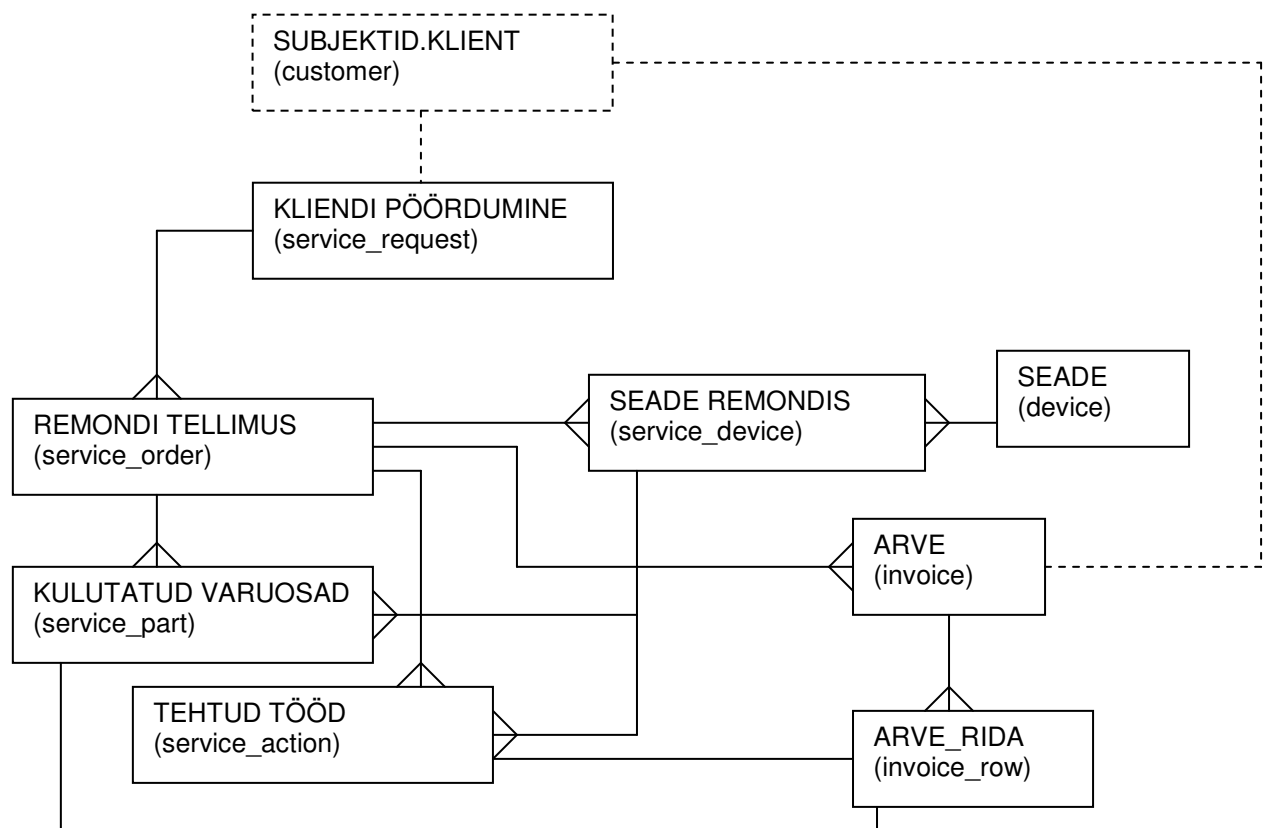


IDU0200

Praktikaülesande „**REMONDITEENUS**” kirjeldus: andmebaasi struktuurid ja rakenduse funktsionaalsuse kirjeldus.

1. Sissejuhatus , ülevaade andmetest ja funktsionaalsusest	2
2. Andmed.	4
2.1. Andmebaasi loomine.	4
2.2. Andmebaasi skeem.....	5
2.3. Andmebaasi ülesehituse üldised põhimõtted.....	5
2.4. Andmetabelite ja väljade selgitused.	6
3. Rakenduselt oodatav funktsionaalsus	24
3.1. Funktsionaalsuste loetelu	24
Kliendi pöördumiste registreerimine ja haldus.	25
Remondi tellimuste registreerimine ja haldus.	25
Arve tegemine.	27
Kliendi sisselogimine süsteemi ja oma tellimuste vaatamine. Märkuste (teadete) sisestamine tellimustele kliendi poolt.Teadete sisestamine tellimustele töötaja poolt.	27
Otsingud.	28
Töötaja autentimine. Sisse- ja väljalogimine.	30
3.2. Ärireeglid ja andmete kontrollid. 10 reeglit.	30
4. Mis teha siis kui mingi osa ülesande funktsionaalsusest tundub ebaselge, kui midagi ei ole täpselt kirjeldatud?	31

1. Sissejuhatus , ülevaade andmetest ja funktsionaalsusest.



Tegemist on remonditöökoja süsteemiga kus peetakse arvet kliendi pöördumiste, vormistatud tellimuste, kulutatud varuosade, tehtud tööde ja kliendile esitatud arvete üle.

Tööprotsess sellises ettevõttes ja tema infosüsteemis käib järgmiselt:

- * tuleb klient kaasas mõned katkised seadmed ja soovib et neid parandatakse.
- * tellimuste vastuvõtja („töötaja” selles süsteemis) registreerib kliendi pöördumise ja paneb tekstina kirja pöördumise sisu (mida parandamiseks toodi, mis kliendi arvates seadmetel viga on, mis tellimuste vastuvõtja arvates seadmetel viga on).
- * töö võidakse tagasi lükata („meie ei remondi selliseid asju”), siis see registreeritakse kliendi pöördumise andmetes.
- * kui töö võetakse tegemiseks (hakatakse kliendi toodud seadmeid remontima) siis regustreeritakse „remondi tellimus”. Tellimusega seotakse seadmed mida parandatakse, kui tegemist on uue seadmega (toodi esimest korda remonti) siis sisestatakse selline seade ka süsteemi seadmete andmebaasi (tabel [device]). Ühe remondi tellimusega võib olla seotud mitu seadet.
- * remondi käigus registreeritakse remondi käigus kulunud varuosad, nende arv ja tüki hind.

* remondi käigus registreeritakse remondi käigus tehtud tööd, nende tööde kogus ja hind. Vaikimisi hinnad võetakse hinnakirjast tabelist [service_type] kus on kirjas töö mõõtühik ja ühiku hind.

* tellimuse kogusumma arvutatakse kulunud varuosade ja tehtud tööde maksumuste summana, seda ekraanivormilt ei sisestata.

* Kui kõik selle tellimuse käigus registreeritud tööd on seisundis „valmis” ja kõik selle tellimuse seadmed on seisundis „töö seadmega lõpetatud” või kinnitavad tellimuse hinna.

* kinnitatud hinnaga tellimusele on võimalik teha arvet. Arve tegemisel saab valida klienti kellele arve tehakse (vaikimisi kliendiks on see klient kellega tellimus (service_order) on seotud. Aga arve juures saab muuta seda) ja sisestada arve maksetähtaega. Muud andmed (arve ridade andmed) võetakse tellimuse andmetest, neid ei sisestata arve ekraanivormilt.

* klient võib ka süsteemi sisse logida ja vaadata seal mingit lihtsat (vähe andmeid) „read-only” nimekirja oma tellimustest. Klient saab tellimuse välja valida ja sisestada selle tellimusega seotud märkust (teadet), samuti peab klient saama lugeda teateid mida on selle tellimusega seoses sisestatud töötaja.

