TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Nikita Rudkov 193396IACB

Otsingumoodul EuroTeQi andmebaasist kompetentside katteks

Bakalaureusetöö

Juhendaja: Vladimir Viies
Dotsent

,

Tallinn 2023

Autorideklaratsioon

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

Autor: Nikita Rudkov

03.01.2023

Annotatsioon

Lõputöös on uuritud erinevat tüüpi andmebaasidest otsingu teostamist. Andmebaaside jaoks kasutati andmetena EuroTeQ konsortsiumiülikoolide õppeaineid ja nendes arendatavaid kompetentside. Esimeses osas pakub nii andmebaasitüüpe kui ka muid andmebaasihaldussüsteeme. Mis on nende juhtmõte, plussid, miinused ja kus neid rakendatakse. Teises osas räägib ülikoolide EuroTeQ konsortsiumist ja sellest, milliseid kompetentse tulevaste inseneride jaoks nad oma uurimis käigus välja tõid. Kolmandas osas kirjeldatakse nii erinevat tüüpi andmebaaside loomise protsessi ja ka otsingu teostamist erinevat tüüpi andmebaasidega prototüüpides.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 44 leheküljel, 3 peatükki, 28 joonist, 8 tabelit.

Abstract

In the thesis, the implementation of search from different types of databases is investigated. The subjects of the EuroTeQ consortium universities and the competencies they develop were used as data for the databases. The first part provides database types as well as other database management systems. What is their basic principle, pros and cons, and where they are applied. The second part tells about the EuroTeQ consortium of universities and what competencies for future engineers they brought out in the course of their research. The third part describes the process of creating different types of databases as well as the implementation of search in prototypes with different types of databases.

The thesis is in Estonian and contains 44 pages of text, 3 chapters, 28 figures, 8 tables.

Lühendite ja mõistete loetelu

1NF	First Normal Form,Esimene normaalvorm
2NF	Second Normal Form,Teine normaalvorm
3NF	Third Normal Form,Kolmas normaalvorm
4NF	Fourth Normal Form,Neljas normaalvorm
5NF	Fifth Normal Form,Viies normaalvorm
6NF	Sixth Normal Form,Kuues normaalvorm
ACID	Atomicity, Consistency, Isolation, Durability;Aatomilisus, Järjepidevus, Isoleeritus ja Vastupidavus
API	Application Programming Interface,Rakenduse programmeerimisliides
BCNF	Boyce-Codd Normal Form,Boyce-Coddi normaalvorm
CSV	Comma-Separated Values,Komaga eraldatud väärtused
DKNF	Domain-key normal form,Domeenivõtme normaalvorm
JSON	JavaScript Object Notation,JavaScripti objekti märkimine
NoSQL	Not only SQL,Mitte ainult SQL
NVRAM	Non Volatile Random Access Memory,Mittelenduv muutmälu
SQL	Structured Query Language,struktureeritudpäringute keel
UNF	Un-Normalised form,Normaliseerimata vorm
VBA	Visual Basic for Applications,Visual Basic rakendustele
XML	Extensible Markup Language,Laiendatav märgistuskeel

Sisukord

Sissejuhatus 9
1.1. Relatsiooni andmebaas (*Relational database*) 10
1.1.1. Seosed tabelite vahel 13
1.1.2. Normaliseerimine 14
1.2. NoSQL andmebaas 17
1.2.1. Tüüpid NoSQL andmebaasi 20
1.3. Teised andmebassid häldussüsteemid 23
2. Vajalikud kompetentsid EuroTeQ 26
2.1. EuroTeQ 26
2.2. Tulevikukompetentsi 27
2.2.1. Sotsiaal-kommunikatiivsed kompetentsid 27
2.2.2. Isiklikud kompetentsid 28
2.2.3. Tehnilised-metodoloogilised kompetentsid 29
3. Rakendus “kompetent” prototüüpid 30
3.1. Rakendus relatsiooni andmebaasiga 31
3.2. Rakendus NoSQL andmebaasiga 34
Kokkuvõte 38
Kasutatud kirjandus 39
Lisa 1 Andmed andmebaasi 41

Jooniste loetelu

Joonis 1. Relatsiooniandmebaasi näiteks 12
Joonis 2. Seos üks-ühele näiteks..... 13
Joonis 3. Seos üks-mitmele näiteks 14
Joonis 4. Seos mitu-mitmele näiteks 14
Joonis 5. Erinevus relatsioonilise ja NoSQL andmebaasi vahel 18
Joonis 6. Tüübid NoSQL andmebaas 19
Joonis 7. Graafikuandmebaasi näiteks 21
Joonis 8. Dokumenti andmebaasi näiteks..... 22
Joonis 9. Võtmeväärtuse andmebassi näiteks..... 22
Joonis 10. Kolonn-perekonna andmebaasi näiteks..... 23
Joonis 11. EuroTeQ skeem..... 27
Joonis 12. Otsingu tulemus..... 30
Joonis 13. Menüü, juhul kui ei valitud kompetentsi..... 30
Joonis 14. Kujundus režiim tabeli Ulikoolid..... 31
Joonis 15. Kujundus režiim tabeli Ained..... 32
Joonis 16. Kujundus režiimtabeli Seotus..... 32
Joonis 17. Seosed tabelite vahel 32
Joonis 18. Taotlus andmebassi 32
Joonis 19. Osa koodist rakenduse relatsiooniandmebaasiga käivitamisel..... 33
Joonis 20. Osa koodist rakenduse relatsiooniandmebaasiga nupu klõpsamisel 33
Joonis 21. Struktuurskeem rakenduse relatsiooni andmebaasiga..... 34
Joonis 22. Käsud kollektsiooni loomiseks..... 34
Joonis 23. Osa käsust, et lisada dokumentide kollektsioonis 35
Joonis 24. Osa dokumentidest kollektsioonis..... 35
Joonis 25. Osa koodist rakenduse NoSQL andmebaasiga käivitamisel 36
Joonis 26. Osa koodist rakenduse NoSQL andmebaasiga, et lisa andmeid tabelis 36
Joonis 27. Osa koodist rakenduse NoSQL andmebaasiga nupu klõpsamisel..... 36
Joonis 28. Struktuurskeem rakenduse NoSQL andmebaasiga 37

Tabelite loetelu

Tabel 1. Näiteks normaliseerimata vorm 15
Tabel 2. Näiteks esimene normaalvorm 15
Tabel 3. Näiteks teine normaalvorm 16
Tabel 4. Tabel inimeste kolmandal normaalvormil 16
Tabel 5. Tabel erialade kolmandal normaalvormil 16
Tabel 6. Tabel õpilaste 16
Tabel 7. Tabel ainete 17
Tabel 8. Suhtlemine õpilaste ja õppeainete vahel 17

Sissejuhatus

Üliõpilase palkamisel huvitab tööandjat pigem mitte see, milliseid aineid ta läbis, vaid milliseid omadusi ja oskusi õpilane koolitusel omandas. Õppeainete nimed ise ütlevad tööandjale üliõpilase kohta vähe. Käesoleva töö autor soovis teha andmebaasi rakendust mis aitas luua õppekava vajalike omaduste ja oskuste omandamine valitud ettevõtte jaoks, milles oleksid õppekava ained ning milliseid oskusi ja omadusi see aine arendab. Eesmärk on tuua selleks prioriteetsed õppeained õpilasteni, kes otsustavad töötada enda valitud ettevõttes.

Lõputöö esimeses osas analüüsin, mis on andmebaas, millised tüübid on olemas, millised need on, millised on nende eelised ja puudused ning millistes valdkondades neid kasutatakse.

Töö teises osas räägib autor, milliseid kompetentse tööandjad võivad tahta lähtudes EuroTeQ Professionali projekt, kust selle sai ja mis pärast töötlemist juhtus.

Kolmandas osas tuuakse välja kaks prototüüpi erinevate andmebaasitüüpidega põhifunktsionaalsus ning antakse ülevaade sellest, kuidas otsing realiseeriti, EuroTeQ samuti antakse ülevaade prototüüpimise käigus loodud andmebaasist.

1. Andmebaas

Andmebaas on struktureeritud teabe või andmete järjestatud kogum, mis tavaliselt salvestatakse elektrooniliselt arvutisüsteemi. Andmebaasi haldab tavaliselt andmebaasihaldussüsteem.

Kaasaegsete andmebaaside levinumate tüüpide andmed salvestatakse tavaliselt ridade ja veergude kujul, mis moodustavad tabeli. Neid andmeid saab hõlpsasti hallata, muuta, ajakohastada, jälgida ja korrastada. Andmebaase on mitut tüüpi, millel on oma eelised ja puudused ning mida kasutatakse teatud ülesannete jaoks. [1]

1.1. Relatsiooni andmebaas (*Relational database*)

See on tüüp, milles teavet salvestatakse tabelite kujul, mis koosnevad veergudest ja ridadest. Iga sellise andmebaasi tabelis sisalduv rida on kordumatu identifikaatoriga kirje, mida nimetatakse võtmeks. Tabeli veergudel on andmeatribuudid ja iga kirje sisaldab tavaliselt iga atribuudi väärtust, mis muudab andmeelementide vaheliste seoste loomise lihtsaks. Relatsiooniandmebaasi on mugav kasutada, kui on vaja andmeelemente omavahel siduda ning neid turvaliselt ja usaldusväärselt hallata, kuna relatsioonimudelis eeldab see loogilist andmestruktuuri: tabeleid, vaateid ja indekseid.[2] Relatsiooni andmebaasid skaleeruvad vertikaalselt. See tähendab, et andmemahu suurenemisega suurendavad need arvutusprotsessorite võimsust ja serveri koormust.[8]

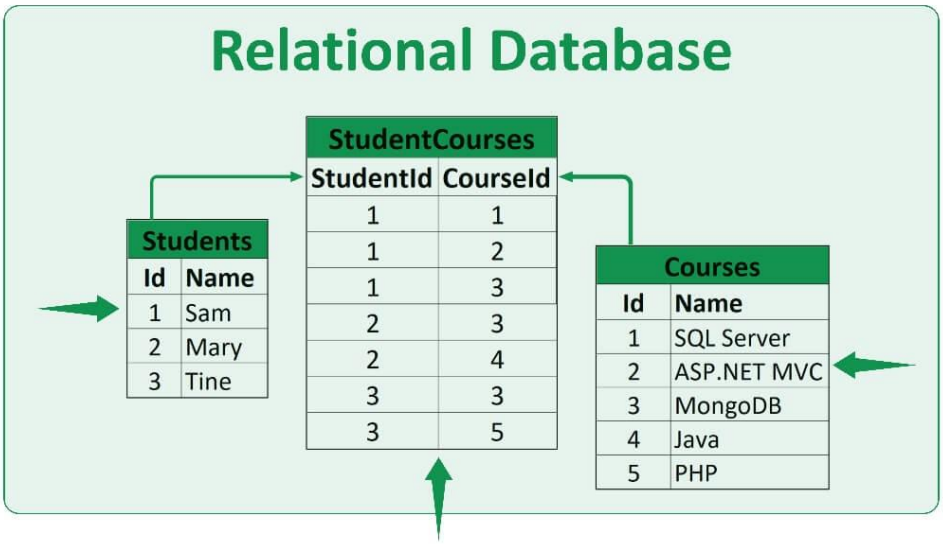
Loogiline struktuur erineb kirje füüsilisest struktuurist selle poolest, et võimaldab töötada füüsilise salvestussüsteemiga ja seega mitte muuta tabelites olevaid andmeid. Füüsilised ja loogilised struktuurid erinevad toimingute poolest, mis on täpselt määratletud toimingud andmete ja andmebaasi struktuuridega. Tõeväärtuse toimingud võimaldavad rakendustel määratleda nõuded nõutavale sisule. Füüsilised toimingud määravad, kuidas andmetele juurde pääsetakse ja ülesandeid täidetakse. Selleks, et relatsiooni andmebaasid oleksid täpsemad ja ligipääsetavamad, on vaja rakendada normaliseerimis meetodeid. Relatsioonimudel säilitab kõige tõhusamalt andmete terviklikkuse rakenduste ja andmebaasi eksemplaride vahel. Relatsiooniandmebaasid sobivad suurepäraselt andmete terviklikkuse säilitamiseks mitmes andmebaasi eksemplaris korraga.

