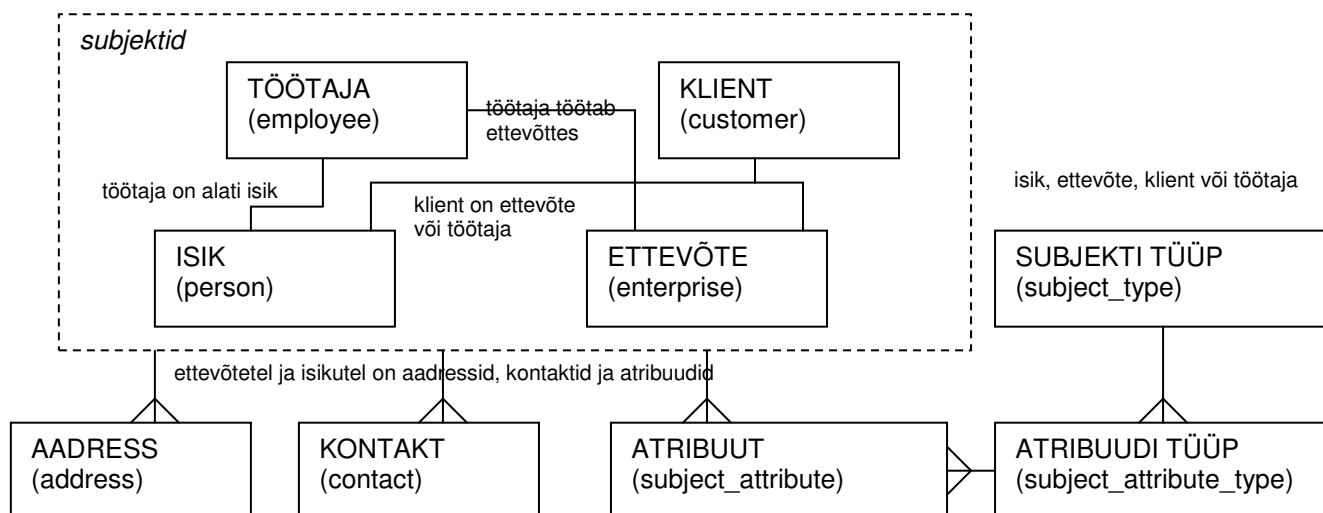


1. Sissejuhatus , ülevaade andmetest ja funktsionaalsusest.....	1
2. Andmed.	2
2.1. Andmebaasi loomine.....	2
2.2. Andmebaasi skeem.....	3
2.3. Andmebaasi ülesehituse üldised põhimõtted.....	3
2.4. Andmetabelite ja väljade selgitused.	5
3. Rakenduselt oodatav funktsionaalsus.....	16
3.1. Funktsionaalsuste loetelu	16
Subjekti tüüpide ettevalmistamine INSERT-lausetega:	16
Andmete lisamine:	17
Atribuutide lisamine subjektile peale subjekti andmebaasi lisamist.Lisamise vormi pealt andmete muutmise vormile.Subjekti tegemine „kliendiks” ja „töötajaks”.	17
Otsing:	19
Tegevused töötajaga:	23
Andmete muutmine:	23
Andmete kustutamine:	28
Autentimine. Sisse- ja väljalogimine.....	28
3.2. Ärireeglid ja andmete kontrollid. 10 reeglit.	28
4. Mis teha siis kui mingi osa ülesande funktsionaalsusest tundub ebaselge, kui midagi ei ole täpselt kirjeldatud?	29

1. Sissejuhatus , ülevaade andmetest ja funktsionaalsusest.



Tegemist on süsteemiga kus hoitakse ettevõtete ja isikute andmeid. Ettevõtte ja isik võivad olla kliendid. Isik võib olla töötaja. Isiku, ettevõtte, töötaja ja kliendiga on seotud tunnuste tüübid. Kui andmebaasi lisatakse subjekt (isik, ettevõtte, klient või töötaja) siis vaadatakse atribuudi tüüpide tabeliust millised tunnused (atribuudid) on sellele subjekti tüübile vaja tabelisse ATRIBUUT lisada. Töötaja ja kliendi tabelid sisaldavad põhimõtteliselt ainult identifikaatoreid millele viidatakse ATRIBUUDI tabelist (ja kõikidest teistest tabelitest kust

on vaja viidata töötajale või kliendile). Aadressid ja kontaktid lisatakse ainult isikule või ettevõttele. Kui isik on klient siis saadakse selle kliendi aadress päringus kätte mööda seoseid **klient->isik->isiku aadress**. Sama kontaktidega.

Rakendus peaks võimaldama

- sisestada muuta ja kustutada isikute ja ettevõtete andmeid, sealhulgas aadresse, kontakte ja atribuute (atribuute peab saama ainult muuta)
- seostada isikuid ettevõtetega
- teha isikuid töötajateks ja lisada töötajatele kontosid
- teha otsinguid subjektide (isikute, ettevõtete, töötajate, klientide) andmete (aadressid, kontaktid, atribuudid) alusel .

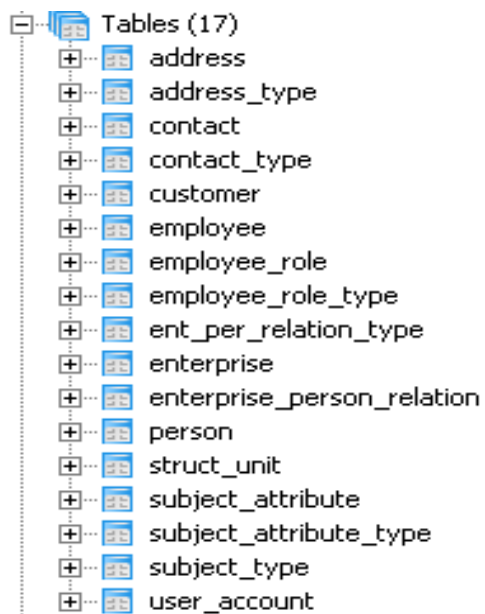
Ülesande sisut ja andmestruktuuridest arusaamiseks vaadake kindlasti ka testandete sisestamise ja näite-päringute skripte (INSERT_DATA_SUBJEKT.txt, SQL_EXAMPLES_SUBJEKT.txt) ja nendest skriptides sisalduvadi kommentaare.

2. Andmed.

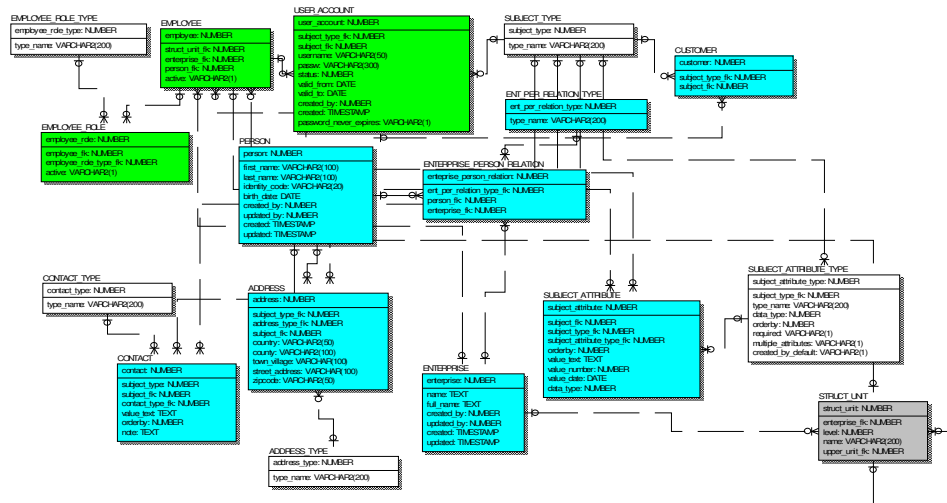
2.1. Andmebaasi loomine.

Selle ülesande andmebaasi loomiseks tuleb andmebaasis käivitada järgmiste failide sisu:

SUBJEKTID/CREATE_DB_SUBJEKT.txt
SUBJEKTID/INSERT_DATA_SUBJEKT.txt



2.2. Andmebaasi skeem.



„sinised” ja „rohelistes” on sellised tabelid kus läbi teie tehtava rakenduse peaks olema võimalik andmeid lisada, muuta, kustutada.

„rohelistes” tabelitega seotud funktsionaalsuse võivad need kes teevad tööd üksi kõrvale jätta.