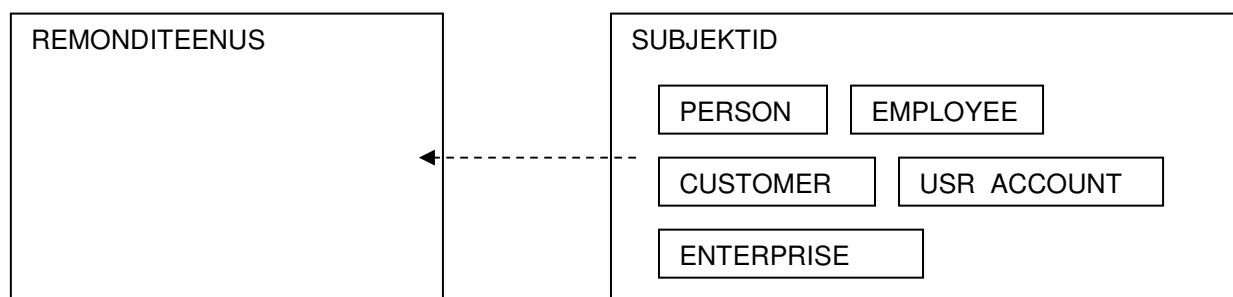
2. Andmed.

2.1. Andmebaasi loomine.

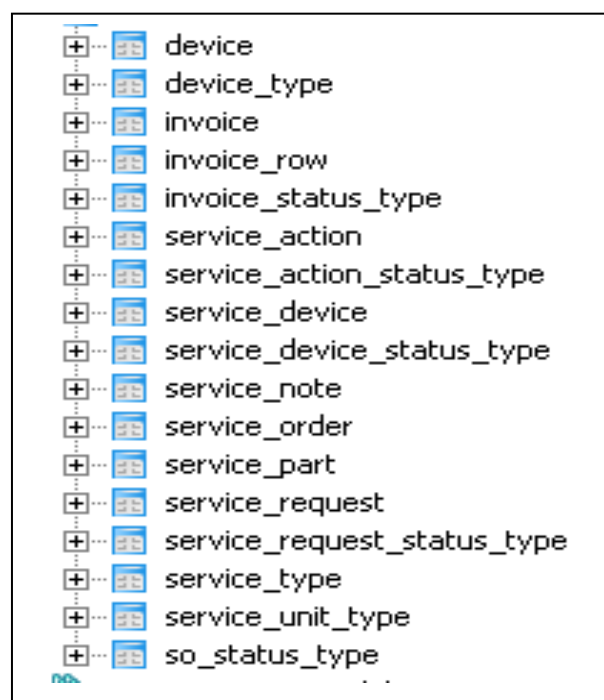
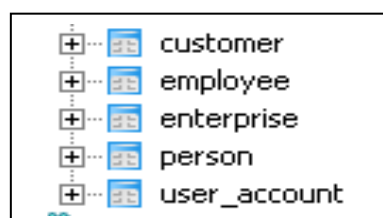
Selle ülesande andmebaasi loomiseks tuleb andmebaasis käivitada järgmiste failide sisu:

SUBJEKTID/CREATE_DB_SUBJEKT.txt
SUBJEKTID/INSERT_DATA_SUBJEKT.txt

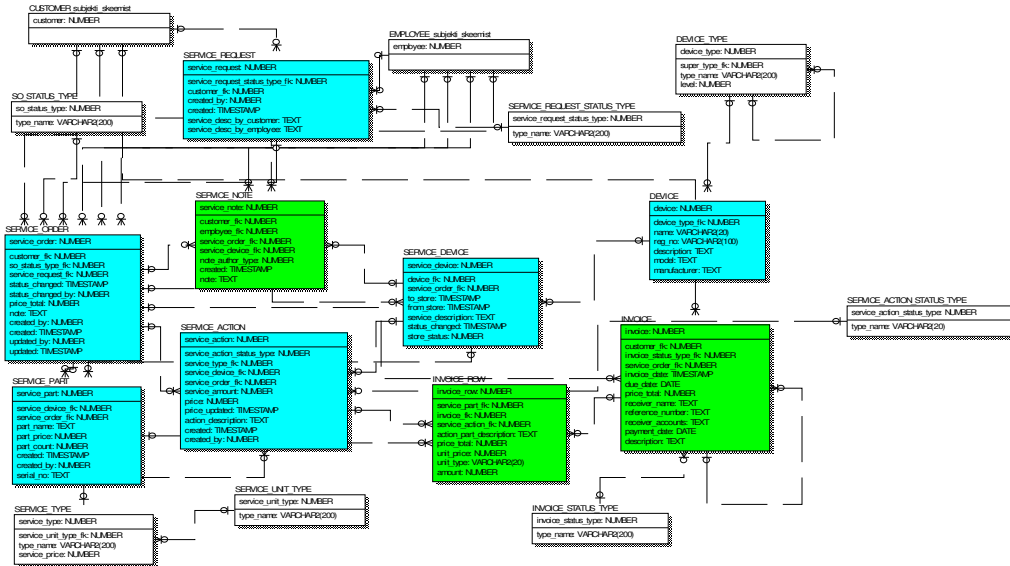
REMONDITEENUS/CREATE_DB_REMONDITEENUS.txt
REMONDITEENUS/INSERT_DATA_REMONDITEENUS.txt



Lisaks ülesande enda andmebaasi skeemi tabelitele (CREATE_DB_REMONDITEENUS) kasutatakse selles ülesandes tabelleid ülesande SUBJEKTID andmebaasi skeemist: PERSON, CUSTOMER, ENTERPRISE, EMPLOYEE, USR_ACCOUNT



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)
);
```

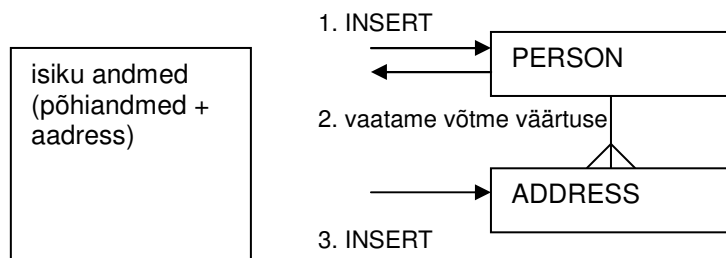
* suurema osa tabelite puhul tehakse tabelite võtmeväljade väärtused andmebaasisüsteemi poolt (nn. autonumbrid). Erandiks on ainult need tabelid milles on vähe kirjeid ja kuhu kirjeid rakenduse töö jooksul ei lisata (liigid, tüübi – sellised klassifikaatorite tabelid kus on 5-6 kirjet)

```
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.

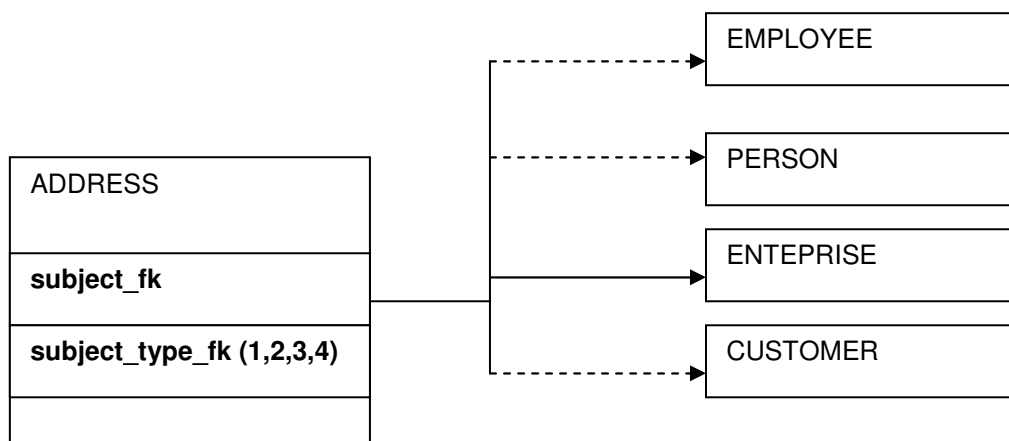


Näiteks : ühe andmevormi pealt isiku ja tema aadressis andmeid sisestades tuleb esimesena sisestada kirje PERSON tabelisse, siis (programselt) vaadata mis sai PERSON-kirje võtmeväli väärtuseks ja siis sisestada PERSON-iga seotud kirje ADDRESS-tabelisse.