Neli olulist omadust määratlevad relatsiooniandmebaasi tehingud: aatomilisus, järjepidevus, isoleeritus ja vastupidavus – mida tavaliselt nimetatakse ACID-ks.

- Aatomilisus määratleb kõik elemendid, mis moodustavad täieliku andmebaasi tehingu.
- Järjepidevus määratleb reeglid andmepunktide õiges olekus hoidmiseks pärast tehingut.
- Isolatsioon hoiab segaduse vältimiseks tehingu mõju teistele nähtamatuks kuni selle sooritamiseni.
- Vastupidavus tagab, et andmete muudatused muutuvad pärast tehingu sooritamist püsivaks. [7]

Relatsiooniandmebaasid luuakse andmebaasihaldussüsteemis nimega MySQL. SQL on kõige levinum standardiseeritud keel, mida kasutatakse andmebaasidele juurdepääsuks. SQL põhineb relatsioonialgebra algoritmidel ja selgel matemaatilisel struktuuril. ANSI SQL standardit toetavad kõik populaarsed relatsiooniandmebaasi mootorid. Mõned tuumad sisaldavad ka ANSI SQL-standardi laiendusi, mis toetavad nende tuumade spetsiifilisi funktsioone. SQL-i kasutatakse andmeridade lisamiseks, värskendamiseks ja kustutamiseks, tehingute töötlemise ja analüütiliste rakenduste andmekogumite toomiseks ning andmebaasi kõigi aspektide haldamiseks.[6][9] Microsoft Access on ka relatsiooniline andmebaasihaldussüsteem loodud Microsoft Corporationi, millel on lai valik funktsioone, sealhulgas lingitud päringud, lingid välistele tabelitele ja andmebaasidele. Tänu sisseehitatud VBA keelele saate kirjutada rakendusi, mis töötavad andmebaasidega Accessis.[16]

Relatsiooni andmebaasid on nõudlikud erinevates organisatsioonides, kus neid kasutatakse raamatupidamiseks, laos olevate kaupade arvu jälgimiseks, tarnijate, partnerite ja klientide personali andmete haldamiseks. Samuti on nad leidnud oma rakenduse Interneti-tehnoloogiates registreeritud kasutajakontode salvestamise korraldamiseks, sõnumitekstide salvestamiseks Interneti erinevates piirkondades, näiteks foorumites, suhtlusvõrgustikes ja uudistevoogudes.[28]



Joonis 1. Relatsiooniandmebaasi näiteks

Plussid:

- Relatsioonilise andmebaasi mudel ei nõua keerulisi päringuid, kuna sellel pole päringute töötlemist ega struktureerimist, nii et andmete käsitlemiseks piisab lihtsatest SQL-päringutest.
- Kasutajad saavad hõlpsasti juurde pääseda oma nõutavale teabele või seda hankida mõne sekundi jooksul, ilma andmebaasi keerukusest järele andmata. SQL kasutatakse keerukate päringute täitmiseks.
- Relatsiooniandmebaasid on rangelt määratletud ja hästi organiseeritud, nii et andmed ei dubleerita. Relatsiooniandmebaasid on nende struktuuri tõttu täpsed, ilma andmete dubleerimiseta.
- Relatsiooniandmebaasi haldussüsteeme kasutatakse laialdaselt ka andmete terviklikkuse tagamiseks, kuna need tagavad järjepidevuse kõigis tabelites. Andmete terviklikkus tagab sellised funktsioonid nagu täpsus ja kasutuslihtsus.
- Andmed on turvalised, kuna relatsiooniandmebaasi haldussüsteem võimaldab andmetele otse juurdepääsu ainult volitatud kasutajatel. Ükski volitamata kasutaja ei pääse teabele juurde.
- Andmebaasi juurde pääseb mitu kasutajat, et saada teavet korraga ja isegi siis, kui andmeid värskendatakse.
- Andmebaasi normaliseerimine tagab ka selle, et relatsiooniandmebaasi struktuur pole erinevus ja seda saab täpselt manipuleerida.

Miinused:

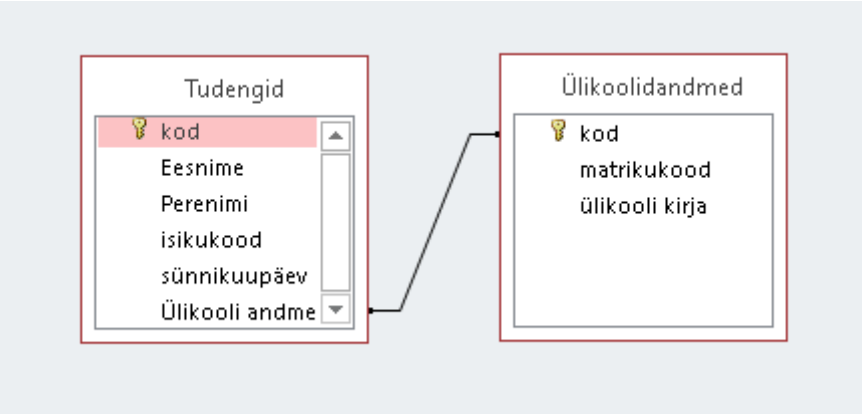
- Relatsiooniandmebaasi haldamine muutub aja jooksul andmete suurenemise tõttu keeruliseks.

- Relatsiooniandmebaas koosneb ridadest ja veergudest, mis nõuab palju füüsilist mälu, kuna iga sooritatav toiming sõltub eraldi salvestusruumist. Füüsilise mälu nõuded võivad suureneda koos andmete arvu suurenemisega.
- Kui kasutada relatsiooniandmebaasi mitmes serveris, muutub selle struktuur ja muutub raskesti käsitletavaks, eriti kui andmete hulk on suur. Tänu sellele ei ole andmed erinevatel füüsilistel salvestusserveritel skaleeritavad. Lõppkokkuvõttes mõjutab see selle jõudlust, st andmete ja laadimisaja puudumine jne. Kuna andmebaas muutub suuremaks või jaotub suurema hulga serveritega, on sellel negatiivsed tagajärjed, nagu latentsus- ja kättesaadavusprobleemid, mis mõjutavad üldist jõudlust.
- Relatsiooniandmebaasid suudavad salvestada andmeid ainult tabeli kujul, mis muudab keerukate objektidevaheliste suhete esitamise keeruliseks. See on probleem, sest paljud rakendused nõuavad rohkem kui ühte tabelit, et salvestada kõik vajalikud andmed, mida nende rakendusloogika nõuab. [11]

1.1.1. Seosed tabelite vahel

Seosed kahe või enama tabeli vahel luuakse primaar- ja võõrvõtmete abil. Primaarvõti on väli, mida kasutatakse iga tabeli kirje unikaalseks tuvastamiseks. Võõrvõti on ühe tabeli väli või mitu välja, mis viitavad teise tabeli primaarvõtmele. Tabelite vahel on kolme tüüpi seoseid: üks-ühele, üks-mitmele, mitu-mitmele.

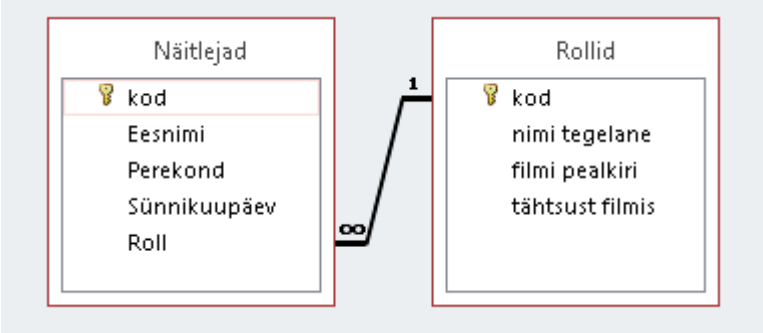
Üks-ühele tähendab, et ühe tabeli rea saab siduda teise tabeli ühe reaga. Näitena saab luua tabeli "tudengid", kus on õpilase täisnimi ja sünniaeg ning teine tabel "Ülikoolidandmed" sisaldab tema maatrikskoodi ja ülikooli kirja.



Joonis 2. Seos üks-ühele näiteks

Sellist ühendust peetakse üks-ühele, kuna igal üliõpilasel on oma unikaalne maatrikskood ja e-post, samuti kuuluvad iga ülikooli andmed ainult ühele üliõpilasele.

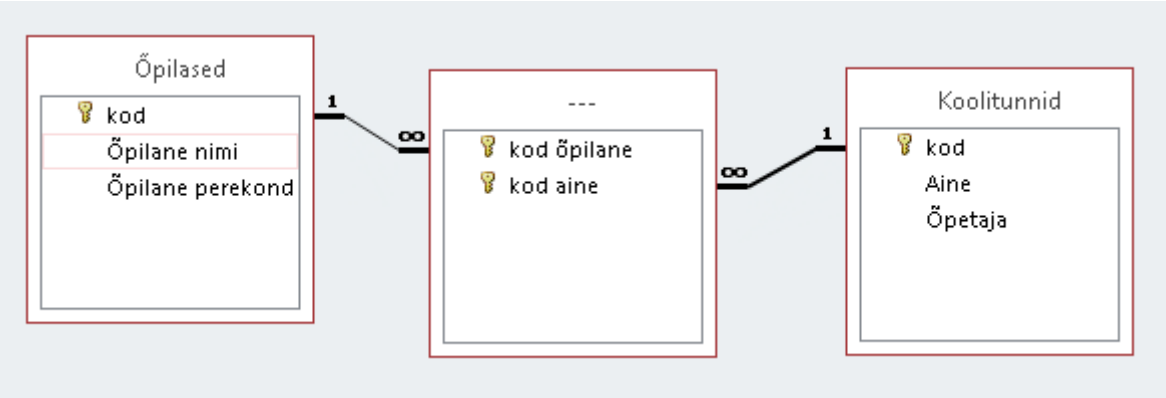
Üks-mitmele tähendab, et ühe tabeli rea saab siduda teise tabeli paljude ridadega. Näitena võib tuua tabeli "Näitlejad", kus asuvad näitleja isikuandmed ja "Rollid", kus on tegelase nimi ja millises filmis see tegelane oli.



Joonis 3. Seos üks-mitmele näiteks

Seda suhet peetakse üks-mitmele, kuna näitleja võib erinevates filmides mängida rohkem kui üht tegelast, kuid kogu filmi jooksul saab tegelast mängida ainult üks inimene.

