eest saadavad punktid.

Kuuendas kodutöös loovad tudengid funktsioone, protseduure ja indekseid. Funktsioonid on „Klubisuurus“, „Nimi“, „Mängija punktid turniiril“, „Mängija koormus“ ja „Mängija võite turniiril“. Protseduurid on „Uus isik“, „Top10“, „Võit-viik-kaotus“ ja „Infopump“. Indeksid on tabeli „Turniirid“ veerul „Alguskuupäev“ ja tabeli „Partiid“ veergudel „Musta tulemus“ ja „Valge tulemus“.

Seitsmendas kodutöös on põhirõhk trigerite loomisel, aga nendele lisandub ka üks tabel, 14 vaadet ja üks protseduur. Trigerid on „Lisa klubi“, „Kustuta klubi“ ja „Kustuta klubi isikutega“.

4. Kodutöö hindamine

Kodune töö koosneb andmebaasifailist laiendiga .db ja logifailist laiendiga .log. Tudeng pakib need failid zip-failiks ning esitab kokkupakitud faili Moodle'isse.

Logifaili on algsel kujul võimatu lugeda, aga SQL Central programmiga saab logifaili muuta tavaliseks tekstifailiks. Seejärel saab tekstifailist kuupäeva ja kellaaja kaupa vaadata, missuguseid käske tudeng kasutas, et luua andmebaasis olevad tabelid ning muud seosed.

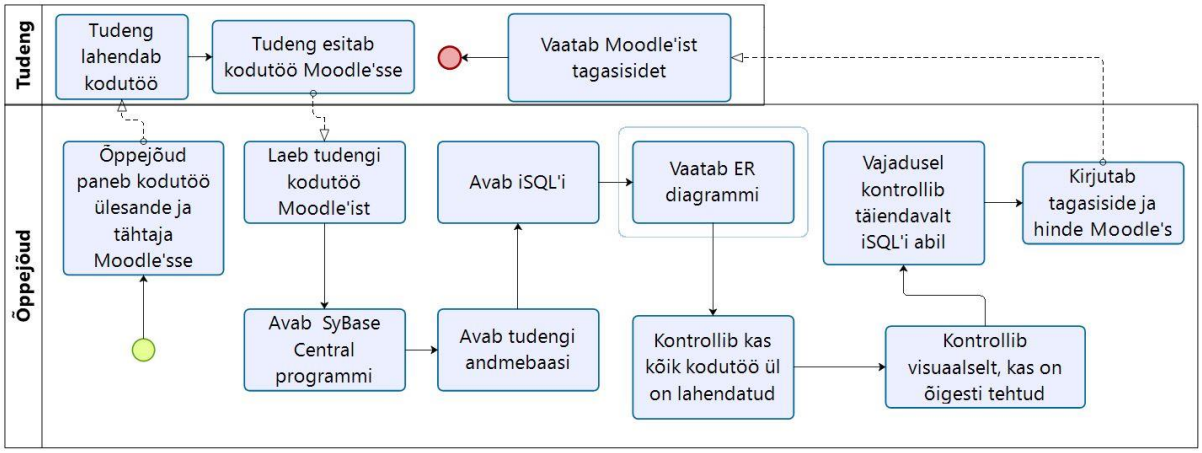
Kui tudengi kodutöö kontrollimisel tekib kahtlus, et töö on plagiaat ehk kellegi teise kodutöö pealt maha kirjutatud, siis selle tõestamiseks on võimalik õppejõul kontrollida logifailist tudengi sisestatud käskude aegasid. Näiteks kui tudeng sisestas ühe käsu kell 15:30 ja järgmise juba 15:31 ning selle käsu väljamõtlemine võtaks loogiliselt kauem aega kui 5 minutit, siis on tõenäoline, et tegu on plagieeritud kodutööga.

4.1. Vana hindamisprotsess

Vana hindamisprotsess toimus käsitsi ja oli võrdlemisi ajamahukas (joonis 1). Ettevalmistuseks laadis õppejõud alla tudengi kodutöö ja pakkis selle lahti. Seejärel avas õppejõud SQL Central programmi, mis on graafilise liidesega andmebaasiklient andmebaaside haldamiseks. Järgmisena võttis ta SQL Central'i vahendusel ühendust tudengi andmebaasiga ning käivitas Interactive SQL'i (edaspidi iSQL), mida kasutatakse objektide loomiseks ja päringute esitamiseks.

Kui õppejõud alustas kodutöö kontrollimist, vaatas ta esmalt ER-diagrammi. ER-diagramm annab hea ülevaate andmebaasis olevatest tabelitest ja tabelitevahelistest seostest ehk välisvõtmetest.

Seejärel vaatas õppejõud tabeleid. Kontrollis, kas kõik veerud on olemas ja kas neid on õige arv. Seejärel vaatas, kas veergudel on vajalikud kitsendused, näiteks *unique* või *check*, ja kontrollib, kas tabeli välisvõtmed on õigesti tehtud. Vaateid kontrollis õppejõud visuaalselt – kas vaade näeb korrektne välja ja kas andmed on õiged. Kui midagi tundus vigane, kontrollis ta vaate loomise SQL päringut. Funktsioone ja protseduure sai õppejõud kontrollida ainult neid funktsioone ja protseduure kasutades. Trigereid sai õppejõud kontrollida ainult iSQL'is päringute abil.



Joonis 1. Vana hindamisprotsessi mudel.

4.1.1. Vana hindamisviisiga kaasnevad probleemid

Kõiki neid vahendeid tuleb kasutada, et saada tudengi lahendusest täit ülevaadet ja kuna lahendused on mahukad, jääb tihti midagi kahe silma vahele, eriti funktsioonide, protseduuride ja trigerite kontrollimisel. Vana hindamisviisiga oli arukas kontrollida ainult konkreetse kodutöö ülesandeid, sest terve andmebaasi kontroll oleks võtnud liiga palju aega.

Kodutööde hindamiskvaliteet kannatab vanal hindamisviisil hindamisel kõige enam just seepärast, et õppejõud võib eri tudengitel hinnata erinevaid lahenduse elemente või unustada midagi kontrollimata. Kuna õppeainet õpetab tavaliselt 2-10 õppejõudu, pole hindamine alati ühtsel tasemel ja seetõttu võivad tekkida tudengite hinnetes suured erinevused.

Tagasiside andmine võib õppejõu jaoks kujuneda väga töömahukaks protsessiks, kui õppejõud ei saa täpselt vea põhjusest aru ja peab selle täpsustamiseks tegema täiendavaid päringuid või kui vigu on nii palju, et nende kõigi kirjapanek võtaks väga kaua aega.