* „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.

device

Seade, remonti toodud seade. Remoindi tellimuse **[service_order]** registreerimisel peaks olema võimalik registreerida ka seadmed. Tähele tuleks panna seda et **[device]** on remondi tellimusega **[service_order]** seotud vahetabeli **[service_device]** kaudu. See tähendab et kui sama seade tuleb remonti teist korda siis ei sisestata seda teist korda **[device]** tabelisse vaid lisatakse täiendav kirje **[service_device]** tabelisse. Järelikult peaks remondi tellimuse juures olema võimalik tellimust seadmetega seostada kahel viisil:

1. on võimalik sisestada uus seade (lisatakse kirje tabelisse **[device]** ja kirje tabelisse **[service_device]**)
2. on võimalik otsida seadmete andmebaasist seadmeid (**[device]** tabelist) ja seostada neid seadmeid tellimusega – siis lisatakse ainult kirje tabelisse **[service_device]**

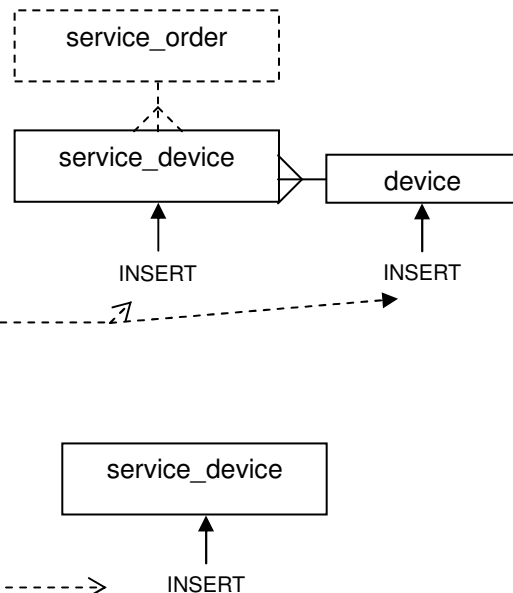
Tellimuse vorm
- lisa uus seade
- otsi seadmeid

1.

SEADME LISAMINE	
nimi:	<input type="text" value="Pesumasin Whirlpool"/>
model:	<input type="text" value="WK88-89T"/>
kirjeldus:	<input type="text" value="see suure trumliga"/>
tootja:	<input type="text" value="Whirlpool"/>
seerianumber:	<input type="text" value="wp6277301289"/>
seadme tüüp:	<input type="text" value="...pesumasinad"/>
<input type="button" value="Lisa"/>	

2.

SEADME OTSING	
nimi:	<input type="text" value="pesumasin"/>
model:	<input type="text" value="WK"/>
seerianumber:	<input type="text"/>
seadme tüüp:	<input type="text" value="...pesumasinad"/>
kliendi nimi:	<input type="text"/>
<input type="button" value="Otsi"/>	
leitud:	
Pesumasin WP768 WK768 lisa seade tellimusse	
Pesumasin Whirlpool WK778 lisa seade tellimusse	



Olemasolevaid seadmeid tuleks otsida ka kliendi nime järgi. Tabel **[device]** ei ole otseselt klientidega seotud – see tähendab et kui otsinguvormi sisestatakse ka kliendi nimi siis see tähendab eraisikust kliendi perekonnanime (SUBJEKTID.[person].last_name) ja ettevõttest kliendi nime (SUBJEKTID.[enterprise].name) ja päringusse tuleb sellisel juhul lisada tabelid

[device] + [service_device] + [service_order] + SUBJEKTID.[customer] ->(SUBJEKTID.[person] + SUBJEKTID.[enterprise])

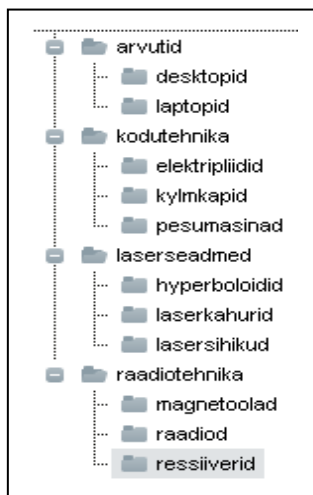
Sest seadmed on klientidega seotud läbi remondi tellimuse **[service_order]** tabeli ja klient võib olla nii isik kui ettevõtte. Sisuliselt tähendab see et otsime seadmete hulgast mis on seotud sellise nimega klientidega ehk seadmete hulgast mille kohta on sellise nimega kliendile koostatud remonditellimus. Sellise päringu tekst tuleb juba üsna pikk, näite SQL-lausetes failis sellise päringu näidet ei ole – tuleb ise teha.


Kui kliendi nime otsinguvormi pole sisestatud siis päringus tabelleid **[service_device]** , **[service_order]** , **[customer]** , **[person]** ja **[enterprise]** ei ole.

Seadmete seosamise remondi tellimusega võib eeldada et remondi tellimus ise on tabelisse

[service_order] juba sisestatud ja tellimuse id on teada – seega peab kasutaja (uue) tellimuse enne andmebaasi ära salvestama – peale seda võib talle anda võimaluse selle tellimuse seadmetega tegelema hakata.

Uue seadme lisamisel tuleb määrata seadme tüüp (seadmete otsingul võib seda määrata aga ei pea). Seadmete tüübid on tabelis [device_type] ja seal on need tüüpid mitmetasemelise hierarhiana („puuna“) salvestatud. Et oleks lihtsam teeme sellise eelduse – seadmete „puu“ on ainult kahetasemeline ja seadme tüübiks sobivad selle „puu“ teise taseme elemendid. See tähendab et seadme lisamisel peab seadme tüübiks valima mitte „kodutehnika“ vaid „pesumasinad“. Eeldame et seadme tüüpide loetelu ei ole väga pikk ja et seda saab „joonistada“ ühe select-boxi valikuteks.



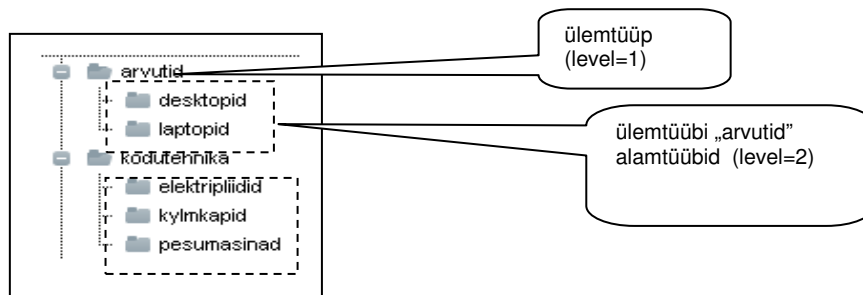
 device	võtmeväli, sisu autonummerdub
device_type_fk	Viit seadme tüübile tabelisse [device_type] . Et oleks lihtsam siis eeldame et seadme tüüpide „puu“ on kahetasemeline ja et konkreetse seadme puhul saab seadme tüübiks olla teise taseme tüüp ([device_type] .level = 2)
name	Seadme nimi, suvaline tekst
reg_no	Seadme registreerimisnumber. Seadme mudel. Seadme mudel on enamasti mingi ühest sõnast koosnev tähtede ja numbrite kombinatsioon ja kui rakenduses teha otsingut seadme mudeli järgi siis võiks see olla nii et otsitakse sõna alguse järgi (WHERE model LIKE „wk%“)
description	Seadme kirjeldus, suvaline tekst
model	Seadme mudel. Seadme mudel on enamasti mingi ühest sõnast koosnev tähtede ja numbrite kombinatsioon ja kui rakenduses teha otsingut seadme mudeli järgi siis võiks see olla nii et otsitakse sõna alguse järgi (WHERE model LIKE „wk%“)
manufacturer	Seadme tootja, tootja nimi


device_type

seadme tüüp. Seadmetel tabelis [device] on tüüp.