Mitu-mitmele tähendab et ühe tabeli rida võib olla seotud paljude teise tabeli ridadega ja vastupidi. Sellist ühendust aga otse luua ei saa. Seda tehakse läbi kolmanda tabeli. Mitu-mitmele suhte näiteks on tabelid "Õpilased" ja "Kooliained" vastavalt sellele, mitu õpilast saab ühes tunnis olla ja üks õpilane käib mitmes tunnis. [17]



Joonis 4. Seos mitu-mitmele näiteks

1.1.2. Normaliseerimine

Normaliseerimine on andmebaasi teisendustehnika, mis minimeerib andmete liiasust ja kõrvaldab liiasuse soovimatud tagajärjed, nagu erinevad kõrvalekalded, jõudluse

halvenemine, ning muudab andmehalduse vähem paindlikuks ja mugavamaks. Üleliigsuse kõrvaldamisel peate jagama suure tabeli väiksemateks ja ühendama need suhete abil. Normaliseerimata kujul võivad tabeli veerud ja väärtused salvestada teavet kahe või enama olemi kohta. Normaliseerimisel on 9 vormi, mis sisaldavad oma reegleid ja kriteeriume. Viib tabeli järgmisele vormile ja järgides reegleid, on võimalik vältida erinevaid anomaaliaid ja viia tabel miinimumi liiasuseni. [18][19]

Normaliseerimata vorm (UNF): Sellel kujul on tabelis palju kõrvalekaldeid ja dubleerivaid väärtusi.[29]

Tabel 1. Näiteks normaliseerimata vorm

Inimine	**E-mail**
Kentaro Miura	Isiklik post berserk1989@gmail.com; Töö post Miura.Kentaro@gmail.com
Edward Elric	Ed.ferrum@gmail.com
Edward Elric	Ed.ferrum@gmail.com
Thorkell Long	IamBig@gmail.com; Thorkell.Long@gmail.com

Esimene normaalvorm (1NF): Sellel kujul ei tohi tabelis olla topeltridu, igal lahtril peab olema mitteliitväärtus ja iga kirje peab olema kordumatu.[30]

Tabel 2. Näiteks esimene normaalvorm

Inimene	**E-mail**	**Tüpp**
Kentaro Miura	berserk1989@gmail.com	Isiklik post
Kentaro Miura	Miura.Kentaro@gmail.com	Töö post
Edward Elric	Ed.ferrum@gmail.com	
Thorkell Long	IamBig@gmail.com	
Thorkell Long	Thorkell.Long@gmail.com	

Teine normaalvorm (2NF): Sellel kujul peab tabel olema esimesel kujul ja sellel peab olema üks veerg primaarvõtmega.[31]

Tabel 3. Näiteks teine normaalvorm

Number	**Inimene**	**Eriala**	**Kirjeldus**
1	Kentaro Miura	Kunstnik	Joonistab Jaapani koomikseid
2	Edward Elric	Keemik	Uurib aine koostist ja selle omadusi
3	Thorkell Long	Ehitaja	Ehitab hooneid

Kolmas normaalvorm (3NF): Sellel kujul peab tabel olema teisel kujul ja sellel ei tohi olla transitiivset sõltuvust, see tähendab, et võtmeta veerud ei tohi sõltuda muude mittevõtmeteta veergude väärtustest.[32]

Tabeli 3 tõlkimiseks kolmandale vormile tuleb see jagada kaheks tabelisse, kuna elukutse kirjeldus ei ole seotud inimese endaga.

Tabel 4. Tabel inimeste kolmandal normaalvormil

Number	**Inimene**	**Eriala**
1	Kentaro Miura	1
2	Edward Elric	2
3	Thorkell Long	3

Tabel 5. Tabel erialade kolmandal normaalvormil

Number eriala	**Eriala**	**Kirjeldus**
1	Kunstnik	Joonistab Jaapani koomikseid
2	Keemik	Uurib aine koostist ja selle omadusi
3	Ehitaja	Ehitab hooneid

Boyce-Coddi normaalvorm (BCNF): Sellel kujul peab tabel olema kolmandal kujul ja liitvõtme võtmeatribuudid ei tohi sõltuda võtmeatribuutidest.[33]

Tabel 6. Tabel õpilaste

Number õpilane	**Õpilane täisnimi**
1	Kentaro Miura
2	Edward Elric
3	Thorkell Long

Tabel 7. Tabel ainete

Number aine	**Aine**	**Õpetaja**
1	Matemaatika	E.Vasiljeva
2	Keemia	A.Kuld
3	Kehalise kasvatus	K.Miyamoto

Tabel 8. Suhtlemine õpilaste ja õppeainete vahel

Number õpilane	**Number aine**
1	1
1	3
2	1
2	2
3	3

Neljas normaalvorm (4NF): Sellel kujul peab tabel olema kolmandal kujul (kui primaarvõti on lihtne) või Boyce-Coddi kujul ning sellel tabelis ei pea olema mitmeväärtuslikud sõltuvused.[34]

Viies normaalvorm (5NF): Sellel kujul peab tabel olema neljandal kujul ja selles tabelis iga mittetriviaalne seossõltuvus selles on määratud selle seose potentsiaalse võtmega [35]

Domeenivõtme normaalvorm (DKNF): Sellel kujul peab tabel olema viiendal kujul ja et iga tabelile kehtestatud piirang on loogiline tagajärg teatud väärtuse atribuudile kehtestatud piirangutele ja piirangutele, mis on kehtestatud sellele atribuudile, mis näivad olevat kandidaatvõti.[36]

Kuues normaalvorm (6NF): Sellisel kujul olevat tabelit ei saa ilma kadudeta edasi lagundada.[37]

1.2. NoSQL andmebaas

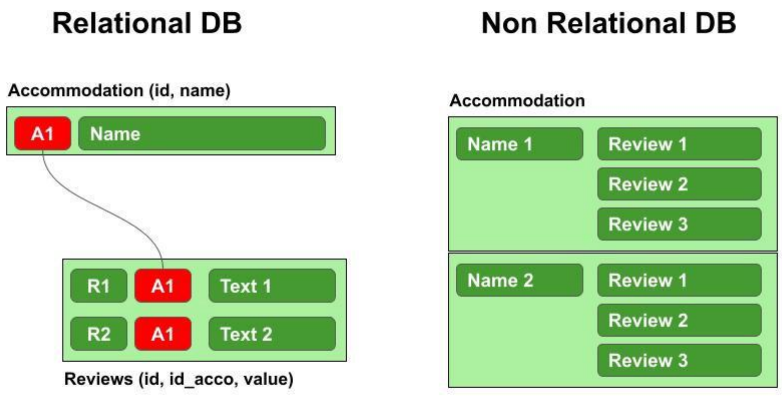
NoSQL-andmebaase nimetatakse ka mitterelatsioonilisteks andmebaasideks, mis salvestavad andmeid muus vormingus kui relatsioonitabelid, kuid NoSQL-andmebaasid suudavad käsitleda tohutul hulgal kiiresti muutuvaid struktureerimata andmeid. NoSQL-i

andmebaasidest saab aga päringuid teha idiomaatiliste keelte API-de, deklaratiivsete struktureeritud päringukeelte ja näidispäringukeelte abil. NoSQL-i andmebaasi peamine eelis on skaleeritavus ja paindlikud skeemid. Kui relatsiooni andmebaasid skaleeruvad vertikaalselt, siis NoSQL-i andmebaasid horisontaalselt. See tähendab, et andmete suurenedes lisatakse täiendavaid andmebaase või jaotatakse andmebaas väiksematele sõlmedele.

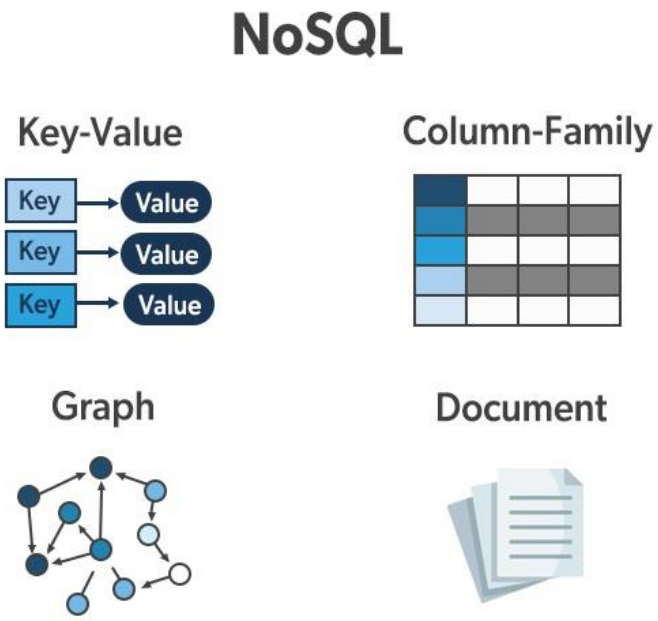
Relatsioonilise andmebaasi haldussüsteem on aluseks SQL-keelele, mis võimaldab kasutajatel kõrgelt struktureeritud tabelites andmetele juurde pääseda ja neid töödelda. See on andmebaasisüsteemide, nagu MS SQL Server, IBM DB2, Oracle ja MySQL, aluseks olev mudel. Kuid NoSQL-i andmebaaside puhul võib andmetele juurdepääsu süntaks erinevates andmebaasides olla erinev.

Erinevalt relatsioonilistest andmebaasihaldussüsteemidest saavad NoSQL-i andmebaasid andmeid salvestada ilma eelnevalt skeemi määratlemata, võimaldades teil kiiresti navigeerida ja itereerida, määratledes andmemudeli töö käigus. See lähenemisviis võib olla kasulik konkreetsete ärinõuete nt graafikud, veerud, dokumendid täitmiseks või võtmeväärtuste hoidlana kasutamiseks.

NoSQL-i andmebaase on sõltuvalt nende andmemudelist erinevat tüüpi. Peamised tüübid on dokument, võtmeväärtus, kolonn-perekond ja graafik.



Joonis 5. Erinevus relatsioonilise ja NoSQL andmebaasi vahel



Joonis 6. Tüübid NoSQL andmebaas

Plussid:

NoSQL-i andmebaasid on väga paindlikud, kuna need võivad salvestada ja kombineerida mis tahes tüüpi andmeid, nii struktureeritud kui ka struktureerimata, erineva19

- lt relatsioonilistest andmebaasidest, mis suudavad andmeid salvestada ainult struktureeritud viisil.
- NoSQL-i andmebaasid võimaldavad teil skeemi dünaamiliselt värskendada, et see muutuvate nõuetega muutuks, tagades samal ajal, et see ei põhjusta teie rakendusele katkestusi ega seisakuid.
- Selle asemel, et suurendada serverite lisamisega, saavad NoSQL-i andmebaasid standardse riistvaraga horisontaalselt skaleerida. Scale-out võimaldab NoSQL-i andmebaasidel saada suuremaks ja võimsamaks, nii et need on eelistatud valik andmekogumite kasvatamiseks.
- NoSQL-i andmebaasid kopeerivad andmeid automaatselt mitme serveri, andmekeskuse või pilveressursside vahel. See omakorda vähendab kasutajate latentsust olenemata nende asukohast. See funktsioon aitab vähendada ka andmebaasihalduse koormust, mis vabastab aega ja võimaldab keskenduda muudele prioriteetidele.