2.3. Andmebaasi ülesehituse üldised põhimõtted.

* tabelite võtmeväljad on number-tüüpi ja sama nimega mis tabeli nimi.

```
CREATE TABLE doc_catalog_type
( doc_catalog_type numeric(10,0) NOT NULL ,
...
CONSTRAINT doc_catalog_type_pk PRIMARY KEY (doc_catalog_type)
);
```

* suurema osa tabelite puhul tehakse tabelite võtmeväljade väärtused andmebaasisüsteemi poolt (nn. autonumbrid). Erandiks on ainult need tabelid milles on vähe kirjeid ja kuhu kirjeid rakenduse töö jooksul ei lisata (liigid, tüübi – sellised klassifikaatorite tabelid kus on 5-6 kirjet)

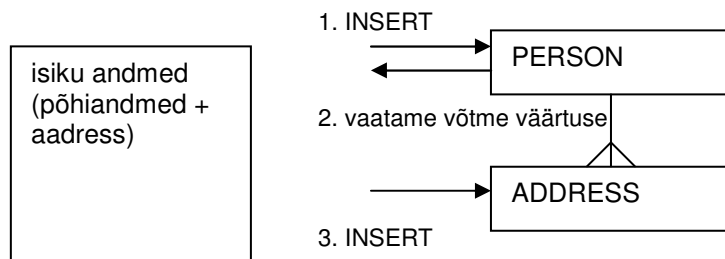
```
CREATE SEQUENCE document_id ;
```

```

CREATE TABLE document
( document numeric(10,0) NOT NULL DEFAULT nextval('document_id'),
..
CONSTRAINT document_pk PRIMARY KEY (document)
);

```

* Autonumbrite tõttu tuleb andmete sisestamisel rakenduses järgida teatud andmete sisestamise (INSERT-lausete) järjekorda – enne tuleb sisestada andmed nendesse tabelutesse millele on vaja teistes kirjetes viidata.

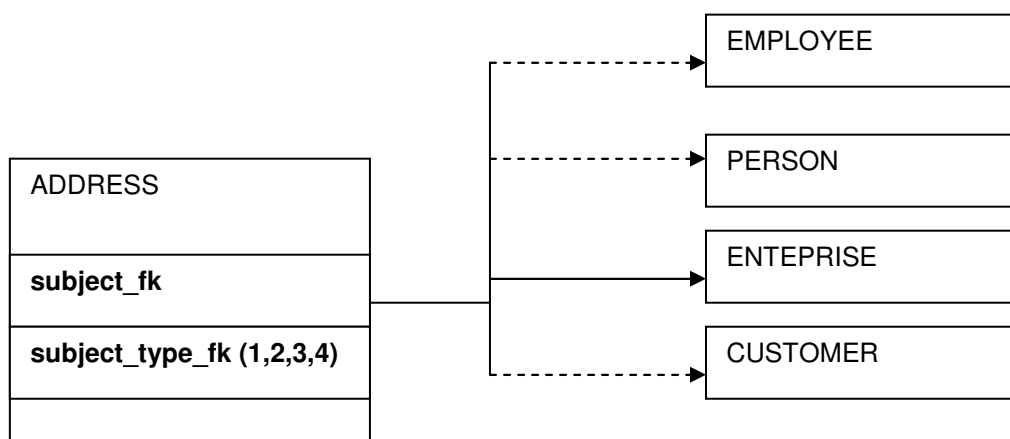


Näiteks : ühe andmevormi pealt isiku ja tema aadressis andmeid sisestades tuleb esimesena sisestada kirje PERSON tabelisse, siis (programmselt) vaadata mis sai PERSON-kirje võtmevälja väärtuseks ja siis sisestada PERSON-iga seotud kirje ADDRESS-tabelisse.


* „foreign key” piiranguid andmebaasides ei ole.

* andmeväljad mille sisu viitab teistele tabelitele (relatsioonid) on nimetatud enamasti nii et nende lõpus on „_fk”.

* mõned relatsioonid võivad viidata erinevatel juhtudel erinevatele andmebaasi tabelitele.



2.4. Andmetabelite ja väljade selgitused.

address	
<p>selles tabelis hoitakse ettevõtete (enterprise) ja isikute (person) aadresse, ühel subjektil (ettevõttel, isikul) võib olla mitu aadressi. Isikul võib olla üks põhiaadress (address_type_fk=1) ja palju lisa-aadresse (address_type_fk=2). Ettevõttel (enterprise) võib olla üks juriidiline aadress (address_type_fk=3) ja palju lisa-aadresse (address_type_fk=2)</p>	
 address	võtmeväli, sisu autonummerduv
subject_type_fk	<p>Viit tabelisse [subject_type] . Näitab millisest tabelist tuleb otsida aadressi omanikku. Kui subject_type_fk= 1 siis viitab väli „subject_fk” tabelisse [person], kui subject_type_fk=2 siis viitab väli „subject_fk” tabelisse [enterprise] . Klientidega (customer) ja töötajatega (person) ei ole vaja rakenduses aadresse siduda, aadressid on alati ettevõtte või isiku „küljes”.</p>
address_type_fk	<p>Viit aadressi tüübile tabelisse [address_type] . Näitab kas tegemist on isiku põhiaadressiga (1), isiku või ettevõtte lisa-aadressiga (2) või ettevõtte juriidilise aadressiga (3)</p>
subject_fk	<p>Viit tabelisse [person] või [enterprise] , näitab kelle/mille aadressiga on tegemist. Millisesse tabelisse konkreetse aadressi kirje puhul viidatakse seda näitab välja „subject_type_fk” sisu.</p>
country	Riik (n. „Eesti”)
county	maakond/rajoon/provints/jne. (n. „Harju”)
town_village	Linn või küla või mõni muu asustatud punkt millel on nimi (n. „Tallinn”)
street_address	Tänav, maja ja korter (n. „Liivalaia 34-15”)
zipcode	postiindeks


address_type

selles tabelis on aadressi tüüpide loetelu.

address_type=1 : isiku põhiaadress

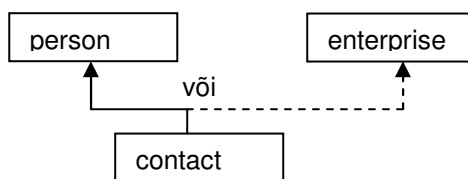
address_type=2 : isiku või ettevõtte lisa-aadress


address_type=3 : ettevõtte juriidiline aadress

 address_type	võtmeväli. sisu ei ole autonummerduv
type_name	Aadressi tüübi nimetus

contact


selles tabelis hoitakse ettevõtete või isikute kontakte. Kontaktideks võivad olla telefonunumbrid või meiliaadressid (seda mis tüüpi kontaktiga on tegemist näitab contact_type_fk).



 contact	võtmeväli, sisu autonummerduv
subject_type	Viit tabelisse [subject_type] . Näitab millisest tabelist tuleb otsida kontakti omanikku. Kui subject_type_fk= 1 siis viitab väli „subject_fk” tabelisse [person] , kui subject_type_fk=2 siis viitab väli „subject_fk” tabelisse [enterprise] . Klientidega (customer) ja töötajatega (person) ei ole vaja rakenduses kontakte siduda, kontaktid on alati ettevõtte või isiku „küljes”.
subject_fk	Viit tabelisse [person] või [enterprise] , näitab kelle/mille aadressiga on tegemist. Millisesse tabelisse konkreetse aadressi kirje puhul viidatakse seda näitab välja „subject_fk” sisu.
contact_type	Viit tabelisse [contact_type] , näitab mis tüüpi kontaktiga on tegemist – kas meiliaadress või telefoninumber
value_text	kontakti sisu : meiliaadress või telefoninumber
orderby	Järjestus mille alusel järjestatakse isiku või ettevõtte kontakte rakenduse ekraanivormidel, väiksema „orderby” väärtusega kontakte näidatakse eespool. Uue kontakti lisamisel arvutatakse valemiga: „uue kontakti order_by”=max(selle isiku viimase seda tüüpi kontakti „orderby”) + 1
note	Märkus, suvaline tekst

contact_type

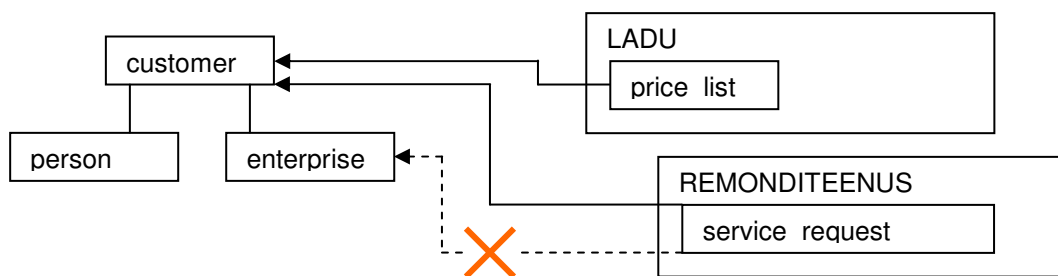
kontakti tüübid. Vaikimisi sisaldab kahte tüüpi – „telefon” ja „meiliaadress”. Kes soovib võib tüüpe sellesse tabelisse juurde lisada.

 contact_type	võtmeväli, sisu ei ole autonummerduv
type_name	Kontakti tüübi nimetus

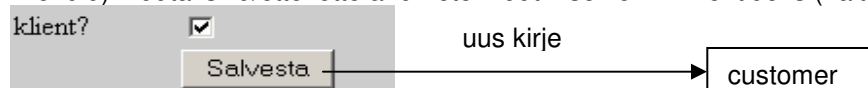
customer

klient. Kui isiku või ettevõtte on klient siis tuleb andmebaasi lisada kirje sellesse tabelisse. „subject_fk” ja „subject_type_fk” näitavad kas tegemist on isiku või ettevõttega ja viitavad vastavale kirjele **[person]** või **[enterprise]** tabelis.