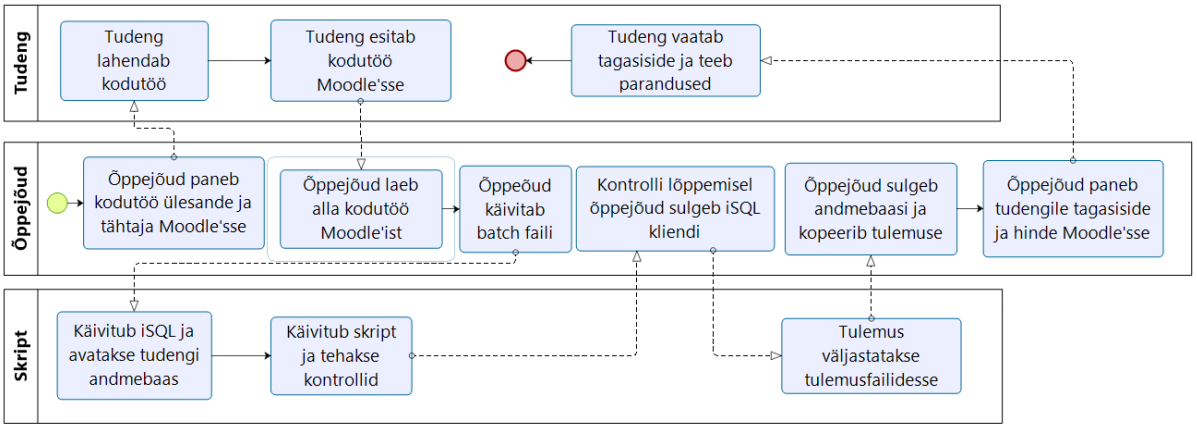
Probleemiks on ka kontrollimise ajaline kestvus, sest tudengeid on väga palju ja ühe kodutöö kontrollimine võtab aega umbes 10 - 40 minutit olenevalt sellest, kui hästi või halvasti kodutöö tehtud on.

4.2. Uus hindamisprotsess

Kõigepealt tuleb ette valmistada keskkond, mis on ühekordne töö. Tuleb luua kaust, kus hakatakse kodutöid kontrollima, siis pakkida lahti kodutöö kontrollimise pakett, mis sisaldab skripti, skripti käivitamiseks vajalikku pakkfaili ja kontrolliks vajalikke tekstifaile. Lõpetuseks loob õppejõud C-kettale kausta nimega *"TEMP"* (kui seda seal veel pole).

Siis on õppejõul kõik valmis tudengite koduste tööde kontrollimise alustamiseks (joonis 2). Selleks laeb ta alla tudengi kodutöö Moodle'ist, pakib selle äsja loodud kaustas lahti ja käivitab pakkfaili. Pakkfail avab tudengi andmebaasi iSQL kliendis ning käivitab skripti. Skript teeb vajalikud kontrollid ning näitab tulemust iSQL keskkonnas vigade tabelina. Kui õppejõud tahab lähemalt uurida, miks esinesid just sellised vead, siis võib ta tudengi andmebaasi vaadata SyBase Centralis ning kodutöö üle kontrollida.

Peale vigade üle vaatamist peab õppejõud iSQL kliendi sulgema ning seejärel kopeerib pakkfail saadud vigade tabeli failidesse „tulemus.txt" ja „tulemus.csv". Need leiab õppejõud samast kaustast, kus on pakkfail. Kui tulemusfailid saadud, tuleb avada neist meelepärane. Avatud failist tuleb terve tabel või tekst kopeerida ning panna Moodle'isse tudengile kodutöö punktid ja tagasiside.



Joonis 2. Uue hindamisprotsessi mudel.

4.2.1. Järgmiste tudengite kodutööde kontrollimine

Kui õppejõud soovib hinnata järgmise tudengi kodutööd, siis peab ta väljuma eelmise tudengi andmebaasist SQL Central'is ning kustutama failid, mis tulid tudengi esitatud zip-failist.

Seejärel võtab õppejõud järgmise tudengi zip-faili ja pakib selle lahti uuesti samas kaustas, kus eelmise ning edasi jätkub kõik samamoodi.

4.2.2. Uue hindamisviisi eelised

Autori loodud automaatkontrolli kõige märgatavam eelis on kodutööde kontrollimisel ajakulu vähenemine. Kui vana hindamisprotsessi puhul oli ühe tudengi kodutöö hindamise ajakuluks 10-40 minutit, siis automaatkontrolliga väheneb see aeg 2-5 minuti peale. Põhiline ajavõit tuleb sellest, et kontroll toimub automaatselt, mõne sekundi jooksul, ja tagasiside antakse hetkega.

Automaatkontroll on väga põhjalik, sest kontrollib kõiki tingimusi ja objekte, kaasa arvatud neid, milleks õppejõul varem aega polnud või mida ei tulnud mõttessegi kontrollida. Igale objektile ja sellega seotud tingimusele on määratud hindepunkt ja pärast kogu töö kontrollimist liidetakse saadud hindepunktid kokku ning saadakse tudengi kodutööle vastav hinne. Seeläbi on hindamine ühtlustunud. Lisaks tekib kontrollimise lõpus selge struktuuriga ja arusaadavate selgitustega vigade tabel, mis annab tudengile selge ja ülevaatliku info iga vea kohta, mille automaatkontroll leidis.

Uue hindamisviisi plussiks on ka see, et kontrollitakse ka neid tingimusi ja objekte, mis olid vajalikud eelmise kodutöö edukaks lahendamiseks. Ehk viiendas kodutöös kontrollitakse, kas tudeng tegi parandused kolmandas, kuuendas kontrollitakse ka kolmas ja viies ning seitsmendas kontrollitakse kolmas, viies ja kuues kodutöö. Sellise kontrollimisviisi eesmärgiks on, et igal tudengil oleks korrektne andmebaas, mille põhjal lahendatakse eksamile saamise ülesanne.

5. Skripti ülevaade

Skript kirjutatakse SQL skriptifailina ja selle käivitamine toimub pakkfaili abil. Pakkfailis on olemas vajalikud tekstiread, et käivitada iSQL, avada tudengi andmebaas ja käivitada skript, mis kontrollib tudengi kodutöö ning väljastab tulemuse teksti- või csv-failis.

5.1. Vajalikud programmid andmebaasi kontrollimiseks

Tudengi kodutöö kontrollimiseks on vaja Windows operatsioonisüsteemi, SQL Central programmi ja Interactive SQL ehk iSQL programmi. Pakkfail on tehtud Windows operatsioonisüsteemi reeglite järgi. Kui pakkfail ümber kirjutada, siis saab loodud kodutöö kontrolli käivitada ka Linux'is või Mac'is.

