TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Informaatikainstituut
Infosüsteemide õppetool

SQL lausete veateadete headuse analüüs kolme erineva andmebaasisüsteemi näitel

Bakalaureusetöö

Üliõpilane: Klaus-Kristjan Põlluste
Üliõpilaskood: 112177IAPB
Juhendaja: dotsent Erki Eessaar

Tallinn
2015

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

---------- ----------

(*kuupäev*) (*allkiri*)

Annotatsioon

SQL lausete veateadete headuse analüüs kolme erineva andmebaasisüsteemi näitel

Töö eesmärk on uurida hulka populaarseid SQL-andmebaasisüsteeme ja teha kindlaks, kui informatiivseid veateateid need esitavad erinevate SQL lausetes tehtud vigade puhul või lausete tulemusena tekkinud erandolukordade puhul. Töös võrreldakse veateateid Oracle (Database 12c Enterprise Edition Release 12.1.0.1.0), PostgreSQL (9.3.0) ja Microsoft Access (2013) andmebaasisüsteemides.

Töös luuakse Oracle, PostgreSQL ja MS Access andmebaasisüsteemide abil sama struktuuriga andmebaasid, kuhu lisatakse identsed testandmed. Seejärel võrreldakse nendes andmebaasisüsteemides SQL lausete täitmisel tekkinud veateateid. Veateadete headust hindab töö autor üksinda. Töö tulemusena leitakse uuritud andmebaasisüsteemide pingerida veateadete headuse seisukohalt ning pakutakse kõige väheinformatiivsematele veateadetele sisukamad alternatiivid. Samuti üritatakse veateadete probleeme üle erinevate andmebaasisüsteemide üldistada.

Töö olulisem tulemus oleks välja tuua järjestus võrreldavatest andmebaasisüsteemidest lähtuvalt nende veateadete kuvamise informatiivsusest.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 66 leheküljel, 4 peatükki, 5 joonist, 7 tabelit.

Abstract

Analysis of the Goodness of SQL Error Messages in the Example of Three Different Database Management Systems

The aim of this work is to compare a set of popular SQL Database Management Systems and find out informativeness of error messages they produce to malformed SQL statements or exceptions caused by SQL statements. Oracle (Database 12c Enterprise Edition Release 12.1.0.1.0), PostgreSQL (9.3.0), and Microsoft Access (2013) database management systems (DBMSs) are used in this comparison.

In this work, Oracle, PostgreSQL, and MS Access DBMSs are used to implement logically identical databases with identical test data. After that, we compare error messages of executed SQL statements. The author does it alone. As a result of this work, a ranking of the examined DBMSs will be constructed based on the quality of error messages and preferable alternatives will be provided for the most uninformative error messages. Likewise, a generalization of problems occurred in the error messages is attempted.

The most important outcome of this work is a ranking of the DBMSs based on informativeness of error messages they display.

The thesis is in Estonian and contains 66 pages of text, 4 chapters, 5 figures, 7 tables.

Jooniste nimekiri

Joonis 1. Disaini täpsusega andmemudel 13
Joonis 2. Näide Oracle andmebaasisüsteemi veateatest (Oracle Application Express) 25
Joonis 3. Näide PostgreSQL andmebaasisüsteemi veateatest (pgAdmin III) 25
Joonis 4. Näide veateatest MS Access andmebaasisüsteemis 26
Joonis 5. Andmebaasi programmeerimiskeskkond 0xDBE 52

Tabelite nimekiri

Tabel 1. Populaarseimad andmebaasisüsteemid (märts 2015) 12
Tabel 2. Atribuutide detailsed kirjeldused 14
Tabel 3. Tabeli *Inventar* veergude tüübid 16
Tabel 4. Tabeli *Tuba* veergude tüübid 16
Tabel 5. Tabeli *Inventar_toas* veergude tüübid 16
Tabel 6. Veateadete hindamisskaala 24
Tabel 7. Uuringu koondtulemus 45

Sisukord

Sissejuhatus 9
1. Eksperimendi kirjeldus 12
1.1 Eksperimendi andmebaasi projekteerimine 12
1.2 Disaini täpsusega andmemudel 13
1.3 Atribuutide detailsed kirjeldused 14
1.4 Baastabelite veergude tüübid 15
1.5 Testülesannete kirjeldused 17
1.5.1 Süntaktiliselt ebakorrektsed laused 17
1.5.2 Süntaktiliselt korrektsed laused, mis aga ei vasta andmebaasi struktuurile 17
1.5.3 Vead andmekirjelduskeele lausetes 18
1.5.4 Tegevus, mis pole kooskõlas kasutaja õigustega 18
1.5.5 Andmemuudatused, mis tekitatavad vastuolu kitsendustega 19
1.6 Andmebaasi tabelite realisatsioon 19
1.6.1 Oracle 19
1.6.2 PostgreSQL 20
1.6.3 MS Access 22
1.7 Testandmed 23
1.8 Testpäringute realisatsioon 23
1.9 Veateadete hindamiskriteeriumid 23
2. Eksperimendi tulemused ja järeldused 25
2.1 Veateadete näited ekraanitõmmistena 25
2.2 Eksperimendi tulemused 26
2.2.1 Süntaktiliselt ebakorrektsed laused 26
2.2.2 Süntaktiliselt korrektsed laused, mis aga ei vasta andmebaasi struktuurile 34
2.2.3 Vead andmekirjelduskeeles 39
2.2.4 Tegevus, mis pole kooskõlas kasutaja õigustega 41
2.2.5 Andmemuudatused, mis tekitavad vastuolu kitsendustega 41
2.2.6 Tulemuste kokkuvõte 45
2.3 Veateadete üldistus andmebaasisüsteemides ning tarkvaras üldiselt 51
2.4 Kõige väheinformatiivsemate veateadete alternatiivid 53
3. Kokkuvõte 56
4. Summary 57

Kasutatud kirjandus 58
Lisa 1 60

Sissejuhatus

SQL (Structured Query Language) andmebaasikeel on üldiselt deklaratiivne keel, mille abil saab muuhulgas andmebaasis teha päringuid või andmemuudatusi. Enne SQL lause täitmist andmebaasisüsteemi poolt see sõelutakse (parsitakse), mille käigus avastatakse süntaktilised vead. Pärast sõelumist kontrollitakse, kas lause on semantiliselt õige. See tähendab näiteks, et kontrollitakse, et lauses ei oleks viiteid andmebaasiobjektidele, mida pole olemas ning et lause käivitajal on lause käivitamise õigus. Kui SQL lause ei läbi neid kontrolle, siis tekib SQL lauses viga ja selle tagajärjel kuvatakse andmebaasisüsteemi poolt veateade.

Üldiselt on väga hea praktika süsteemi disainimine nii, et kasutajal ei ole võimalik tekitada veaolukorda (erandit), millele peaks veateatega vastama. Andmebaasisüsteemides on see variant välistatud, sest paratamatult tekivad SQL lausete koostamisel vead, millele peab lause täitmisel veateatega reageerima. Veateade on vastus kasutajale, kui süsteemis on tekkinud viga e erandolukord e erand. Erandolek, erandolukord, e erand on „programmi täitmisel normaalses käsuvoos tekkinud viga, mis on tingitud näit. ületäitumisest, välisseadme tõrkest vms.“ [10] Sellised vead tekivad näiteks kasutaja erakorraliselt käitumisest, millele programm ei oska adekvaatselt reageerida. Kui kasutajale veast teada ei anta, ei saa kasutaja aru, mis põhjusel programmi mingi ülesanne täitmata jäeti. Sellisel juhul ei oska kasutaja järgmisel korral oma tegevust parandada ja jääb vigu üha uuesti kordama. Lisaks võib juhtuda, et ilma veateadet tagasi saamata ei saa kasutaja üldse aru, et midagi läks valesti ning elab edasi ebaõige teadmisega, et tema tegevused õnnestusid. Üldjuhul kuvatakse teated ainult vigade puhul, kuid kui on tegemist juhusega, kus kasutaja peaks tingimata saama tagasisidet oma sooritatud käskude kohta (nt maksekorralduse kinnituse kuvamine internetipangas koheselt pärast makse kinnitamist), siis tuleks ka kuvada teade operatsiooni õnnestumisest.

Vigade kuvamise praktika andmebaasisüsteemides on sarnane nagu enamustes süsteemides – vigadele vastatakse veateadetega ning õnnestunud operatsiooni korral kuvatakse selle kohta kinnitus. Paljud andmebaasisüsteemid kuvavad veateadetes erinevate vigade korral unikaalseid veakoode, mille abil saavad kasutajad vajadusel otsida lisainformatsiooni. Kui vea täpne asukoht leitakse SQL lause sõelumisel üles, siis kuvatakse antud asukoht ka veateates.

Mõned SQL lausetes tehtavad vead Eessaar [1] kohaselt on näiteks.

- Tabeli või veeru nimi on valesti kirjutatud
- WHERE/HAVING klauslis olev loogikaavaldis on valesti koostatud
- WHERE klauslisse kirjutatakse avaldis, mis tuleks kirjutada HAVING klauslisse või vastupidi
- Ei arvestata, et alampäring võib tabelites olevate andmete muutudes tagastada rohkem kui ühe rea
- Lausetes ei arvestata NULLidega
- CREATE TABLE lauses valed andmetüübid

Nagu on välja toonud Tognazzini oma artiklis First Principles of Interaction Design [3], siis kolm suhtlusprintsiipi, millele peaks kasutajaliidest disainides mõtlema on:

- selgita probleemi,
- ütle kasutajale täpselt, kuidas oma viga parandada,
- valmistu selleks, et süsteem võib kuvada suvalise veateate enda sisemise probleemi tõttu.

Harv juhus on aga see, et veateated kõiki neid nõudmisi täidaks. Tavaline on see, et enamus veateated isegi ei selgita probleemi, nagu näiteks „Juhtus erakorraline sündmus. Jätkamiseks vajuta OK“.

Bolton märgib oma artiklis [4], et veateadete kirjutajad unustavad tihtilugu ära, et need, kes veateateid loevad, ei ole kursis süsteemi sisemise funktsioneerimisega ja seega ei pruugi arendajale mõistlikuna tunduvad veateated olla arusaadavad kasutajale. Teisalt võib lõppkasutajale nähtav liiga tehniliselt detailne veateade anda liigset infot süsteemi suhtes pahatahtlikult meelestatud inimestele, kes tahavad seda rünnata. Seda tuleb silmas pidada, kui rakendus lõppkasutajale veateadet edastab. See ei peaks tähendama seda, et andmebaasisüsteem peab andma meelega välja ebainformatiivse veateate.

Nagu öeldud, siis võivad veateadete kasutajateks olla lõppkasutajad, tehniline tugi, testija ja programmi hooldusega tegelev arendaja. Andmebaasisüsteemi korral on üheks kasutajate tüübiks ka arendajad, kes kirjutavad programme, mis oma toimimiseks peavad andmebaasi kasutama ning andmebaasi disainerid ja programmeerijad, kes kavandavad ja realiseerivad andmebaase. Sisukad veateated on ka väga olulised programmi ja selles kasutatava keele (antud juhul andmebaasisüsteemi ja SQLi) õppijale. Kuna kõik eelnevalt mainitud inimesed on veateadete lugejad, siis väidab ta, et tuleks veateated ka koos nendega välja töötada. Kui

veateade kirjutatakse korra, siis seda loetakse ning püütakse selle abil lahendust leida tuhandeid kordi. Seega, kui kohe tegeleda korralike veateadete väljatöötamisega, hoitakse kokku palju programmi kasutajate ja hooldajate aega ja tõstetakse kasutajate rahulolu.

Töö eesmärk on uurida hulka populaarseid SQL-andmebaasisüsteeme ja teha kindlaks, kui informatiivseid veateateid need esitavad erinevate SQL lausetes tehtud vigade puhul või lausete tulemusena tekkinud erandolukordade puhul. Töö tulemusena leitakse uuritud andmebaasisüsteemide pingerida veateadete headuse seisukohalt ning pakutakse kõige väheinformatiivsematele veateadetele sisukamad alternatiivid. Samuti üritatakse veateadete probleeme üle erinevate andmebaasisüsteemide üldistada.

Töö on vajalik arendajatele, kes kasutavad andmebaasisüsteeme Oracle, PostgreSQL, MS Access. Nad saavad valida veateadete kuvamise informatiivsuse ja kasutajasõbralikkuse põhjal, millist süsteemi kasutada ning mida nendelt süsteemidelt oodata. Iseenesest võiks sellise uuringu tulemused pakkuda huvi ka andmebaasisüsteemide arendajatele, kes saaksid selle alusel veateateid parandada. Kui Oracle ja MS Access on suletud lähtekoodiga kommertssüsteemid, siis PostgreSQLi lähtekood on muudatuste tegemiseks avatud, mis tähendab, et soovi korral on laial arendajate ringil võimalik PostgreSQLi veateadetesse parandusi teha. Samuti võiks see töö huvi pakkuda kasutatavuse spetsialistidele, kes peavad rakenduste jaoks veateateid välja töötama. Kuigi uuritavad veateated väljastatakse andmebaasisüsteemi poolt, siis koorub nende analüüsist välja üldisemaid põhimõtteid selle kohta, millised on head ja halvad veateated.

1. Eksperimendi kirjeldus

Veateadete võrdlemiseks on autor valinud kolm andmebaasisüsteemi, mille põhjal viiakse läbi veateadete võrdlemine. Andmebaasisüsteemideks on Oracle (Database 12c Enterprise Edition Release 12.1.0.1.0), PostgreSQL (9.3.0) ja Microsoft Access (2013) andmebaasisüsteemid. Autor valis need andmebaasisüsteemid, sest neid kasutatakse tema poolt läbitud õppeainetes „Andmebaasid I“ ja „Andmebaasid II“ ning autor on nende andmebaasisüsteemidega kõige rohkem kokku puutunud. Valiku sobilikkust kinnitab ka andmebaasisüsteemide populaarsuse edetabel (DB-Engines Ranking, 16.03.2015) [5]. Kõik kolm valitud andmebaasisüsteemi kuuluvad 257-st enimkasutatavast andmebaasisüsteemi seas esimese seitsme hulka (vt Tabel 1).