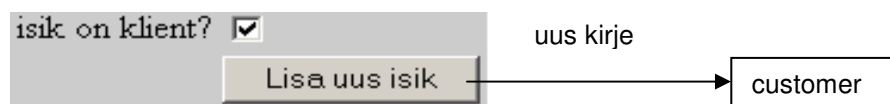
Kõik seosed mis viitavad kõikides ülesande variantides kliendile peavad viitama tabelisse [customer].




Ülesandes SUBJEKTID võib andmebaasis juba olemasolevaid isikuid ja ettevõtteid (kes ei ole veel kliendid) muuta isiku/ettevõtte andmete muutmise vormil klientideks (näiteks nii:



) , sellisel juhul lisatakse tabelisse customer vastav kirje. Aga võib küsida juba uue kirje lisamisel – „kas see isik/ettvõte on ka klient?”

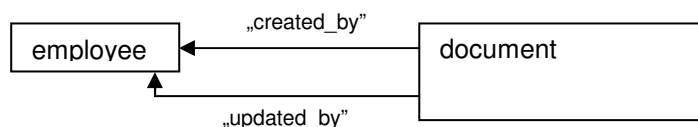


Kui isik/ettevõte on juba klient (customer) siis seda võimalust pole vaja teha et teda klientide hulgast (st. **[customer]** tabelist) uuesti maha saaks võtta – kes on juba tehtud klientideks need jäävad klientideks.

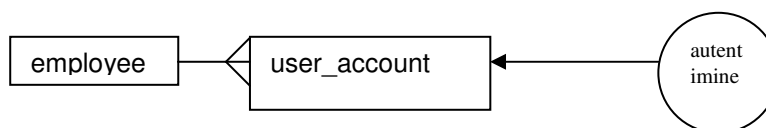
 customer	võtmeväli, sisu autonummerduv
subject_type_fk	Viit tabelisse [subject_type] . Näitab millisest tabelist tuleb otsida kliendi andmeid. Kui subject_type_fk= 1 siis viitab väli „subject_fk” tabelisse [person] , kui subject_type_fk=2 siis viitab väli „subject_fk” tabelisse [enterprise] .
subject_fk	Viit tabelisse [person] või [enterprise] , viitab

	ettevõttele või isikule kes on klient. Millisesse tabelisse konkreetse aadressi kirje puhul viidatakse seda näitab välja „subject_type_fk” sisu.

employee




töötaja andmed. Kõikides ülesande variantide andmebaasides kus on tabelites väljad „updated_by” ja „created_by” viitavad need väljad **[employee]** tabelisse.



autentimisel leitakse
[user_account] kaudu
töötaja id


Kõik ülesande variandid kasutavad sisselogimiseks tabelite **[employee]** ja **[user_account]** andmeid – kui kasutaja sisestab kasutajanime ja parooli siis otsitakse seda kasutajanime ja parooli tabelist **[user_account]** ja kui leitakse siis leitakse ka sellele kasutajakontole vastava **[employee]** identifikaator (**[user_account].employee_fk**). Sisselogimisel leitud viidet „employee” tabeli kirjele kasutatakse vajadusel rakenduses „created_by” ja „updated_by”väljade täitmiseks.

Üldiselt on soovitatav parooli väljas („passw”) hoida paroole krüpteeritult (nt. MD5)

 employee	võtmeväli, sisu autonummerduv
struct_unit_fk	Ei ole kasutusel
enterprise_fk	Viit tabelisse [enterprise] , viit ettevõttele mille töötajaga on tegemist
person_fk	Viit tabelisse [person] , viit isikule kes on töötaja
active	Kas töötaja on selles ettevõtte praegu töötaja (,Y') või on seal kunagu olnud töötaja (,N')


employee_role

töötaja rollid ettevõttes. näitab millistel töötajatel on millised rollid. Selle tabelis sisu otseselt kuskil kasutada (nt. õiguste arvestamisel või sisse logimisel) pole vaja, ülesandes SUBEKTID peab aga rakendus võimaldama töötajatele rolle lisada või neid ära võtta.

 employee_role	võtmeväli, sisu autonummerduv
employee_fk	Viit tabelisse [employee] , viit töötajale kelle rolliga on tegemist
employee_role_type_fk	Viit tabelisse [employee_role_type] , viit rollile (rolli tüübile) mis töötajal on , näiteks „esindaja”, „direktor”, „müügiesindaja” vms.
active	,Y' – roll on aktiveeritud, ,N'- roll on deaktiveeritud.


employee_role_type

ettevõtte töötajate rollide loetelu.

 employee_role_type	võtmeväli, sisu ei ole autonummmmerduv
type_name	Rolli nimi


ent_per_relation_type

ettevõtte ja isiku vahelise seose tüüp. Põhimõtteliselt võiksime seostada isiku ettevõttega ka [employee] tabeli kaudu aga oletame et me tahame (SUBJEKTIDE ülesandes) lisaks siduda ettevõttega ka isikuid kes ei ole ettevõtte töötajad (on näiteks selle ettevõttega seotud advokaadid või pankrotihaldurid) või kelle kohta meil pole oluline mida nad ettevõttes teevad, meid huvitab see mis rolli nad täidavad meie (ettevõttega) suheldes – on näiteks meie jaoks selle ettevõtte „kontaktisikud”. See seose tüüp näitab mis rollis see isik antud ettevõttega seotud on – nt. „pankrotihaldur”, „meie kontaktisik selles ettevõttes”, „juht”, „ettevõtet esindav jurist” vms.

 ent_per_relation_type	võtmeväli, sisu ei ole autonummerduv
type_name	seose tüübi nimi (pankrotihaldur”, „meie kontaktisik selles ettevõttes”, „juht”, „ettevõtet esindav jurist”)

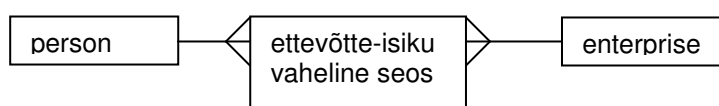
enterprise

ettevõtte

 enterprise	Võtmeväli, ettevõtte id, sisu on autonummmmerduv
name	Ettevõtte nimi
full_name	Ettevõtte pikk, ametlik nimi

created_by	Rakendusse sisseloginud töötaja kelle poolt ettevõtte sisestatud , viit tabelisse [employee]
updated_by	sisseloginud töötaja kelle poolt ettevõtte andmed (sealhulgas aadressid, kontaktid või atribuudid) on viimasena muduetud, viit tabelisse [employee]
created	Ettevõtte sisestamise aeg (andmebaasi) (uuendatakse koos väljaga „created_by“)
updated	Ettevõtte andmete muutmise aeg (uuendatakse koos väljaga „updated_by“)

enterprise_person_relation




„ettevõtet esindav jurist”
„pankrotihaldur”
„meie kontaktisik ettevõttes”

ettevõtte ja isiku vahelise seos. Põhimõtteliselt võiksime seostada isiku ettevõttega ka [employee] tabeli kaudu aga oletame et me tahame (SUBJEKTIDE ülesandes) lisaks siduda ettevõttega ka isikuid kes ei ole ettevõtte töötajad (on näiteks selle ettevõttega seotud advokaadid või pankrotihaldurid) või kelle kohta meil pole oluline mida nad ettevõttes teevad, meid huvitab see mis rolli nad täidavad meie (ettevõttega) suheldes – on näiteks meie jaoks selle ettevõtte „kontaktisikud”. See seose tüüp näitab mis rollis see isik antud ettevõttega seotud on – nt. „pankrotihaldur”, „meie kontaktisik selles ettevõttes”, „juht”, „ettevõtet esindav jurist” vms.