- NoSQL-i andmebaasid on mõeldud hajutatud andmesalvedele, mis nõuavad suurt salvestusruumi. Just see teeb NoSQL-ist ideaalse valiku suurandmete ja reaalajas veebirakenduste jaoks.[8]

Miinused:

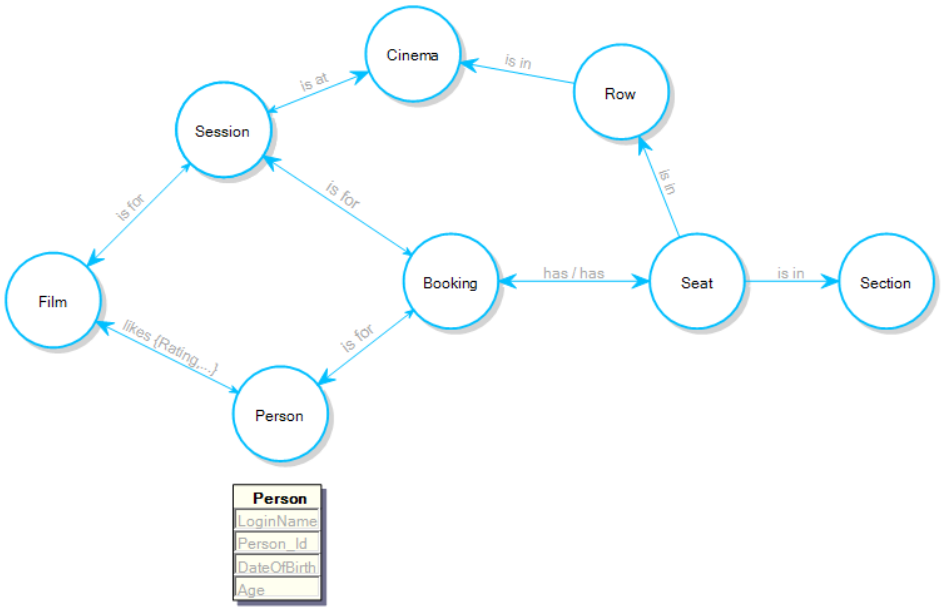
- Puudub standard, mis määratleks NoSQL-i andmebaaside reeglid ja rollid. NoSQL-i andmebaaside kujundus- ja päringukeeled on erinevate NoSQL-toodete puhul väga erinevad – palju laiemalt kui traditsiooniliste SQL-andmebaaside puhul.
- Varukoopiad on NoSQL-i andmebaaside puuduseks. Kuigi mõned NoSQL-i andmebaasid, nagu MongoDB, pakuvad varundamiseks mõningaid tööriistu, pole need tööriistad piisavalt küpsed, et tagada andmete täielik varunduslahendus.
- NoSQL seab esikohale skaleeritavuse ja jõudluse, kuid kui rääkida andmete järjepidevusest, siis NoSQL ei võta palju arvesse, nii et see muudab selle relatsiooniandmebaasiga võrreldes vähe ebaturvaliseks, nt NoSQL-i andmebaasides, kui sisestate sama andmekomplekti uuesti, see võtab seda ilma vigadeta, samas kui relatsiooniandmebaasid tagavad, et andmebaasidesse ei satuks dubleerivaid ridu. [12]

1.2.1. Tüüpid NoSQL andmebaasi

Graafiku andmebaas (*Graph database*) on tüüpi andmebaas, mis salvestab andmeid sõlmedevaheliste linkide abil. Sõlmed on graafiku andmepunktid, mis on andmetega ühendatud servade kaudu ja serva kasutatakse andmete vaheliste suhete salvestamiseks.[2] Serval on alati algussõlm, lõppsõlm, tüüp ja suund. Servad saavad kirjeldada vanemate ja laste suhteid, toiminguid, omandiõigust jne.[38]

Need andmebaasid on väga paindlikud, mistõttu on andmete linkimine lihtne, kuid neil on halb skaleeritavus.[13] Graafikandmebaasid sobivad sotsiaalvõrgustike loomiseks ja saavad hakkama ka finantstehingutega.[3][38]

Näiteks: Infinite Graph, TITAN, Sparksee, GraphBase, ArangoDB, Neo4J.[14]



Joonis 7. Graafikuandmebaasi näiteks

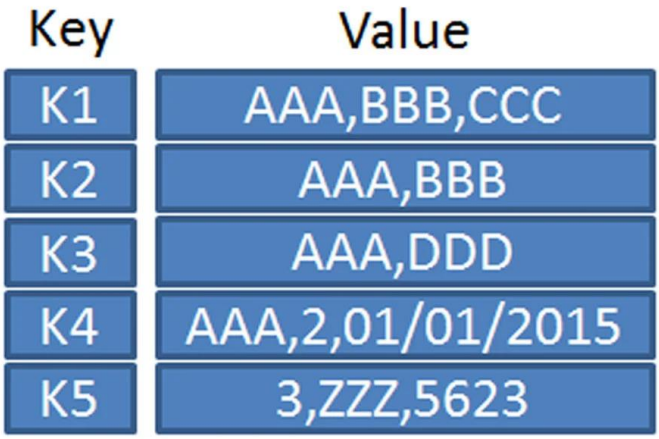
Dokumenti andmebaas (*Document database*) on andmetüüp, mis salvestab andmed JSON- või XML-dokumentides. Dokumente hoitakse hoidlates ja need ei vaja skeeme. Kui dokumente lisatakse hoidlatesse, sisestatakse need andmebaasi indeksitesse, mis võimaldab kiiret otsingut ka keerulise salvestusstruktuuriga. Kuna andmebaas ei vaja skeemi, võib iga üksik dokument koosneda suvalisest arvust unikaalsetest väljadest, mis võib põhjustaa skaleerimise probleeme. Dokumendiandmebaasid näitavad end hästi teabe salvestamisel kataloogi, näiteks loovad elektroonikaseadmete katalooge veebipoes. [3][4][10]

Näiteks: CouchDB, Couchbase, MongoDB, ArangoDB, Elastic. [14]

Joonis 8. Dokumenti andmebaasi näiteks

Võtmeväärtuse andmebaas (*Key-value database*) on tüüpi andmebaas, milles igale väärtusele on määratud oma kordumatu võti. Selline andmebaas ei pea looma skeeme ja väärtuste vahelisi seoseid. Sellistel andmebaasidel on väga hea skaleeritavus, mis võimaldab töötada suurte andmemahtudega. Sellised andmebaasid pole aga mõeldud kiireteks otsinguteks, kuna see nõuab kogu kollektsiooni skannimist või üksikute indeksiväärtuste loomist. Selliseid andmebaase kasutatakse objektide vahemäludena.[3][15]

Näiteks: ArangoDB, Redis, DynamoDB, Couchbase, Aerospike. [14]

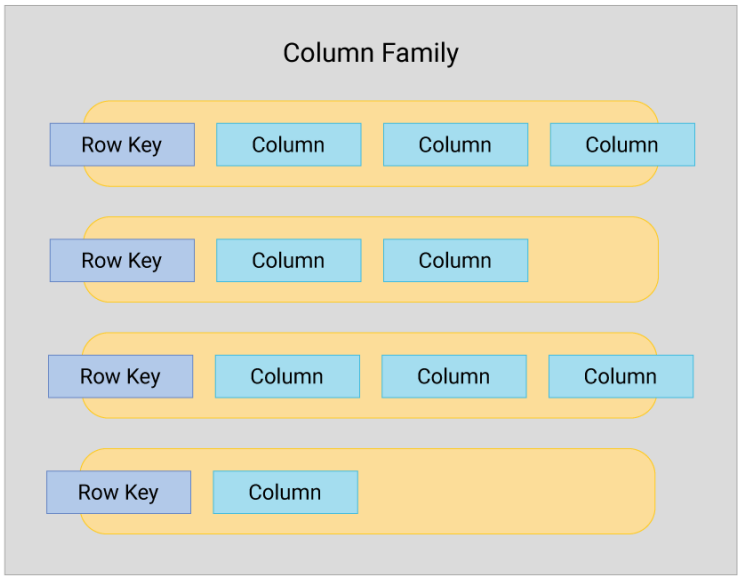


Joonis 9. Võtmeväärtuse andmebassi näiteks

Veerupere või laia veeruga andmebaas (*Column family database*) on tüüp, kus andmed salvestatakse veergudesse, mis seejärel kombineeritakse alamrühmadesse. Erinevalt relatsiooniandmebaasidest, kus tabelid on skeemide sees lingitud, saab laiades veergudes salvestada iga veeru eraldi või kui on sarnased seotud veerud, siis kombineeritakse need veeruperekondadeks, misjärel salvestatakse need üksteisest eraldi.

Veeruperekond on jagatud kahte tüüpi. See on standardne veerg, milles on võtmeveerg ja väärtustega veerud (sarnaselt tabeliga). Ja superveerud, kus igal superveerul on nimi ja väärtus, mis kaardistab superveeru mitme erineva veeruga. Sellistel andmebaasidel on hea skaleeritavus, kuid need ei ole võrgutehingute töötlemisel kuigi tõhusad.[5] Neid kasutatakse konkreetse ülesande jaoks, näiteks mitme miljoni artikli pealkirjade loetlemiseks.[13]

Näiteks: Apache Flink, Hypertable, Cassandra, MonetDB. [14]



Joonis 10. Kolonn-perekonna andmebaasi näiteks

1.3. Teised andmebassid häldussüsteemid

Residentide andmebaas (mälusisene andmebaas (*in-memory database*) või põhimälu andmebaas (*main memory database*)) on andmebaasihaldussüsteem, mis salvestab arvuti andmed arvuti põhimällu, mitte kettale, et tagada kiirem reageerimisaeg. Sellised andmebaasid on kiiremad kui kettale mälu salvestavad andmebaasid, kuna juurdepääs kettale

on palju aeglasem ja sisemised optimeerimisalgoritmid on lihtsamad ja täidavad vähem protsessori käske.

Andmed laaditakse alla tihendatud ja mitterelatsioonilises vormingus ning need on vormingus, mida on lihtne otse kasutada ilma tihendamise või krüptimise takistusteta. See võimaldab otsest navigeerimist registrist reale või veergu ja on kirjutuskaitstud süsteem.

Kuigi elanike andmebaas suudab kiiresti teavet kuvada, on sellel märkimisväärne puudus. Krahhi või ühenduse katkemise korral lähevad andmed kaotsi. Andmete kadumise vältimiseks peate tegema andmetest hetktõmmiseid, kirjutama tehingud kõvakettale, installima NVRAM ja kasutama välkmälu, kuid sellel on välkmälu kustutamise ja ülekirjutamise arv piiratud. mälu. Residentide andmebaas on leidnud tee näiteks pankades enne laenu väljastamist klienti analüüsima, kindlustusseltsides riske arvutama või interaktiivseid online mänge, sest võimaldab reaalajas andmeid analüüsida ja aruandeid koostada. [20][21][22]

Näiteks: Redis, Apache Ignite, Tarantool. [14]

Mitme mudeliga andmebaas (*Multi-model database*) on haldussüsteem, mis ühendab mitut tüüpi andmebaase ühe taustaprogrammiga ning salvestab, pärib ja indekseerib erinevate mudelite andmeid. Mitme mudeliga andmebaasid pakuvad mitmekeelse püsivuse modelleerimise eeliseid, ilma et peaks leidma võimalusi erinevate mudelite kombineerimiseks. Paindlik lähenemine võimaldab salvestada andmeid erineval viisil.

Mitme mudeliga andmebaasid võivad teisendada andmeid ühest vormingust teise, mis annab täiendavat paindlikkust ja hõlbustab konkreetsete projektinõuete täitmist.[23]

Näiteks: ArangoDB, OrientDB, FoundationDB. [14]

Objektorienteeritud andmebaas (*Object Oriented Database*) on andmebaasihaldussüsteem, kus andmed on esitatud objektidena, mida kasutatakse objektorienteeritud programmeerimiskeeltes. Igal objektil on oma omadused ja meetodid ning seda hallatakse koos objektorienteeritud programmeerimiskeelega, et hõlbustada objektorienteeritud andmete salvestamist ja otsimist. Objektid moodustuvad lihtsatest objektidest tänu konstruktoritele. Lihtsamad objektid on: numbrid, sümbolid, suvalise pikkusega märgistringid, tõeväärtuslikud muutujad jne. Komplekssed objektikonstruktorid on:

korteežid, hulgad, multihulgad, loendid ja massiivid. Komplektid esindavad reaalse maailma objektide kogusid loomulikul viisil. Kordad pakuvad viisi olemite atribuutide esitamiseks. Loendid ja massiivid esindavad reaalse maailma objektide elamiskõlblikkust ning esindavad ka laia valikut matemaatilisi objekte.