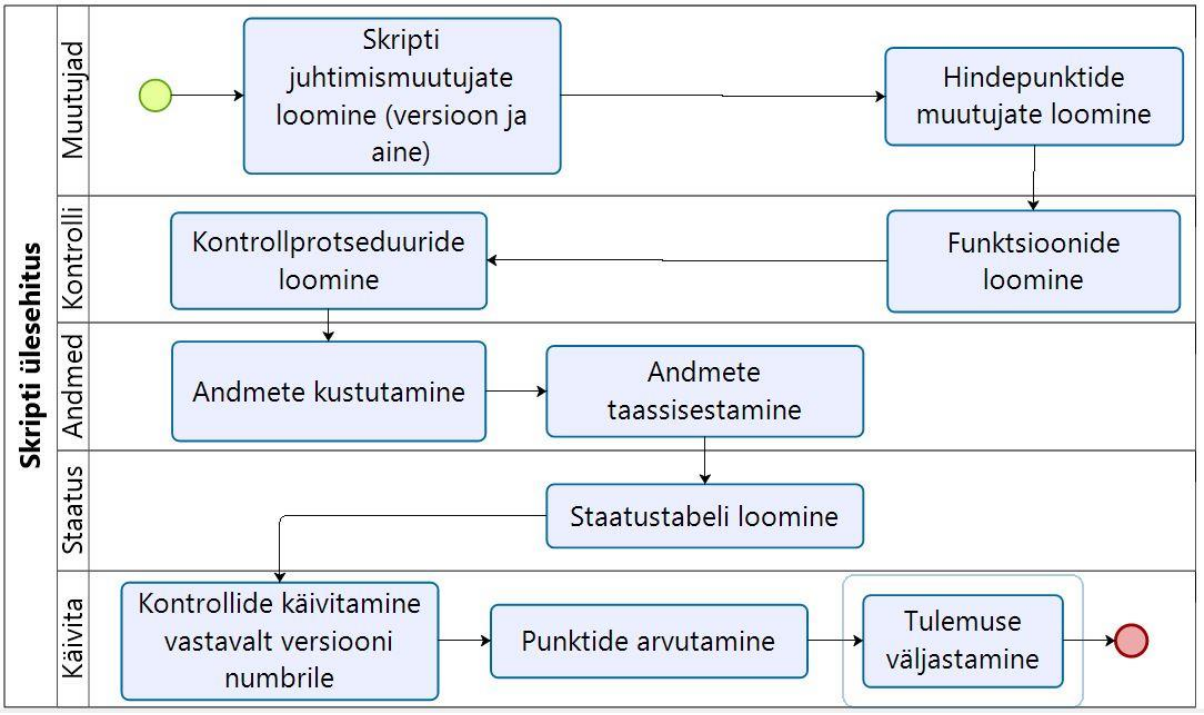
ISQL on kodutöö kontrollimiseks kõige olulisem programm. Seal avatakse tudengi andmebaas ja käivitatakse skript, mis teeb vajalikud kontrollid. SQL Central'it kasutavad õppejõud selleks, et vaadata tudengi andmebaasi juhul, kui peaks tekkima küsimusi, miks kontrolli tulemusel teatud tagasiside tuli.

5.2. Skripti testimine arenduse ajal

Skripti testimine toimus 2019/2020 sügis- ja kevadsemestril „Andmebaasid" ainet läbivate tudengite esitatud tööde põhjal.

6. Skripti ülesehitus

Skriptis on viis olulist osa: muutujate loomine, kontrollprotseduuride loomine, kirjete kustutamine ja uuesti sisestamine tudengi andmebaasis, staatustabeli loomine ning vajalike kontrollide käivitamine (joonis 3).



Joonis 3. Skripti ülesehituse mudel.

6.1. Muutujate loomine

Skripti alguses luuakse kaks muutujat - versioon ja aine. Muutuja „versioon“ määrab ära, millist kodutööd hakatakse kontrollima. See on vajalik kohe alguses määrata, sest ülesanded muutuvad kodutööde lõikes ja punktide arvud muutuvad olenevalt kodutöö numbrist. „Versiooniks“ on kodutöö number, milleks võib olla 3, 5, 6 või 7.

Muutuja „aine“ määrab ära, missuguse aine kodutöö kontroll käivitatakse. Kui muutuja väärtuseks on „A“, on tegu ainega „Andmebaasid“ ja kui väärtuseks on „S“, on tegu ainega „Sissejuhatus andmebaasidesse“. Seda on vajalik eristada, sest mõnda ülesannet kasutatakse mõlemas aines, mõnda aga mitte.

Teisena luuakse hindepunktide muutujad. Selles osas toimub muutujate loomine, kus iga muutuja vastab mingile objektile, näiteks tabeli veerule, või tingimusele, mida kontrollitakse.

Hindepunktide muutujad on ära jagatud kodutööde kaupa, mis omakorda on ära jagatud objektide kaupa.

6.2. Kontrollprotseduuride loomine

Kodutööde lahendustena tekivad objektid. Sõltuvalt sellest, mida iga kodutöö lahendamine õpetab, tuleb kontrollida objektide ja nende tingimuste olemasolu ja kvaliteeti, näiteks nime ja tüübi õigsust. Sellepärast oli loogiline luua hulk protseduure ja funktsioone, mis kontrollivad kõiki neid objekte ja tingimusi ning nende kvaliteeti. Nende kontrollimine käib süsteemsete tabelite kaudu.

Kontrollprotseduurid jagunevad kaheks: objektide kontrollid ja tingimuste kontrollid. Objektideks on tabelid, vaated, funktsioonid, protseduurid ja trigerid. Tingimusteks on veerud, veergude kitsendused, välisvõtmed ja indeksid.

6.3. Kirjete kustutamine ja uuesti sisestamine

Skripti selles osas toimub tudengi andmebaasis kirjete kustutamine ja uuesti sisestamine. Kirjete kustutamine ja taas sisestamine on vajalik, et tekitada kõigi tudengite kodutöödes võrdne seis, et skript kontrolliks vaid andmebaasi objektide ja nende omaduste vastavust ülesannetest tulenevatele funktsionaalsetele nõuetele. Seda tehakse etalonandmebaasi kaudu, kasutades andmete võrdlemist mahu ja kui vaja, ka järjestuse mõttes. See tagab, et iga tudengi kodutöö hindamine on objektiivne.

See ei kehti aga kolmanda kodutöö kontrollimisel, sest selle kodutöö kontrolli osaks on vaadata, kas tudengid sisestasid tabelitesse andmed õigesti. Viiendas ja kuuendas kodutöös kustutatakse kirjed ära põhitabelites, milleks on: „Isikud“, „Klubid“, „Turniirid“ ja „Partiid“. Seitsmendas kodutöös kustutatakse kirjed lisaks põhitabelitele ka tabelist „Asulad“.

Kirjete kustutamisel ja sisestamisel esineb tihti probleeme, sest tudengitel on tabelite objektidel midagi valesti ja seitsmenda kodutöö puhul võivad loodud trigerid olla vigased. Seepärast on lisatud automaatkontrolli vea tagasiside, kus antakse teada ebaõnnestunud kirjete kustutamisest või lisamisest ja ebaõnnestumise võimalik põhjus. Veateade on õppejõule selleks, et ta saaks kontrollida, milles on probleem, ja anda tudengile põhjalikuma tagasiside.