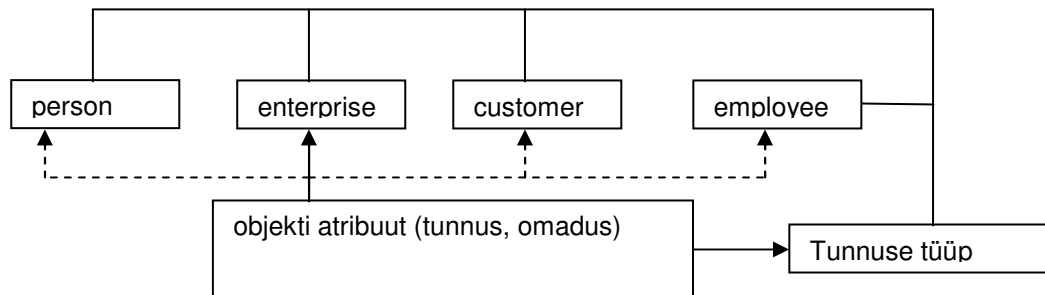
Väli „ent_per_relation_type_fk” näitab mis tüüpi ettevõtte-isiku vahelise seosega on tegemist. Seose tüüpe võib tabelisse **[ent_per_relation_type]** INSERT-lausetega lisada soovi korral, lisaks nendele tüüpidele mis seal (peale testandmete sisestamise skripti käivitamist) juba sees on.

enterprise_person_relation	võtmeväli, sisu autonummerdub
ent_per_relation_type_fk	Viit seose tüübile tabelisse [ent_per_relation_type] , näitab mis rollis antud isik antud ettevõttega seotud on
person_fk	Viit isikule tabelisse [person]
enterprise_fk	Viit ettevõttele millega isik seotud on, viit tabelisse [enterprise]

person	
tabel isikute andmete hoidmiseks	
 person	võtmeväli, sisu autonummerduv
first_name	eesnimi
last_name	perekonnanimi
identity_code	isikukood
birth_date	sünniaeg
created_by	sisselloginud töötaja kelle poolt isik sisestatud , viit tabelisse [employee]
updated_by	sisselloginud töötaja kelle poolt isiku andmed (sealhulgas aadressid, kontaktid või atribuudid) on viimasena muduetud, viit tabelisse [employee]
created	Isiku andmebaasi sisestamise aeg (uuendatakse koos väljaga „created_by“)
updated	Isiku andmete viimase muutmise aeg (uuendatakse koos väljaga „updated_by“)

struct_unit	
ei ole kasutuses	
struct_unit	
enterprise_fk	
level	
name	
upper_unit_fk	

subject_attribute



Ettevõttele, isikule, kliendile ja töötajale on süsteemis (andmebaasis) võimalik defineerida teatud hulga lisatunnuseid, mingeid omadusi mille kohta saab antud objektitüübi (ettevõtte, isiku,..) puhul andmeid salvestada. Näiteks tahame teatud klientide puhul hoida andmeid allahindluse protsendi kohta. See ei ole ettevõtte ega isiku omadus vaid see on kliendi omadus – seega sisestame „Tunnuse tüübi” tabelisse (**[subject_attribute_type]**) uue tüübi – „allahindluse %” ja ütleme et see tunnus käib klientide kohta (subject_type_fk=4).

Kui nüüd kasutaja sisestab andmebaasi uue kliendi (see tähendab kui luuakse uus kirje tabelisse [customer]) siis peab rakendus otsima tunnuse tüüpide tabelist millised tunnused on seotud klientidega (**[subject_attribute_type]**.subject_type_fk=4) ja sisestama seda tüüpi tunnuste kirjed tabelisse **[subject_attribute]**.

Kui nüüd kasutaja (SUBJEKTIDE ülesandes) võtab ette isiku andmete muutmise vormi (ja see isik on ühtlasi ka [customer]) siis peaks see andmete muutmise vorm lisaks isiku põhiaandmete muutmisele (tabelist **[person]**) võimaldama muuta ka selle isiku kui kliendi andmeid mis tegelikult on kirjetena tabelis **[subject_attribute]**. Kasutaja peaks saama muuta **[subject_attribute]** tabeli väljade „value_text”, „value_number” või „value_date” sisu – sõltuvalt sellest mis on selle atribuudi tüübi andmetüüp (**[subject_attribute_type]**.data_type).

data_type=1 (teksti tüüp)

data_type=2 (number tüüp)

data_type=3 (date-tüüp)

Atribuudi lisamisel võib tema andmetüübi dubleerida ka atribuudi enda tabelisse (**[subject_attribute]** tabelis on ka väli „data_type”, selle sisu võetakse vastava atribuudi tüübi andmetest , tabelist **[subject_attribute_type]**)

subject_attribute_type [PK] numeric(10,0)	subject_type numeric(10,0)	type_name character varying(255)	data_type numeric(1,1)	orderby numeric(10,0)	required character varying(255)
1	2	tegevusalad	1	1	N
2	2	asutamise aasta	2	2	N
3	1	Eesti resident	1	1	N
4	2	laste arv	2	2	N
5	4	kliendi huvialad	1	1	N
6	4	allahindluse %	2	2	N

Isiku või ettevõtte 2kliendiks tegemisel” peame küsima:

```
SELECT subject_attribute_type,type_name,data_type,required,orderby FROM
subject_attribute_type WHERE subject_type_fk=4;
```

	subject_attribute_type numeric(10,0)	type_name character varying(200)	data_type numeric(1,0)	required character varying(1)	orderby numeric(10,0)
1	6	allahindluse %	2	N	2
2	5	kliendi huvialad	1	N	1

ehk „anna mulle kõik atribuudi tüübid mis käivad kliendi kohta”.

Ja peame tegema (lisaks kirje sisestamisele **[customer]** tabelisse) 2 INSERT-lauset tabelisse **[subject_attribute]**.


INSERT INTO subject_attribute (subject_fk,subject_attribute_type_fk,subject_type_fk, value_text,data_type,orderby) VALUES (<customer id>,5,4,NULL,1,2);

INSERT INTO subject_attribute (subject_fk,subject_attribute_type_fk,subject_type_fk, value_number,data_type,orderby) VALUES (<customer id>,6,4,NULL,2,2);


Siin on <customer_id> **[customer]** tabelisse sisestatud kliendi kirje identifikaator

Atribuudi väärtuste väljadele („value_text” ja „value_number”) sisestame NULL-id sest nende väljade sisu sisestab/muudab andmete muutmise vormil kasutaja.

Eesnimi:	<input type="text" value="Juhan"/>
Perenimi:	<input type="text" value="Juurikas"/>
----- kliendi andmed -----	
kliendi huvialad:	<input type="text" value="mootorrattad"/>
allahindluse %:	<input type="text" value="10"/>
<input type="button" value="Salvesta"/>	

 subject_attribute	võtmeväli, sisu autonummerdub
subject_fk	Viit tabelitesse [person], [enterprise],[customer] või [employee] – viitab objektile millele antud antribuut kuulub
subject_type_fk	Viit subjekti tüübile tabelisse [subject_type] . subjekti tüüpe on 4 : subject_type_fk= 1 [person] subject_type_fk=2 [enterprise] subject_type_fk=3 [employee] subject_type_fk=4 [customer]
subject_attribute_type_fk	Viit tunnuse tüübile tabelisse [subject_attribute_type]
order_by	Järjestus mille alusel järjestatakse objekti atribuute rakenduse ekraanivormidel, väiksema „orderby” väärtusega atribute näidatakse eespool. Uue atribuudi lisamisel leitakse „orderby” tabelist [subject_attribute_type] – selle tabeli „orderby” väljas on kirjas millises järjekorras tuleks atribuudid järjestada:
value_text	Väärtus mille täidab vormil kasutaja kui atribuudi andmetüüp(data_type) on 1
value_number	Väärtus mille täidab vormil kasutaja kui atribuudi andmetüüp(data_type) on 2
value_date	Väärtus mille täidab vormil kasutaja kui atribuudi andmetüüp(data_type) on 1
data_type	Atribuudi andmetüüp:


	1 - string 2 - number 3 - date

subject_attribute_type objekti tunnuse tüüp. Tunnuse tüüp on seotud objekti tüübiga millele ta kuulub (subject_type_fk). On ettevõtte tunnused (atribuudi tüübid), isiku kohta käivad tunnused, kliendi kohta käivad tunnused ja töötaja kohta käivad tunnused	
 subject_attribute_type	võtmeväli, sisu autonummerdub
subject_type_fk	Subjekti tüüp, näitab millise subjekti tüübi tunnusega on tegemist 1 – [person] (isiku tunnus/omadus) 2 - [enterprise] (ettevõtte tunnus/omadus) 3 – [employee] (töötaja tunnus/omadus) 4 – [customer] (kliendi tunnus/omadus)
type_name	Atribuudi tüübi nimi (näiteks „laste arv”, „ettevõtte asutamise aasta”)
data_type	Andmetüüp 1- string 2- number 3- date sõltuvalt atribuudi tüübi andmetüübist salevstatakse vastava atribuudi ([subject_attribute]) väärtused erinevatesse väljadesse – kas „value_text”, „value_number” või „value_date”
orderby	Järjestus mille alusel järjestatakse objekti atribuute rakenduse ekraanivormidel, väiksema „orderby” väärtusega atribuute näidatakse eespool. Uue atribuudi lisamisel leitakse „orderby” tabelist [subject_attribute_type] – selle tabeli „orderby” väljas on kirjas millises järjekorras tuleks subjekti atribuudid järjestada:
required	,Y’ – väärtus nõutud, ,N’- väärtus ei ole nõutud. Selle „required” välja sisu peavad tuleb vaadata rakenduses kui kasutaja salvestab ekraanivormil andmed – kui antud atribuudi tüübil on required=Y siis peab kasutajal olema selle atribuudi väärtus (sõltuvalt andmetüübist value_text/value_number/value_date) olema täidetud. Rakendus peaks salvestamisel kontrollima et väärtus oleks täidetud. <div data-bbox="805 1747 1404 1971"> Eesnimi: <input type="text" value="Juhan"/> Perenimi: <input type="text" value="Juurikas"/> ----- kliendi andmed----- kliendi huvialad: <input type="text" value="mootor-rattad"/> kliendi grupp: <input type="text"/> * täitmata! <input type="button" value="Salvesta"/> </div>

multiple_attributes	Ei ole kasutusel
created_by_default	Alati „Y” – see tähendab et uue subjekti (isik, ettevõtte, töötaja, klient) luuakse alati vaikimisi kohe kõik atribuudi mis sellele subjekti tüübile tabelis [subject_attribute_type] määratud on