Objektorienteeritud andmebaasid toetavad kapseldamist, mis peidab andmeid, ja programmeerimiskoode andmetega manipuleerimiseks. Objekt on jagatud liidese- ja teostusosadeks. Liidese osa on objektil lubatud toimingute komplekti spetsifikatsioon ja see objekti osa on nähtav teiste objektide meetoditele. Rakendusosa koosneb andmetest, mis kirjeldavad objekti olekut, ja protseduuridest, mis teostavad objektil toiminguid. Samuti on olemas pärimise tugi, mis võimaldab luua uusi klasse kasutades teiste klasside andmeid ja meetodeid.

Üks peamisi eeliseid relatsioonilise andmebaasihaldussüsteemi ees on see, et objektorienteeritud andmebaasid suudavad luua kiireid seoseid keeruliste andmetega ja keeruliste seostega, kuna andmebaasi struktuur on programmeerimisobjektidele väga lähedane. Kuid kuna igal objektorienteeritud programmeerimiskeelel on oma päringu süntaks ja puuduvad üldtunnustatud standardid, peate erinevalt SQL-i kasutavast relatsioonilise andmebaasi haldussüsteemist kohanema iga keelega.[25][26][27]

Näiteks: db4o, ZODB, ObjectDB. [14]

2. Vajalikud kompetentsid EuroTeQ

Antud töö põhineb Euroopa tehnika- ja teadusülikoolist koosneva konsortsiumi EuroTeQ uurimisel tööstusettevõtetes üle kogu Euroopa. Kuna tehnoloogiad arenevad kiiresti ja muudavad sotsiaalset keskkonda, mille tõttu seniste spetsialistide vaja arenevad oskused muutuvad, on projekti EuroTeQ Professional eesmärk muuta koolituse formaati ja arendada tulevaste spetsialistide tulevikukompetentsi nii, et need oleksid kogu aeg nõutud.

Tulevikukompetentsid võib jagada kolme rühma: isiklikud pädevused, sotsiaalsed ja kommunikatiivsed pädevused, tehnilised ja metoodilised pädevused, samas kui need ristuvad omavahel. Tulevased professionaalsed insenerid peavad neid pädevusi arendama, et teha koostööd keerulistes keskkondades.

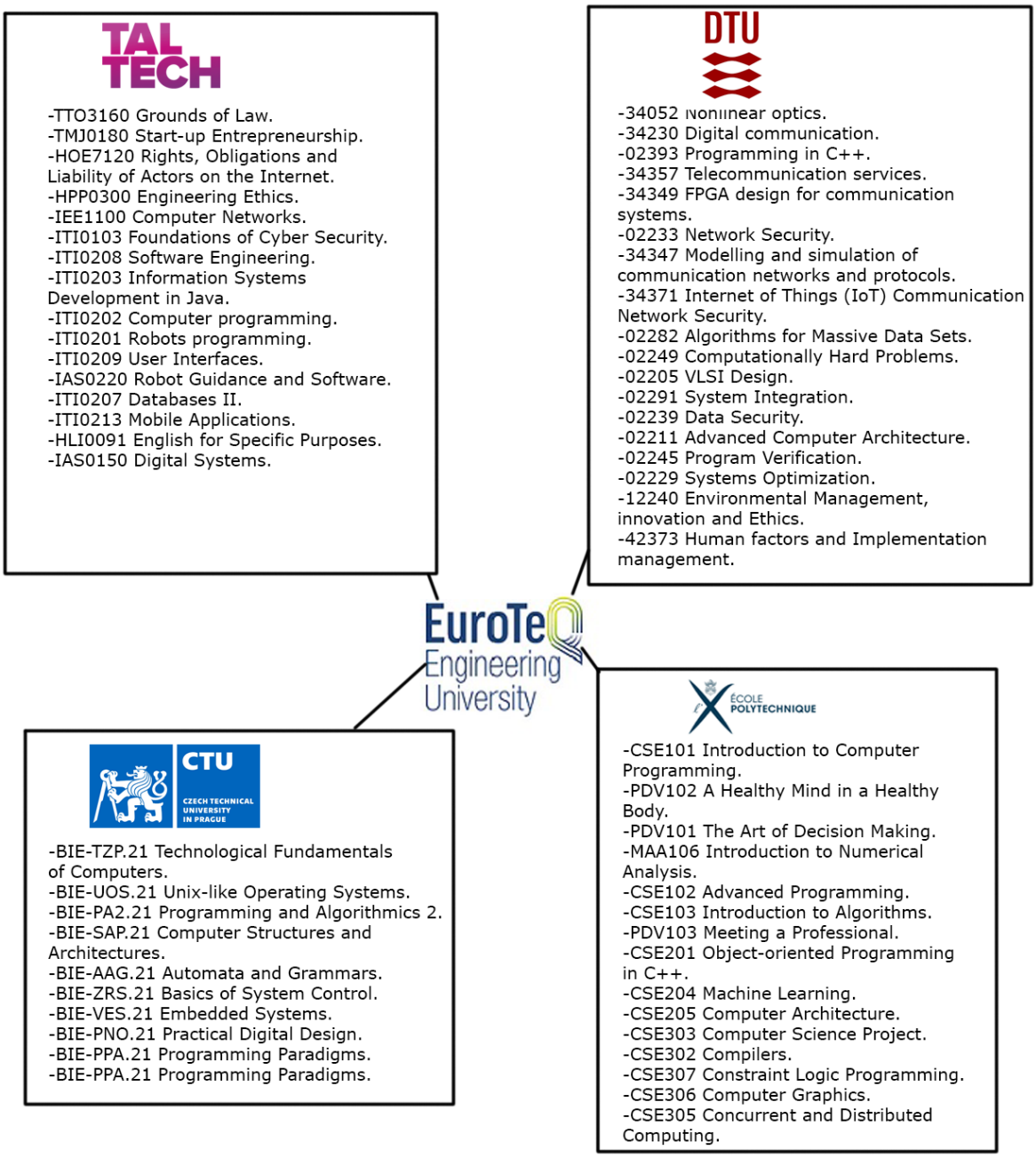
See võimaldab neil lahendada suuri sotsiaalseid probleeme, jääda konkurentsivõimeliseks, uuendusmeelseks, omandada uusimaid tehnoloogiaid, mõista uuenduslike lahenduste ja võimaluste tehnilisi aspekte.

2.1. EuroTeQ

EuroTeQ on Euroopa ülikoolide liit, mis on arenenud teisest ülikoolide liidust EuroTech. Euroopa ülikoolide EuroTeQ alliansi kuuluvad: Müncheni tehnikaülikool (TUM), Taani tehnikaülikool (DTU), Eindhoveni Tehnikaülikool (TLÜ/e), École Polytechnique (l'X), Tšehhi tehnikaülikool Prahas (CTU), Tallinna Tehnikaülikool (TalTech) ja 45 assotsieerunud projektipartnerit tööstusest ja ühiskonnast. Kõik kuuest juhtivast teadus- ja tehnoloogiaülikoolist, mis asuvad laiali üle Euroopa, asuvad uuenduslikes ökosüsteemides ja millel on pikk koostöö ajalugu.

EuroTeQ eesmärk on, et umbes 2030. aastaks saaksid nad moodustada juhtiva üleeuroopalise ökosüsteemi kaasvastutustundliku väärtuse loomisel tehnoloogia vallas, saada eeskujuks ülikoolide võrgustikuks, mis hõlmab kõiki tehnoloogilise innovatsiooni protsesside asjakohaseid sidusrühmi, ning saada juhtivaks rahvusvaheliseks teadus- ja tehnoloogiaalase hariduse võrgustikuks. , teadmiste ruudu fikseerimine kooskõlas euroopalike väärtustega. Oma eesmärkide saavutamiseks on EuroTeQ valmis kohandama asjakohaseid struktuure ja protsesse, et luua enneolematuid koostööraamistike ja tegevuste vorme ning avada meie arendusprotsess ja sellega seotud teadmised nii nende

institutsioonide liikmetele kui ka meie vastavate sidusrühmadele. ökosüsteemides Euroopas ja mujal.[24]



Joonis 11. EuroTeQ skeem

2.2. Tulevikukompetentsi

2.2.1. Sotsiaal-kommunikatiivsed kompetentsid

Sotsiaal-kommunikatiivsed kompetentsid eeldavad oskust suhelda ja leida teiste inimestega ühist keelt, samuti arendada juhtimisoskusi ja oskust töötada meeskonnas. Sotsiaal-

kommunikatiivsed kompetentsid viitavad meie ühiskondades ja keskkondades aktsepteeritud käitumisreeglite ja suhtlusreeglite tundmisele, samuti oskustele, mis võimaldavad inimesel erinevates keskkondades konstruktiivselt suhelda, teha koostööd, pidada läbirääkimisi, olla tolerantne, väljendada ja mõista erinevaid seisukohti ja luua enesekindlust. Sotsiaalselt kompetents olemine tähendab ka koostööhoiaku kasvatamist, inimeste mitmekesisust austamist, eelarvamuste ületamist ja kompromisside tegemist ühiskonnaelus osalemise kaudu.

See valdkond hõlmab suhtlemist ja koostööd. Suhtlemine arendab inimeses oskust edastada infot, kirjeldada tulemusi ja meetodeid ning põhjendada teistele tehtud järeldusi erinevates vormides, samuti kuulata ja tajuda teistelt vestluses osalejatelt saadavat informatsiooni vestluspartneri edukama suhte nimel. Koostöö on vajalik selleks, et inimesed saaksid antud ülesandega tõhusalt töötada, samuti osataks tööd osalejate vahel jagada nii, et iga osaleja oskused oleksid õiges valdkonnas rakendatavad.

Millises aines neid pädevusi arendatakse, saab näidata Tallinna Tehnikaülikooli õppeaine "Start-up ettevõtlus (TMJ0180)" näitel, kus rubriigis õppeaine õpiväljundid on kirjas, et üliõpilane saab meeskonnatöö kogemuse valdkonna arendamisel. äriidee ja selle elluviimise võimaluste väljaselgitamine. Siit võib järeldada, et selle aine lõpetanud üliõpilane tõstab koostöö kompetentsi ja kuna ta suhtleb meeskonnaliikmetega, siis areneb ka suhtlemine.

2.2.2. Isiklikud kompetentsid

Isiklikud kompetentsid pakuvad igal inimesel enda kui isiksuse arendamiseks, õitsenguks ja oma potentsiaali realiseerimiseks. Isiklikud pädevused hõlmavad teadmisi tervest vaimust, kehast ja elustiilist; oskust tulla toime keerukuse, ebakindluse ja stressiga, otsida vajadusel tuge ja säilitada vastupidavust, samuti arendada iseseisva töö ja karjääri juhtimise oskust. See hõlmab ka visadust, ausust, enesemotivatsiooni, probleemide lahendamise oskust muutustega toimetulekuks ning üldiselt positiivset suhtumist isikliku, sotsiaalse ja füüsilise heaolu edendamisse.

Isikliku kompetentsi näite võib võtta õppeainest "Tarkvaratehnika (ITI0208)", mis annab tudengile teadmisi erinevatest arendusprotsessidest ja oskab töötada agiilses arendusmeeskonnas. See tähendab, et meeskonnaga paigas olev tudeng kohanevad erinevate arenguprotsessidega. Nii arendab ta kohanemisvõimet, mis võimaldab olla paindlik ja kohaneda muutuvate tegurite, tingimuste või keskkonnaga.