Selles tabelis hoitakse tüüpide hierarhiat – on ülemtüübid (level=1) ja nendele viitavad alamtüübid (level=2, super_type_fk viitab ülemtüübile)

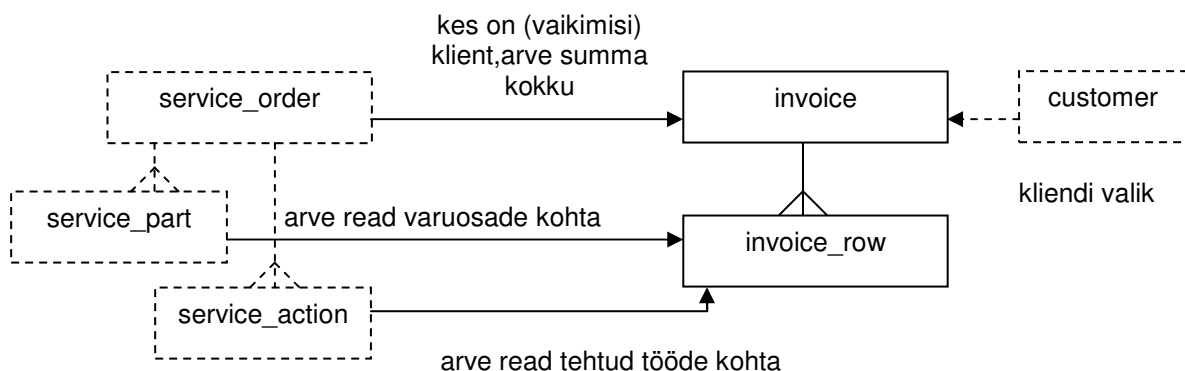
Et oleks lihtsam siis eeldame et seadme tüüpide „puu” on kahetasemeline ja et konkreetse seadme puhul saab seadme tüübiks olla teise taseme tüüp (**[device_type]** .level = 2)



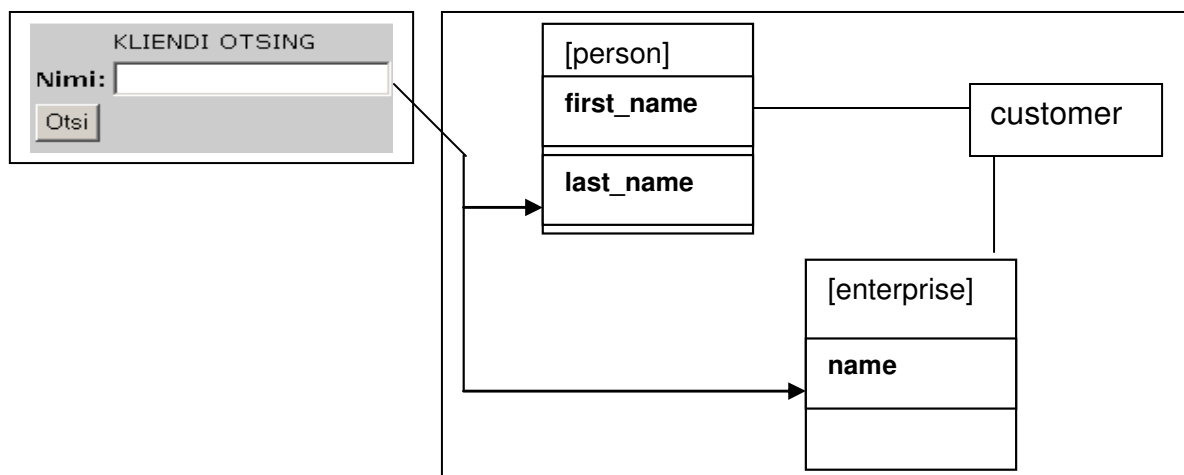
 device_type	võtmeväli, sisu autonummerduv
super_type_fk	Viit ülemtüübile samasse tabelisse [device_type] (kui level=1 siis on 0)
type_name	Seadme tüübi nimi
level	Seadme tüübi tase tüüpide „puus”

invoice

Remondi tellimuse [**service_order**] alusel tehtud arve. Arvega on seotud arve read tabelis [**invoice_row**], iga arve rida sisaldab infot kulutatud varuosa või tehtud töö kohta. Arve tegemisel infot selle kohta mille eest arve esitatakse ei sisestata (summasid ja mille eest) – see info võetakse arve ridadesse ja päisesse (arve summa kokku) remondi tellimuse tabelist [**service_order**], selles tellimuses kulutatud varuodade tabelist [**service_part**] ja selles tellmuses tehtud tööde tabelist [**service_action**]. Ühe remondi tellimusega võib olla seotud mitu tööd ja mitu erinevat tüüpi varuosa. Arve tegemisel läheb iga remondi tellimusega seotud töö ja iga eri tüüpi varuosa eraldi arve reaks – nii et kui remondi tellimuse käigus tehtud 3 tööd ja kulutatu 2 eri tüüpi varuosa siis tekib arvele tabelisse [**invoice_row**] 5 rida.



Arve tegemisel on kasutajal võimalik valida klienti (SUBJEKTID.[customer]) kellele arve tehakse. Vaikimisi võib selleks kliendiks olla see klient kellega on seotud vastav remondi tellimus [**service_order**] aga seda peab saama ka muuta. Uue kliendi valimiseks peab kasutaja saama kliente otsiga aga see otsing võib olla väga lihtne, ainult nime järgi.




Selline nime järgi otsing tähendab et otsitakse ettevõtete hulgast selliseid millel „name” on selline nagu otsinguvormil ja isikute hulgast selliseid kelle „last_name” on selline nagu otsinguvormil. Nii **[person]** kui **[enterprise]** peavad olema kliendid, st. peab olema nendele viitav kirje tabelis **[customer]** (kui asi jääb arusaamatuks lugege ülesande SUBJEKTID juhendist täpsemalt).

Arvet saab teha tellimusele mille seisundiks on 3 („hinnastatud” – ehk et tellimuse hinnad on tabelites [service_part] ja [service_action] üle vaadatud ja kinnitatud).

Kui tellimuse seisund andmebaasis on „hinnastatud” siis näidatakse tellimuse vormil nuppu „Tee arve”. Kas ühele tellimusele võiks arvet lubada teha mitu korda – seda otsustage ise.