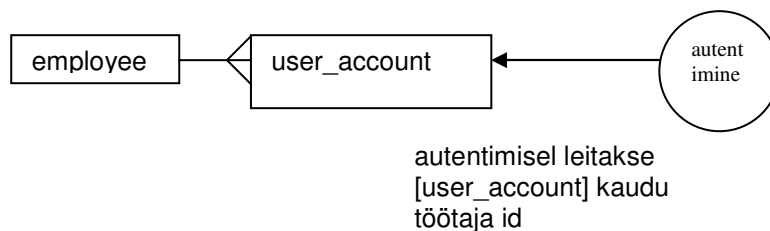
subject_type

subjekti tüüp. Siin on 4 kirjet – subjekt võib olla kas isik mis on salvestatud tabelisse **[person]**, ettevõtte mis on salvestatud tabelisse **[enterprise]**, töötaja (kes alati on ka isik) mis on salvestatud tabelisse **[employee]**, klient (kes alati on kas ettevõtte või isik) mis on salvestatud tabelisse **[customer]**

 subject_type	võtmeväli, sisu ei ole autonummerduv
type_name	Subjekti tüübi nimi („isik, „ettevõtte, „töötaja”, „klient”)


user_account

töötaja kasutajakonto. Selleks et rakendusse sisse logida peavad tabelisse **[employee]** olema sisestatud töötajad ja nendel töötajatel (kes saavad sisse logida) peab olema üks konto tabelis **[user_account]**



Kõik ülesande variandid kasutavad sisselogimiseks tabelite **[employee]** ja **[user_account]** andmeid – kui kasutaja sisestab kasutajanime ja parooli siis otsitakse seda kasutajanime ja parooli tabelist **[user_account]** ja kui leitakse siis leitakse ka sellele kasutajakontole vastava **[employee]** identifikaator (**[user_account].employee_fk**). Sisselogimisel leitud viidet „employee” tabeli kirjele kasutatakse vajadusel rakenduses „created_by” ja „updated_by”väljade täitmiseks.

Üldiselt on soovitatav parooli väljas („passw”) hoida paroole krüpteeritult (nt. MD5)

 user_account	võtmeväli, sisu autonummerduv
subject_type_fk	alati 3 (see tähendab et „subject_fk” viitab alati tabelisse [employee])
Subject_fk	Viit tabelisse [employee] – töötaja kelle kontoga on tegemist

username	kasutajanimi
passw	Parool. Soovitav hoida krüpteeritult. Kui parooli hoitakse andmebaasis krüpteeritult siis autentimisel krüpteeritakse sama algoritmiga (nt. MD5) ära ka kasutaja sisestatud parool ja kontrollitakse nende võrdsust.
status	Kas konto on kehtiv või mitte
valid_from	Konto kehtivus algus (võib langeda kokku konto loomise ajaga)
valid_to	Konto kehtivuse lõpp (kui täitmata siis tähendab et kehtib määramata aja)
created_by	Kelle poolt konto on loodud , viit tabelisse [employee]
created	Konto loomise aeg
password_never_expires	,Y' – parool ei aegu, 'N' – parool aegub (mingi aja pärast, millal aegub pole oluline siin)

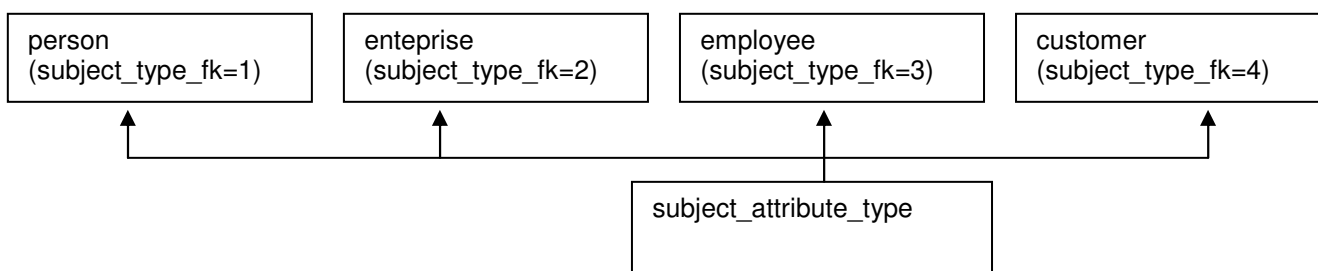
3. Rakenduselt oodatav funktsionaalsus

3.1. Funktsionaalsuste loetelu

Rakendus peaks sisaldama ekraanivorme ja funktsionaalsust mis võimaldavad teha alljärgvaid tegevusi:

Subjekti tüüpide ettevalmistamine INSERT-lausetega:

Lisage tabelisse [subject_attribute_type] iga subjekti tüüpi (see tähendab isiku, ettevõtte, töötaja ja kliendi) jaoks vähemalt 5 atribuudi tüüpi.



Kui teie rakendus sisestab kirje tabelitesse [person], [enterprise], [customer] või [employee] siis ta loeb tabelist [subject_attribute_type] millised atribuudi peavad olema sellel subjekti tüübil ja lisab vastavad atribuudid tabelisse [subject_attribute]

Ütleme näiteks et kõikidel klientidel on andmebaasis atribuudid „kliendi huvialad” ja „allahindluse %”.


```
INSERT INTO subject_attribute_type
(subject_type_fk,type_name,data_type,multiple_attributes,orderby,required) VALUES (4,'kliendi
huvialad',1,'N',1,'N');
INSERT INTO subject_attribute_type
(subject_type_fk,type_name,data_type,multiple_attributes,orderby,required) VALUES (4,'allahindluse
%',2,'N',2,'N');
```

Atribuudi tüüpe ei pea saama rakendusest sisestada ega muuta, pange INSERT-lausetega otse andmebaasi iga subjekti tüübi jaoks 4-5 atribuudi tüüpi.

Andmete lisamine:

1. Subjektide lisamine. subjektide (isikute ja ettevõtete) sisestamine (lisamine). Subjektide kui töötajate ja klientide lisamine.

1.1. Subjektidele aadresside lisamine.

1.2. Subjektidele kontaktide lisamine koos kontaktitüübi valiku võimusega.

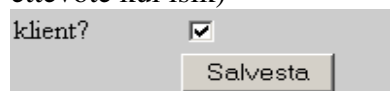
Subketi lisamisel peab koos subjekti põhiandmetega küsima ka ühte aadressi. Andmete sisestamisel andmebaasi saab isiku puhul sellest esimesest aadressist põhiaadress (address_type_fk=1) ja ettevõtte puhul saab põhivormi-aadressist juriidiline aadress (address_type_fk=3). Hiljem lisatavad aadressis saavad isiku ja ettevõtte puhul lisa-aadressideks (address_type_fk=2)

2. Ettevõtete sidumine isikutega. Ettevõttega peab olema võimalik siduda isikuid kes on slele ettevõttega kuidagi seotud. Ekraanivormil näeks see välja umes nii et ettevõtte andmete muutmise vormil on võimalik otsida isikuid ja nad lisada ettevõtte juurde, öeldes mis rollis rad selle ettevõttega seotud on (näiteks: isik on ettevõtte „pankrotihaldur”). Andmed sisestatakse tabelisse [**enterprise_person_relation**] .

Atribuutide lisamine subjektile peale subjekti andmebaasi lisamist.Lisamise vormi pealt andmete muutmise vormile.Subjekti tegemine „kliendiks” ja „töötajaks”.

Subjekti lisamisel võiks kasutaja kõigepealt määrata ära mis tüüpi subjekti ta sisestab. Selle alusel saab genereerida subekti tüübile (isikule, ettevõttele) sobiliku andmete sisestamise vormi. Andmete sisestamise vormil peaks olema võimalik sisestada kliendi põhiandmeid (neid mis lähevad tabelitesse [person] või [enterprise]) ja põhiaadressi või (ettevõttel) juriidilise aadressi andmed.

Andmete lisamisel võib olla valikuks ainult „isik” „ettevõtte” ja „töötaja”. Kliendiks võib muuta juba andmebaasis olevaid isikuid ja ettevõtteid hiljem, andmete muutmise vormil (sisestamisel läheb tülikaks kohe kliendina andmeid sisestada kuna klient võib olla nii ettevõtte kui isik)



Peale andmete sisestamist võiks ekraanivormil näidata kohe sama isiku/ettevõtte/töötaja andmete muutmise vormi.