2.2.3. Tehnilised-metodoloogilised kompetentsid

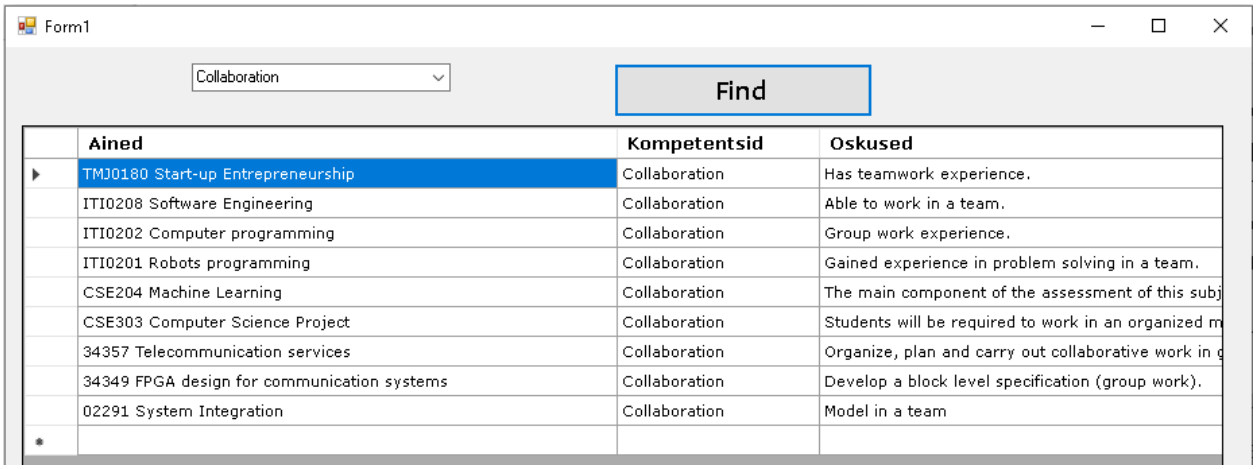
Tehnilised-metodoloogilised kompetentsid on oskus omada teavet konkreetse teema või aine kohta ning oskus seda kasutada õiges kohas ja ajal. Näiteks mitmed erinevad insenerierialad, aga ka IT-ga seotud tehnilised oskused või nutikate projektide loomisega seotud tehnilised oskused nagu programmeerimine, andmete kogumine, eelarvestamine ning didaktilised teadmised või õppejõudude töövahendid.

Tehnilisest-metodoloogilisest valdkonnast võib välja tuua kompetentsi tarkvara tundmine, mis võib kasutada teadmisi mõne arvutiprogrammi ja tööriista kohta. Näiteks õppeaines "Programmeerimise põhikursus (ITI0202)" töötavad õpilased Java keeles. Ja pärast seda ainet saab õpilane teadmised Java programmeerimiskeelest.

3. Rakendus “kompetent” prototüüpid

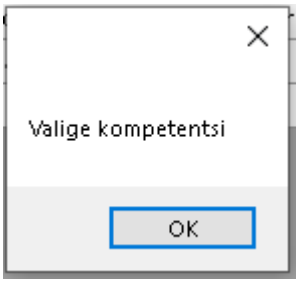
Mõlemad prototüübid realiseerivad otsingu andmeid oma andmebaasis. Mõlemad prototüübid on loodud Visual Studio keskkonnas Visual Basic programmeerimiskeeles. Esimene prototüüp võtab andmed Microsoft Aceessis loodud relatsiooniandmebaasist. Teine prototüüp võtab andmeid NoSQL-i andmebaasist tüüpi dokumendi andmebaasi, mis loodi MongoDB haldussüsteemis. Kui käivitate rakendused spetsiaalsel väljal tabeli jaoks, kuvatakse kõik andmed andmebaasist, millega see on ühendatud.

Otsinguprotsessi toimumiseks peab kasutaja valima spetsiaalsel real pakutavast loendist sobiva kompetentsi. Nupu vajutamisel tehakse otsingu protsessi ja pärast protsessi lõppu on spetsiaalsel väljal tabeli jaoks näitab kõik andmeid valitud kompetentsiga.



Joonis 12. Otsingu tulemus

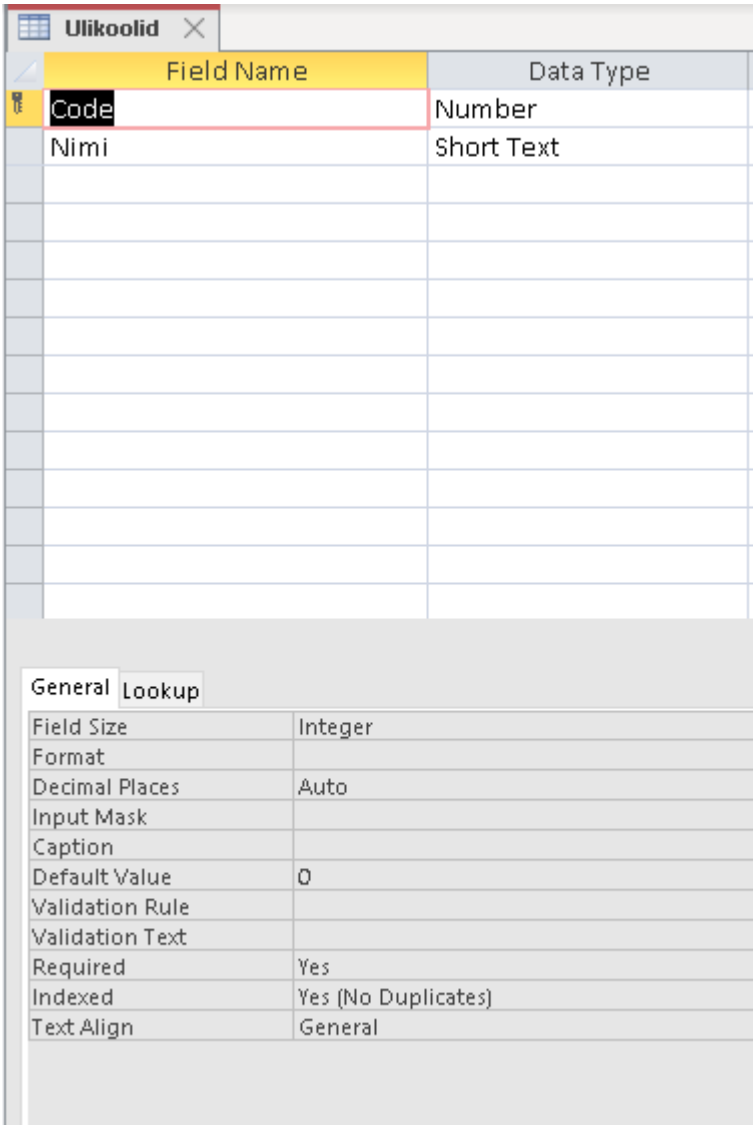
Juhul kui kasutaja klõpsas nupul ja ei valinud kompetentsi, siis ilmub menüü, mis palub teil seda teha.



Joonis 13. Menüü, juhul kui ei valitud kompetentsi

3.1. Rakendus relatsiooni andmebaasiga

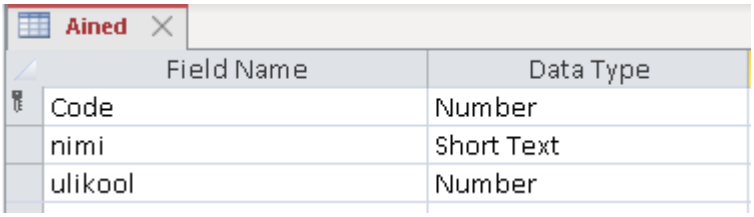
Relatsiooniandmebaasi loomiseks loodi tabelid ja igasse tabelisse salvestati osa andmete. Iga tabel loodi kujundus režiimis ja tabeli loomisel anti veeru nimi ja mis tüüpi andmed selles veerus oleks. Näiteks tabelis "Ulikoolid" on kaks veergu: kood ja nimi. Veerg kood on selle tabeli primaarvõti, mida kasutatakse tabeli iga kirje tuvastamiseks ja mis salvestab täisarvu andmetüübi. Nimi veergu kasutatakse ülikoolide nimetuste teksti salvestamiseks.



Joonis 14. Kujundus režiim tabeli Ulikoolid

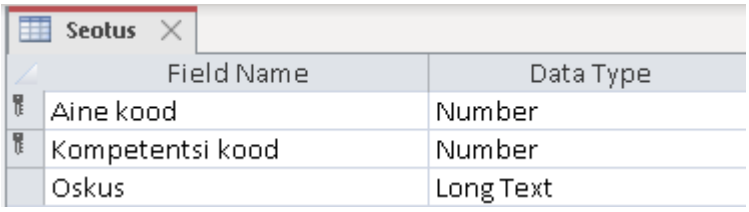
Tabelite "Ained" ja "Kompetentsid" seaded on sarnased, kuid tabelis "Ained" on veerg nimega ülikoolid ja täisarvuline andmetüüp. Seda tehakse selleks, et edastada tabeli "Ulikoolid" kirjed tabelisse "Ained" kasutades esimese tabeli primaarvõtit ja seeläbi saab

sellest teise tabeli võõrvõti. Need kaks tabelit moodustavad üks-mitmele seose, kuna ühes ülikoolis võib olla mitu õppeainet, kuid ühte ainet saab õpetada ainult ühes ülikoolis.

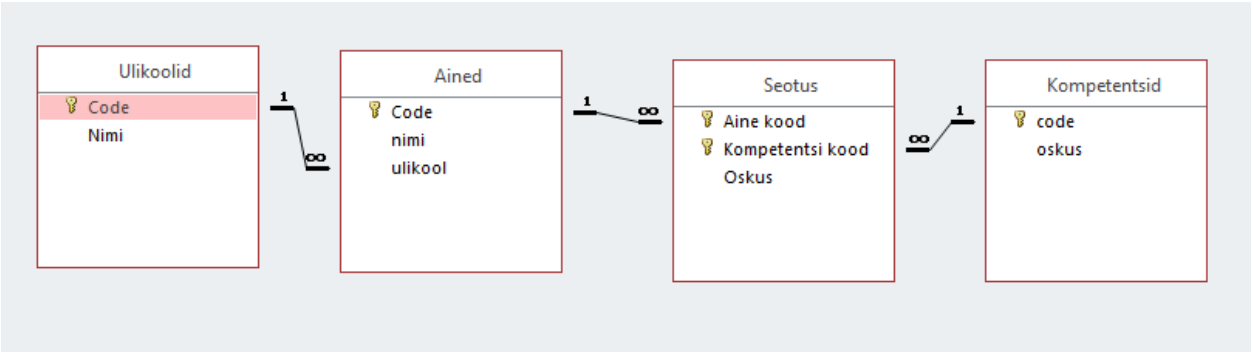


Joonis 15. Kujundus režiim tabeli Ained

Loogiliselt võttes võib ühe aine koolitusel arendada mitut pädevust ja ühte pädevust mitmes õppeaines. Seetõttu peab tabelite "Ained" ja "Kompetentsid" vahel olema seos palju-mitmele ning seose loomiseks tehti vahetabel nimega “Seotus”, mis salvestab mõlema tabeli primaarvõtmed.

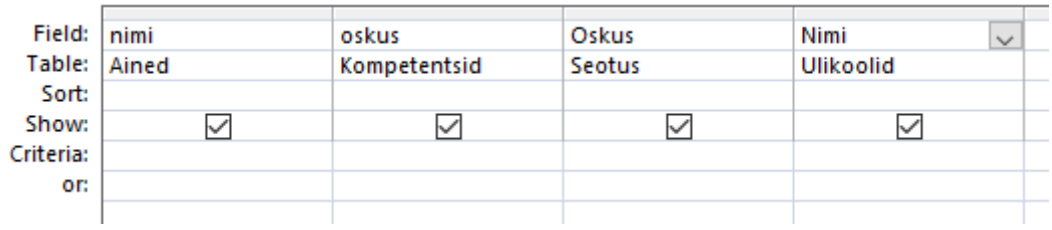


Joonis 16. Kujundus režiimtabeli Seotus



Joonis 17. Seosed tabelite vahel

Selleks et rakendusele väljastada primaarvõtme kirje, mitte võtme enda number, tehti sellekohane taotlus ja rakendus võtab selle kaudu teavet.