6.4. Staatustabeli loomine

Kodutöö lahenduse kontrolli käigus koondatakse kõik tulemused staatustabelisse. Seda isegi siis, kui kodutöös vigu ei leidu, sest staatustabeli abil arvutatakse tudengile kodutöö eest punktid (tabel 1).

Tabel 1. Staatustabeli selgitus.

Veeru pealkiri	**Veeru selgitus**
Nimi	Mis objekti kontrolliti?
Veerg	Mis tingimust kontrolliti?
Tagasiside	Veerg, kus on kirjas, mis valesti on või kui kõik on korras, siis lihtsalt märgitud side kriipsuga '-'
Olek	Tagasiside tüüp („OK", „VIGA", „Hindepunktid" või „AEG").
Punktid	Veerg, kuhu on märgitud, kui palju punkte sai tudeng selle kontrolli osa eest.
Max_punktid	Veerg, kuhu on märgitud maksimaalsed punktid, mis selle kontrolli osa eest saab.
Soovitus	Veerg, kuhu on kirjutatud soovitusi ja küsimusi tudengile, et nad saaksid aru, miks see vale on või kas nad said ülesandest õigesti aru.

6.5. Lahenduse kontrollimine

Konkreetse kodutöö lahenduse kontroll toimub protseduuris „käivita". See protseduur kutsutakse välja skripti viimasel real ning sellesse sisestatakse muutuja „versioon" väärtus ehk kodutöö number. Kodutöö number määrab ära, millised kontrollprotseduurid käivitatakse, millised hindepunktide muutujad kasutusele võetakse ning mis on maksimumpunktid terve kodutöö eest. Kui toimub kolmanda kodutöö kontrollimine, siis käivitatakse ainult kolmanda kodutöö kontrollprotseduurid. Kui viienda, siis käivitatakse nii kolmanda kui ka viienda kodutöö kontrollprotseduurid jne. Selline dünaamiline ülesehitus on kasulik selleks, et kui tulevikus kodused ülesanded muutuvad, siis lahenduse kontrollimise ja hindamise muutmine on lihtsam.

6.6. Tulemuse väljastamine

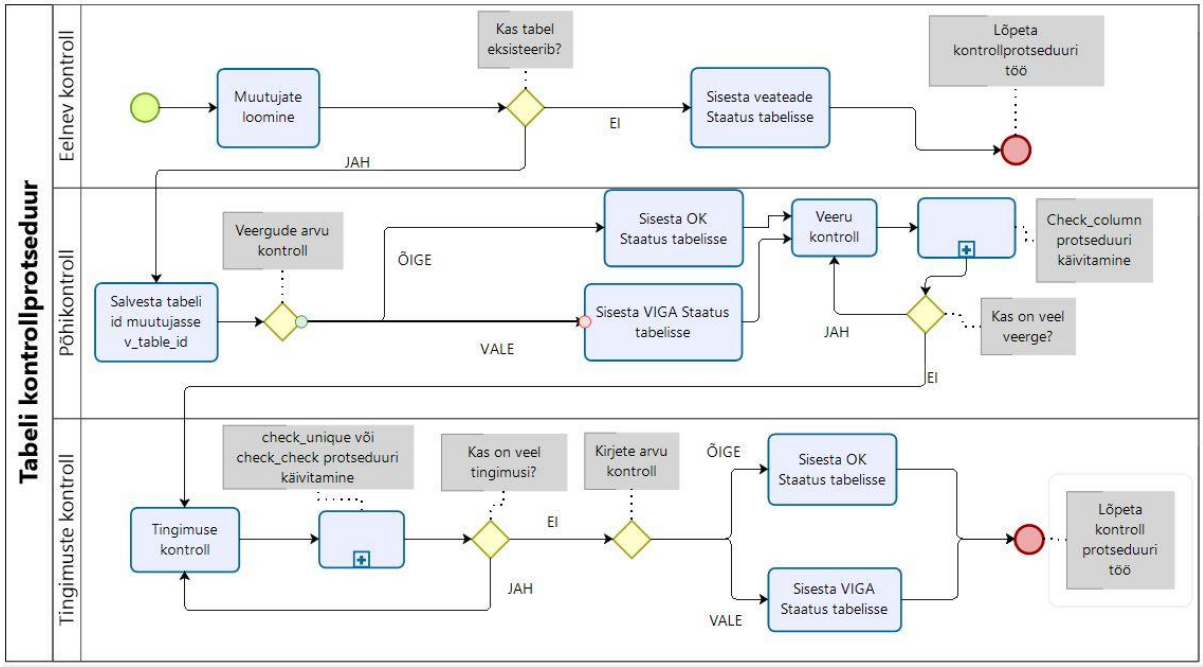
Peale vajalike kontrollprotseduuride tegemist protseduuris „käivita“ toimub punktide arvutamine. Selleks kasutatakse protseduuri „arvuta_punktid“, mis võtab muutujaks kodutöö numbri, mis määrab kodutöö maksimumpunktid. „Arvuta_punktid“ liidab kokku punktide summa staatustabeli veerust „punktid“ ning sisestab saadud summa staatustabelis punktide veeru lõppu. Seejärel toimub tulemuse väljastamine failidesse ning siis ka ekraanile.

7. Tähtsamate kontrollprotseduuride ülesehitused

Kontrollprotseduure on üheksa: tabel, veeru ja tabeli kitsendus, välisvõti, vaade, funktsioon, protseduur, triger, indeks, süsteemne triger. Kui kontrolli käigus vigu ei ilmne, sisestatakse staatustabelisse kirje „OK“ ja tudeng saab konkreetse osa eest maksimumpunktid. Vea korral sisestatakse staatustabelisse kirje „VIGA“, tagasiside vea asukoha ja kirjeldusega ning tudeng saab konkreetse osa eest null punkti.

7.1. Tabeli kontrollprotseduuri ülesehitus

Tabelite kontrollprotseduur on selleks, et kodutöödes kontrollida tabelite olemasolu, veergusid ja nende tingimusi ning kirjete arvu (joonis 4).



Joonis 4. Tabeli kontrollprotseduuri mudel

Tabelite kontrollprotseduurid kasutavad muutujat „versioon“ ehk kodutöö numbrit. Mõnel tabelil on eri kodutöödes erinev arv veerge ja seetõttu on oluline määrata, missugust kodutööd kontrollitakse. Tabeli kontroll koosneb kuuest osast: muutujate loomine, tabeli olemasolu kontroll, veergude arvu kontroll, veergude kontroll, veerukitsenduste kontroll ja kirjete arvu kontroll.