Tabel 1. Populaarseimad andmebaasisüsteemid (märts 2015)

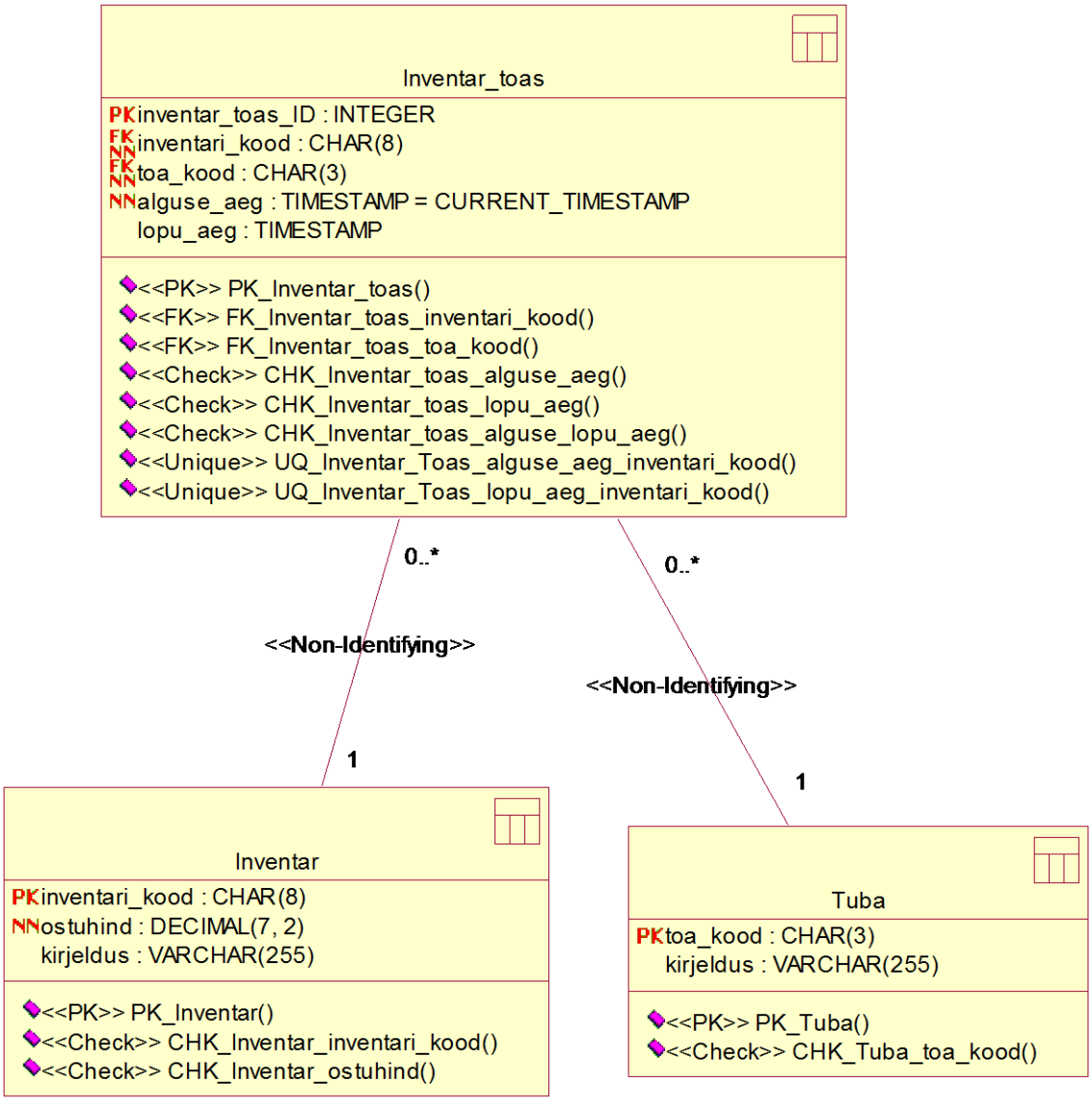
Koht	***Andmebaasisüsteem***
1	**Oracle**
2	MySQL
3	Microsoft SQL Server
4	MongoDB
5	**PostgreSQL**
6	DB2
7	**Microsoft Access**

Eksperimendi eesmärgiks on välja selgitada nende kolme andmebaasisüsteemi erinevused SQL lausete täitmisel tekkinud veateadete seisukohalt. Töös luuakse Oracle, PostgreSQL ja MS Access andmebaasisüsteemide abil sama struktuuriga andmebaasid, kuhu lisatakse identsed testandmed. Seejärel võrreldakse nendes andmebaasisüsteemides SQL lausete täitmisel tekkinud veateateid.

1.1 Eksperimendi andmebaasi projekteerimine

Antud töös kasutatav disaini täpsusega andmemudel (vt Joonis 1) on üks fragment Andmebaasid II [7] projektist, mis on edasiarendus Andmebaasid I aineprojektist [6]. See on loodud kasutades Rational Rose modelleerimistarkvara.

1.2 Disaini täpsusega andmemudel



Joonis 1. Disaini täpsusega andmemudel

1.3 Atribuutide detailsed kirjeldused

Alljärgnev atribuutide kirjelduste tabel (vt Tabel 2) on fragment Andmebaasid II aineprojektist [7]. See atribuutide kirjeldus oli osaks kontseptuaalsest andmemudelist, mille põhjal loodi omakorda eelmises punktis kirjeldatud tabelid.

Tabel 2. Atribuutide detailsed kirjeldused

Tabeli nimi	Atribuudi nimi	Atribuudi definitsioon	Näiteväärtus
Inventar	inventari_kood	Inventari unikaalselt identifitseeriv kood, mida kasutatakse inventari viitamiseks ka väljapool andmebaasi. {Inventari unikaalne identifikaator. Registreerimine on kohustuslik. Inventari kood koosneb täpselt kaheksast numbrist.}	65754687
Inventar	ostuhind	Inventari ostuhind eurodes sisaldades endas ka riigis kehtivaid makse. Hind, mis oli selle ostmise hetkel. {Registreerimine kohustuslik. Ostuhind sisaldab numbreid ning punkte (komakohti, täpsusega 2 kohta pärast koma). Väärtus on vahemikus 0.00 kuni 99999.99.}	44.04
Inventar	kirjeldus	Inventari lühikirjeldus. Kirjelduse järgi peab olema aru saada, mille inventariga on tegu. "{Kirjeldus ei tohi olla tühi string või ainult tühikutest koosnev string.}"[8]	Lamp, millel on lüliti ning 2 meetrine juhe, mille otsas on pistik.
Inventar_toas	alguse_aeg	Kuupäev ja kellaaeg, millal inventari paigutati sellega seotud tuppa. {Registreerimine on kohustuslik. Kuupäev peab olema vahemikus 1. jaanuar 2000 ja tänane kuupäev, formaadis YYYY-MM-dd, peab olema varasem kuupäev, kui lopu_aeg.}	2011-12-24 12:34:21

Tabeli nimi	Atribuudi nimi	Atribuudi definitsioon	Näiteväärtus
Inventar_toas	lopu_aeg	Kuupäev ja kellaaeg, millal inventar eemaldati sellega seotud toast. {Registreerimine on kohustuslik. Kuupäev peab olema vahemikus 1. jaanuar 2000 ja tänane kuupäev, formaadis YYYY-MM-dd, peab olema hilisem kuupäev kui alguse_aeg.}	2011-12-25 12:34:21
Tuba	toa_kood	„Ruumi hoone piires unikaalselt identifitseeriv kood, mis on ka kirjutatud ruumi sisenemiseks mõeldud ukse juurde. Seda koodi kasutatakse ruumile viitamiseks ka väljapool andmebaasi. {Registreerimine on kohustuslik. Ruumi kood koosneb kolmest numbrist, millest esimene näitab korrust, kus tuba asub, toa_kood ei tohi olla tühi string või ainult tühikutest koosnev string. Ühes hoones ei tohi olla kahte või rohkem samasuguse koodiga ruumi.}“ [8]	409
Tuba	kirjeldus	Toa lühikirjeldus. Kirjelduse järgi peab olema arusaadav, mille ruumiga on tegu. “{Kirjeldus ei tohi olla tühi string või ainult tühikutest koosnev string.}”[8]	1-toaline, vaade pargile.

1.4 Baastabelite veergude tüübid

Järgnevalt (vt Tabel 3 – Tabel 5) on kirjeldatud baastabelite (tabelite) *Inventar_toas*, *Inventar* ja *Tuba* veergude tüübid erinevate andmebaasisüsteemide korral. Igas andmebaasisüsteemis on oma andmetüüpide (tüüpide) hulk, mida seal saab veergude kirjeldamiseks kasutada.

Tabel 3. Tabeli *Inventar* veergude tüübid

Inventar			
Veeru nimi	Oracle	PostgreSQL	MS Access
inventari_kood	VARCHAR2(8)	CHAR(8)	CHAR(8)
ostuhind	DECIMAL(7, 2)	DECIMAL(7, 2)	DECIMAL(7, 2)
kirjeldus	VARCHAR2(255)	VARCHAR(255)	VARCHAR(255)

Tabel 4. Tabeli *Tuba* veergude tüübid

Tuba			
Veeru nimi	Oracle	PostgreSQL	MS Access
toa_kood	VARCHAR2(3)	CHAR(3)	CHAR(3)
kirjeldus	VARCHAR2(255)	VARCHAR(255)	VARCHAR(255)

Tabel 5. Tabeli *Inventar_toas* veergude tüübid

Inventar_toas			
Veeru nimi	Oracle	PostgreSQL	MS Access
inventar_toas_ID	NUMBER	INTEGER	INTEGER
inventari_kood	VARCHAR2(8)	CHAR(8)	CHAR(8)
toa_kood	VARCHAR2(3)	CHAR(3)	CHAR(3)
alguse_aeg	TIMESTAMP	TIMESTAMP	TIMESTAMP
lopu_aeg	TIMESTAMP	TIMESTAMP	TIMESTAMP

1.5 Testülesannete kirjeldused

Järgnevalt esitan testülesannete üldised kirjeldused, milledele luuakse andmebaasi iseärasusi arvesse võttes erinevad realisatsioonid. Testülesanded sellesse töösse on võetud selle järgi, milliste vigadega on autor ise kokku puutunud. Selleks, et leida veel rohkem ülesandeid, käisin intervjueerimas juhendajat ning selle tulemusena leidsin veel lisanduvaid ülesandeid. Testülesanded on grupeeritud lause tüübi järgi. Vigade järjekord on juhuslik ja ei viita nende esinemise sagedusele.

1.5.1 Süntaktiliselt ebakorrektsed laused

1. SELECT lauses (päringus) puudub FROM klausel
2. Lauses puudub sulg (nt alampäringu ümbert)
3. Lauses on tühik tabeli nime ja veeru nime vahel (Tabel. veerg)
4. AND loogikaoperaatori asemel kasutatakse ja (päringus kasutatakse sellise nimega operaatorit, mida süsteemis pole defineeritud)
5. SELECT lauses kasutatakse funktsiooni, mida süsteemis pole defineeritud (nt Upper *vs*.Ucase)
6. Võtmesõnade valesti kirjutamine
7. Klauslid lauses vales järjekorras
8. Lauses kasutatakse sõne (string) tähistamiseks jutumärke
9. FROM klauslis olevale alampäringule pole antud aliast
10. GROUP BY klausli puudumine lauses, kus see on nõutud
11. HAVING klauslis kasutatakse aliast
12. WHERE klauslisse kirjutatakse tingimus, mis tuleks kirjutada HAVING klauslisse
13. Kokkuvõttefunktsiooni argumendiks alampäring
14. Kokkuvõttefunktsiooni argumendiks on teise kokkuvõttefunktsiooni poole pöördumine
15. ORDER BY klausli kasutamine DELETE lauses
16. UNION päringus pole alampäringute tulemustel ühesugune struktuur
17. CREATE VIEW lauses puudub AS klausel

1.5.2 Süntaktiliselt korrektsed laused, mis aga ei vasta andmebaasi struktuurile

18. Päring tabelist, mida andmebaasis pole.
19. Päringus on SELECT klauslis vale veeru nimi

20. Päringus on WHERE klauslis vale veeru nimi
21. Päringus on ORDER BY klauslis vale veeru nimi
22. WHERE klauslis olev otsingutingimuses on veeru nimed jäänud korrektselt välja kirjutamata (nt ruumi_nr = 1 OR 2)
23. WHERE klausli tingimuses kasutatav arvu esitav literaal ei esita väärtust, mis kuulub veeru tüüpi (tüüp kui nime omav väärtuste hulk)
24. WHERE klausli tingimuses kasutatav kuupäevana esitav literaal ei esita väärtust, mis kuulub veeru tüüpi (tüüp kui nime omav väärtuste hulk)
25. Alarmpäring tagastab rohkem kui ühe rea olukorras, kus see peaks tagastama maksimaalselt ühe rea.
26. Üritatakse luua tabel nimega, mis on juba olemas
27. Üritatakse kustutada tabelit, mida pole olemas
28. Kustutamine läbi JOIN klausli (pole selge, millisest tabelist peaks ridu kustutama)

1.5.3 Vead andmekirjelduskeele lausetes

29. Tabeli loomisel pole kahe kirjelduse osa vahel koma
30. Tabeli loomisel on veerule jäänud määramata andmetüüp
31. Tabeli loomisel on identifikaatoris (nt tabeli nimes, veeru nimes, kitsenduse nimes) tühik (kui nimi on piiritlemata identifikaator)
32. Välisvõtme deklareerimisel viidatakse tabelile, mida pole loodud
33. Mittedeterministliku CHECK kitsenduse kirjeldamine (samade andmete kontroll erinevatel ajahetkedel võib anda erineva tulemuse)

1.5.4 Tegevus, mis pole kooskõlas kasutaja õigustega

Siin jaotises nimetatavad laused on süntaktiliselt korrektsed, kuid neid ei täideta kasutaja ebapiisavate õiguste hulga tõttu.

34. Kasutajal puudub õigus päringut teostada

1.5.5 Andmemuudatused, mis tekitatavad vastuolu kitsendustega

Siin jaotises nimetatavad laused on süntaktiliselt korrektsed, kuid nende abil viiakse läbi andmemuudatus, mis läheb vastuollu mõne andmebaasis deklareeritud kitsendusega. Andmebaasisüsteem peab sellise kitsenduse tagasi lükkama – see on tema kohus, kuid küsimus on, kui sisukaid veateateid sellisel juhul väljastatakse.

35. FOREIGN KEY vastu eksimine
36. PRIMARY KEY vastu eksimine
37. NOT NULL vastu eksimine
38. CHECK kitsenduse vastu eksimine
39. UNIQUE kitsenduse vastu eksimine

1.6 Andmebaasi tabelite realisatsioon

Kontseptuaalne andmemudel on aluseks, et luua realisatsioonid erinevates andmebaasisüsteemides. Mudelil kasutatakse üldjuhul SQL standardile vastavaid andmetüüpe, et mudel oleks võimalikult universaalne. Mitte ükski tänapäevane SQL-andmebaasisüsteem ei realiseeri SQL standardit täies mahus. Erinevad andmebaasisüsteemid pakuvad igaüks oma versiooni SQL andmekäitluskeelest ja andmekirjelduskeelest ja seetõttu ei ole need süntaktiliselt ühilduvad. Andmebaasisüsteemide erinevused teatul juhtudel (nt FROM klausli nõutavus või mitte) tuuakse igal konkreetsel juhul välja. Antud töös koostab autor ühe andmemudeli alusel kolm erinevat realisatsiooni. Erinevad realisatsioonid tuleb luua nii tabelite kui ka testpäringute kohta. Testandmete lisamise laused on kõigi uuritavate andmebaasisüsteemide puhul ühesugused.