Eesnimi:	<input type="text" value="Juhan"/>
Perenimi:	<input type="text" value="Juurikas"/>
Sünniaasta:	<input type="text"/>
isikukood:	<input type="text"/>
-----põhiaadress-----	
riik:	<input type="text" value="Eesti"/>
maakond:	<input type="text"/>
linn/küla:	<input type="text"/>
aadress:	<input type="text"/>
postindeks:	<input type="text"/>

Subjekti tüüp:	<input type="text" value="isik"/>
<input type="button" value="Lisa"/>	

```
INSERT INTO person
SELECT FROM subject_attribute_type WHERE subject_type_fk=1
INSERT INTO subject_attribute
```

Kui juba andmebaasi sisestatud isiku andmeid näidatakse andmete muutmise vormil siis tuleks näidata sellel vormil ka juba selle subjekti (antud juhul isiku) atribuutide andmeid.

Eesnimi:	<input type="text" value="Juhan"/>
Perenimi:	<input type="text" value="Juurikas"/>
Sünniaasta:	<input type="text" value="1990"/>
isikukood:	<input type="text" value="62637211"/>
-----põhiaadress-----	
riik:	<input type="text" value="Eesti"/>
maakond:	<input type="text" value="Tartu"/>
linn/küla:	<input type="text" value="Tartu"/>
aadress:	<input type="text" value="Kuuse 78"/>
postindeks:	<input type="text" value="112324"/>
-----isiku atribuudid-----	
Eesti resident:	<input type="text" value="jah"/>
klient?	<input checked="" type="checkbox"/>
<input type="button" value="Salvesta"/>	

Kui isiku on „kliendiks” tehtud ja salvestatud siis lisatakse andmebaasi selle isiku kui kliendi atribuudid ja seotakse kirjega tabelis [customer]

```
INSERT INTO customer
SELECT FROM subject_attribute_type WHERE subject_type_fk=4
INSERT INTO subject_attribute
```

Andmete muutmise vormil on nüüd näha ka kliendi atribuudid.

Eesnimi:	Juhan
Perenimi:	Juurikas
Sünniaasta:	1990
isikukood:	62637211
-----põhiaadress-----	
riik:	Eesti
maakond:	Tartu
linn/küla:	Tartu
aadress:	Kuuse 78
postindeks:	112324
-----isiku atribuudid-----	
Eesti resident:	jah
-----isiku kui kliendi atribuudid-----	
kliendi huvialad:	
allahindluse %:	
klient?	jah
Salvesta	

Isiku atribuudid, viitavad **[person]** tabelisse

kliendi atribuudid, viitavad **[customer]** tabelisse

Muidugi võib ka kohe esimese, andmete lisamise vormil näidata subjekti atribuute (kui kasutaja on subjekti tüübi ära valinud) ja anda kasutajale võimalust määrata et tegemist on kliendiga. Kes peab seda siiski liiga keerukaks siis tehke nii nagu on näidatud nendel ekraanipildidel – atribuute näitate alles andmete muutmise vormil.

Ka töötajaks võib teha juba sisestatud isikut andmete muutmise vormil .

Kui ettevõtte või isik on juba klient siis sellest kliendiks olemisest teda vabastatada ei ole enam vaja (see tähendab anda võimalust kustutada tabelist [customer] kirjet nii et tabelisse [person] või [enterprise] jääb kirje alles). Kes on kliendiks tehtud see jääb kliendiks.

Otsing:

3. Subjektide otsing.subjectide otsing subjekti põhiaandmete (nimi, sünniaeg, isikukood) , aadressi andmete , kontaktide ja atribuudi andmete järgi. Subjektide otsingul peab olema võimalik piirata otsingut nii et otsitakse ainult:

- klientide hulgast
- tootajate hulgast
- isikute hulgast
- ettevõtete hulgast

Korraga peab kehtima ainult üks piirang subjekti tüübi järgi (see tähendab ei pea saama otsida subjektide hulgast kes on isikud ja samaaegselt kliendid).

Kui on otsingul valitud subjekti tüüp (isik, ettevõte, klient, töötaja) siis peab saama otsida ka selle subjektitüübi atribuutide väärtuste hulgast (st. tabelist subject_attribute). Näiteks:

Kui on valitud et otsitakse klientide hulgast ja meil on süsteemis seadistatud nii et klientidel on tabelis [subject_attribute] kaks atribuuti („huvialad” ja „allahindluse %”) siis peab saama otsida ka nende järgi – ja otsinguvorm peaks välja nägema umbes nii:

Eesnimi:
Nimi:
kliendi huvialad:
allahindluse %:
Subjekti tüüp: klient

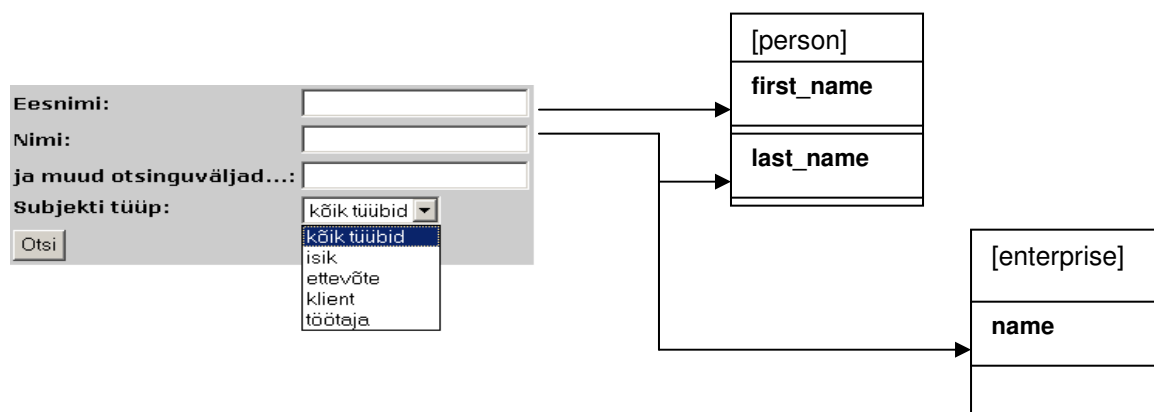
Kuna atribuudi tüübid on võimalik teada saada (ja ekraanil kuvada) alles siis kui kasutaja on subjekti tüübi valinud siis võib vabalt teha nii et kui subjekti tüübi „combo-box-i” sisu muudetakse (valitakse mõni teine subjekti tüüp) siis laaditakse veebileht ümber ja genereeritakse uus otsinguvorm. Aga siin on muidugi ka hea võimalus kasutada AJAX-it et teha selline otsinguvormi väljade muudatus ära ilma veebilehte ümber laadimata.

Kui atribuudi andmetüübiks on number või kuupäev siis nende puhul võib teha ka nii et kasutaja sisestab vahemiku:

allahindluse % vahemikus:

Kohustust ühegi atribuudi andmetest vahemiku järgi otsida siiski ei ole.

Kui otsitakse subjektide hulgast üldiselt või klientide hulgast siis tähendab „Nimi” otsinguparameetrina et otsitakse ettevõtete tabelist [enterprise] väljast „name” järgi ja isikute tabelist [person] väljast „last_name” järgi.



NB! Kõik otsingukriteeriumid päringutes on seotud „AND” tingimusega!

Kõikide otsingukriteeriumite vahel kõikidel otsinguvormidel on „JA” („AND”). See tähendab et siin alumisel vormil toodud otsinguparameetrite korral ei leita andmebaasist midagi.

„Otsi andmebaasist subjekti mille nimi on (alga),Toru’ **JA** mille eesnimi on (algab) ,Juhan’ „

Selline päring tähendab et [enterprise] tabelist ei olegi mõtet midagi otsida kuna ettevõtte ei ole eesnime.

Eesnimi:	<input type="text" value="Juhan"/>
Nimi:	<input type="text" value="Toru"/>
ja muud otsinguväljad...:	<input type="text"/>
Subjekti tüüp:	<input type="text" value="kõik tüübid"/>
<input type="button" value="Otsi"/>	

Kui täidame ära ainult ühe välja („Nimi”) siis saame andmebaasist kätte 2 rida andmeid – ühe ettevõtte ja ühe isiku.

Eesnimi:	<input type="text"/>
Nimi:	<input type="text" value="Toru"/>
ja muud otsinguväljad...:	<input type="text"/>
Subjekti tüüp:	<input type="text" value="kõik tüübid"/>
<input type="button" value="Otsi"/>	

```
SELECT P.subject_id ,P.subject_name, P.subject_type FROM (SELECT person AS subject_id, 'isik' AS subject_type, last_name AS subject_name FROM person WHERE UPPER(last_name) LIKE UPPER('toru%') UNION SELECT enterprise AS subject_id, 'ettevõtte' AS subject_type, name AS subject_name FROM enterprise WHERE UPPER(name) LIKE UPPER('toru%') ) AS P;
```

	subject_id numeric(10,0)	subject_name character varying	subject_type text
1	6	Toru	isik
2	2	Torupood	ettevõtte

NB !Kui mingi otsinguväli on täitmata siis sellist tingimust päringu kriteeriumutesse ei lisatagi

Otsingukriteeriumiteks peab saama sisestada ka kliendi aadresside ja kontaktide andmeid (st. otsida kontaktide ja aadresside järgi).