Kõigepealt luuakse kolm muutujat: v_table_id, v_size, kirje_count, mis on kõik täisarvulist tüüpi. Seejärel kontrollitakse, kas tabel eksisteerib süsteemses tabelis (*systable)*. Kui tabelit ei

eksisteeri, siis sisestatakse staatustabelisse veateade ning kontrollprotseduuri töö lõpetatakse. Kui tabel eksisteerib, siis salvestatakse tabeli „id“ v_table_id muutujasse.

Järgmisena kontrollitakse veergude arvu tabelis. Kui veergude arv on vale, sisestatakse staatustabelisse veateade. Seejärel toimub veergude kontroll. See toimub *check_column* protseduuri abil, mille sisestusandmeteks on v_table_id, veeru nimi, veeru *default* tingimus, veeru *primary key* tingimus, veeru *null* tingimus, veeru suurus, punkti muutuja nimi.

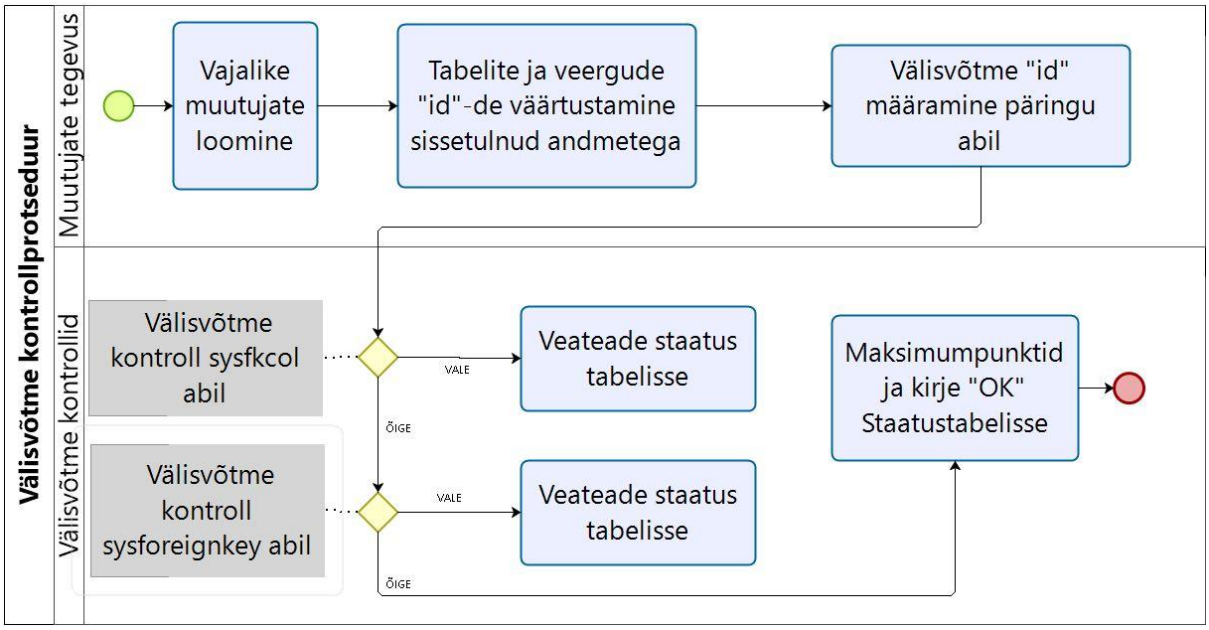
Seejärel tuleb erinevate kitsenduste (*unique* ja *check)* kontroll. *Unique* kitsenduse parameetrid on tabeli nimi, veeru nimi ja hindepunktide muutuja. *Check* kitsenduse parameetrid on kitsenduse nimi, tabeli nimi, veeru nimi ja hindepunktide muutuja. Viimasena toimub kirjete arvu kontroll.

7.2. Veeru ja tabeli kitsenduse kontrollprotseduuri ülesehitus

Veerukitsendus sisaldab tingimuses vaid ühe veeru nime, tabelikitsenduses on veerge rohkem. Kontrollitavad kitsendused on tüübilt *unique* või *check*. Automaatkontrollis kontrollitakse vaid kitsenduste olemasolu. *Unique* kitsenduse kontrollimine toimub süsteemse tabeli (*sysindex*) abil. Sellest tabelist tehakse päring, millega loetakse, kas vastava kitsenduse nimele leidub kitsendus. Kui ei leidu, kirjutatakse staatustabelisse veateade. *Check* kitsenduse kontrollimine toimub süsteemse tabeli (*syscheck*) abil. *Check* kontroll toimub täpselt samamoodi nagu *unique* kontroll, ainult süsteemne tabel on teine.

7.3. Välisvõtme kontrollprotseduuri ülesehitus

Välisvõtme kontrollprotseduur on selleks, et kontrollida kodutöödes välisvõtmete olemasolu ja seda, kas välisvõti on tehtud õigete tabeli veergude vahel (joonis 5). Kontrollprotseduuri sissetulevateks andmeteks on peamise tabeli nimi, välistabeli nimi, peamise tabeli veeru nimi, välistabeli veeru nimi ja punktid, mis selle välisvõtme õige loomise eest saab. Välisvõtme kontroll koosneb kolmest osast: muutujate loomine, muutujate väärtuse määramine ja välisvõtme olemasolu kontrollid kahest erinevast süsteemsest tabelist.



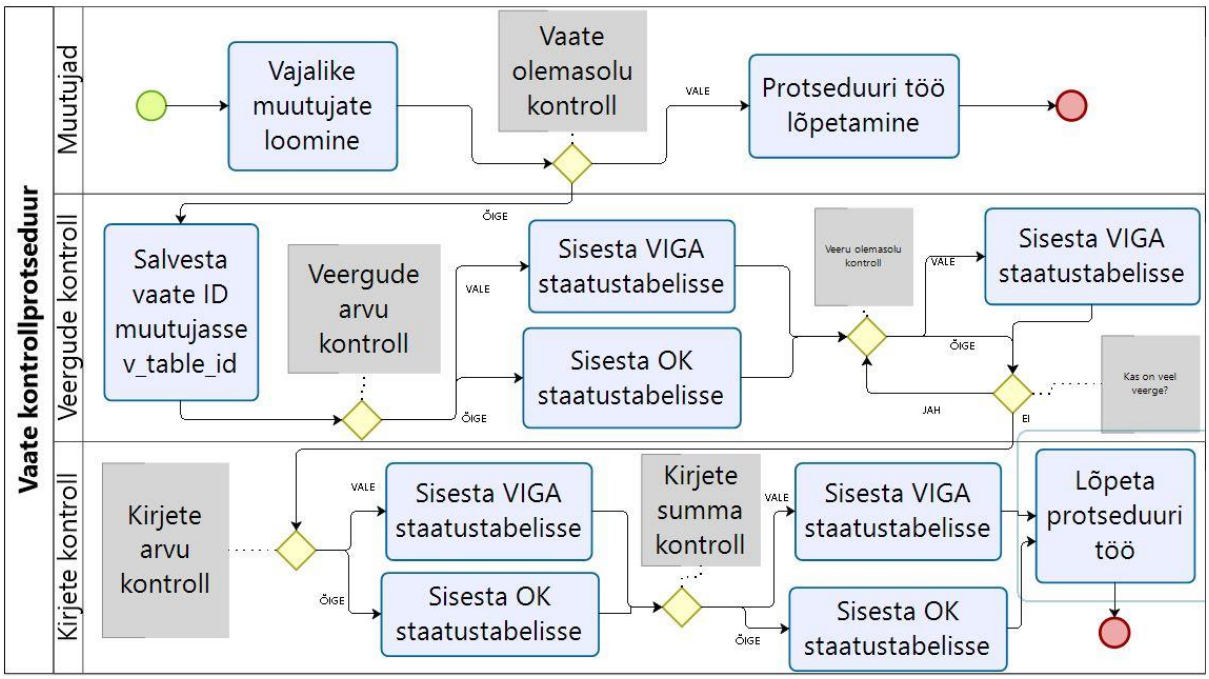
Joonis 5. Välisvõtme kontrollprotseduuri mudel