1.6.1 Oracle

```
CREATE TABLE Inventar (
      inventari_kood VARCHAR2(8) NOT NULL,
      ostuhind DECIMAL(7, 2) NOT NULL,
      kirjeldus VARCHAR2(255),
      CONSTRAINT PK_Inventar PRIMARY KEY (inventari_kood),
      CONSTRAINT CHK_Inventar_inventari_kood CHECK (
            REGEXP_LIKE (inventari_kood, '^[0-9]{8}$') )
);

CREATE TABLE Tuba (
      toa_kood VARCHAR2(3) NOT NULL,
      kirjeldus VARCHAR2(255),
      CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood)
);

CREATE TABLE Inventar_Toas (
```

```
      inventar_toas_ID NUMBER GENERATED ALWAYS AS IDENTITY START WITH 1
            INCREMENT BY 1,
      inventari_kood VARCHAR2(8) NOT NULL,
      toa_kood VARCHAR2(3) NOT NULL,
      alguse_aeg TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL,
lopu_aeg TIMESTAMP,

      CONSTRAINT PK_Inventar_Toas PRIMARY KEY (inventar_toas_ID),

      CONSTRAINT UQ_Inventar_Toas_alguse_aeg UNIQUE (
            alguse_aeg, inventari_kood),

      CONSTRAINT UQ_Inventar_Toas_lopu_aeg UNIQUE (
      lopu_aeg, inventari_kood),

      CONSTRAINT CHK_Inventar_Toas_algus CHECK (alguse_aeg < lopu_aeg)
);

ALTER TABLE Inventar_Toas ADD CONSTRAINT FK_Inventar_Toas_toa_kood
      FOREIGN KEY (toa_kood) REFERENCES Tuba (toa_kood)
      INITIALLY DEFERRED DEFERRABLE;


ALTER TABLE Inventar_Toas ADD CONSTRAINT FK_Inventar_Toas_inventar_kood
      FOREIGN KEY (inventari_kood) REFERENCES Inventar (inventari_kood)
      INITIALLY DEFERRED DEFERRABLE;


CREATE OR REPLACE TRIGGER triger_alguse_lopu_aeg
BEFORE INSERT OR UPDATE ON Inventar_toas
  FOR EACH ROW
BEGIN
  IF( :new.alguse_aeg > CURRENT_TIMESTAMP)
  THEN
    RAISE_APPLICATION_ERROR( -20001,
         'Alguse aeg ei tohi olla tulevikus.');
  END IF;
  IF( :new.lopu_aeg > CURRENT_TIMESTAMP )
  THEN
    RAISE_APPLICATION_ERROR( -20002,
        'Lõpu aeg ei tohi olla tulevikus.' );
  END IF;
END;
```

1.6.2 PostgreSQL

```
CREATE TABLE Inventar (

      inventari_kood CHAR(8) NOT NULL,

      ostuhind DECIMAL(7, 2) NOT NULL,

      kirjeldus VARCHAR(255),

      CONSTRAINT PK_Inventar PRIMARY KEY (inventari_kood),

      CONSTRAINT CHK_Inventar_inventari_kood CHECK (
```

```
            inventari_kood ~ '^[0-9]{8}$') ,

      CONSTRAINT CHK_Inventar_ostuhind CHECK (
            ostuhind BETWEEN 0.00 AND 99999.99)

);

CREATE TABLE Tuba (

      toa_kood CHAR(3) NOT NULL,

      kirjeldus VARCHAR(255),

      CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood)

);

CREATE TABLE Inventar_Toas (

      inventar_toas_ID SERIAL NOT NULL,

      inventari_kood CHAR(8) NOT NULL,

      toa_kood CHAR(3) NOT NULL,

      alguse_aeg TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL,

            lopu_aeg TIMESTAMP,

      CONSTRAINT PK_Inventar_Toas PRIMARY KEY (inventar_toas_ID),

      CONSTRAINT UQ_Inventar_Toas_alguse_aeg_inventari_kood
            UNIQUE (alguse_aeg, inventari_kood),

      CONSTRAINT UQ_Inventar_Toas_lopu_aeg_inventari_kood
            UNIQUE (lopu_aeg , inventari_kood),

      CONSTRAINT CHK_Inventar_Toas_alguse_aeg CHECK (
      alguse_aeg BETWEEN '2000-01-01 00:00:00' AND CURRENT_TIMESTAMP),

      CONSTRAINT CHK_Inventar_Toas_lopu_aeg CHECK (
      lopu_aeg BETWEEN '2000-01-01 00:00:00' AND CURRENT_TIMESTAMP),

      CONSTRAINT CHK_Inventar_Toas_alguse_lopu_aeg CHECK (
      alguse_aeg < lopu_aeg));
```

```
ALTER TABLE Inventar_Toas ADD CONSTRAINT FK_Inventar_Toas_toa_kood
      FOREIGN KEY (toa_kood) REFERENCES Tuba (toa_kood)
      ON DELETE NO ACTION ON UPDATE CASCADE;

ALTER TABLE Inventar_Toas ADD CONSTRAINT FK_Inventar_Toas_inventari_kood
      FOREIGN KEY (inventari_kood) REFERENCES Inventar (inventari_kood)
      ON UPDATE CASCADE;
```

1.6.3 MS Access

```
CREATE TABLE Inventar (

    inventari_kood CHAR(8) NOT NULL,

    ostuhind DECIMAL(7, 2) NOT NULL,

    kirjeldus VARCHAR(255),

    CONSTRAINT PK_Inventar PRIMARY KEY (inventari_kood),

    CONSTRAINT CHK_Inventar_inventari_kood CHECK (
      inventari_kood LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),

    CONSTRAINT CHK_Inventar_ostuhind CHECK (
      ostuhind BETWEEN 0.00 AND 99999.99) );

CREATE TABLE Tuba (

      toa_kood CHAR(3) NOT NULL,

      kirjeldus VARCHAR(255),

      CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood)

);

CREATE TABLE Inventar_Toas (

      inventar_toas_ID AUTOINCREMENT NOT NULL,

      inventari_kood CHAR(8) NOT NULL,

      toa_kood CHAR(3) NOT NULL,

      alguse_aeg TIMESTAMP DEFAULT NOW() NOT NULL,

            lopu_aeg TIMESTAMP,

      CONSTRAINT PK_Inventar_Toas PRIMARY KEY (inventar_toas_ID),

      CONSTRAINT UQ_Inventar_Toas_alguse_aeg_inventari_kood
            UNIQUE (alguse_aeg, inventari_kood),

      CONSTRAINT UQ_Inventar_Toas_lopu_aeg_inventari_kood
            UNIQUE (lopu_aeg, inventari_kood),

      CONSTRAINT CHK_Inventar_Toas_alguse_aeg CHECK (
            alguse_aeg BETWEEN '2000-01-01 00:00:00' AND NOW()),

      CONSTRAINT CHK_Inventar_Toas_lopu_aeg CHECK (
      lopu_aeg BETWEEN '2000-01-01 00:00:00' AND NOW()),
```

```
      CONSTRAINT CHK_Inventar_Toas_alguse_lopu_aeg CHECK (
            alguse_aeg < lopu_aeg)

);

ALTER   TABLE   Inventar_Toas   ADD   CONSTRAINT   FK_Inventar_Toas_toa_kood
      FOREIGN KEY (toa_kood) REFERENCES Tuba (toa_kood)
      ON DELETE NO ACTION ON UPDATE CASCADE;

ALTER  TABLE  Inventar_Toas  ADD  CONSTRAINT  FK_Inventar_Toas_inventari_kood
      FOREIGN  KEY  (inventari_kood)  REFERENCES  Inventar  (inventari_kood)
      ON UPDATE CASCADE;
```

1.7 Testandmed

Testandmete tabelitesse lisamise realisatsioon on kõigis kolmes andmebaasisüsteemis identne.

```
INSERT  INTO  Inventar  (inventari_kood,  ostuhind,  kirjeldus)  VALUES
('12345678', '292.29', 'Diivan Lux');
```

```
INSERT  INTO  Inventar  (inventari_kood,  ostuhind,  kirjeldus)  VALUES
('22345678', '1532', 'Vann suur');
```

```
INSERT INTO Tuba (toa_kood, kirjeldus) VALUES ('303', 'Suur tuba');
```

```
INSERT INTO Tuba (toa_kood, kirjeldus) VALUES ('11', 'Väike tuba');
```

```
INSERT    INTO    Inventar_toas    (inventari_kood,    toa_kood)    VALUES
('12345678','303');
```

1.8 Testpäringute realisatsioon

Testpäringute realisatsioonid on välja toodud jaotuses Lisa 1.

1.9 Veateadete hindamiskriteeriumid

Suurepärane veateade peab vastama järgmistele kriteeriumitele.

- Kirjeldus. Kirjeldab tekkinud probleemi.
- Põhjus. Näitab, mis probleemi põhjustas.
- Lahendus. Pakub lahendust, mis aitab kasutajal viga parandada.
- Asjakohasus. Veateade ei tohi olla eksitav.

Töös hinnatakse iga veateadet käesoleva töö autori poolt kuue palli skaalal. Järgnevalt (vt Tabel 6) kirjeldatakse iga hinde puhul, mis tingimustele peab veateade vastama, et sellist hinnet saada.

Tabel 6. Veateadete hindamisskaala

Hinne	Kirjeldus
5	Veateade sisaldab täpset vea kirjeldust (vähemasti märksõnana), näitab vea asukohta (sh enne või pärast viga ühe sõne või sümboli kaugusel näidatud vea asukoht) ja pakub õigeid vihjeid vea parandamiseks.
4	Veateade sisaldab täpset vea kirjeldust (vähemasti märksõnana), näitab vea asukohta (sh enne või pärast viga ühe sõne või sümboli kaugusel näidatud vea asukoht), aga ei paku õigeid vihjeid vea parandamiseks.
3	Veateade sisaldab täpset vea kirjeldust (vähemasti märksõnana), aga ei näita vea asukohta (sh enne või pärast viga ühe sõne või sümboli kaugusel näidatud vea asukoht).
2	Veateade ei sisalda täpset vea kirjeldust (vähemasti märksõnana), aga näitab vea asukohta (sh enne või pärast viga ühe sõne või sümboli kaugusel näidatud vea asukoht).
1	Veateade ei sisalda täpset vea kirjeldust (vähemasti märksõnana) ja ei näita vea asukohta (sh enne või pärast viga ühe sõne või sümboli kaugusel näidatud vea asukoht).
0	Veateade ei sisalda täpset vea kirjeldust (vähemasti märksõnana), ei näita vea asukohta (sh enne või pärast viga ühe sõne või sümboli kaugusel näidatud vea asukoht) ning on eksitav või lause õnnestub ebakorrektselt ilma, et veateadet kuvatakse.

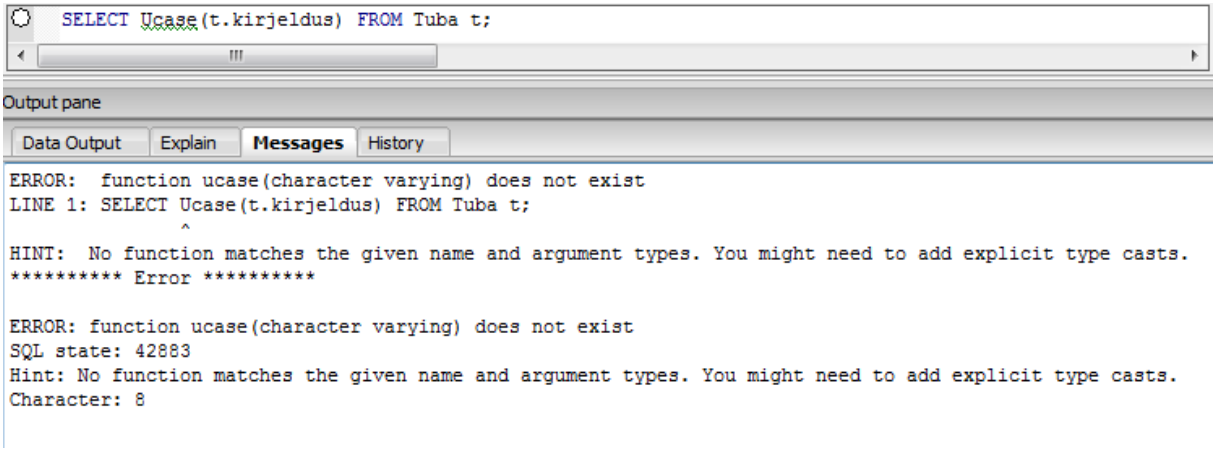
2. Eksperimendi tulemused ja järeldused

Selles peatükis esitatakse eksperimendi tulemused ning nende analüüsimise tulemusena tehtud järeldused. Oracle puhul kasutati lausete käivitamiseks sellega kaasatulevat veebirakenduste kiirprogrammeerimise keskkonda Oracle Application Express. PostgreSQL puhul kasutati lausete käivitamiseks administreerimisprogrammi PgAdmin. MS Access korral käivitati lauseid *Query* aknas. Vea koodi esitamist veateate hindamisel ei arvestata. Veateade peaks olema ka ilma selleta piisavalt arusaadav.

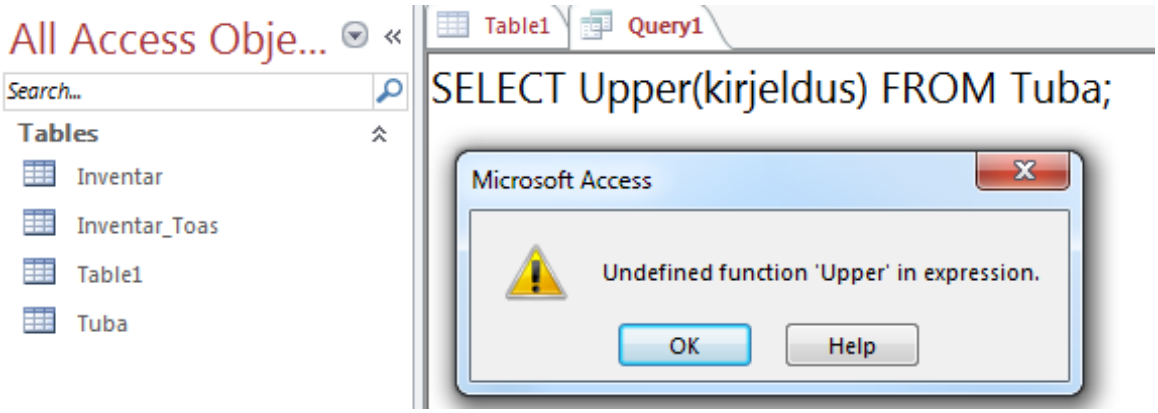
2.1 Veateadete näited ekraanitõmmistena

Joonis 2 – Joonis 4 illustreerivad seda, kuidas näeb veateadet andmebaasisüsteemi kasutaja. Nagu näete, siis Oracle ja PostgreSQL korral on veateatega seotud unikaalne kood, kuid MS Accessi korral sellist koodi ei näidata.