Arve tegemise vormile

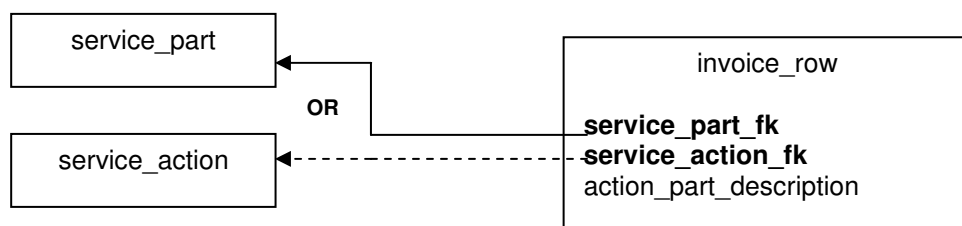
Toimereegel: arvet peaks saama teha vaid siis kui vastava remonditellimuse [service_order] kõik tööd tabelis [service_action] on valmis (st. [service_action].service_action_status_type=2)

 invoice	võtmeväli, sisu autonumbriduv. See on siis arve number selles süsteemis
customer_fk	Viit kliendile skeemis SUBJEKTID, viit tabelisse [customer]
invoice_status_type_fk	Arve staatus , viit tabelisse [invoice_status_type] . Kasutusel on staatused „poolleli” ja „kinnitatud”. Arve vormil võiks olla nupp „Kinnita arve” ja kui sellele nupule vajutada siis muduetakse staatus=2 („kinnitatud”).
service_order_fk	Viit remondi tellimusele tabelisse [service_order] – millise remonditellimuse arvega on tegemist.
invoice_date	Arve kinnitamise kuupäev (timestamp) – täidetakse siis kui vajutatakse nuppu „kinnita arve”
due_date	Arve maksmise tähtaeg kliendile
price_total	Are summa. Ei sisestata vaid võetakse tabelist [service_order].price_total


receiver_name	Arve saaja nimi stringina ([customer] -i nimi mis võetakse kas tabelist [person].first_name + last_name või tabelist [enterprise].name)
reference_number	Arve viitenumber (siia võib panna näiteks remondi tellimuse numbri või kliendi id tabelist [customer])
receiver_accounts	Suvalised stringid – arve saatja (st. „meie” remondiettevõtte arvete numbrid erinevates pankades, kõik ühes väljas. Ei pea sisestama ekraanivormilt, võiks olla mingi konstant mida näidatakse ekraanivormil ja sisestatakse andmebaasi.
payment_date	Arve maksmise kuupäev. Selles ülesandes ei kasutata.
description	Arve selgitus. Võib võtta tabelist [service_order].note aga kasutajal peaks oelma võimalus arve vormi peal seda teksti muuta.

invoice_row


arve rida. Iga arve rida viitab ühele kirjele tabelis **[service_part]** (tellmuses kulunud varuosad) või **[service_action]** (tellimuse käigus tehtud tööd mille eest küsitakse kliendi käest raha)



Andmeid mis sellesse tabelisse pannakse ei sisestata ekraanivormilt vaid võetakse **[service_part]** ja **[service_action]** tabelitest, ekraanivormidel neid muuta ei saa, ainult näidatakse vastava arve vormi peal.

 invoice_row	võtmeväli, sisu autonummerdub
service_part_fk	Viit tellimuses kulunud varusosa kirjele (kui invoice_row_type=1) tabelisse [service_part]
invoice_fk	Viit arvele mille reaga on tegemist , viit tabelisse [invoice]
service_action_fk	Vii tellimuse tööle (kui invoice_row_type=2) tabelisse [service_action]
action_part_description	Arve rea kirjeldus. Kui tegemist on varusosa reaga siis võetakse väärtus tabelist [service_part].part_name . Kui tegemist on tellimuse töö reaga siis võetakse väärtus tabelist [service_action].action_description Ekraanivormil peab selle välja sisu ainult näitama, muuta ei pea saama.

price_total	Arve rea summa kokku. Ainult näitamiseks, muuta ei saa.sisu võetakse tabelist [service_part].part_price * part_count või [service_action].price * service_amount . Ainult näitamiseks, muuta ei saa
unit_price	Ühe ühiku (varuosa või teenuse ühiku) hind. sisu võetakse tabelist [service_part].part_price või [service_action].price Ainult näitamiseks, muuta ei saa
amount	Hulk. Kui tegemist on varuosa reaga siis võetakse sisu tabelist [service_part].part_count , kui tegemist on tellimuse töö reaga siis võetakse sisu tabelist [service_part].service_amount Ainult näitamiseks, muuta ei saa
unit_type	Sisu sõltub arve rea tüübist. Kui on tegemist varuosade reaga siis on siin alati string „tk.” (tükki). Kui on tegemist tellimuse töö reaga siis tuleb tabelist [service_action] vaadata milles seda tööd mõõdetakse ([service_action].unit_type) ja võtta ühikute tabelist mõõtühiku nimetus („tundi”, „minutit”, „tööperatsioon” vms.) Ainult näitamiseks, muuta ei saa
invoice_row_type	1 – siis viitab tabelisse [service_part] (varuosade arve rida) , 2 – siis viitab tabelisse [service_action] (tehtud tööde arve rida). Ekraanivormidel ei näidata.

invoice_status_type	
arve staatus. Selles ülesandes on kasutusel kaks seisundit – 1 („koostamisel”) ja 2 („kinnitatud”).	
 invoice_status_type	võtmeväli, sisu ei ole autonummerduv
type_name	Arve staatuse tüübi nimi

service_action

tellimuse täitmise käigus tehtud töö. Ühe tellimuse käigus võidakse teha mitmu erineva hinnaga tööd, konkeertse töö liik (teenuse liik) on kirjas väljas **service_type_fk**. On seotud tellimusega tabelis **[service_order]** ja võib olla seotud ka tellimusega seotud seadmega tabelis **[service_device]** (et oleks teada millise seadmega seoses see töö tehti). **[service_device]** tabeliga on **[service_part]** seotud siis kui kasutaja tellimuse vormil selle kulutatud varuosa tellimuse seadmega seob

vali tellimuse seade: külmkapp KM788
külmkapp KM788
elektripliit PL9

„Tellimuse töö” andmeid saab muuta (ja uusi töid lisada) tellimuse vormilt – saab muutatöö ühiku hinda ja töö mahtu („kogus”). Koguhind arvutatakse siis süsteemi (rakenduse) poolt.

tellimuse seadme valikuvõimalus ka kuhugi varuosade reale

tellimus nr. 13

summa kokku: 300.00
...muud tellimuse andmed...

töö:	trumpli vahetus	teenus:	raske remont	kogus:	2	[tundi]	ühiku hind:	50	hind kokku:	100
osa:	trummel WT5666			kogus:	1	[tk.]	ühiku hind:	200	hind kokku:	200

Pesumasin Wihrlpool WK778


tellimuse staatus: Hinnastatud Salvesta tellimus Tee arve

service_action


teenuse ühik, tabelitest
[service_type]-
[service_unit_type]

„default” väärtus
[service_type] tabelist, aga
saab muuta tellimuse vormil

kogus * ühiku hind

 service_action	võtmeväli, sisu on autonummerdud
service_action_status_type_fk	Viit tegevuse seisundi tüübile tabelisse [service_action_status_type] , näitab kas töö on veel tegemisel („pooleli”) või on töö tehtud („valmis”). Peaks saama tellimuse vormilt muuta.
service_type_fk	Viide teenuse tüübile (töö liigile, hinnakirjale)