Kõigepealt luuakse kontrolliks vajalikud muutujad. Järgmisena määratakse tabelite ja veergude „id“ muutujatele väärtused. Need saadakse sissetuleva info, milleks on tabelite ja veergude nimed, abil süsteemsetest tabelitest (*systable* ja *syscolumn)*.

Seejärel kontrollitakse, kas vastavatele andmetele leidub üks kindel välisvõti ja salvestatakse selle „id“ väärtus muutujasse. Siis järgneb kahes erinevas süsteemses tabelis (*sysfkcol* ja *sysforeignkey*) välisvõtme koguse kontroll. Kui mõlemas tabelis on sellise välisvõtme „id“-ga välisvõtmeid vaid üks, on välisvõti korras. Kui selliseid välisvõtmeid pole või on neid rohkem kui üks, lisatakse staatustabelisse veateade.

7.4. Vaate kontrollprotseduuri ülesehitus

Vaate kontrollprotseduur on selleks, et kontrollida kodutöös vaate olemasolu, veergude olemasolu ja kogust ning vaate kirjetega seotud tingimusi. Vaate kontrollprotseduur koosneb viiest osast: muutujate loomine, vaate olemasolu kontroll, vaate veergude arvu kontroll, vaate veergude kontroll ja üks või kaks vaate kirjetega seotud kontrolli (joonis 6).



Joonis 6. Vaate kontrollprotseduuri mudel

Kõigepealt luuakse muutujad v_table_id – vaate „id“, v_size – veergude kogus, kirje_count – kirjete kogus, punkti_summa – kirjete summa. Järgmisena kontrollitakse vaate olemasolu. Kui vaadet ei eksisteeri, siis kirjutatakse staatustabelisse veateade ja kontrollprotseduuri töö lõpetatakse. Kui kõik on korras, siis salvestatakse muutujasse „v_table_id“ vaate „id“.

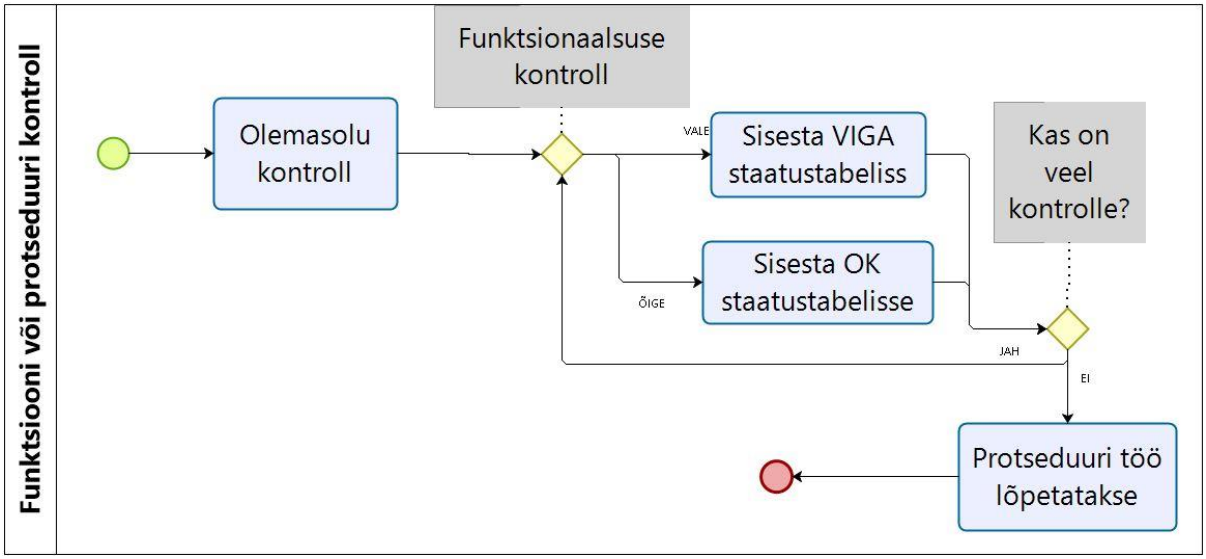
Seejärel on veergude koguse kontroll. Kui veergude arv on vale, kirjutatakse staatustabelisse veateade. Seejärel kontrollitakse süsteemses tabelis (*syscolumn*), kas vaatel on konkreetsed veerud olemas.

Viimasena on kirjetega seotud kontrollid: kirjete kontroll ja summa kontroll. Kirjete kontrolli puhul loetakse kokku, mitu kirjet on vaates ning vastavalt tulemuse õigsusele kirjutatakse staatustabelisse selle kohta kommentaar. Summa kontrolli on vaja selleks, et mõnes ülesandes on olulisel kohal kirjete väärtused, näiteks punktide või klubide arv. Seega kontrollitakse, kas vaates on õige summa või mitte, ning tulemusele vastav kommentaar kirjutatakse staatustabelisse.

7.5. Funktsiooni ja protseduuri kontrollprotseduuri ülesehitus

Funktsiooni ja protseduuri kontrollprotseduur on selleks, et kontrollida kodutöödes olevate funktsioonide ja protseduuride olemasolu ja funktsionaalsust. Nende kontrollimine toimub

kahes osas: olemasolu kontroll ja funktsionaalsuse kontroll (joonis 7). Funktsionaalsuskontrolle on alati vähemalt üks.