Joonis 2. Näide Oracle andmebaasisüsteemi veateatest (Oracle Application Express)



Joonis 3. Näide PostgreSQL andmebaasisüsteemi veateatest (pgAdmin III)



Joonis 4. Näide veateatest MS Access andmebaasisüsteemis

2.2 Eksperimendi tulemused

Järgnevalt kirjeldatakse eksperimendi tulemusi. Iga päringu korral tuuakse välja veateade, mis antud andmebaasisüsteemis kuvati. Iga veateade on hinnatud autori poolt skaalal 0-5. Päringuid, mis õnnestuvad korrektselt, ei hinnata. Päringud, mis õnnestuvad ebakorrektselt (st lause täitmine toimub kui tegelikult ei peaks toimuma), ilma veateateta, hinnatakse hindega 0.

2.2.1 Süntaktiliselt ebakorrektsed laused

1. SELECT päringus puudub FROM klausel		
Lause näide (PostgreSQL, Oracle, MS Access): `SELECT * Tuba`		
Oracle	ORA-00923: FROM keyword not found where expected	3
PostgreSQL	ERROR: syntax error at or near "Tuba" LINE 1: SELECT * Tuba; ^	2
Access	Syntax error (missing operator) in Query expression '* Tuba'	1

Oracle veateade on teistest parem, sest sel juhul kirjeldati täpselt vea märksõna. PostgreSQL näitab vea asukohta aga MS Access mainib ainult seda, et operaator on puudu.

2. Lauses puudub sulg
Lause näide (PostgreSQL, Oracle, MS Access): `INSERT INTO Tuba (toa_kood, kirjeldus) VALUES ('303', 'Suur tuba'`

Oracle	ORA-00917: missing comma	0
PostgreSQL	ERROR: syntax error at end of input LINE 1: ...T INTO Tuba (toa_kood, kirjeldus) VALUES ('303', 'Suur tuba' ^	2
MS Access	Syntax error in INSERT INTO statement	1

Ainult PostgreSQL leidis vea asukoha üles ja teavitas sellest kasutajat.

3. Lauses on tühik tabeli nime ja veeru nime vahel (Tabel. veerg)		
Lause näide (PostgreSQL, Oracle, MS Access): `SELECT i. ostuhind FROM Inventar i WHERE inventari_kood = '22345678'`		
Oracle	Päring õnnestub korrektselt	-
PostgreSQL	Päring õnnestub korrektselt	-
MS Access	Invalid use of ’.’, ’!’, or ’()’. in query expression ’i. ostuhind’	0

Nii Oracle kui PostgreSQL süsteemides ei kuvatud selle SQL lause käivitamisel ühtki veateadet, see tähendab andmebaasisüsteem leidis nendel juhtudel kasutaja tehtud vea ning parandas selle kasutajat teavitamata.

4. AND loogikaoperaatori asemel kasutatakse & (päringus kasutatakse sellise nimega operaatorit, mida süsteemis pole defineeritud)		
Lause näide (PostgreSQL, Oracle, MS Access): `SELECT * FROM Inventar_toas WHERE toa_kood = '303' & inventari_kood = '12345678'`		
Oracle	ORA-00933: SQL command not properly ended	0
PostgreSQL	ERROR: operator does not exist: unknown & character LINE 1: ...ELECT * FROM Inventar_toas WHERE toa_kood = '303' & inventar... ^	4
MS Access	Päring õnnestub ebakorrektselt. Veateadet ei kuvata.	0

Oracle veateade väidab, et SQL lause on lõpetamata või puudub semikoolon lause lõpus. Arvestades, et kõik töös olevad andmebaasisüsteemid võimaldavad käivitada SQL lauseid, mis ei lõpe semikooloniga, on need veateated kasutajat eksitavad (antud veateated kuvatakse ka lausetes 6, 15, 28).

MS Accessis käivitatud päring tagastas tühja rea tabelist *Inventar_toas*. „&“ on MS Accessis stringide konkatenatsiooni operaatori nimi. Ilmselt täitis MS Access lõpuks päringu

```
SELECT * FROM Inventar_toas WHERE toa_kood = '303inventari_kood
= 12345678'
```

PostgreSQL veateade on hinde 4 väärilne, sest veateade sisaldab täpset seletust veast ning samuti näidatakse selle vea asukohta. Õige vihje selle vea parandamiseks oleks olnud, et kasutage & sümboli asemel AND operaatorit, see oleks andnud veateatele viis punkti.

5. SELECT lauses kasutatakse funktsiooni, mida süsteemis pole defineeritud (nt Upper vs. Ucase)		
Lause näide (Oracle, PostgreSQL): `SELECT Ucase(t.kirjeldus) FROM Tuba t` Lause näide (MS Access): `SELECT Upper(t.kirjeldus) FROM Tuba t`		
Oracle	ORA-00904: "UCASE": invalid identifier	3
PostgreSQL	ERROR: function ucase(character varying) does not exist LINE 1: SELECT Ucase(t.kirjeldus) FROM Tuba t; ^ HINT: No function matches the given name and argument types. You might need to add explicit type casts.	5
MS Access	Undefined function ’Upper’ in expression.	3

PostgreSQL veateade saab hinnangu 5, sest pakutakse välja ka vihje vea parandamiseks.

6. Võtmesõnade valesti kirjutamine		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * FROM Tuba WHRE toa_kood = '303'`		
Oracle	ORA-00933: SQL command not properly ended	0

PostgreSQL	ERROR: syntax error at or near "toa_kood" LINE 1: SELECT * FROM Tuba WHRE toa_kood = '303';	2
MS Access	Syntax error in FROM clause.	0

Oracle veateade on tüüpiline eksitav veateade (vaata ka lause nr 4). Kõik andmebaasisüsteemid jäid antud juhul hätta, kõige parema veateate kuvas PostgreSQL, sest kuvati vähemasti vea oletatav asukoht.

7. Klauslid lauses vales järjekorras		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * WHERE Inventar FROM inventari_kood = '12345678'`		
Oracle	ORA-00923: FROM keyword not found where expected	3
PostgreSQL	ERROR: syntax error at or near "FROM" LINE 1: SELECT * WHERE Inventar FROM inventari_kood = '12345678'; ^	1
MS Access	Syntax error (missing operator) in query expression '* WHERE Inventar'	2

Oracle kuvas kõige parema veateate, sest mainitakse, et FROM klausel on puudu, kus see on nõutud (ehk pärast SELECT klauslit), kuid ei ütle kuhu see tuleks kirjutada. PostgreSQL veateade väidab, et FROM klausli juures on süntaktiline viga. Parem oleks see siis kui see väidaks, et ka WHERE klausli juures on viga. MS Access näitab õiget vea asukohta, aga mitte täpset vea märksõna.

8. Lauses kasutatakse sõne (string) tähistamiseks jutumärke		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * FROM Inventar WHERE inventari_kood = "12345678"`		
Oracle	ORA-00904: "12345678": invalid identifier	3
PostgreSQL	ERROR: column "12345678" does not exist LINE 1: SELECT * FROM Inventar WHERE inventari_kood = "12345678"; ^	4
MS Access	Päring õnnestub ebakorrektselt.	0

Oracle ja PostgreSQL puhul tähendab jutumärkides sõna antud kontekstis veeru nime ning veateated osundavad sellele korrektselt. MS Accessi SQL keele murrak lubab tekstilisi väärtusi esitada nii apostroofide kui jutumärkide vahel, seetõttu õnnestub ka antud päring justkui oleks tingimuses mõeldud tekstilist väärtust '12345678'. Samas pole selline käitumine SQL standardiga kooskõlas ning õige oleks, kui andmebaasisüsteem selle lause tagasi lükkaks.

9. FROM klauslis olevale alampäringule pole antud aliast		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * FROM (SELECT * FROM Inventar)`		
Oracle	Päring õnnestub korrektselt.	-
PostgreSQL	ERROR: subquery in FROM must have an alias LINE 1: SELECT * FROM (SELECT * FROM Inventar); ^ HINT: For example, FROM (SELECT ...) [AS] foo.	5
MS Access	Päring õnnestub korrektselt.	-

Kolmest andmebaasisüsteemist ainult PostgreSQL nõuab aliast FROM klauslis olevale alampäringule. Selle veateade on vormistatud väga hästi ning hindan seda 5 väärilseks.

10. GROUP BY klausli puudumine lauses, kus see on nõutud		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT count(toa_kood), kirjeldus FROM Tuba`		
Oracle	ORA-00937: not a single-group group function	1
PostgreSQL	ERROR: column "tuba.kirjeldus" must appear in the GROUP BY clause or be used in an aggregate function LINE 1: SELECT count(toa_kood), kirjeldus FROM Tuba; ^	5
MS Access	Your query does not include the specified expression 'kirjeldus' as part of an aggregate function.	3

PostgreSQL veateade annab täpselt teada, kus puudub vajalik andmebaasi tabeli veerg. MS Access leidis küll vea üles aga ei näita täpset asukohta ega ka täpseid vihjeid vea parandamiseks.

11. HAVING klauslis kasutatakse aliast		
Lause näide (MS Access): `SELECT Inventar_toas.inventari_kood, toa_kood, alguse_aeg AS algus FROM Inventar_toas INNER JOIN Inventar ON Inventar_toas.inventari_kood = Inventar.inventari_kood GROUP BY Inventar_toas.inventari_kood, Inventar_toas.toa_kood, Inventar_toas.alguse_aeg HAVING algus < NOW()`		
Oracle	ORA-00904: "ALGUS": invalid identifier	3
PostgreSQL	ERROR: column "algus" does not exist LINE 6: HAVING algus < CURRENT_TIMESTAMP; ^	3
Access	Päring õnnestub	-

MS Accessis päring õnnestub korrektselt pärast ette kuvatud hüpikaknasse aliase algus väärtuse sisestamist. MS Access toetab parameetritega päringuid ja käsitleb antud kontekstis sõna *algus* kui parameetrit (kohahoidjat), mille asemel pannakse kasutaja sisestatud väärtus.

12. WHERE klauslisse kirjutatakse tingimus, mis tuleks kirjutada HAVING klauslisse		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT Inventar.inventari_kood, ostuhind, kirjeldus, Count(*) AS inventaride_arv FROM Inventar INNER JOIN Inventar_toas ON Inventar.inventari_kood = Inventar_toas.inventari_kood WHERE Count(*) > 1 GROUP BY Inventar.inventari_kood, ostuhind, kirjeldus`		
Oracle	ORA-00934: group function is not allowed here	1
PostgreSQL	ERROR: aggregate functions are not allowed in WHERE LINE 4: WHERE Count(*) > 1 ^	4

12. WHERE klauslisse kirjutatakse tingimus, mis tuleks kirjutada HAVING klauslisse		
MS Access	Cannot have aggregate function in WHERE clause (Count(*)>1)	4

Oracle veateade ütleb, et kokkuvõttefunktsioon ei ole lubatud kasutada, seega on see eksitav, sest WHERE klausli viga ei märgitud.

13. Kokkuvõttefunktsiooni argumendiks alampäring		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT Max(SELECT ostuhind FROM Inventar) FROM Inventar`		
Oracle	ORA-00936: missing expression	0
PostgreSQL	ERROR: syntax error at or near "SELECT" LINE 1: SELECT Max(SELECT ostuhind FROM Inventar) FROM Inventar; ^	2
MS Access	At most one record can be returned by this subquery	4

Oracle veateade ei ole informatiivne. MS Accessi veateade täidab enamusi hea veateate nõudmisi: viga (alampäring tagastab rohkem kui ühe rea) ning asukoht (alampäring).

14. Kokkuvõttefunktsiooni argumendiks on teise kokkuvõttefunktsiooni poole pöördumine		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT Max(Avg(ostuhind)) AS koige_kallim FROM Inventar`		
Oracle	ORA-00978: nested group function without GROUP BY	0
PostgreSQL	ERROR: aggregate function calls cannot be nested LINE 1: SELECT Max(Avg(ostuhind)) AS koige_kallim FROM Inventar; ^	4
MS Access	Cannot have aggregate function in expression (Max(Avg(ostuhind)))	4

Oracle kuvab eksitava veateate.

15. ORDER BY klausli kasutamine DELETE lauses
Lause näide (Oracle, PostgreSQL, MS Access):

15. ORDER BY klausli kasutamine DELETE lauses		
`DELETE FROM Inventar_toas WHERE toa_kood = '303' ORDER BY inventari_kood`		
Oracle	ORA-00933: SQL command not properly ended	0
PostgreSQL	ERROR: syntax error at or near "ORDER" LINE 3: ORDER BY inventari_kood; ^	2
MS Access	Missing semicolon (;) at end of SQL statement	0

Oracle ja MS Accessi veateated väidavad, et SQL lause on lõpetamata või puudub semikoolon lause lõpus. Arvestades, et kõik töös olevad andmebaasisüsteemid toetavad SQL lauseid, mis ei lõpe semikooloniga, on need veateated kasutajat eksitavad (Oracle puhul sama veateade ka lausetes 4, 6, 28). PostgreSQL kuvab veateate, mis ütleb, et ORDER BY klausli juures on süntaktiline viga. Vähemasti näitab see veateade vea asukohta.

16. UNION päringus pole alampäringute tulemustel ühesugune struktuur		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * FROM Tuba UNION SELECT * FROM Inventar`		
Oracle	ORA-01789: query block has incorrect number of result columns	4
PostgreSQL	ERROR: each UNION query must have the same number of columns LINE 1: SELECT * FROM Tuba UNION SELECT * FROM Inventar; ^	5
MS Access	The number of columns in the two selected tables or queries of a union query do not match	4

Kõik andmebaasisüsteemid tuvastasid antud päringus oleva vea ning kuvasid ka informatiivsed veateated.

17. CREATE VIEW lauses puudub AS klausel		
Lause näide (Oracle, PostgreSQL, MS Access): `CREATE VIEW luksus_inventar SELECT kirjeldus, ostuhind FROM Inventar WHERE ostuhind > 1000`		
Oracle	ORA-00905: missing keyword	1

17. CREATE VIEW lauses puudub AS klausel		
PostgreSQL	ERROR: syntax error at or near “SELECT” LINE 1: CREATE VIEW luksus_inventar SELECT * FROM Inventar WHERE ost… ^	2
MS Access	Invalid SQL syntax – expected token: AS.	3

Oracle leidis, et puudu on võtmesõna, aga milline ja kus see puudu on, ei ole välja toodud. PostgreSQL pakub küll asukohta, kus viga on, aga ei ütle milles viga seisneb. MS Access on ainus, mille veateade sisaldas vea märksõna, samas aga ei näita see veateade vea asukohta.

2.2.2 Süntaktiliselt korrektsed laused, mis aga ei vasta andmebaasi struktuurile

18. Päring tabelist, mida andmebaasis pole.		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * FROM Invetar`		
Oracle	ORA-00942: table or view does not exist	3
PostgreSQL	ERROR: relation “invetar” does not exist LINE 1: SELECT * FROM Invetar ^	4
MS Access	The Microsoft Access database engine cannot find the input table or query ‘Invetar’. Make sure it exists and that its name is spelled correctly.	5

Oracle veateade väidab üldistatult, et tabel või vaade ei eksisteeri. PostgreSQL ja Accessi veateated väidavad, et tabelit või vaadet Inventar ei eksisteeri, seejuures MS Accessi veateade lisab ka vihje, kuidas võiks viga parandada.