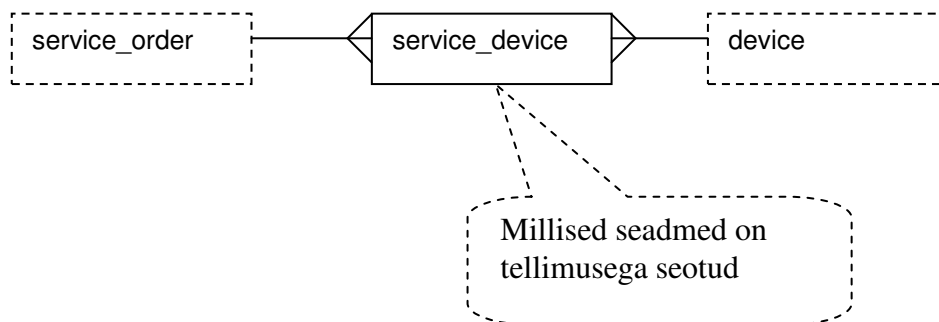
	tabelisse [service_type] . Kui tellimuse tööle on service_type_fk valitud (salvestatud) siis saab tabelist [service_type] võtta vaikimisi töö ühiku hinna.
service_device_fk	Viit tellimuse seadmele mille remontimise käigus see töö tehti, tabelisse [service_device] , võib ka täitmata olla
service_order_fk	Viit remondi tellimusele tabelisse [service_order_fk]
service_amount	Töö maht (ühikutes). Mitu tundi, minutit, tööoperatsiooni kulus
price	Töö ühiku hind. Vaikimisi väärtus võetakse tabelist [service_type] , seda saab teha siis kui kasutaja on teenuse tüübi antud tööle välja valinud (tellimuse vormil). Vaikimisi ühiku hinda võib tellimuse vormil muuta.
action_description	töö lühike kirjeldus
created	Töö sisestamise aeg, timestamp-tüüpi
created_by	Viit sisseloginud töötajale tabelisse SUBJEKTID. [employee] kes tellimuse varuosa sisestas
price_updated	Ei ole selles ülesandes kasutuses

service_action_status_type	
töö seisundi tüüp. Tööl ([service_action]) on kaks tüüpi – „pooleli” või „valmis”	
 service_action_status_type	võtmeväli, sisu ei ole autonummerdud
type_name	Töö seisundi tüübi nimi

service_device

„tellimuse seade”. Näitab

- millised seadmed on tellimusega seotud (see tähendab – milliseid seadmeid selle tellimuse käigus remonditakse)
- näitab milline on seadme remontimise staatus („vastu võetud”, „töö seadmega tehtud”, „kliendile tagastatud”)
- näitab millal on seade ettevõttesse vastu võetud ja millal ta on kliendile tagasi antud



Ilmselt on tellimuste vormile mõistlik teha eraldi alamvorm või „tab” nimega „**tellimuse seadmed**” ja sealt saab siis näha/muuta/lisada tellimusega seotud seadmeid. Kui tellimuse ridades seostatakse mingi tellimuse rida (töö või varuosa) tellimuse seadmega siis hea oleks selliselt „tellimuse seadmete” vormilt näha millised tellimuse read on seotud milliste seadmetega („read-only” vaates , näiteks


- tellimuse seade: külmkapp WP899

Varuosa : trummel TR253 hind 12.- kogus: 1 tk. Hind kokku: 12.-

Varuosa : rihm R11 hind 10.- kogus: 2 tk. Hind kokku: 20.-


Selline tellimuse seadmete vorm oleks nagu teine vaade tellimusele, sellel vormil saaks lisada tellimusele seadmeid ja kustutada selliseid seadmeid mille küljes ei ole varuosasid ja töösid (tabelites [service_part] ja [service_action]), samuti peaks selle vormil saama sistada ja muuta seadme saabumise ja väljaandmise kuupäevi ja seadme remondi staatust.

Toimereegel: seadme staatuseks saab salvestada „tööd seadmega on lõpetatud” ainult siis kui kõik selle seadmega seotud tööd tabelis [service_action] on valmis.

 service_device	võtmeväli, sisu on autonummerduv
service_device_status_type_fk	Viit seadme remondi staatusele tabelisse [service_device_status_type] , peaks saam ekraanivormilt muuta.
device_fk	Viit seadmele tabelisse [device]
service_order_fk	Viit tellimusele tabelisse [service_order]
to_store	Seadme vastuvõtmise aeg, timestamp-tüüpi (kuupäev + kellaaeg minuti täpsusega)
from_store	Seadme väljaandmise aeg, timestamp tüüpi (kuupäev + kellaaeg minuti täpsusega)
store_status	Ei ole selles ülesandes kasutusel
service_description	Ei ole selles ülesandes kasutusel
status_changed	Aeg millal muudeti tellimuse seadme staatust, timestamp tüüpi

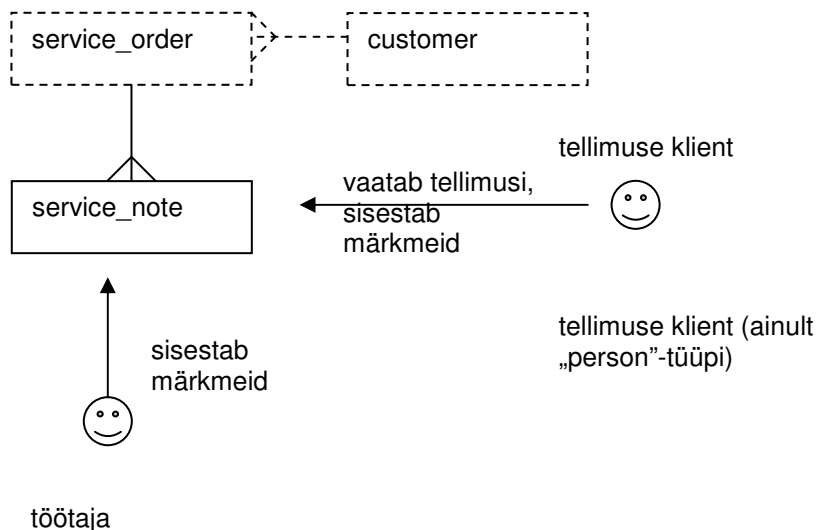
service_device_status_type

seadme remondi staatus. Tellimusega seotud seadme remondi staatuse tüüp – kas seade on „vastu võetud”, „töö seadmega lõpetatud” või „seade kliendile tagastatud”.

 service_device_status_type	võtmeväli, sisu ei ole autonummerduv
type_name	Tellimusega seotud seadme remontimise staatuse nimetus

service_note

kliendi või töötaja poolt sisestatud märkus remondi tellimuse kohta.



Sellesse remonditeenuse süsteemi peaks saama sisse logida ka klient, lihtsuse mõttes eeldame et ainult selline klient kes on eraisik (st. kelle andmed on tabelis „person”). Sellisele kliendile tuleks SUBJEKTIDE skeemi [user_account] tabelisse lisada üks kasutajakonto, näiteks nii:

```
INSERT INTO user_account (subject_type_fk, subject_fk, username, passw,status) VALUES (4,1,'kaarel','kmmm89',1);
```

subject_fk – viit tabelisse [customer], sellisele kliendile kes on eraisik ja kelle andmed omakorda on tegelikult tabelis [person]

Selline klient peaks saama süsteemi sisse logida ja autentimisparing oleks selline:

```
SELECT C.customer,UA.user_account, P.first_name, P.last_name FROM customer C INNER JOIN user_account UA ON C.customer = UA.subject_fk INNER JOIN person P ON C.subject_fk = P.person WHERE C.subject_type_fk = 1 AND UA.subject_type_fk = 4 AND UA.username='kaarel' AND UA.passw='kmmm89';
```

Siin on 'kaarel' ja 'kmmm89' – sisselogimisvormist saadud kasutanime ja parooli andmed (lihtsuse mõttes teeme siin näites praegu nii et parool ei ole andmebaasis krüpteeritud).

UA.subject_type_fk=4 tähendab et tegemist on kliendi kasutajakontoga ja subject_fk viitab tabelile **[customer]**.