Joonis 18. Taotlus andmebassi

Selleks, et rakendus saaks suhelda Accessi andmebaasiga, tuleb kutsuda rakenduste programmeerimisliides OleDB. Kui rakendus käivitub, kuvab see kõik päringus määratud andmed spetsiaalsel väljal

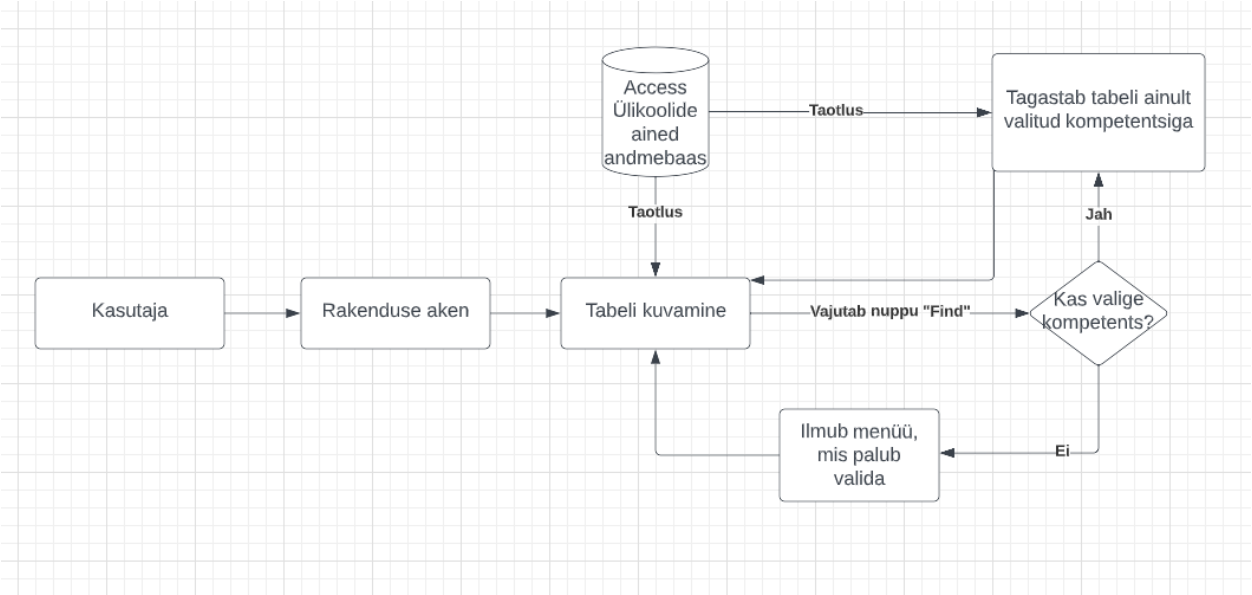
```
Imports System.Data.OleDb

ссылка: 1
Public Class Form1
    Dim con As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\Users\Никита\source\repos\GraduateWorkPrototype\GraduateWorkPrototype\loputoo1.accdb")
    Dim cmd As New OleDbCommand("Select * From Taotlus", con)
    Dim da As New OleDbDataAdapter(cmd)
    Dim dt As New DataTable

    Ссылок: 0
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'When loading the application, the entire table from the query is displayed
        da.Fill(dt) 'Adds rows in a specified range in the DataSet
        DataGridView1.DataSource = dt
        DataGridView1.Columns(0).Width = 400 'A table column is given a width
        DataGridView1.Columns(1).Width = 150
        DataGridView1.Columns(2).Width = 430
        DataGridView1.Columns(3).Width = 200
        DataGridView1.Columns(0).HeaderText = "Ained" 'The table column is given a name
        DataGridView1.Columns(1).HeaderText = "Kompetentsid"
        DataGridView1.Columns(2).HeaderText = "Oskused"
        DataGridView1.Columns(3).HeaderText = "Ülikoolid"
    End Sub
```

Joonis 19. Osa koodist rakenduse relatsiooniandmebaasiga käivitamisel

Pärast seda, kui kasutaja valib huvipakkuva kompetentsi ja klõpsab otsingunupul, määrab programm esmalt päringutabelis olevad väärtused, seejärel määrab päringule "kompetents" käsustringi, kuid ainult valitud kompetentsiga elementist ja seejärel kuvage väärtus vastloodud tabelist.

```
Private Sub BtnFind_Click(sender As Object, e As EventArgs) Handles BtnFind.Click 'When the button is pressed, a new table is created with the selected word
    If CBKompetentsid.Text IsNot "" Then
        'If competency has been selected
        cmd.CommandType = CommandType.Text
        cmd.CommandText = "select * from Taotlus where kompetents = '" + CBKompetentsid.Text + " ' " 'Returns the command string for the specified data source
        dt = New DataTable()
        da = New OleDbDataAdapter(cmd)
        da.Fill(dt)
        DataGridView1.DataSource = dt
    Else
        MessageBox.Show("Valige kompetentsi") 'A window appears with a message that you need to select a competency
    End If
End Sub
```

Joonis 20. Osa koodist rakenduse relatsiooniandmebaasiga nupu klõpsamisel



Joonis 21. Struktuurskeem rakenduse relatsiooni andmebaasiga

3.2. Rakendus NoSQL andmebaasiga

Dokumendile orienteeritud andmebaasi loomiseks tuleb MongoDB Shelli programmis luua kogu, millesse tuleb dokumendid sisestada, et hiljem kogust, kasutades programmi MongoDB Compass, väljastada fail CSV-vormingus. Koos andmetega lisati dokumendile ka id väärtus, et edaspidi oleks soovitud dokumenti lihtne leida ja sellega suhelda.

```
test> show dbs
admin      40.00 KiB
config    116.00 KiB
database  144.00 KiB
local      72.00 KiB
test> use database
switched to db database
database> db.createCollection("databaseCollction")
{ ok: 1 }
```

Joonis 22. Käsud kollektsiooni loomiseks

Joonis 23. Osa käsust, et lisada dokumentide kollektsioonis

Joonis 24. Osa dokumentidest kollektsioonis

Kui rakendus käivitub, hakkab programm määratud failist andmeid lugema. CSV-failist otsides kirjutatakse andmed komadega eraldatuna, seejärel kantakse rida alguses muutujasse "sline". Seejärel tuleb sellest muutujast string jagada komadega ja kanda stringi massiivi muutujasse "words" ning seejärel lisada väärtus DataTable tüüpi muutujale ja seejärel tabeli jaoks spetsiaalsele väljale. See jätkub seni, kuni programm läbib faili kõik read.

```
Dim table As New DataTable("table")
Ссылок: 0
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    Try
        Dim fname As String = "C:\Users\Никита\source\repos\GraduateWorkNoSQL\loputoo2.csv"
        Dim reader As New StreamReader(fname, Encoding.Default) 'Open file for reading
        Dim sline As String = ""
        table.Columns.Add("Ained", Type.GetType("System.String"))
        table.Columns.Add("Kompetentsid", Type.GetType("System.String"))
        table.Columns.Add("Oskused", Type.GetType("System.String"))
        table.Columns.Add("Ülikoolid", Type.GetType("System.String"))
        sline = reader.ReadLine
        Do
            sline = reader.ReadLine
            If sline Is Nothing Then Exit Do
            Dim words() As String = sline.Split(",") 'Divides the string by commas and passes it to the array of strings
            AddToTable(words)
        Loop
        reader.Close()
        DataGridView1.DataSource = table
        DataGridView1.Columns(0).Width = 400 'A table column is given a width
        DataGridView1.Columns(1).Width = 150
        DataGridView1.Columns(2).Width = 430
        DataGridView1.Columns(3).Width = 200
    Catch ex As Exception
        MessageBox.Show("Valitud faili ei leitud")
    End Try
End Sub
```

Joonis 25. Osa koodist rakenduse NoSQL andmebaasiga käivitamisel

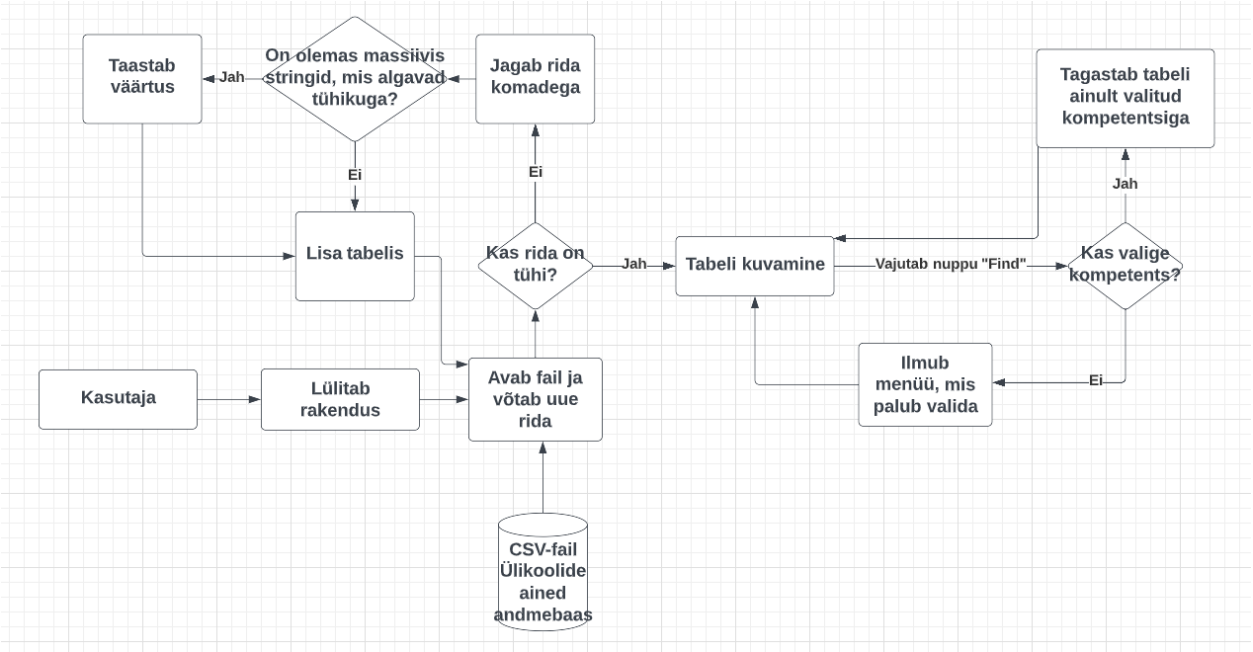
Tekib aga probleem, et nagu on näidatud joonisel 3.14, on "Oskus" kategoorias oleva id 15-ga dokumendi väärtuses koma ja programm jagab selle ka ära. Seega, et tabelis oleks õiged andmed, tuleb kontrollida, millise märgiga mingi massiivi väärtus algab. Kui see algab tühikuga, siis tuleb taastada ja seejärel lisada väärtus massiivist teatud indeksite alla.

```
Function AddToTable(ByVal words As String())
    If words(1).Chars(0) = " " Then 'If a value is shared in the "Ained" category
        words(0) = words(0).Substring(1, Len(words(0)) - 1) + "," + words(1).Substring(0, Len(words(1)) - 2) 'Restore value and remove quotes
        table.Rows.Add(words(0), words(2), words(3), words(4))
    ElseIf words(3).Chars(0) = " " Then 'If a value is shared in the "Oskused" category
        words(2) = words(2).Substring(1, Len(words(2)) - 1) + "," + words(3).Substring(0, Len(words(3)) - 2)
        table.Rows.Add(words(0), words(1), words(2), words(4))
    Else
        table.Rows.Add(words(0), words(1), words(2), words(3))
    End If
End Function
```