19. Päringus on SELECT klauslis vale veeru nimi		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT kirjelus FROM Inventar WHERE inventari_kood ='22345678'`		
Oracle	ORA-00904: “KIRJELUS”: invalid identifier	3
PostgreSQL	ERROR: column “kirjelus” does not exist	4

19. Päringus on SELECT klauslis vale veeru nimi		
	LINE 1: SELECT kirjelus FROM Inventar WHERE inventari_kood = ‘223456… ^	
MS Access	Päring õnnestub ebakorrektselt	0

MS Access toetab parameetritega päringuid ja käsitleb antud kontekstis sõna *kirjelus* kui parameetrit (kohahoidjat), mille asemel pannakse kasutaja sisestatud väärtus. MS Accessis päringu täitmisel küsitakse hüpikaknas ’kirjeluse’ väärtust, mille sisestamisel täidetakse päring, nii, et tagastatakse üks veerg ja rida, mille väärtus on, see mille hüpikakna tekstikasti sisestasid.

20. Päringus on WHERE klauslis vale veeru nimi		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * FROM Inventar WHERE kirjelus =’Vann suur’`		
Oracle	ORA-00904: “KIRJELUS”: invalid identifier	3
PostgreSQL	ERROR: column “kirjelus” does not exist LINE 1: SELECT * FROM Inventar WHERE kirjelus =’Vann suur’ ^	4
MS Access	Päring õnnestub ebakorrektselt	0

MS Accessis päringu täitmisel küsitakse hüpikaknas ’kirjeluse’ väärtust, mille sisestamisel täidetakse lause nii, et kui sisestatud väärtus on ’Vann suur’ siis täidetakse päring (tulemuses kõik tabeli read):

```
SELECT * FROM Inventar WHERE TRUE
```

ja kui midagi muud, siis täidetakse päring (tulemuses pole ühtegi rida):

```
SELECT * FROM Inventar WHERE FALSE
```

21. Päringus on ORDER BY klauslis vale veeru nimi
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * FROM Inventar ORDER BY kirjelus`

21. Päringus on ORDER BY klauslis vale veeru nimi		
Oracle	ORA-00904: “KIRJELUS”: invalid identifier	3
PostgreSQL	ERROR: column “kirjelus” does not exist LINE 1: SELECT * FROM Inventar ORDER BY kirjelus ^	4
MS Access	Päring õnnestub ebakorrektselt	0

MS Accessis päringu täitmisel küsitakse hüpikaknas ’kirjeluse’ väärtust, mille sisestamisel täidetakse päring, nii, et tagastatakse kõik veerud tabelist *Inventar*, sorteerimist ei rakendata.

22. WHERE klauslis olev otsingutingimuses on veeru nimed jäänud korrektselt välja kirjutamata		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * FROM Inventar_toas WHERE inventar_toas_id = 1 OR 2`		
Oracle	ORA-00920: invalid relational operator	1
PostgreSQL	ERROR: argument of OR must be type boolean, not type integer LINE 1: SELECT * FROM Inventar_toas WHERE inventar_toas_id = 1 OR 2 ^	2
MS Access	Päring õnnestub ebakorrektselt	0

PostgreSQLi veateade ütleb, et OR operaatori mõlemad argumendid peavad olema tõeväärtustüüpi, s.t leiti üles viga, mis ei ole esmane. MS Accessi päring õnnestub, näidatakse kõiki ridu tabelist *Inventar_toas*, tagasisidet kasutajale kuvamata.

23. WHERE klausli tingimuses kasutatav arvu esitav literaal ei esita väärtust, mis kuulub veeru tüüpi (tüüp kui nime omav väärtuste hulk)		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * FROM Inventar WHERE inventari_kood = 1`		
Oracle	Päring õnnestub ebakorrektselt	0
PostgreSQL	ERROR: operator does not exist: character = integer LINE 1: SELECT * FROM Inventar WHERE inventari_kood = 1 ^	4

MS Access	Data type mismatch in criteria expression	3

Oracle puhul kuvatakse tulemusse ’no data found’, st andmebaasisüsteemi poolt tehti teisendus, mis teisendas inventari koodi arvuliseks väärtuseks ning otsiti selle järgi inventari. Kui WHERE klauslisse panna arvuna selline inventari kood, mis on andmebaasis olemas tekstina, siis leitakse see rida üles. Sellistest tegevustest tuleks andmebaasisüsteemil kasutajat teavitada.

24. WHERE klausli tingimuses kasutatav kuupäevana esitav literaal ei esita väärtust, mis kuulub veeru tüüpi (tüüp kui nime omav väärtuste hulk)		
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * FROM Inventar_toas WHERE alguse_aeg = 2015-04-13`		
Oracle	ORA-00932: inconsistent datatypes: expected TIMESTAMP got NUMBER	3
PostgreSQL	ERROR: operator does not exist: timestamp without time zone = integer LINE 1: SELECT * FROM Inventar_toas WHERE alguse_aeg = 2015-04-13 ^	4
MS Access	Päring õnnestub ebakorrektselt	0

MS Accessi päringu puhul tagastatakse tühi rida tabelist *Inventar_toas* ilma ühegi veateateta.

25. Alarmpäring tagastab rohkem kui ühe rea olukorras, kus see peaks tagastama maksimaalselt ühe rea.		

Lause näide (Oracle, PostgreSQL, MS Access):
`SELECT inventari_kood FROM Inventar_toas`
`WHERE inventari_kood = (SELECT inventari_kood FROM Inventar)`		

Oracle	ORA-01427: single-row subquery returns more than one row	3
PostgreSQL	ERROR: more than one row returned by a subquery used as an expression	3
MS Access	At most one record can be returned by this subquery.	3

Kõik andmebaasisüsteemid tagastavad antud päringu puhul kasutajale rahuldava ja suhteliselt sarnase veateate.

26. Üritatakse luua tabel nimega, mis on juba olemas		
Lause näide (Oracle, PostgreSQL, MS Access): `CREATE TABLE Tuba (toa_kood CHAR(3) NOT NULL, kirjeldus VARCHAR(255), CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood));`		
Oracle	ORA-00955: name is already used by an existing object	3
PostgreSQL	ERROR: relation “tuba” already exists	4
MS Access	Table ‘Tuba’ already exists	4

PostgreSQL ja MS Accessi veateated toovad välja tabeli nime, mida üritati luua ning mis on juba olemas, Oracle seda ei tee.

27. Üritatakse kustutada tabelit, mida pole olemas		
Lause näide (Oracle, PostgreSQL, MS Access): `DROP TABLE Tubad`		
Oracle	ORA-00942: table or view does not exist	3
PostgreSQL	ERROR: table "tubad" does not exist	4
MS Access	Table ‘Tubad’ does not exist	4

PostgreSQL ja Accessi veateated toovad välja tabeli nime, mida üritati kustutada ning mida pole olemas, Oracle seda ei tee.

28. Kustutamine läbi JOIN klausli (pole selge, millisest tabelist peaks ridu kustutama)		
Lause näide (Oracle, PostgreSQL, MS Access): `DELETE FROM Inventar_toas INNER JOIN Inventar ON Inventar_toas.inventari_kood=Inventar.inventari_kood WHERE toa_kood = '303'`		
Oracle	ORA-00933: SQL command not properly ended	0
PostgreSQL	ERROR: syntax error at or near "INNER" LINE 2: INNER JOIN Inventar ^	1

28. Kustutamine läbi JOIN klausli (pole selge, millisest tabelist peaks ridu kustutama)		
MS Access	Specify the table containing the records you want to delete.	5

Oracle veateade väidab, et SQL lause on lõpetamata, kuigi nagu juba enne mainitud, kõik antud töös uuritavate andmebaasisüsteemide SQL keele murrakut toetavad semikooloni puudumist lause lõpust (antud veateated kuvatakse ka lausetes 4, 6, 15). MS Accessi veateade antud päringule on sellise vea puhul väga informatiivne ja abistav.

2.2.3 Vead andmekirjelduskeeles

29. Tabeli loomisel pole kahe kirjelduse osa vahel koma		
Lause näide (Oracle, PostgreSQL, MS Access): `CREATE TABLE Tuba ( toa_kood CHAR(3) NOT NULL, kirjeldus VARCHAR(255) CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood) );`		
Oracle	ORA-00907: missing right parenthesis	0
PostgreSQL	ERROR: syntax error at or near "(" LINE 4: CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood) ^	0
MS Access	Syntax error in CREATE TABLE statement	1

Oracle veateade väidab, et kuskil on puudu parempoolne sulg (sama veateade ka lauses 31). PostgreSQL veateade väidab, et primaarvõtme veeru viitamisel on tehtud süntaktiline viga, ning osutab veaga seoses valele kohale lauses. Erinevalt eelmisest kahest ei ole MS Accessi veateade otseselt eksitav, sest see on väga üldine.

30. Tabeli loomisel on veerule jäänud määramata andmetüüp		

Lause näide (Oracle, PostgreSQL, MS Access):
`CREATE TABLE Tuba ( toa_kood NOT NULL, kirjeldus VARCHAR(255),`
`CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood) );`		

Oracle	ORA-02263: need to specify the datatype for this column	3
PostgreSQL	ERROR: syntax error at or near "NOT" LINE 2: toa_kood NOT NULL, ^	2
MS Access	Syntax error in field definition.	3*

*MS Accessi puhul kuvatakse pärast veateate hüpikakna sulgemist SQL lausete kirjutamise aknas aktiivsena NOT võtmesõna, mis näitab vea oletatavat asukohta. Selle tõttu saab antud veateadet hinnata hindega 3. PostgreSQL veateade ei täpsusta, et viga on veeru defineerimise, vaid mainib ainult süntaktilist viga.

31. Tabeli loomisel on identifikaatoris (nt tabeli nimes, veeru nimes, kitsenduse nimes) tühik (kui nimi on piiritlemata identifikaator)		
Lause näide (Oracle, PostgreSQL, MS Access): `CREATE TABLE Tuba ( toa kood CHAR(3) NOT NULL, kirjeldus VARCHAR(255), CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood) );`		
Oracle	ORA-00907: missing right parenthesis	0
PostgreSQL	ERROR: syntax error at or near "CHAR" LINE 2: toa kood CHAR(3) NOT NULL, ^	2
MS Access	Syntax error in field definition.	3*

Oracle veateade väidab, et kuskil on puudu parempoolne sulg (sama veateade ka lauses 29). PostgreSQL leidis vea oletatava asukohta üles, küll aga ei mainita vea täpset märksõna. *MS Accessi puhul kuvatakse pärast veateate hüpikakna sulgemist SQL päringu aknas aktiivsena märksõna „kood“, mis näitab vea oletatavat asukohta, selle tõttu saab antud veateadet hinnata hindega 3.

32. Välisvõtme deklareerimisel viidatakse tabelile, mida pole loodud		
Lause näide (Oracle): `ALTER TABLE Inventar_Toas ADD CONSTRAINT FK_Inventar_Toas_toa_kood FOREIGN KEY (toa_kood) REFERENCES Tubad (toa_kood) INITIALLY DEFERRED DEFERRABLE;`		
Oracle	ORA-00942: table or view does not exist	1
PostgreSQL	ERROR: relation "tubad" does not exist	3
MS Access	Cannot find table or constraint	1

Oracle ja MS Access ei kuva veateates tabeli nime, mida ei eksisteeri andmebaasis.

33. Mittedeterministliku CHECK kitsenduse kirjeldamine		
Lause näide (PostgreSQL): `ALTER TABLE Inventar_Toas ADD CONSTRAINT CHK_Inventar_Toas_alguse_aeg CHECK (alguse_aeg < CURRENT_TIMESTAMP);`		
Oracle	ORA-02436: date or system variable wrongly specified in CHECK constraint	4
PostgreSQL	Lause täitmine õnnestub	-
MS Access	Lause täitmine õnnestub	-

PostgreSQL ja MS Accessi andmebaasisüsteemide puhul on mittedeterministlik CHECK kitsendus lubatud.

2.2.4 Tegevus, mis pole kooskõlas kasutaja õigustega

34. Kasutajal puudub õigus päringut teostada		

Lause näide (PostgreSQL):
`CREATE USER vigade_haldaja WITH PASSWORD 'DSA18JL';`
`GRANT CONNECT ON DATABASE errors TO vigade_haldaja;`
`GRANT USAGE ON SCHEMA public TO vigade_haldaja;`
`GRANT SELECT ON Tuba TO vigade_haldaja;`
`INSERT INTO Tuba (toa_kood, kirjeldus) VALUES ('213', 'Remonditud tuba');`		

Oracle	ORA-01031: insufficient privileges	3
PostgreSQL	ERROR: permission denied for relation tuba	4
MS Access	-	-

Andmebaasikasutaja defineerimine MS Access keskkonnas on seotud Windowsi sisese *workspace*-ga seotud, seetõttu jäi Accessi veateade sellest punktis välja.

2.2.5 Andmemuudatused, mis tekitavad vastuolu kitsendustega

35. FOREIGN KEY vastu eksimine
Lause näide (Oracle, PostgreSQL, MS Access):

35. FOREIGN KEY vastu eksimine		
`INSERT INTO Inventar_toas ( inventari_kood, toa_kood ) VALUES ( '12342678', '303' );`		
Oracle	ORA-02091: transaction rolled back ORA-02291: integrity constraint (C##TUD10. FK_INVENTAR_TOAS_ INVENTAR_KOOD) violated - parent key not found	3
PostgreSQL	ERROR: insert or update on table "inventar_toas" violates foreign key constraint "fk_inventar_toas_inventari_kood" DETAIL: Key (inventari_kood) = (12342678) is not present in table "inventar".	5
MS Access	Microsoft Access can't append all the records in the append query. Microsoft Access set 0 field(s) to Null due to a type conversion failure, and it didn't add 1 record(s) to the table due to key violations, 0 records(s) due to lock violations, and 0 record(s) due to validation rule violations. Do you want to run the action query action anyway? To ignore the error(s) and run the query, click Yes. For an explanation of the causes of the violations, click Help.	1

MS Accessi veateade on musternäide, milline veateade ei tohi olla. Ilmselt on üritatud luua universaalne dünaamiline veateate muster, mis vastaks mitmele erinevale veale (sama väikeste erinevustega veateade ka lausetes 36, 37, 38, 39).

36. PRIMARY KEY vastu eksimine		
Lause näide (Oracle, PostgreSQL, MS Access): `INSERT INTO Tuba (toa_kood, kirjeldus) VALUES ('303', 'Suur tuba');`		
Oracle	ORA-00001: unique constraint (C##TUD10.PK_TUBA) violated	3
PostgreSQL	ERROR: duplicate key value violates unique constraint "pk_tuba" DETAIL: Key (toa_kood) = (303) already exists.	5
MS Access	Microsoft Access can't append all the records in the append query. Microsoft Access set 0 field(s) to Null due to a type conversion failure, and it didn't add 1 record(s) to the table due to key	1

36. PRIMARY KEY vastu eksimine		
	violations, 0 records(s) due to lock violations, and 0 record(s) due to validation rule violations. Do you want to run the action query action anyway? To ignore the error(s) and run the query, click Yes. For an explanation of the causes of the violations, click Help.	

MS Accessi dünaamiline veateade kuvatakse lausete 35, 36, 37, 38, 39 korral.