C.subject_type_fk=1 näitab et tegemist on sellise kliendiga kes on eraisik ja [customer].subject_fk viitab tabelisse **[person]**.

	customer numeric(10,0)	user_account numeric(10,0)	first_name character varying(100)	last_name character varying(100)
1	1	4	Kaarel	Klient

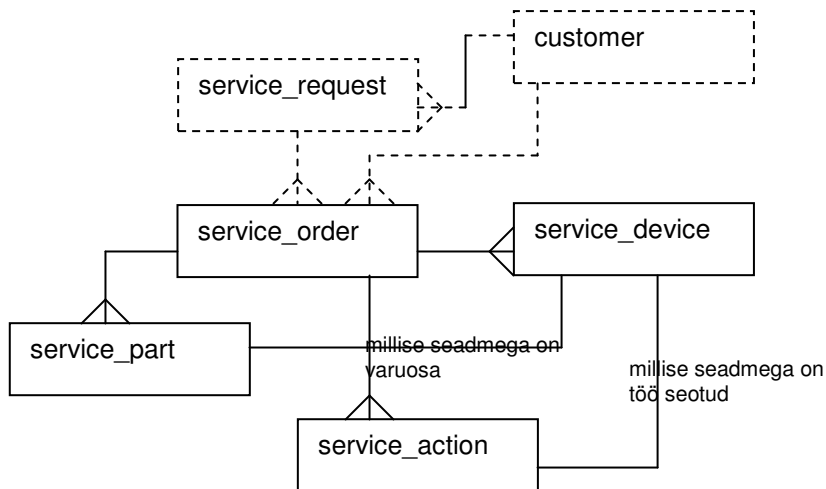
Selline sisselogitud eraisikust klient peaks nägema oma tellimuste kohta andmeid (mõelge mida ta võiks oma tellimuste kohta näha – tellimuse staatust, kogu summat, tellimusega seotud arvete andmeid – otsustage ise mida oleks kliendil kasulik näha).

Kui klient on mingi tellimuse välja valinud siis peaks ta nähema selle tellimuse kohta töötajate poolt siia tabelisse sisestatud märkmeid (sisuliselt mingid teated) ja peaks saama ka oma tellimuse kohta saada märkmeid sisestada (see on siispõhimõtteliselt väga lihtne infovahetuse võimalus tellimuse täitja ja kliendi vahel).

Teiselt poolt – ka töötajal peaks olema võimalik tellimuse juurde sisestada märkemdi (teateid).

 service_note	võtmeväli, sisu on autonummerduv
customer_fk	Viit kliendile kes on teate sisestaja, viit tabelisse SUBJEKTID. [customer] . Täidetud siis kui tegemist on kliendi teatega.
employee_fk	Viit sisselogitud töötajale kes on teate sisestaja, viit tabelisse SUBJEKTID. [employee] . Täidetud siis kui tegemist on töötaja sisestatud teatega.
service_order_fk	Viit remondi tellimusele tabelisse [service_order]
service_device_fk	Ei ole selles ülesandes kasutusel
note_author_type	1 – kliendi teade , 2 – töötaja teade
created	Kirje sisestamise aeg, timestamp tüüpi
note	Teate sisu, mingi tekst

service_order



remondi tellimus. Selle tabeli kohta võib öelda et see on sele andmebaasi kõige tähtsab objekt, selle süsteemi põhiohjekte hoitakse siin tabelis.

Kui otsustatakse kliendi pöördumise (**[service_request]**) alusel seade või seadmed remonti võtta siis vormistatakse remondi tellimus. Remondi tellimuse registreerimise käigus registreeritakse:

- klient kelle tellimusega on tegemist
- seadmed mis on tellimusega seotud (vaata tabelite **[device]** ja **[service_device]** selgitusi
- seos remondi tellimusega (see tekib rakenduse kasutaja poolt vaadates automaatselt sest remondi tellmust saab hakata vormistama kliendi pöördumise vormilt)

kliendi pöördumine nr. 40

Klient: Juhan Juurikas

.....

Kliendi kirjeldus: Kolkub hirmsasti, ei pese ka eriti

vastuvõtja kirjeldus: Pesumasin whirlpool wp788 , tundub et trummel läbi

Tellimuse tegemise vormile

Hiljem (kui mõelda reaalse tööprotsesis mõttes, ajaliselt), tellimuse täitmise käigus registreeritakse lisaks tellimuse käigus tehtud tööd ja nende maksumused (tabelisse **[service_action]**) ja kulunud varuosad (tabelisse **[service_part]**) tehtud tööde ja kulunud varuosade maksumuste summa annab tellimuse kogumaksumuse. Tellimuse vormi lihtsustatud pilt võiks olla umbes selline:

tellimus nr. 13

summa kokku: 300.00
...muud tellimuse andmed...

töö: trumli vahetus teenus: raske remont kogus: 2 [tundi] ühiku hind: 50 hind kokku: 100
osa: trummel Wt5666 kogus: 1 [tk.] ühiku hind: 200 hind kokku: 200
Pesumasin Whirlpool WK778

tellimuse staatus: Töö vastu võetud Salvesta tellimus

Töö vastu võetud
Valmis
Hinnastatud
Arve tehtud
Seade tagastatud

ei sisesta vormilt, arvutatav


Tellimuse vormil peaks saama muuta tellimuse real ühiku hinda ja kogust, „hind kokku” peaks olema vormil mittemuudetav vaid arvutatav (kogus * ühiku hind). Arvutamist ei pea tegema brauseris (st. Javascriptis) vaid need „hind kokku” väärtused ja „summa kokku” võib arvutada (uuesti arvutada) siis kui kasutaja on tellimuse andmed salvestanud.

Tellimuse vormil peaks olema võimalik näha ka tellimuse seadmeid (andmed võetakse tabelitest **[device]** ja **[service_device]** , sellel tellimuse vormil peaks olema võimalik seadmeid lisada ja kustutada (tellimuse „küljest”).

Tellimuse ridade juures tuleb anda kasutajale võimalus siduda konkreetset tellimuse rida (tööd või kulutatud varuosa) tellimuse seadmega (et oleks võimalik aru saada millise seadme juures mingit varuosa kasutati või tööd tehti. Kuna tellimuses on vähe seadmeid (1-2) siis seadme valik võiks olla tehtud nii:

vali tellimuse seade: külmkapp KM788
külmkapp KM788
elektripliit PL9

Kui tellimuse real on seade (seadmed) valitud ja tellimuse salvestatakse siis tehakse UPDATE lause tellimuse rea tabelites **[service_part]** ja **[service_action]** , uuendades nendes tabelites välja **service_device_fk** väärtust.

 service_order	võtmeväli, sisu on autonummerduv, tellimuse number selles süsteemis
customer_fk	Viit kliendile tabelisse SUBJEKTID.[customer]. Väärtus võetakse tellimusele vaikselt vastava [service_request] kirje andmetest – aga peab saama muuta (teist klienti valida)
so_status_type_fk	Viit tellimuse staatuse tüübile tabelisse [so_status_type] .
service_request_fk	Viit kliendi pöördumisele tabelisse [service_request]
status_changed	Tellimuse staatuse muutmise aeg, timestamp. Muduetakse suse kui tellimus salvestatakse ja tellimuse staatus on salvestamisel muutunud
status_changed_by	Viit sisseloginud töötajale tabelisse SUBJEKTID. [employee] kelle salvestus viimati tellimuse staatust muutis
price_total	Tellimuse hind (kliendile) kokku. Ei sisestata kasutajavormilt vaid arvutatakse tellimuse käigust tehtud tööde ja kulutatud varuosade summast