NB! Otsingu puhul peab kõiki otsingukriteeriumeid saama sisestada ÜHELE otsinguvormile ja otsingi tegemiseks peab olema üks nupp – otsing tehakse siis korraga kõiki TÄIDETUD otsingukriteeriume arvestades, otsingukriteeriumid on omavahel seotud „AND” tingimusega.

Eesnimi:	<input type="text"/>
Nimi:	<input type="text" value="Juurikas"/>
kliendi huvialad:	<input type="text"/>
allahindluse %:	<input type="text" value="10"/>

riik:	<input type="text" value="Eesti"/>
maakond:	<input type="text"/>
linn/küla:	<input type="text"/>
aadress:	<input type="text"/>
postindeks:	<input type="text"/>

kontakt:	<input type="text" value="juurikas@hotmail"/>
Subjekti tüüp:	<input type="text" value="kõik tüübid"/>
<input type="button" value="Otsi"/>	

Kuidas otsida:

Kui tegemist on selliste tekstiväljadega mis sisaldavad ühte stringi (nimi, isikukood, ..) siis võiks otsida sõna alguse järgi , LIKE-tüüpi päringuga.

Kuna PostgreSQL-i LIKE-otsing on vaikimisi tõstutundlik (case-sensitive) siis on mõistlik WHERE-klausli mõlemad pooled näiteks UPPER-funktsiooniga teisendada, tekstiotsingus me tavaliselt ei taha tõstutundlikkust.

Eesnimi:	<input type="text" value="Juh"/>	WHERE UPPER(eesnimi) LIKE UPPER(,Juh%)
----------	----------------------------------	---

Kui tegemist on selliste tekstiväljadega mis sisaldavat pikemat teksti (mitu sõna) siis võiks otsida täistekstiotsinguga mis tähendab et otsitakse kas otsitavad sõnad sisalduvad väljas.Näiteks aadressid, märkused, tegevusalad vms.

kliendi huvialad:	<input type="text" value="vedrud liblikad kastid"/>	WHERE (to_tsvector(SA.value_text) @@ to_tsquery('vedrud'))
-------------------	---	---

Kui tegemist on kuupäevade ja numbritega (summad, aastaarvud) mille puhul on mõistlik otsingul ette anda vahemik siis võiks seda teha (kuid kohustust nii teha ei ole).Seega võib teha nii:

allahindluse % vahemikus:	<input type="text" value="10"/>	<input type="text" value="20"/>	WHERE allahindlus >= 10 AND allahindlus <= 20
---------------------------	---------------------------------	---------------------------------	--

Aga võib teha ka ni:

allahindluse %:

10

WHERE allahindlus = 10

Otsinguvormi sisestatud andmete säilitame pärast otsingu nupule vajutamist:

Otsinguvormile sisestatud andmed peaksid olema sellel otsinguvormil alles ka pärast nupule „Otsi” vajutamist (pärast veebilehe ümberlaadumist). Otsinguvorm ise võiks ka muidugi ekraanile alles jääda. Et uue otsingu tegemiseks ei peaks hakkama otsinguparameetreid uuesti sisestama.

Tegevused töötajaga:

4. Töötajatele (**[employee]**) kasutajakontode tegemine (andmed lähevad tabelisse **[user_account]**). Piisab kui igale töötajale saab teha ühe kasutajakonto. Mõistlik on teha nii et kasutajakonto tegemisel sisestatud parool pannakse andmebaasi (**[user_account].passwd**) krüpteeritult. Mis tähendab ühtlasi ka seda et kasutaja parool tuleb sisestamisel meelde jätta (kui seda näiteks soovitakse praktikatöö kaitsmisel kasutada rakendusse sisse logimisel) sest andmebaasist seda vaadata ei saa, seal on parool juba krüpteeritud kujul. Kasutajakontot peab saama ka kustutada.

5. Töötajatele (**[employee]**) peab olema võimalik lisada rolle tabelisse **[employee_role]** . Rollide loetelu võetakse tabelist **[employee_role_type]** . Rolle peab saama ka kustutada.

Andmete muutmine:

6. Otsingu tulemusel (vt. punkt 3.) leitud subjektide andmeid peab saama teha järgmisi andmete muutmisega seitud tegevusi:

- subjekti põhiandmete, sealhulgas atribuutide väärtuste muutmine (tabelites **[person]** , **[enterprise]** , **[subject_attribute]** ja **[address]**). Põhivormil näidatakse isiku puhul põhiaadressi ja ettevõtte puhul juriidilist aadressi. Rakenduses tuleb teha nii et ettevõtte/isiku/kliendi/töötaja muutmisel peab olema võimalik muuta ka nende atribuute (mis loodi vahetult peale isiku või ettevõtte sisestamist andmebaasi ja vahetult peale isiku/ettevõtte lisamist töötajate/klientide hulka.)

See tähendab et kui subjekt mis tehakse andmete muutmise vormil lahti on isik (tabelis **[person]**) siis tema atribuudi tüübid on:

```
SELECT subject_attribute_type, type_name FROM subject_attribute_type WHERE subject_type_fk=1 ORDER BY orderby ;
```

	subject_attribute_type	type_name
	numeric(10,0)	character varying(200)
1	3	Eesti resident

Kui see isik on lisaks ka veel klient (leidub temale viitav kirje tabelis **[customer]**) siis lisandub talle veel atribuudi tüüpe:

```
SELECT subject_attribute_type, type_name FROM subject_attribute_type WHERE subject_type_fk=4
ORDER BY orderby ;
```

	subject_attribute_type numeric(10,0)	type_name character varying(200)
1	5	kliendi huvialad
2	6	allahindluse %

Kokkuvõttes tähendavad sellised atribuudi päringud et kui isiku sisestamine ja tema „kliendiks” tegemine on rakenduses õigesti tehtud siis peab antu isiku/kliendiga olema seotud andmebaasis 3 atribuuti , tabelist **[subject_attribute]** .

Võtame näiteks andmebaasist välja isiku nr. 4 (kes on ka klient **[customer]** tabelis) atribuudid. Päring on siin nüüd mõnevõrra keerukam, esialgu võime võtta andmed välja kahe päringuga – kliendi atribuudid ja isiku atribuudid.

Isiku atribuudid:

```
SELECT PERSON_ATTRIBUTES.subject_attribute AS attribute_id, PAT.type_name AS type_name,
PERSON_ATTRIBUTES.value_text AS text_value, PERSON_ATTRIBUTES.value_number AS
number_value
FROM person P LEFT JOIN subject_attribute PERSON_ATTRIBUTES ON P.person =
PERSON_ATTRIBUTES.subject_fk
LEFT JOIN subject_attribute_type PAT ON
PERSON_ATTRIBUTES.subject_attribute_type_fk= PAT.subject_attribute_type
WHERE (PERSON_ATTRIBUTES.subject_type_fk=1 OR PERSON_ATTRIBUTES.subject_type_fk IS
NULL)
AND P.person=4;
```

Isiku kui kliendi atribuudid:

```
SELECT CUSTOMER_ATTRIBUTES.subject_attribute AS attribute_id, CAT.type_name AS
type_name,
CUSTOMER_ATTRIBUTES.value_text AS text_value, CUSTOMER_ATTRIBUTES.value_number AS
number_value FROM person P LEFT JOIN customer C ON P.person = C.subject_fk
LEFT JOIN subject_attribute CUSTOMER_ATTRIBUTES ON C.customer =
CUSTOMER_ATTRIBUTES.subject_fk
LEFT JOIN subject_attribute_type CAT ON
CUSTOMER_ATTRIBUTES.subject_attribute_type_fk=CAT.subject_attribute_type
WHERE (C.subject_type_fk=1 OR C.subject_type_fk IS NULL)
AND (CUSTOMER_ATTRIBUTES.subject_type_fk=4 OR
CUSTOMER_ATTRIBUTES.subject_type_fk IS NULL )
AND P.person=4;
```