37. NOT NULL vastu eksimine		
Lause näide (Oracle, PostgreSQL, Access): `INSERT INTO Tuba ( kirjeldus ) VALUES ( 'Suur tuba' );`		
Oracle	ORA-01400: cannot insert NULL into ("C##TUD10"."TUBA"."TOA_KOOD")	4
PostgreSQL	ERROR: null value in column "toa_kood" violates not-null constraint DETAIL: Failing row contains (null, Suur tuba).	5
MS Access	Microsoft Access can't append all the records in the append query. Microsoft Access set 0 field(s) to Null due to a type conversion failure, and it didn't add 0 record(s) to the table due to key violations, 0 records(s) due to lock violations, and 1 record(s) due to validation rule violations. Do you want to run the action query action anyway? To ignore the error(s) and run the query, click Yes. For an explanation of the causes of the violations, click Help.	1

MS Accessi dünaamiline veateade kuvatakse lausete 35, 36, 37, 38, 39 korral.

38. CHECK kitsenduse vastu eksimine

Lause näide (Oracle, PostgreSQL, MS Access):
`INSERT INTO Inventar (inventari_kood, ostuhind, kirjeldus)`
`VALUES ('abcd5678', '292.29', 'Diivan Lux');`

38. CHECK kitsenduse vastu eksimine		
Oracle	ORA-02290: check constraint (C##TUD10.CHK_INVENTAR_INVENTARI_KOOD) violated	4
PostgreSQL	ERROR: new row for relation "inventar" violates check constraint "chk_inventar_inventari_kood" DETAIL: Failing row contains (abcd5678, 292.29, Diivan Lux).	4
MS Access	Microsoft Access can't append all the records in the append query. Microsoft Access set 0 field(s) to Null due to a type conversion failure, and it didn't add 0 record(s) to the table due to key violations, 0 records(s) due to lock violations, and 1 record(s) due to validation rule violations. Do you want to run the action query action anyway? To ignore the error(s) and run the query, click Yes. For an explanation of the causes of the violations, click Help.	1

MS Accessi dünaamiline veateade kuvatakse lausete 35, 36, 37, 38, 39 korral.

39. UNIQUE kitsenduse vastu eksimine		
Lause näide (PostgreSQL): `INSERT INTO Inventar_toas (inventari_kood, toa_kood, alguse_aeg) VALUES ('12345678', '303', '2015-04-13 18:09:19.743733');`		
Oracle	ORA-00001: unique constraint (C##TUD10.UQ_INVENTAR_TOAS_ALGUSE_AEG) violated	4
PostgreSQL	ERROR: duplicate key value violates unique constraint "uq_inventar_toas_alguse_aeg_inventari_kood" DETAIL: Key (alguse_aeg, inventari_kood) = (2015-04-13 18:09:19.743733, 12345678) already exists.	5
MS Access	Microsoft Access can't append all the records in the append query. Microsoft Access set 0 field(s) to Null due to a type conversion failure, and it didn't add 1 record(s) to the table due to key violations, 0 records(s) due to lock violations, and 0 record(s) due to validation rule violations.	1

39. UNIQUE kitsenduse vastu eksimine		
	Do you want to run the action query action anyway? To ignore the error(s) and run the query, click Yes. For an explanation of the causes of the violations, click Help.	

MS Accessi dünaamiline veateade kuvatakse lausete 35, 36, 37, 38, 39 korral.

2.2.6 Tulemuste kokkuvõte

Tabelis 7 esitatakse uuringu koondtulemus. Tabelis on toodud välja katsetatud SQL lausete üldised kirjeldused ning nende kirjelduste põhjal tehtud SQL lausete täitmisel tekkinud veateadete hinnangud skaalal 0-5 (v.a juhtumid, kui lause täitmine õnnestus korrektselt, siis tähistatakse tulemust sümboliga ’ - ’). Igale kirjeldusele võis vastata kolm erinevat SQL lauset, kuna kõikides vaadeldavates andmebaasisüsteemides (Oracle, PostgreSQL, MS Access) on kasutusel erinevad SQL keele murrakud (kõik SQL laused on välja toodud lisas Lisa 1). Olenemata keelelistest eripäradest on kõik erinevad murrakud piisavalt sarnased ning sellise töö kontekstis võrreldavad.

Tabel 7. Uuringu koondtulemus

	Hinded veateadetele		
Mittekorrektse SQL lause kirjeldus	Oracle	PostgreSQL	MS Access
1. SELECT päringus puudub FROM klausel	3	2	1
2. Lauses puudub sulg	0	2	1
3. Lauses on tühik tabeli nime ja veeru nime vahel (Tabel. veerg)	-	-	0
4. AND loogikaoperaatori asemel kasutatakse & (päringus kasutatakse sellise nimega operaatorit, mida süsteemis pole defineeritud)	0	4	0
5. SELECT lauses kasutatakse funktsiooni, mida süsteemis pole defineeritud (nt Upper vs. Ucase)	3	5	3

	Hinded veateadetele		
Mittekorrektse SQL lause kirjeldus	Oracle	PostgreSQL	MS Access
6. Võtmesõnade valesti kirjutamine	0	2	0
7. Klauslid lauses vales järjekorras	3	1	2
8. Lauses kasutatakse sõne (string) tähistamiseks jutumärke	3	4	0
9. FROM klauslis olevale alampäringule pole antud aliast	-	5	-
10. GROUP BY klausli puudumine lauses, kus see on nõutud	1	5	3
11. HAVING klauslis kasutatakse aliast	3	3	-
12. WHERE klauslisse kirjutatakse tingimus, mis tuleks kirjutada HAVING klauslisse	1	4	4
13. Kokkuvõttefunktsiooni argumendiks alampäring	0	2	4
14. Kokkuvõttefunktsiooni argumendiks on teise kokkuvõttefunktsiooni poole pöördumine	0	4	4
15. ORDER BY klausli kasutamine DELETE lauses	0	2	0
16. UNION päringus pole alampäringute tulemustel ühesugune struktuur	4	5	4
17. CREATE VIEW lauses puudub AS klausel	1	2	3
Süntaktiliselt ebakorrektsed laused – veateadete keskmine hinne	1,47	3,25	1,93
18. Päring tabelist, mida andmebaasis pole.	3	4	5
19. Päringus on vale veeru nimi SELECT klauslis	3	4	0
20. Päringus on vale veeru nimi FROM klauslis	3	4	0
21. Päringus on vale veeru nimi ORDER BY klauslis	3	4	0

	Hinded veateadetele		
Mittekorrektse SQL lause kirjeldus	Oracle	PostgreSQL	MS Access
22. WHERE klauslis olev otsingutingimuses on veeru nimed jäänud korrektselt välja kirjutamata	1	2	0
23. WHERE klausli tingimuses kasutatav arvu esitav literaal ei esita väärtust, mis kuulub veeru tüüpi (tüüp kui nime omav väärtuste hulk)	0	4	3
24. WHERE klausli tingimuses kasutatav kuupäevana esitav literaal ei esita väärtust, mis kuulub veeru tüüpi (tüüp kui nime omav väärtuste hulk)	3	4	0
25. Alarmpäring tagastab rohkem kui ühe rea olukorras, kus see peaks tagastama maksimaalselt ühe rea.	3	3	3
26. Üritatakse luua tabel nimega, mis on juba olemas	3	4	4
27. Üritatakse kustutada tabelit, mida pole olemas	3	4	4
28. Kustutamine läbi JOIN klausli (pole selge, millisest tabelist peaks ridu kustutama)	0	1	5
Süntaktiliselt korrektsed, aga ei vasta andmebaasi struktuurile – veateadete keskmine hinne	2,27	3,45	2,18
29. Tabeli loomisel pole kahe kirjelduse osa vahel koma	0	0	1
30. Tabeli loomisel on veerule jäänud määramata andmetüüp	3	2	3
31. Tabeli loomisel on identifikaatoris (nt tabeli nimes, veeru nimes, kitsenduse nimes) tühik (kui nimi on piiritlemata identifikaator)	0	2	3

	Hinded veateadetele		
Mittekorrektse SQL lause kirjeldus	Oracle	PostgreSQL	MS Access
32. Välisvõtme deklareerimisel viidatakse tabelile, mida pole loodud	1	3	1
33. Mittedeterministliku CHECK kitsenduse kirjeldamine	4	-	-
Vead andmekirjelduskeeles - veateadete keskmine hinne	1,6	1,75	2
34. Kasutajal puudub õigus päringut teostada	3	4	-
Tegevus, mis pole kooskõlas kasutaja õigustega – veateadete keskmine hinne	3	4	-
35. FOREIGN KEY vastu eksimine	3	5	1
36. PRIMARY KEY vastu eksimine	3	5	1
37. NOT NULL vastu eksimine	4	5	1
38. CHECK kitsenduse vastu eksimine	4	4	1
39. UNIQUE kitsenduse vastu eksimine	4	5	1
Andmemuudatused, mis tekitavad vastuolu kitsendustega – veateadete keskmine hinne	3,6	4,8	1
Kõikide lausete üldtulemus			
	Oracle	PostgreSQL	MS Access
Väga halbade veateadete osakaal (hinne 0)	27%	3%	**29%**
Rahuldavate veateadete osakaal (hinded 3-5)	59%	**68%**	43%
Väga heade veateadete osakaal (hinne 5)	0%	**22%**	4%
Punktide kogusumma	76	125	66
Hinnatuid veateateid	37	37	35

	Hinded veateadetele		
Mittekorrektse SQL lause kirjeldus	Oracle	PostgreSQL	MS Access
Veateadete keskmine hinne	**2,05**	**3,38**	**1,89**

Uuringust tuli välja, et kõige paremaid veateateid uuritud lausete puhul kuvab PostgreSQL andmebaasisüsteem, keskmiseks hindeks kujunes 3,38. Oracle ja MS Access andmebaasisüsteemid kuvasid headuselt suhteliselt võrdseid veateateid. Oracle sai keskmiseks hindeks 2,05 ja MS Access 1,89. Olenemata sellest, et Oracle ja MS Access kuvasid mõlemad võrdselt väga halbu veateateid võib siiski väita nii keskmise hinde kui ka rahuldavate veateadete osakaalu põhjal, et Oracle kuvab uuritud lausete puhul paremaid veateateid kui MS Access.

Vaadates tulemusi veateadete grupeerimise järgi, siis saab väita, et kõige suurema valimiga grupi, süntaktiliste vigadega SQL lausete veateadete puhul osutus parimaks samuti PostgreSQL, kuid märkimisväärne on see, et antud jaotises esineb Oracle andmebaasisüsteem üllatavalt halvasti, keskmiseks skooriks 1,47. MS Access sai antud jaotises skooriks 1,93, mis ei ole küll väga hea, kuid siiski tuntavalt parem skoor kui Oracle andmebaasisüsteemil.

Kõige kehvemaid veateateid kuvasid uuritud andmebaasisüsteemid jaotises „Vead andmekirjelduskeeles“, kus kõige paremat keskmist tulemust näitas MS Access, saades skooriks 2, järgnes PostgreSQL 1,75-ga ja kõige kehvemini esines selles jaotuses Oracle 1,6-ga. Halbu tulemusi võib põhjendada sellega, et antud alajaotises on suurema fataalsusega vead, need võivad olla nii süntaktilised kui ka semantilised ebakorrektsed ning andmebaasisüsteem ei pruugi leida mitmest veast üles algset veapõhjust.

„Süntaktiliselt korrektsed, aga ei vasta andmebaasi struktuurile“ – ainus jaotus, kus kõik andmebaasisüsteemid said keskmiseks skooriks üle 2. See tundub ka mõistetav, sest vead, mis on vastuolus andmebaasi struktuuriga peaksidki olema kõige paremini tuvastatavad, sellest tulenevad ka paremad veateated.

„Andmemuudatused, mis tekitavad vastuolu kitsendustega“ – jaotuse tulemustest võib järeldada, et MS Accessi andmebaasisüsteemi veateadetele on teatud sarnastele vealiikidele üritatud luua ühtne mall mitme veateate kuvamiseks kasutajale. See tähendab, et üks veateade,

küll minimaalsete muudatustega, kuvatakse mitme erineva vea korral. Sama probleem on ka Oracle andmebaasisüsteemis, kus võis näha isegi kuidas erinevatele vigadele kuvati sama veateadet (vaata SQL laused nr 4, 6, 15, 28).

Oracle andmebaasisüsteemis on veateated üldiselt väga napisõnalised. Seda näitab ka uuring, 1/3 Oracle veateadetest on eksitavad. Oracles on iga veateade oma unikaalse identifikaatoriga (nt ORA-00901), aga kui vea algset põhjust ei leitud, siis selle koodi põhjal edasi uurides võib kasutaja veelgi rohkem segadusse sattuda.

MS Accessi puhul saab välja tuua, et antud süsteemis on loodud veateated, mis oleks võimalikult sarnased tavalistele lausetele, n-ö inimloetavad veateated (nt lause nr. 18). Hästiloetavate lausete kõrval on aga unustatud lisada veateadete unikaalsed koodid, mida asendab „Abi“ nupp veateate hüpikaknas, millele klõpsamisel viiakse Microsofti veateadete keskusse, kus võid loota täiendavat abi. MS Accessi eripära võrreldes teiste uuringus olnud andmebaasisüsteemidega on see, et mõnede situatsioonide korral, täpsemalt tundmatutele veergudele viidates, küsitakse hüpikaknas selle „tundmatu“ veeru väärtust. Mõnel juhul sellise stsenaariumi täitmisel päring õnnestub korrektselt, mõnel korral mitte (vaata laused 11, 19, 20, 21, 30, 31). Siin on põhjuseks MS Accessi toetus parameetrite kasutamisele andmekäitluskeele lausetes. Selline funktsionaalsus on potentsiaalne vigade allikas, sest kasutaja kes kirjutas veeru nime valesti ei saa selle kohta tagasisidet. Teisalt on jälle andmebaasisüsteemil võimatu kindlaks teha, milline oli lause kirjutaja tahtlus – kas ta tahtis kirjutada veeru nime ja kirjutas selle valesti või tahtis kirjeldada parameetrit.

Võib väita, et PostgreSQL on uuritud kolmest andmebaasisüsteemist veateadete kuvamises kõige parem. Enamus PostgreSQL veateated sisaldavad endas leitud vea täpset asukohta, samuti kohati kuvatakse eraldi DETAIL ja HINT märksõnad, milles antakse vihjeid, kuidas vigu parandada. PostgreSQL leidis 2/3 juhtudel üles vähemasti täpse vea märksõna. Samuti on igal veal PostgreSQL süsteemis oma unikaalne identifikaator, mille abil vajadusel otsida lisainformatsiooni vea parandamiseks. Lisaks on PostgreSQL ka avatud lähtekoodiga, mis muudaks võimalikuks soovijal veateadete süsteemi veelgi rohkem täiustada.