Joonis 26. Osa koodist rakenduse NoSQL andmebaasiga, et lisa andmeid tabelis

```
Private Sub BtnFind_Click(sender As Object, e As EventArgs) Handles BtnFind.Click
    If CBKompetentsid.Text IsNot "" Then
        TryCast(DataGridView1.DataSource, DataTable).DefaultView.RowFilter =
        String.Format("Kompetentsid like '%" & CBKompetentsid.Text & "%'")
    Else
        MessageBox.Show("Valige kompetentsi")
    End If
End Sub
```

Joonis 27. Osa koodist rakenduse NoSQL andmebaasiga nupu klõpsamisel



Joonis 28. Struktuurskeem rakenduse NoSQL andmebaasiga

Kokkuvõte

Töö põhieesmärgiks oli andmebaasiotsingu rakendamine ülikooli õppeainete ja tulevaste professionaalsete inseneride kompetentside näitel. Käesolevas töös analüüsiti peamisi andmebaaside tüüpe ja mõningaid andmebaasihaldussüsteemide tüüpe. Nagu ka meetodid relatsiooniandmebaaside ja nende tüüpide kujundamiseks. Käesolevas töös selgitati ka ülikoolide EuroTeQ konsortsiumi ja tulevikukompetentside gruppe. Töö tulemusena valmisid prototüübid, mis tuvastati Visual Studio programmis Visual Basic programmeerimiskeeles ja täidavad sarnast ülesannet, kuid kasutavad erinevat tüüpi andmebaase. Esimene kasutab relatsiooniandmebaasi, mis tehti Microsoft Accessis. Ja teine kasutab MongoDB-s tehtud ja CSV-failiks teisendatud dokumendibaasi.

Need prototüübid toovad välja valitud kompetentsi ained ja aitavad arendada tulevastel professionaalsetel inseneridel soovitud pädevusi. Edaspidi võiks proovida teostada otsingut teist tüüpi andmebaaside abil. Samuti on võimalik lisada uusi andmeid olenevalt uute õppeainete ilmumisest, uutest tehnikaülikoolidest konsortsiumis ning olemasolevate kompetentside ümbermõtestamisest ja uute tekkimisest.

Kasutatud kirjandus

[1] https://www.oracle.com/cis/database/what-is-database/#WhatIsDBMS (veebiartikkel)
[2] https://jino.ru/journal/articles/7-baz-dannyh/ (veebiartikkel)
[3] https://veesp.com/ru/blog/sql-or-nosql/ (veebiartikkel)
[4] https://aws.amazon.com/ru/nosql/document/ (veebiartikkel)
[5] https://www.techtarget.com/searchdatamanagement/tip/NoSQL-database-types-explained-Column-oriented-databases (veebiartikkel)
[6] https://aws.amazon.com/ru/relational-database/ (veebiartikkel)
[7] https://www.oracle.com/cis/database/what-is-a-relational-database/ (veebiartikkel)
[8] https://www.oracle.com/cis/database/nosql/what-is-nosql/ (veebiartikkel)
[9] https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html (veebiartikkel)
[10] https://en.wikipedia.org/wiki/Document-oriented_database (veebiartikkel)
[11] https://databasetown.com/relational-database-benefits-and-limitations/ (veebiartikkel)
[12] https://technologypoint.in/advantages-and-disadvantages-of-nosql-databases/ (veebiartikkel)
[13] https://tproger.ru/translations/types-of-nosql-db/ (veebiartikkel)
[14] https://hostingdata.co.uk/nosql-database/ (veebiartikkel)
[15] https://www.freecodecamp.org/news/the-pros-and-cons-of-different-data-formats-key-values-vs-tuples-f526ad3fa964/ (veebiartikkel)
[16] https://ru.wikipedia.org/wiki/Microsoft_Access#cite_note-2 (veebiartikkel)
[17] https://www.youtube.com/watch?v=pYRPYfqWiUo&t=223s (video)
[18] https://info-comp.ru/database-normalization (veebiartikkel)
[19] https://www.guru99.com/database-normalization.html (veebiartikkel)
[20] https://www.heavy.ai/technical-glossary/in-memory-database (veebiartikkel)
[21] https://habr.com/ru/company/headzio/blog/505792/ (veebiartikkel)
[22] https://en.wikipedia.org/wiki/In-memory_database (veebiartikkel)
[23] https://phoenixnap.com/kb/multi-model-database (veebiartikkel)
[24] https://euroteq.eurotech-universities.eu/ (veebiartikkel)
[25] https://elearning.sumdu.edu.ua/free_content/lectured:eaeb836f0e4d45b9a509d6c3f79a9ecf6ab05395/20141129091116/49214/index.html (veebilehekülg loengu)
[26] https://www.mongodb.com/databases/what-is-an-object-oriented-database (veebiartikkel)
[27] https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85 (veebiartikkel)
[28] http://mech.math.msu.su/~shvetz/54/inf/databases/chApplications.xhtml (veebillehekülg)
[29] https://info-comp.ru/zero-normal-form (veebiartikkel)
[30] https://info-comp.ru/first-normal-form (veebiartikkel)
[31] https://info-comp.ru/second-normal-form (veebiartikkel)
[32] https://info-comp.ru/third-normal-form (veebiartikkel)
[33] https://info-comp.ru/boyes-codd-normal-form (veebiartikkel)
[34] https://info-comp.ru/fourth-normal-form (veebiartikkel)

[35] https://info-comp.ru/fifth-normal-form (veebiartikkel)
[36] https://info-comp.ru/domain-key-normal-form (veebiartikkel)
[37] https://info-comp.ru/sixth-normal-form (veebiartikkel)
[38] https://aws.amazon.com/ru/nosql/graph/ (veebiartikkel)

Lisa 1 Andmed andmebaasi

Tallinna TehnikaÜlikool:

1. TTO3160 Grounds of Law: Systems thinking.
2. TMJ0180 Start-up Entrepreneurship: Communication, Collaboration, Project management.
3. HOE7120 Rights, Obligations and Liability of Actors on the Internet: Systems thinking.
4. HPP0300 Engineering Ethics: Sustainability, Systems thinking, Social awareness.
5. IEE1100 Computer Networks: Computer science, Social awareness.
6. ITI0103 Foundations of Cyber Security: Sustainability, Systems thinking, Critical Thinking, Computer science.
7. ITI0208 Software Engineering: Communication, Collaboration, Adaptability, Computer science, Sustainability.
8. ITI0203 Information Systems Development in Java: Computer science, Software knowledge, Problem solving.
9. ITI0202 Computer programming: Communication, Collaboration, Problem solving, Interdisciplinairy skills, Computer science, Software knowledge.
10. ITI0201 Robots programming: Communication, Collaboration, Artificial intelligence literacy, Adaptability, Critical Thinking.
11. ITI0209 User Interfaces: Sustainability, Social awareness.
12. IAS0220 Robot Guidance and Software: Software knowledge, Artificial intelligence literacy, Problem solving, Critical Thinking.
13. ITI0207 Databases II: Data Science, Entrepreneurial mindset, Computer science, Software knowledge.
14. ITI0213 Mobile Applicatioьns: Software knowledge.
15. HLI0091 English for Specific Purposes: Communication, Critical Thinking.
16. IAS0150 Digital Systems: Computer science, Modelling.

Technical University of Denmark(DTU):

1. 34052 Nonlinear optics: Modelling, Critical Thinking, Communication.
2. 34230 Digital communication: Computer science, Software knowledge, Critical Thinking, Communication.

3. 02393 Programming in C++: Computer science, Software knowledge, Critical Thinking, Entrepreneurial mindset, Communication, Data Science.
4. 34357 Telecommunication services: Collaboration, Interdisciplinairy skills, Critical Thinking, Communication.
5. 34349 FPGA design for communication systems: Communication, Computer science, Software knowledge, Collaboration, Critical Thinking, Modelling.
6. 02233 Network Security: Computer science, Sustainability, Problem solving, Critical Thinking, Social awareness.
7. 34347 Modelling and simulation of communication networks and protocols: Communication, Modelling, Critical Thinking.
8. 34371 Internet of Things (IoT) Communication Network Security: Communication, Critical Thinking, Innovative thinking.
9. 02282 Algorithms for Massive Data Sets: Communication, Critical Thinking, Data Science, Interdisciplinairy skills.
10. 02249 Computationally Hard Problems: Critical Thinking, Foresight thinking, Problem solving, Entrepreneurial mindset.
11. 02205 VLSI Design: Computer science, Critical Thinking, Modelling.
12. 02291 System Integration: Communication, Collaboration, Modelling, Emotional Intelligence, Critical Thinking.
13. 02239 Data Security: Innovative thinking, Foresight thinking, Problem solving, Critical Thinking.
14. 02211 Advanced Computer Architecture: Communication, Critical Thinking, Computer science.
15. 02245 Program Verification: Communication, Critical Thinking, Computer science, Software knowledge.
16. 02229 Systems Optimization: Communication, Computer science, Problem solving, Modelling, Critical thinking, Entrepreneurial mindset.
17. 12240 Environmental Management, innovation and Ethics: Communication, Innovative thinking, Sustainability, Systems thinking.
18. 42373 Human factors and Implementation management: Systems thinking, Critical Thinking, Sustainability, Self-awareness.

Czech Technical University in Prague (CTU):

1. BIE-TZP.21 Technological Fundamentals of Computers: Computer science, Modelling, Software knowledge.
2. BIE-UOS.21 Unix-like Operating Systems: Computer science, Software knowledge.
3. BIE-PA2.21 Programming and Algorithmics 2: Data Science, Interdisciplinairy skills, Software knowledge.
4. BIE-SAP.21 Computer Structures and Architectures: Computer science.
5. BIE-AAG.21 Automata and Grammars: Computer science.
6. BIE-ZRS.21 Basics of System Control: Computer science, Modelling.
7. BIE-VES.21 Embedded Systems: Computer science, Modelling, Software knowledge.
8. BIE-PNO.21 Practical Digital Design: Modelling, Software knowledge.
9. BIE-PPA.21 Programming Paradigms: Computer science, Critical Thinking.
10. BIE-PST.21 Probability and Statistics: Critical Thinking.

Institut polytechnique de paris:

1. CSE101 Introduction to Computer Programming: Computer science, Software knowledge, Data Science.
2. PDV102 A Healthy Mind in a Healthy Body: Self-awareness.
3. PDV101 The Art of Decision Making: Self-awareness.
4. MAA106 Introduction to Numerical Analysis: Data Science, Computer science, Software knowledge.
5. CSE102 Advanced Programming: Computer science, Software knowledge, Interdisciplinairy skills.
6. CSE103 Introduction to Algorithms: Data Science, Computer science, Interdisciplinairy skills.
7. PDV103 Meeting a Professional: Communication, Self-awareness.
8. CSE201 Object-oriented Programming in C++: Software knowledge, Computer science.
9. CSE204 Machine Learning: Communication, Collaboration, Data Science, Software knowledge, Artificial intelligence literacy, Interdisciplinairy skills.
10. CSE205 Computer Architecture: Computer science, Interdisciplinairy skills.
11. CSE303 Computer Science Project: Communication, Collaboration, Interdisciplinairy skills.

12. CSE302 Compilers: Computer science, Software knowledge, Interdisciplinairy skills.
13. CSE307 Constraint Logic Programming: Data Science, Computer science, Software knowledge, Interdisciplinairy skills.
14. CSE306 Computer Graphics: Computer science, Interdisciplinairy skills.
15. CSE305 Concurrent and Distributed Computing: Computer science, Interdisciplinairy skills.