Tegelikult saame need kaks päringut ühendada UNION-iga ka üheks:

```
SELECT U.person_or_client_attribute, U.attribute_id, U.type_name, U.text_value, U.number_value
FROM
(
SELECT 1 AS person_or_client_attribute, CUSTOMER_ATTRIBUTES.subject_attribute AS
attribute_id, CAT.type_name AS type_name,
```



```

CUSTOMER_ATTRIBUTES.value_text AS text_value,CUSTOMER_ATTRIBUTES.value_number AS
number_value FROM person P LEFT JOIN customer C ON P.person = C.subject_fk
LEFT JOIN subject_attribute CUSTOMER_ATTRIBUTES ON C.customer =
CUSTOMER_ATTRIBUTES.subject_fk
LEFT JOIN subject_attribute_type CAT ON
CUSTOMER_ATTRIBUTES.subject_attribute_type_fk=CAT.subject_attribute_type
WHERE (C.subject_type_fk=1 OR C.subject_type_fk IS NULL)
AND (CUSTOMER_ATTRIBUTES.subject_type_fk=4 OR
CUSTOMER_ATTRIBUTES.subject_type_fk IS NULL )
AND P.person=4
UNION
SELECT 2 AS person_or_client_attribute, PERSON_ATTRIBUTES.subject_attribute
ASattribute_id,PAT.type_name AS type_name,
PERSON_ATTRIBUTES.value_text AS text_value,PERSON_ATTRIBUTES.value_number AS
number_value
FROM person P LEFT JOIN subject_attribute PERSON_ATTRIBUTES ON P.person =
PERSON_ATTRIBUTES.subject_fk
LEFT JOIN subject_attribute_type PAT ON
PERSON_ATTRIBUTES.subject_attribute_type_fk=PAT.subject_attribute_type
WHERE (PERSON_ATTRIBUTES.subject_type_fk=1 OR PERSON_ATTRIBUTES.subject_type_fk IS
NULL)
AND P.person=4 ) AS U ORDER BY U.person_or_client_attribute;

```

Päringu tulemustes „person_or_client_attribute” järgi on võimalik (rakenduses) aru saada kas tegemist on kliendi või isiku atribuudiga et oleks teada millises ekraanivormi osas antud atribuudi andmeid kuvada.

	person_or_client_attribute integer	attribute_id numeric(10,0)	type_name character varying(200)	text_value text	number_value numeric
1	1	6	kliendi huvialad	mootorrrat	
2	1	7	allahindluse %		10
3	2	14	Eesti resident	jah	

Ekraanivorm selliste päringutulemuste korral näeks välja umbes nii:

isiku põhiaandmed-----

Eesnimi:

Perenimi:

isiku atribuudid-----

Eesti resident:

kliendi atribuudid-----

kliendi huvialad:

allahindluse %:

Kõikide otsingutingimuste vahel on alati „JA” (AND).

Ekraanivormile andmete kuvamisel tuleb nüüd arvestada mis on antud atribuudi andmetüüp (**[subject_attribute].data_type**) ja selle alusel otsida päringu tulemuste hulgast õigest väljast (kas „value_text”, „value_number” või „value_date”) väärtus mida vormil kuvada. Andmete salvestamisel tuleb muidugi ka arvestada millisse **[subject_attribute]** andmevälja see tuleb salvestada.

Atribuutide väärtuste muutmisel peab kindlasti arvestama kas selle atribuudi tüübi puhul on väärtus nõutud või mitte (kas **[subject_attribute_type].required = ,Y’** või **,N’**). Kui antud atribuudi tüübi puhul on väärtus nõutud peab kasutaja vormil kindlasti väärtuse kohale midagi kirjutama, kui ta seda ei tee tuleks salvestamisel näidata veateadet .



-----atribuutide jutu lõpp-----

- subjektile lisa-aadresside lisamine, olemasolevate lisa-aadresside andmete muutmine ja subjekti lisa-aadresside kustutamine.
- Subjekti lisa-aadressi muutmine põhiaadressiks (isiku korral) või juriidiliseks aadressiks (ettevõtte korral), olemasolev põhi- või juriidiline aadress muutub sellisel juhul lisa-aadressiks.
- kontaktide lisamine, muutmine, kustutamine.

Aadresside ja kontaktide puhul peaks olema võimalik ühe nupuvajutusega salvestada kõikide aadresside või kontaktide andmeid:

isik	Juhan Juurikas
----- lisa-aadress 1. -----	
riik:	<input type="text" value="Eesti"/>
maakond:	<input type="text" value="Harju"/>
linn/küla:	<input type="text" value="Tallinn"/>
aadress:	<input type="text" value="Metsa 11-3"/>
postindeks:	<input type="text" value="11234"/>
<input type="button" value="Tee põhiaadressiks"/>	
----- lisa-aadress 2. -----	
riik:	<input type="text" value="Eesti"/>
maakond:	<input type="text" value="Tartu"/>
linn/küla:	<input type="text" value="Laeva"/>
aadress:	<input type="text" value="Kuuse 45"/>
postindeks:	<input type="text" value="11444"/>
<input type="button" value="Tee põhiaadressiks"/>	
----- lisa-aadress 3. -----	
riik:	<input type="text" value="Roots"/>
maakond:	<input type="text" value="Stockholm"/>
linn/küla:	<input type="text" value="Stockholm"/>
aadress:	<input type="text" value="Ymama 78"/>
postindeks:	<input type="text" value="55463-378"/>
<input type="button" value="Tee põhiaadressiks"/>	
<input type="button" value="Salvesta"/>	

Salvestab kõikide lisa-aadresside muudatused

Lisa-aadressi peaks saama ühe nupulevajutusega saada teha isiku põhiaadressiks (või ettevõtte juriidiliseks aadressiks), sellisel juhul vahetavad põhiaadress ja lisaaadress kohad – vana põhiaadress muutub lisaaadressiks ja lisaaadress muutub põhiaadressiks. Põhiaadressi andmeid tuleks näidata isiku (või ettevõtte) põhiaadressidega samal vormil (see tähendab vormil kus näidatakse andmeid tabelitest **[person]**/[enterprise] ja **[subject_attribute]**). Põhimõtteliselt muutub kahel aadressil välja „address_type_fk” sisu.

Andmete kustutamine:

Otsingu tulemustena kätte saadud isikuid ja ettevõtteid peab saama ka kustutada. Kustutamise korral tuleks andmebaasist muidugi ka seotud andmed teistest tabelitest [customer],[[subject_attribute], [address],[contact]

Kui isik ([person]) on seotud ettevõttega tabeli [enterprise_person_relation] kaudu siis ei tohiks seda isikut lasta kustutada vaid tuleks kuvada teade et isik on seotud ettevõtte (või ettevõtetega).

Autentimine. Sisse- ja väljalogimine.

Rakendust saab kasutada peale sisselogimist. Kasutajanime ja parooli kontrollimiseks (autentimispäring) kasutatakse andmeid SUBJEKT-i skeemi tabelis [user_account].

Kõik ülesande variandid kasutavad sisselogimiseks tabelite [employee] ja [user_account] andmeid – kui kasutaja sisestab kasutajanime ja parooli siis otsitakse seda kasutajanime ja parooli tabelist [user_account] ja kui leitakse siis leitakse ka sellele kasutajakontole vastava [employee] identifikaator ([user_account].employee_fk). Sisselogimisel leitud viidet „employee” tabeli kirjele kasutatakse vajadusel rakenduses „created_by” ja „updated_by”väljade täitmiseks.

Üldiselt on soovitatav parooli väljas („passwd”) hoida paroole krüpteeritult (nt. MD5).

```
/* tootaja kasutajanimega 'marten' logib süsteemi sisse */  
SELECT E.employee,UA.user_account, P.first_name, P.last_name FROM employee E INNER  
JOIN user_account UA ON E.employee = UA.subject_fk  
INNER JOIN person P ON E.person_fk = P.person  
WHERE UA.subject_type_fk = 3 AND UA.username='marten' AND  
UA.passw='37b4931088193a73b6561bae19bf06d9';
```

Rakenduses peab olema tehtud ka väljalogimine.

3.2. Ärireeglid ja andmete kontrollid. 10 reeglit.

Praktikaülesande üldistes nõuetes on kirjas et tuleb rakendusse teha ka „validaator” tüüpi objektid (klassid) mis kontrollivad sisendandmete õigsust. Millised need reeglid on millele andmed peavad vastama – need mõelge ise välja.

Mõelge välja vähemalt 10 reeglit millele sisendandmed peavad vastama ja kontrollige rakenduses neid reegleid (kui kasutajad andmeid sisestavad või muudavad) ning andke kasutajale ekraanivormidel teada kui sisend-andmed neid kontrolli reegleid ei rahulda.

- millised väljad ei tohi olla tühjad
- milliseid andmeid ei tohi kustutada kuna nende andmetele viidatakse teistes tabelites
- millised summad või kuupäevad peavad jääma teatud piiridesse

- milliste omadustega isikuid või seostada ettevõtetega (tabelisse **[enterprise_person_relation]**), näiteks et sellisel isikul peab olema 2 lisaaadressi ja vähemalt üks meiliaadress
- ja nii edasi, ja nii edasi.

allahindluse %: * ei saa olla üle 100!

Hea oleks kui vähemalt mõned reeglid oleksid ka mõnevõrra keerulisemad ja sisaldaksid näiteks andmebaasi päringuid, et kõik reeglid ei oleks ainult väljade tühjuse või andmeformaadi kontrolliga seotud.

4. Mis teha siis kui mingi osa ülesande funktsionaalsusest tundub ebaselge, kui midagi ei ole täpselt kirjeldatud?

Kui midagi jääb ebaselgeks siis võib alati näiteks harjutustunnis küsida aga võib ka ise otsustada ebaselgetes situatsioonides kuidas rakendus peaks toimima. Ilmselt on mõistlik otsustada nii et peaks vähem programmeerima, et rakenduse toimimine oleks lihtsam .