2.3 Veateadete üldistus andmebaasisüsteemides ning tarkvaras üldiselt

Horning kirjutab oma 1978. aasta artiklis [12]: „Kompilaatori kirjutaja arvab lühinägelikult, et lähtekood on alati korrektne: viga on erand ja tüütus, sest see häirib kompilaatori elegantsete algoritmide tööd. Kui kompilaator võetakse kasutusele ning selle veateated on lootusetud, siis selle järgnevad mitmed parandused, kuni kasutajad hakkavad vähem kaebama. Sellised süsteemid ei saa kunagi kasutajasõbralikeks“. Selliste sõnadega võib ka tänapäeva tarkvaraarendust kirjeldada. Kui programmi loomisel ei arvestata kohe vigade (erandite) haldamisega, siis ei saagi sellest kunagi hästikasutatavat tarkvara. Kasutaja tahab programmi kasutada ning mitte teada, kuidas programm sisemiselt toimib ning mitte ise uurida, kuidas probleeme lahendada.

Kui tol ajal oli probleemiks ka vähene tehnoloogiline suutlikus kuvada veateateid, siis tänapäeva agiilses tarkvaraarenduses võib see jääda tahaplaanile seoses kiiretele tulemustele suunatud arendusega. Vähestele klientidele suudetaks selgeks teha, et oleks tarvis luua eraldi erandite ning veateadete haldamise alamsüsteem, nagu selle vajalikkust mainis juba Brown oma 1983 aasta artiklis [11].

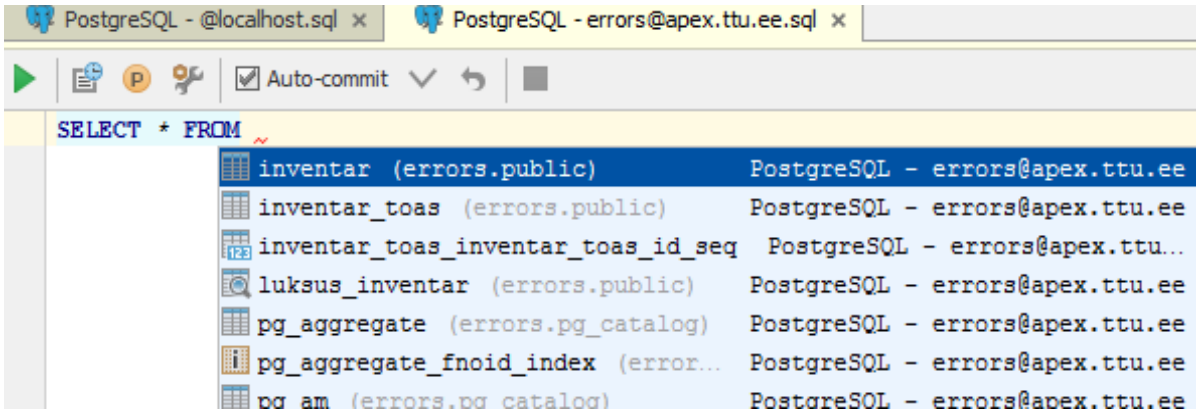
Veateated ei tohiks sisaldada väga spetsiifilist informatsiooni. See võib anda süsteemi ründajatele liiga palju kasulikku informatsiooni. Vastupidiselt, ei tohiks sattuda ka teise äärmusesse, kus veateated on väga üldsõnalised, samas üldsõnaline veateade võib teatud olukordades rohkem kasu anda kui väga spetsiifiline vale veateade (vt nt MS Accessi veateid lausetele 29 ja 31). Selle „vahepealse“ hea veateate vajalikkust mainib ka Brown [11]. Samas ütleb ta, et selleks on vaja läbi viia väga hea analüüs iga võimaliku vea korral.

Igale erinevale veale peaks vastama erinev veateade. Väga halva nii-öelda dünaamilise veateate näidet, kus üks veateade, küll väikeste erinevustega, kuvatakse mitme erineva vea puhul, võib näha MS Accessi veateadetes lausetes 35, 36, 37, 38, 39.

Võimalusel tuleks vea korral üritada sellest aru saada ja seda parandada, kuid sellest tuleb kindlasti kasutajale tagasisidet anda. Ei tohiks üritada viga peita nagu näiteks MS Access, mis tõlgendab valesti kirjutatud veeru nime korral seda parameetrina ja küsib kasutajalt tagasisidet (vt laused 19, 20, 21). Selle asemel tuleks öelda, et antud tabelist ei leitud sellise nimetusega veergu. MS Accessi SQL dialekti tuleks muuta nii, et lauses peaks parameetri olemasolu mingil spetsiifilisemal viisil määrama, et andmebaasisüsteem ei hakkaks vales olukorras arvama nagu kasutaja kirjutanuks lausesse parameetri.

Rakenduste loomisel tuleks mõelda, kas kuvada kasutajale andmebaasisüsteemi poolt pakutud veateateid või esitada rakenduse poole teisendatud veateated. Arvestades, et enamus andmebaasisüsteemide poolt väljastatud veateated ei ole tavakasutajale arusaadavad, siis peaks ikkagi looma igale tarkvarale veateadete töötlemise mooduli. Süsteem peaks andmebaasisüsteemi veateatest ning süsteemi enda tehinguinfost aru saama, mis valesti läks. Nende kahe infoallika põhjal on võimalik selgeks teha vead ning kuvada kasutajale süsteemi poolt arusaadavam veateade. Taolise lähenemise kasuks räägib ka see, et selliseid veateateid on võimalik tõlkida erinevatesse keeltesse. Tõlgitavate veateadete toetuse lisamine on näide süsteemi globaliseerimisest, mis muudab selle kasutatavaks kogu maailmas.

Tänapäeval populaarsed andmebaasisüsteemid on väga suured tarkvaraprojektid ning nende veahaldamise süsteemi muutmine on keeruline ja aeganõudev protsess. Selleks, et andmebaasiarendajate elu lihtsamaks teha, on loodud palju kolmanda osapoole tarkvara. Andmebaasiarendajate jaoks on loodud abistavaid programmeerimiskeskkondi (inglise keeles lühend IDE), et andmebaasisüsteemide kehvade veateadete olukorda parandada. Selliste vahendite ülesanne on vähendada arendaja eksimuste arvu, eelkompileerides SQL koodi lähtekoodi redaktoris ning leides vigu ning pakkudes kiirlahendusi koodi loomise käigus. Üheks selliseks näiteks on varajases arendusfaasis olev JetBrainsi toode 0xDBE [16], millel on väga palju funktsionaalsust just arendaja poolt tekkinud vigade vähendamiseks, näiteks redaktor teab andmebaasi skeemi ning pakub selle abil lahendusi probleemidele ja ka abi lausete koostamisel. Selle funktsionaalsuse näide on toodud joonisel 5.



Joonis 5. Andmebaasi programmeerimiskeskkond 0xDBE

Antud ettepanekud ei ole midagi uut ja innovaatilist, neid kuid paratamatult vaadates käesoleva uuringu tulemusi, peab kinnitama Browni [11] väidet 1983 aasta artiklist, et neid ettepanekuid

rakendatakse siiani väga vähe. Brown [11] uuris üle kolmekümne aasta tagasi veateadete olukorda toonases tarkvaras ja jõudis järeldusele, et need veateated on puudulikud. Paraku ei anna käesoleva uuringu tulemused alust öelda, et olukord oleks kardinaalselt paranenud.

Käesoleva töö piiranguks tuleb lugeda seda, et autor hindas veateateid ainuisikuliselt. Erinevad hindajad võivad anda veateadete headusele erinevaid hinnanguid. Seega oleks töö üheks edasiarenduseks täpsustada veateadete hindamise skaalat ning kaasata hindamisse suurem hulk kasutajaid. Osana kasutatavuse testimisest võiks laborikeskkonnas uurida, kuidas erinevad kasutajad reageerivad samale veateatele, kuhu nad ekraanil silmadega liiguvad ning püüda selle kaudu leida paremaid veateateid. Samuti saab tööd edasi arendada, võttes vaatluse alla veel suurema hulga lauseid ja andmebaasisüsteeme. Üheks võimalikuks töö edasiarenduse suunaks oleks realiseerida mingi tarkvara osana moodul, mis suudab häid ja kasutajale mõistetavaid veateateid väljastada ning toetab veateadete tõlkimist.

2.4 Kõige väheinformatiivsemate veateadete alternatiivid

Antud jaotises tuuakse välja kõige väheinformatiivsematele veateadetele inglisekeelsed alternatiivid. Kui uuringus esitatud mittekorrektsele SQL lausele kuvasid kõik andmebaasisüsteemid maksimaalseks hindeks 2, siis on sellele SQL lausele kuvatud veateade hinnatud autori poolt kõige väheinformatiivsemaks. Sellistele kriteeriumitele vastavad veateated 2, 3, 6, 15, 22, 29. Juhul, kui vähemalt ühe andmebaasisüsteemi veateade kolmest on parema hindega kui 2, siis saab öelda, et sellele veateatele on rahuldav alternatiiv juba olemas.

<table>
<tr><td colspan="3">2. Lauses puudub sulg</td></tr>
<tr><td colspan="3">Lause näide (PostgreSQL, Oracle, MS Access):
<code>INSERT INTO Tuba (toa_kood, kirjeldus) VALUES ('303', 'Suur tuba'</code></td></tr>
<tr><td>Oracle</td><td>ORA-00917: missing comma</td><td>0</td></tr>
<tr><td>PostgreSQL</td><td>ERROR: syntax error at end of input
LINE 1: ...T INTO Tuba (toa_kood, kirjeldus) VALUES ('303',
'Suur tuba'
^</td><td>2</td></tr>
<tr><td>MS Access</td><td>Syntax error in INSERT INTO statement</td><td>1</td></tr>
<tr><td>Alternatiiv</td><td colspan="2">Syntax Error: VALUES ('303', 'Suur tuba'</td></tr>
</table>

2. Lauses puudub sulg	
	Values specified in VALUES clause must be inside parenthesis HINT: VALUES ('303', 'Suur tuba')

3. Lauses on tühik tabeli nime ja veeru nime vahel (Tabel. veerg)		
Lause näide (PostgreSQL, Oracle, MS Access): `SELECT i. ostuhind FROM Inventar i WHERE inventari_kood = '22345678'`		
Oracle	Päring õnnestub korrektselt	-
PostgreSQL	Päring õnnestub korrektselt	-
MS Access	Invalid use of ’.’, ’!’, or ’()’. in query expression ’i. ostuhind’	0
Alternatiiv	Syntax Error: SELECT i. ostuhind No whitespaces are allowed between table i and column ostuhind.	

15. ORDER BY klausli kasutamine DELETE lauses		
Lause näide (Oracle, PostgreSQL, MS Access): `DELETE FROM Inventar_toas WHERE toa_kood = '303' ORDER BY inventari_kood`		
Oracle	ORA-00933: SQL command not properly ended	0
PostgreSQL	ERROR: syntax error at or near "ORDER" LINE 3: ORDER BY inventari_kood; ^	2
MS Access	Missing semicolon (;) at end of SQL statement	0
Alternatiiv	Semantical Error: ORDER BY not allowed in DELETE statement. HINT: ORDER BY inventari_kood.	

22. WHERE klauslis olev otsingutingimuses on veeru nimed jäänud korrektselt välja kirjutamata
Lause näide (Oracle, PostgreSQL, MS Access): `SELECT * FROM Inventar_toas WHERE inventar_toas_id = 1 OR 2`

22. WHERE klauslis olev otsingutingimuses on veeru nimed jäänud korrektselt välja kirjutamata		
Oracle	ORA-00920: invalid relational operator	1
PostgreSQL	ERROR: argument of OR must be type boolean, not type integer LINE 1: SELECT * FROM Inventar_toas WHERE inventar_toas_id = 1 OR 2 ^	2
MS Access	Päring õnnestub ebakorrektselt	0
Alternatiiv	Semantical Error: WHERE inventar_toas_id = 1 OR 2 Missing column reference in search condition HINT: WHERE Inventar_toas_id = 1 OR 2	

29. Tabeli loomisel pole kahe kirjelduse osa vahel koma		
Lause näide (Oracle, PostgreSQL, MS Access): `CREATE TABLE Tuba ( toa_kood CHAR(3) NOT NULL, kirjeldus VARCHAR(255) CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood) );`		
Oracle	ORA-00907: missing right parenthesis	0
PostgreSQL	ERROR: syntax error at or near "(" LINE 4: CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood) ^	0
MS Access	Syntax error in CREATE TABLE statement	1
Alternatiiv	Syntax Error: kirjeldus VARCHAR(255) CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood). Each column definition and table level constraint definition must be separated with comma. HINT: kirjeldus VARCHAR(255), CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood).	

3. Kokkuvõte

Töö eesmärgiks oli uurida hulka populaarseid SQL-andmebaasisüsteeme ja teha kindlaks, kui informatiivseid veateateid need esitavad erinevate SQL lausetes tehtud vigade puhul või lausete tulemusena tekkinud erandolukordade puhul. Töös võrreldi veateateid Oracle (Database 12c Enterprise Edition Release 12.1.0.1.0), PostgreSQL (9.3.0) ja Microsoft Access (2013) andmebaasisüsteemides.

Eksperimendi käigus 39 SQL lause täitmisel tekkinud vigade ning neile kuvatud veateadete põhjal saadud statistika tulemusena leiti uuritud andmebaasisüsteemide pingerida veateadete headuse seisukohalt. Hinnangud veateadetele andis autor ainuisikuliselt. Eksperimendi käigus leitud kõige väheinformatiivsetele veateadetele pakuti sisukamad alternatiivid. Samuti üldistati veateadete probleeme üle erinevate andmebaasisüsteemide ning tarkvaras üldiselt.

Töö tulemusena leiab autor, et kõige paremaid veateateid uuritud lausete puhul kuvas PostgreSQL andmebaasisüsteem, järgnes Oracle ja kõige halvemaid veateateid kuvas MS Access. Uuringust tuli välja, et veateadete informatiivsus on üldiselt väga kehv ning rakendustes ei oleks mõistlik kuvada kasutajale andmebaasisüsteemist pärit veateateid. Uuringust tulid väljad mitmed head ja halvad veateadete mustrid, mida ka töös lähemalt kirjeldatakse.

4. Summary

The aim of this work was to compare a set of popular SQL Database Management Systems and find out informativeness of error messages they produce to malformed SQL statements or exceptions caused by SQL statements. Oracle (Database 12c Enterprise Edition Release 12.1.0.1.0), PostgreSQL (9.3.0), and Microsoft Access (2013) database management systems (DBMSs) are used in this comparison.

In this work, Oracle, PostgreSQL, and MS Access DBMSs were used to implement logically identical databases with identical test data. After that, we compared error messages in case of 39 SQL statements. The author did it alone. As a result of this work, a ranking of examined DBMSs was constructed based on the quality of error messages. Alternative error messages were proposed for the most uninformative error messages that were identified in this experiment.

As a result of this work, author points out that the best error messages for studied SQL statements were provided by PostgreSQL, followed by Oracle, MS Access and DBMS. From studied error messages it can be pointed out that the overall quality of error messages is very poor. Good and bad practices of displaying error messages were gathered and described in this work. Likewise, a generalization of problems occurred in error messages was provided based on DBMSs used in this work and in software general.