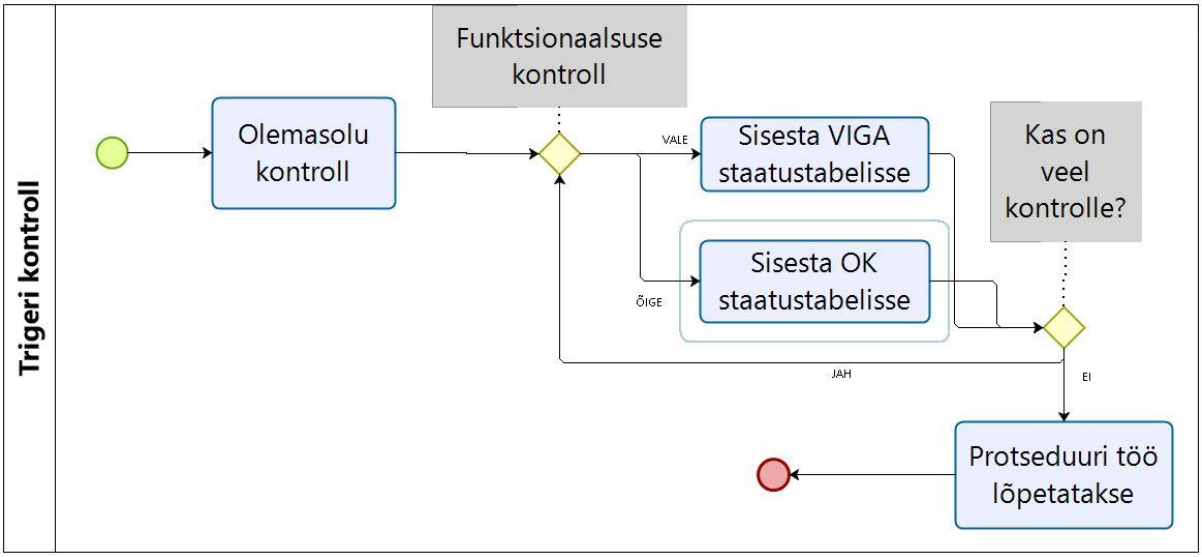


Joonis 7. Funktsiooni ja protseduuri kontrollprotseduuri mudel

Olemasolu kontrollis kontrollitakse süsteemse tabeli (*sysprocedure*) abil, kas õige nimega funktsioon või protseduur on olemas. Funktsiooni või protseduuri funksionaalsust kontrollitakse seda funktsiooni või protseduuri mingite parameetritega kasutades. Kontrolli tulemust kajastav kommentaar kirjutatakse staatustabelisse.

7.6. Trigeri kontrollprotseduuri ülesehitus

Kuna triger on protseduur, toimub kontroll sarnaselt eelmises alapeatükis kirjeldatuga. Trigeri kontrollprotseduur on selleks, et kontrollida kodutöödes olevate trigerite olemasolu ja funktsionaalsust. Kontrolli ülesehitus on täpselt sama, mis funktsioonil ja protseduuril, ainult olemasolu kontrollimine toimub süsteemse tabeli (*systriger*) abil (joonis 8).



Joonis 8. Trigeri kontrollprotseduuri mudel

Trigereid on kolm: „lisa_klubi“, „kustuta_klubi“ ja „kustuta_klubi_isikutega“ ning nende funktsionaalsuse kontroll toimub mitmes osas. Kodutöös vajalikud trigerid on seotud tabelis uute andmete sisestamise või kustutamisega.

Triger „lisa_klubi“ on selleks, kui tabelisse „Klubid“ lisatakse uus klubi ja selle asukoht. Triger kontrollib, kas tabelis „Asulad“ on see asukoht juba olemas ja kui ei ole, lisatakse see sinna. Trigeri funktsionaalsust kontrollitakse uute klubide lisamisega ja neid kontrolle on kaks. Kõigepealt lisatakse uus klubi ja selle klubi asukoht. Seejärel kontrollitakse, kas see asukoht lisati ka tabelisse „Asulad“. Peale seda kontrolli kustutatakse tabelist „Klubid“ lisatud klubi ja tabelist „Asulad“ asukoht. Järgmise kontrolliga lisatakse kaks klubi, mis asuvad samas kohas. Kontrollitakse, kas lisatud klubidel on olemas viide asukoha „id“-le. Peale kontrolli kustutatakse taas lisatud klubid ja asukoht. Vea korral sisestatakse staatustabelisse veateade.

Triger „kustuta_klubi“ on selleks, kui mingit klubi hakatakse kustutama. Kui sellel klubil on asukoht ja see on selles asukohas ainus klubi, siis peab triger kustutama lisaks klubile ka asukoha. Kui see pole selles asukohas ainus klubi, siis ei tohi triger seda asukohta kustutada, küll aga kustutab klubi. Sellel trigeril on kaks kontrolli. Kõigepealt lisatakse tabelisse „Klubid“ kaks uut klubi ja tabelisse „Asulad“ nende asukoht, mis peab olema kummalgi klubil sama. Seejärel kustutatakse üks lisatud klubidest ja kontrollitakse, kas tabelisse „Asulad“ jäi selle asukoht alles. Seejärel kustutatakse teine lisatud klubi ja kontrollitakse, kas asukoht tabelist „Asulad“ kustus, sest rohkem klubisid selles asukohas pole. Juhul kui triger ei tööta korrektselt ning klubisid ja asukohta ei kustutatud kontrolli käigus, tehakse seda kontrolli lõpus.

Triger „kustuta_klubi_isikutega“ on selleks, kui tahetakse kustutada klubi, kus on mängupartiideta mängijaid. Sellisel juhul kustutatakse lisaks klubile ka mängijad. Kui selles klubis olevatel mängijatel on partiisid, ei tohiks triger seda klubi kustutada lasta. Sellel trigeril on kaks kontrolli. Kõigepealt lisatakse uus klubi ja sellesse klubisse uus isik. Seejärel klubi kustutatakse ja kontrollitakse, kas triger lasi seda kustutada. Peale seda kustutatakse lisatud isik ja klubi igaks juhuks uuesti, sest võis juhtuda, et triger ei töötanud korrektselt. Teises kontrollis üritatakse kustutada klubi, kus on mängijad, kellel on mängupartiisid. Triger ei tohiks lasta seda klubi kustutada.

7.7. Indeksite ja süsteemsete trigerite kontrollprotseduurid

Indeks luuakse tabeli veeru peale ja neid on kaht tüüpi: klaster- ja komposiitindeks. Indeks, kus on rohkem kui üks tabeli veerg, nimetatakse komposiitindeksiks, ja see on kasulik siis, kui see luuakse veergudel, millel on omavaheline seos, näiteks eesnimi ja perenimi. Klasterindeks~~is~~ on indeks, kus kirjed sorteeritakse tabelis indeksi poolt määratud järjekorda ehk kirjeid saab järjest lugeda ja see luuakse ühe tabeli veeru peale. (SAP Company, 2016)

Indeksite kontroll on ainult indeksite kokku lugemine. Tabelis „Partiid“ peab olema kaks indeksit ja tabelis „Turniirid“ peab olema üks indeks ning need loetakse kokku tabelist (*sysindex*).