Kasutatud kirjandus

1. Eessaar, E. (2014). Õppeaine Andmebaasid I õppematerjalid, Kommentaarid enne SQL kontrolltööd [WWW] http://maurus.ttu.ee/ained/IDU0220_2014/doc/12/Kommentaarid_enne_SQL_kontrolltood_IDU0220_2014_ver2.ppt (12.05.2015).

2. Eessaar, E. (2014). Õppeaine Andmebaasid II õppematerjalid, Teema 11 konspekt.

3. Tognazzini, T. (2014). *First Principles of Interaction Design* [WWW] http://www.asktog.com/basics/firstPrinciples.html (11.03.2015).

4. Bolton, M. (2003). *A Review of Error Messages* [WWW] http://www.developsense.com/essays/AReviewOfErrorMessages.html (16.03.2015).

5. DB-Engines Ranking (2015) [WWW] http://db-engines.com/en/ranking (16.03.2015).

6. Põlluste, K.K. (2014) *Hotelli infosüsteem - tubade haldamine.* Andmebaasid I (IDU0220) aineprojekt.

7. Põlluste, K.K., Menšenin J., Bystrov D. (2015). *Hotelli infosüsteem - tubade haldamine*. Andmebaasid II (IDU0230) aineprojekt, IDU0220 projekti edasiarendus.

8. Eessaar, E. *Vastuvõtuaegade IS* (2014). [WWW] http://maurus.ttu.ee/ained/IDU0230_2014/doc/12/Naidisprojekt_IDU0230_vastuvotuajad_ver3_19.pdf (13.12.2014).

9. Error Messages (Microsoft) - MSDN [WWW] https://msdn.microsoft.com/en-us/library/windows/desktop/dn742471%28v=vs.85%29.aspx (28.04.2015).

10. Vallaste e-teatmik - ingliskeelsete info- ja sidetehnoloogia terminite seletav sõnaraamat. [WWW] http://vallaste.ee/ (12.05.2015).

11. Brown P.J., *Error messages: the neglected area of the man/machine interface* (1983) [WWW] http://dl.acm.org/citation.cfm?id=358083 (17.05.2015).

12. Horning, J.J. *What the compiler should tell the user. In Bauer and Eickel (eds.), Compiler Construction, Springer-Verlag* (1974).

13. PostgreSQL 9.3 Documentation. [WWW] http://www.postgresql.org/docs/9.3/ (17.05.2015)

14. Oracle Database 12c Release 1 (12.1) Documentation [WWW] http://docs.oracle.com/database/121/index.htm (17.05.2015)

15. Access 2013 Desktop Database Reference [WWW] https://msdn.microsoft.com/en-us/library/office/dn142571.aspx (17.05.2015)

16. 0xDBE - Intelligent IDE for DBAs and SQL Developers by Jetbrains [WWW] https://www.jetbrains.com/dbe/ (18.05.2015)

Lisa 1

Järgnevalt tuuakse välja SQL laused, mida kasutati eksperimendi läbiviimiseks.

1. SELECT lauses (päringus) puudub FROM klausel.

```
SELECT * Tuba;
```

2. Lauses puudub sulg (nt alampäringu ümbert).

```
INSERT INTO Tuba (toa_kood, kirjeldus) VALUES ('303', 'Suur tuba';
```

3. Lauses on tühik tabeli nime ja veeru nime vahel (Tabel. veerg).

```
SELECT i. ostuhind FROM Inventar i WHERE inventari_kood = '22345678';
```

4. AND loogikaoperaatori asemel kasutatakse & (päringus kasutatakse sellise nimega operaatorit, mida süsteemis pole defineeritud).

```
SELECT * FROM Inventar_toas WHERE toa_kood = '303' & inventari_kood =
'12345678'
```

5. SELECT lauses kasutatakse funktsiooni, mida süsteemis pole defineeritud (nt Upper vs.Ucase).

Oracle

```
SELECT Ucase(t.kirjeldus) FROM Tuba t;
```

PostgreSQL

```
SELECT Ucase(t.kirjeldus) FROM Tuba t;
```

MS Access

```
SELECT Upper(t.kirjeldus) FROM Tuba t;
```

6. Võtmesõnade valesti kirjutamine.

```
SELECT * FROM Tuba WHRE toa_kood = '303';
```

7. Klauslid lauses vales järjekorras.

```
SELECT * WHERE Inventar FROM inventari_kood = '12345678';
```

8. Lauses kasutatakse sõne (string) tähistamiseks jutumärke

```
SELECT * FROM Inventar WHERE inventari_kood = "12345678";
```

9. FROM klauslis olevale alampäringule pole antud aliast.

```
SELECT * FROM (SELECT * FROM Inventar);
```

10. GROUP BY klausli puudumine lauses, kus see on nõutud.

```
SELECT count(toa_kood), kirjeldus FROM Tuba;
```

11. HAVING klauslis kasutatakse aliast

PostgreSQL ja Oracle

```
SELECT Inventar_toas.inventari_kood, toa_kood, alguse_aeg
AS algus
FROM Inventar_toas
INNER JOIN Inventar
ON Inventar_toas.inventari_kood = Inventar.inventari_kood
GROUP BY Inventar_toas.inventari_kood, Inventar_toas.toa_kood,
Inventar_toas.alguse_aeg
HAVING algus < CURRENT_TIMESTAMP;
```

Access

```
SELECT Inventar_toas.inventari_kood, toa_kood, alguse_aeg
AS algus
FROM Inventar_toas
INNER JOIN Inventar
ON Inventar_toas.inventari_kood = Inventar.inventari_kood
GROUP BY Inventar_toas.inventari_kood, Inventar_toas.toa_kood,
Inventar_toas.alguse_aeg
HAVING algus  < NOW();
```

12. WHERE klauslisse kirjutatakse tingimus, mis tuleks kirjutada HAVING klauslisse

```
SELECT Inventar.inventari_kood, ostuhind, kirjeldus, Count(*) AS
inventaride_arv
FROM Inventar INNER JOIN Inventar_toas ON
 Inventar.inventari_kood=Inventar_toas.inventari_kood
WHERE Count(*) > 1
GROUP BY Inventar.inventari_kood, ostuhind, kirjeldus;
```

13. Kokkuvõttefunktsiooni argumendiks alampäring

```
SELECT Max(SELECT ostuhind FROM Inventar) FROM Inventar;
```

14. Kokkuvõttefunktsiooni argumendiks on teise kokkuvõttefunktsiooni poole pöördumine

```
SELECT Max(Avg(ostuhind)) AS koige_kallim FROM Inventar;
```

15. ORDER BY klausli kasutamine DELETE lauses

```
DELETE FROM Inventar_toas WHERE toa_kood = '303' ORDER BY inventari_kood;
```

16. UNION päringus pole alampäringute tulemustel ühesugune struktuur

```
SELECT * FROM Tuba UNION SELECT * FROM Inventar
```

17. CREATE VIEW lauses puudub AS klausel

```
CREATE VIEW luksus_inventar SELECT kirjeldus, ostuhind FROM Inventar WHERE
ostuhind > 1000
```

18. Päring tabelist, mida andmebaasis pole.

```
SELECT * FROM Invetar
```

19. Päringus on vale veeru nimi SELECT klauslis

```
SELECT kirjelus FROM Inventar WHERE inventari_kood ='22345678'
```

20. Päringus on vale veeru nimi WHERE klauslis

```
SELECT * FROM Inventar WHERE kirjelus ='Vann suur'
```

21. Päringus on vale veeru nimi ORDER BY klauslis

```
SELECT count(toa_kood), kirjeldus FROM Tuba;
```

22. WHERE klauslis olev otsingutingimuses on veeru nimed jäänud korrektselt välja kirjutamata (nt ruumi_nr = 1 OR 2)

```
SELECT * FROM Inventar_toas WHERE inventar_toas_id = 1 OR 2
```

23. WHERE klausli tingimuses kasutatav arvu esitav literaal ei esita väärtust, mis kuulub veeru tüüpi (tüüp kui nime omav väärtuste hulk)

```
SELECT * FROM Inventar_toas WHERE inventar_toas_id = '1'
```

24. WHERE klausli tingimuses kasutatav kuupäevana esitav literaal ei esita väärtust, mis kuulub veeru tüüpi (tüüp kui nime omav väärtuste hulk)

```
SELECT * FROM Inventar_toas WHERE alguse_aeg = 2015-04-13
```

25. Alarmpäring tagastab rohkem kui ühe rea olukorras, kus see peaks tagastama maksimaalselt ühe rea.

```
SELECT inventari_kood FROM Inventar_toas
WHERE inventari_kood = (SELECT inventari_kood FROM Inventar)
```

26. Üritatakse luua tabel nimega, mis on juba olemas

```
CREATE TABLE Tuba (
      toa_kood CHAR(3) NOT NULL,
      kirjeldus VARCHAR(255),
      CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood)
);
```

27. Üritatakse kustutada tabelit, mida pole olemas

```
DROP TABLE Tubad
```

28. Kustutamine läbi JOIN klausli (pole selge, millisest tabelist peaks ridu kustutama)

```
DELETE FROM Inventar_toas INNER JOIN Inventar ON
Inventar_toas.inventari_kood=Inventar.inventari_kood WHERE toa_kood = '303'
```

29. Tabeli loomisel pole kahe kirjelduse osa vahel koma

```
CREATE TABLE Tuba (
      toa_kood CHAR(3) NOT NULL,
      kirjeldus VARCHAR(255)
      CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood)
);
```

30. Tabeli loomisel on veerule jäänud määramata andmetüüp

```
CREATE TABLE Tuba (
      toa_kood NOT NULL,
      kirjeldus VARCHAR(255),
      CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood)
);
```

31. Tabeli loomisel on identifikaatoris (nt tabeli nimes, veeru nimes, kitsenduse nimes) tühik (kui nimi on piiritlemata identifikaator)

```
CREATE TABLE Tuba (
      toa kood CHAR(3) NOT NULL,
      kirjeldus VARCHAR(255),
```

```
      CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood)
);
```

32. Välisvõtme deklareerimisel viidatakse tabelile, mida pole loodud

PostgreSQL ja Access

Vana kitsenduse eemaldamine

```
ALTER TABLE Inventar_Toas DROP CONSTRAINT FK_Inventar_Toas_toa_kood;
```

Uue kitsenduse lisamine

```
ALTER TABLE Inventar_Toas ADD CONSTRAINT FK_Inventar_Toas_toa_kood FOREIGN
KEY (toa_kood) REFERENCES Tubad (toa_kood) ON DELETE NO ACTION ON UPDATE
CASCADE;
```

Algseisu taastamine

```
ALTER TABLE Inventar_Toas DROP CONSTRAINT FK_Inventar_Toas_toa_kood;

ALTER TABLE Inventar_Toas ADD CONSTRAINT FK_Inventar_Toas_toa_kood FOREIGN
KEY (toa_kood) REFERENCES Tuba (toa_kood) ON DELETE NO ACTION ON UPDATE
CASCADE;
```

Oracle

Vana kitsenduse eemaldamine

```
ALTER TABLE Inventar_Toas DROP CONSTRAINT FK_Inventar_Toas_toa_kood;
```

Uue kitsenduse lisamine

```
ALTER TABLE Inventar_Toas ADD CONSTRAINT FK_Inventar_Toas_toa_kood FOREIGN
KEY (toa_kood) REFERENCES Tubad (toa_kood) INITIALLY DEFERRED DEFERRABLE;
```

Algseisu taastamine

```
ALTER TABLE Inventar_Toas DROP CONSTRAINT FK_Inventar_Toas_toa_kood;

ALTER TABLE Inventar_Toas ADD CONSTRAINT FK_Inventar_Toas_toa_kood FOREIGN
KEY (toa_kood) REFERENCES Tuba (toa_kood) INITIALLY DEFERRED DEFERRABLE;
```

33. Mittedeterministliku CHECK kitsenduse kirjeldamine

PostgreSQL

Vana kitsenduse eemaldamine

```
ALTER TABLE Inventar_Toas DROP CONSTRAINT CHK_Inventar_Toas_alguse_aeg;
```

Uue kitsenduse lisamine PostgreSQL

```
ALTER TABLE Inventar_Toas ADD CONSTRAINT CHK_Inventar_Toas_alguse_aeg CHECK
(alguse_aeg < CURRENT_TIMESTAMP);
```

Uue kitsenduse lisamine Access

```
ALTER TABLE Inventar_Toas ADD CONSTRAINT CHK_Inventar_Toas_alguse_aeg CHECK
(alguse_aeg < NOW());
```

Algseisu taastamine

```
ALTER TABLE Inventar_Toas DROP CONSTRAINT CHK_Inventar_Toas_alguse_aeg;
ALTER TABLE Inventar_Toas ADD CONSTRAINT CHK_Inventar_Toas_alguse_aeg CHECK
(alguse_aeg BETWEEN '2000-01-01 00:00:00' AND CURRENT_TIMESTAMP);
```

34. Kasutajal puudub õigus päringut teostada

PostgreSQL

```
DROP USER IF EXISTS vigade_haldaja;
CREATE USER vigade_haldaja WITH PASSWORD 'DSA18JL';
GRANT CONNECT ON DATABASE errors TO vigade_haldaja;
GRANT USAGE ON SCHEMA public TO vigade_haldaja;
GRANT SELECT ON Tuba TO vigade_haldaja;
INSERT INTO Tuba (toa_kood, kirjeldus) VALUES ('213', 'Merevaatega tuba');
```

Oracle puhul testiti sellise kasutajaga, kellel oli küll õigus andmebaasiga ühenduda, aga ühtki lauset ei olnud lubatud käivitada. Seega kasutasin veateate saamiseks tabeli loomise lauset.

```
CREATE TABLE Tuba (
      toa_kood VARCHAR2(3) NOT NULL,
      kirjeldus VARCHAR2(255),
      CONSTRAINT PK_Tuba PRIMARY KEY (toa_kood)
);
```

35. FOREIGN KEY vastu eksimine

```
INSERT INTO Inventar_toas ( inventari_kood, toa_kood ) VALUES ( '12342678', '303' );
```

36. PRIMARY KEY vastu eksimine

```
INSERT INTO Tuba (toa_kood, kirjeldus) VALUES ('303', 'Suur tuba');
```

37. NOT NULL vastu eksimine

```
INSERT INTO Tuba (kirjeldus) VALUES ('Suur tuba');
```

38. CHECK kitsenduse vastu eksimine

```
INSERT INTO Inventar (inventari_kood, ostuhind, kirjeldus) VALUES ('abcd5678', '292.29', 'Diivan Lux');
```

39. UNIQUE kitsenduse vastu eksimine

PostgreSQL

```
INSERT INTO Inventar_toas (inventari_kood, toa_kood, alguse_aeg) VALUES ('12345678', '303', '2015-04-13 18:09:19.743733');
```

Oracle

```
INSERT INTO Inventar_toas (inventari_kood, toa_kood, alguse_aeg) VALUES ('12345678', '303', '16-MAR-15 10.37.15.823731 PM');
```

Access

```
INSERT INTO Inventar_toas (inventari_kood, toa_kood, alguse_aeg) VALUES ('12345678', '303', '27.04.2015 22:28:03');
```