Süsteemsed trigerid on protseduurid, mis tekivad koos välisvõtmetega. Nende kontroll toimub sarnaselt veeru kitsendustega ehk kontrolliks tehakse päring süsteemsesse tabelisse (*systrigger*) ja loetakse kokku, kas andmebaasis on kokku piisavalt süsteemseid trigereid.

8. Edasiarendus

Käesolev lõputöö katab automaattestidega ainult poole praktikumide materjalist. Lisaks ope andmebaasi käsitlevatele praktikumidele toimub aines „Andmebaasid“ üks praktikum, kus käsitletakse edu andmebaasi. Ka selles antava kodutöö kontrollimist saaks automatiseerida, kuid käesoleva töö raames oleks see muutunud liiga mahukaks. Ülejäänud praktikumides ei ole kodutöid, mida oleks võimalik automaattestidega kontrollida.

8.1. Moodle

Moodle on õppekeskkond, kuhu tudeng esitab kodutöid. Kui kodutööde kontrollimine üle viia Moodle’isse, kaob kontrollimisel suures osas ära õppejõu osa. Moodle’is toimub kontrolltestide käivitamine virtuaalmasinal, kus automaattestid käivitatakse samasuguses keskkonnas nagu käesolevas lõputöös. Välja näeb see nii, et tudeng laeb kodutöö Moodle’isse ja vajutab „käivita“ nuppu. Virtuaalmasinas käivitatakse andmebaas ja avatakse iSQL’i klient, seejärel käivitatakse skript ning tehakse kontrollid. Tulemus väljastatakse tudengile ekraanile. Kui kontrolli käigus tekib viga, teavitatakse sellest tudengit ekraanil.

Kokkuvõte

Käesoleva bakalaureusetöö eesmärk oli luua automaattestid ainele „Andmebaasid“, et kiirendada ja ühtlustada kodutöö ülesannete kontrollimist ja hindamist. Varem võttis ühe tudengi kodutöö hindamine olenevalt vigade rohkusest aega 10-40 minutit, loodud skripti abil saab ühe tudengi töö hinnatud aga 2-5 minutiga. Loodud skript kontrollib iga kodutöö puhul ka eelmises kodutöös läbi viidud parandusi, mida varem ajapuuduse tõttu teha ei jõutud.

Samuti on kodutööde kontrollimine muutunud loodud skripti tõttu tunduvalt objektiivsemaks, sest kõigi tudengite kodutöid hinnatakse samadel alustel, tudengitel võetakse samade vigade eest sama arv punkte maha ja antakse ühesugune tagasiside. Seega sai bakalaureusetöö eesmärk täidetud.

Positiivsena saab välja tuua ka seda, et skripti sai arendamise ajal testida ning ainet õpetavad õppejõud on testimise käigus harjunud skripti kasutama, mis loob head eeldused selleks, et skripti kasutatakse ka edaspidi.

Kasutatud kirjandus

Cybernetica. (2020). *Andmekaitse ja infoturbe leksikon.* Kasutatud 30.03.2020, https://akit.cyber.ee/term/5590-funktsioon

IT Terministandardi projekti (1998-2001) sõnastik. (2001). Kasutatud 30.03.2020, https://www.keeleveeb.ee/dict/speciality/itstandard/dict.cgi?word=kitsendus&lang=et

Guru99. (i.a). *What is Database? What is SQL?* Kasutatud 30.04.2020, https://www.guru99.com/introduction-to-database-sql.html

Guru99. (i.a). *Oracle PL/SQL Stored Procedures & Functions with Examples.* Kasutatud 30.03.2020, https://www.guru99.com/subprograms-procedures-functions-pl-sql.html

Hanson, V., Tavast, A. (2005). *Arvutikasutaja sõnastik.* Kasutatud 30.04.2020, https://www.keeleveeb.ee/dict/speciality/aks/dict.cgi?word=batch&lang=en

Laurie, V. (i.a). *Batch Files (Scripts) in Windows.* Kasutatud 30.04.2020. https://commandwindows.com/batch.htm

Liikane, L., Kesa, M. (2006). *Arvutisõnastik.* Kasutatud 30.04.2020, https://www.keeleveeb.ee/dict/speciality/computer/dict.cgi?word=SQL&lang=en

Liikane, L., Kesa, M. (2006). *Arvutisõnastik.* Kasutatud 30.04.2020, https://www.keeleveeb.ee/dict/speciality/computer/dict.cgi?word=tabel&lang=et

Liikane, L., Kesa, M. (2006). *Arvutisõnastik.* Kasutatud 30.04.2020, https://www.keeleveeb.ee/dict/speciality/computer/dict.cgi?word=veerg&lang=et

Paakspuu, K. (2014). *Automaattestimine – mis see on?* Kasutatud 5.02.2020, http://asaquality.blogspot.com/2014/05/automaattestimine-mis-see-on.html

Sybase. (2005). *Mainframe Connect DirectConnect for z/OS Option 12.6.* Kasutatud 30.04.2020, http://infocenter.sybase.com/help/topic/com.sybase.help.mainframeconnect_12.6.mvsinst/mvsinst.pdf

SAP Company, (2016). *DocCommentXchange.* Kasutatud 30.04.2020, http://dcx.sap.com/index.html#sqla170/en/html/81827ecb6ce21014a3b3e073c3af66f4.html

Sepp, R. (2018). *Automaattestide loomine andmebaasile Edu.* Bakalaureusetöö. Tartu Ülikool ,Arvutiteaduse instituut.

Siiber, M. (2015). *Aine „Andmebaasid“ automaattestimise vahend*. Bakalaureusetöö. Tartu Ülikool, Arvutiteaduse instituut.

W3resource, (2020). *MySQL Triggers.* Kasutatud 30.04.2020, https://www.w3resource.com/mysql/mysql-triggers.php

W3schools, (i.a). *SQL Views.* Kasutatud 30.03.2020, https://www.w3schools.com/sql/sql_view.asp

W3schools, (i.a). *SQL Stored Procedures for SQL Server.* Kasutatud 30.04.2020, https://www.w3schools.com/sql/sql_stored_procedures.asp

W3schools, (i.a). *SQL Foreign Key Constraint.* Kasutatud 30.04.2020, https://www.w3schools.com/sql/sql_foreignkey.asp

Õunamaa, M. (2019). *Automaattestide loomine sessioonõppe ainele „Sissejuhatus andmebaasidesse.* Bakalaureusetöö. Tartu Ülikool, Arvutiteaduse instituut.

Lisad

Lisa 1. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Martti Kakk,

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

   „Automaattestid andmebaaside ainele“,

   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.

Martti Kakk

08.05.2020