	(tabelid [service_action] ja [service_part])
note	
created_by	Viit sisselloginud töötajale tabelisse SUBJEKTID. [employee] kes tellimuse sisestas
created	Tellimuse sisestamise aeg, timestamp
updated_by	Viit sisselloginud töötajale tabelisse SUBJEKTID. [employee] kes viimati tellimuse andmeid salvestas
updated	Tellimuse muutmise aeg, timestamp-tüüpi

service_part

tellimuse täitmise käigus kulutatud varuosa. On seotud tellimusega tabelis **[service_order]** ja võib olla seotud ka tellimusega seotud seadmega tabelis **[service_device]** (et oleks teada millise seadmega seoses see varuosa kulus). **[service_device]** tabeliga on **[service_part]** seotud siis kui kasutaja tellimuse vormil selle kulutatud varuosa tellimuse seadmega seob

vali tellimuse seade:

„Tellimuse varuosa” andmeid saab muuta (ja uusi varuosasid lisada) tellimuse vormilt – saab muuta varuosa hinda ja varuosa arvu! Koguhind arvutatakse siis süsteemi (rakenduse) poolt.


tellimus nr. 13

summa kokku: 300.00
...muud tellimuse andmed...

töö: teenus: kogus: [tundi] ühiku hind: hind kokku:
osa: kogus: [tk.] ühiku hind: hind kokku:
Pesumasin Wihrlpool Wk778
tellimuse staatus:

service_part

kogus * tüki hind

 service_part	võtmeväli, sisu on autonummerdud
service_device_fk	Viit tellimuse seadmele mille remondiks see varuosa kulus, tabelisse [service_device] , võib ka täitmata olla
service_order_fk	Viit remondi tellimusele tabelisse [service_order_fk]
part_name	Varuosa nimi
part_price	Varuosa tüki hind
part_count	Varuosade arv (kui sama nimega varuosa kasutati mitu tükki remondil)
created_by	Viit sisselloginud töötajale tabelisse SUBJEKTID. [employee] kes tellimuse varuosa sisestas

created	Tellimuse varuosa siestamise aeg, timestamp-tüüpi
serial_no	Varuosa seeria number. Kui seeria number on täidetud siis ei tohiks varuosa arv olla suurem kui 1 sest seerianumber identifitseerib reeglina ühte konkreetset varuosa eksemplari („kõvaketas seerianumbriga E525253625”) mitte varuosa tüüpi („krugi M4x8”) ja ühe seerianumbriga ei saa olla rohkem kui üks varuosa. Seega on antud tabeli puhul tegemist tabeliga kus mõnel juhul hoitakse remondil kulunud varuosa tüüpi ja kulunud osade arvu ja teisel juhul konkreetse varuosa eksemplari andmeid (kui seerianumber on täidetud).

service_request

registreeritud kliendi pöördumine. Sisaldab kliendi poolt antud informatsiooni katkiste seadmete kohta ja tellimuse vastuvõtja omapoolset informatsiooni kliendi pöördumise kohta. Kliendi pöördumise registreerimisel eeldame et klient on meie SUBJEKTIDE andmebaasi olemas ja pöördumise registreerimisel otsime nime järgi klientide hulgast sellist keda seostame ekraanivormil antud pöördumisega.

Kliendi otsing toimub nii nagu arve sidumisel kliendiga, vaata tabeli **[invoice]** kirjeldust ülevalt.



service_request

	võtmeväli, sisu on autonummerduv, kliendi pöördumise number selles süsteemis
service_request_status_type_fk	Viit kliendi pöördumise seisundile tabelisse [service_request_status_type] . Näitab mis seisundis on antud kliendi pöördumine – kas pöördumine on registreeritud (algseisund), tellimus on tehtud (tellimuse salvestamisel tuleks automaatselt muuta vastava kliendi pöördumise staatust) või töö on tagasi lükatud („tagasi_lykatud”).

	Tagasi lükkamise (see tähendab – ei võeta sellist seadet parandamiseks) registreerimiseks võib kliendi pöördumise vormi juurde teha nupu „Registreeri tagasilükkamine” vms.
customer_fk	viit kliendile tabelisse SUBJEKTID.[customer]
created_by	Viit sisseloginud töötajale tabelisse SUBJEKTID.[employee] kes kliendi tellimuse registreeris
created	Kliendi pöördumise sisestamise aeg, timestamp tüüpi
service_desc_by_customer	Pöördumise kliendipoolne kirjeldus (mis seadmetel viga on)
service_desc_by_employee	Pöördumise vastuvõtja poolne kirjeldus (mis seadmetel viga on)


service_request_status_type

kliendi pöördumise seisundi tüüp. On kolm tüüpi: „registreeritud”, „tagasi lükatud” (see tähendab et ei võetud tööd ette) ja „tellimus tehtud”.

Pöördumise sisestamisel saab seisundiks „registreeritud”

Kui tehakse pöördumisele vastav tellimus saab pöördumine seisundiks „tellimus tehtud” (st. töö vastu võetud tegemiseks)


Kui kasutaja vajutab pöördumise andmete muutmisel vormil nuppu „tagasi lükatud” siis saab pöördumuse seisundiks „tagasi lükatud”.

 service_request_status_type	võtmeväli, sisu ei ole autonummerdud
type_name	Kliendi pöördumise seisundi nimi

service_type


põhimõtteliselt teenustööde hinnakiri. Sisaldab infot selle kohta


- milliseid tööliike tehakse
- mis on nende tööliikide arvestuse ühikuks („tund”, „minut”, „tööoperatsioon”)
- palju töö ühik kliendile vaikumisi maksab

 service_type	võtmeväli, sisu on autonummerdud
service_unit_type_fk	Viit töö mõõtühikule tabelisse [service_unit_type]
type_name	Töö (teenuse) liigi nimetus
service_price	Töö (teenuse) ühiku (tunni, minuti, tööoperatsiooni) vaikumisi hind (konkreetselt ühiku hinda mingis tellimuses peab saama tellimuse vormil muuta, default-hinnast erinevaks)

service_unit_type

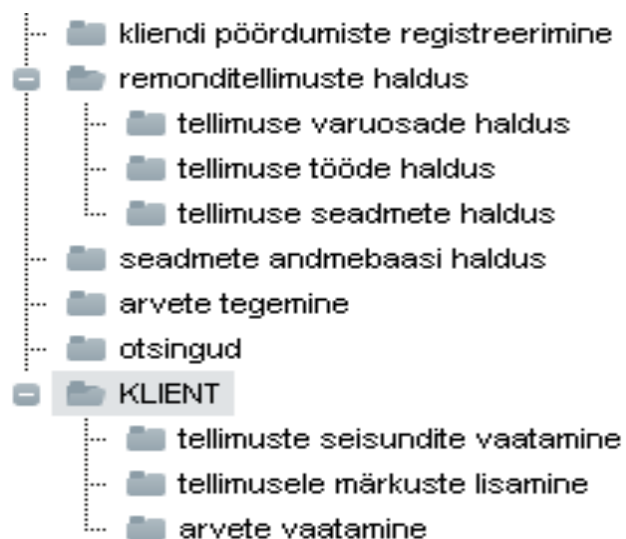
tellimusega seotud tööde mõõtühikute tabel, mõõtühik on seotud töö hinnaga („1 tund teatud teenust maksa 10 EUR-i”) Tööd võib mõõta tundides, minutites või ka näiteks tööoperatsioonidest (st. klient ei maksa mitte töö tegemisele kulutatud aja eest vaid mingi tööoperatsiooni kui terviku tegemise eest, näiteks – „pesumasina trumli vahetamine maksam 100 EUR-i”)

 service_unit_type	võtmeväli, sisu ei ole autonummerdud, ühiku kood
type_name	Töö mõõtühiku nimi („tund“, „minut“, „operatsioon“ vms.)

so_status_type	
tellimuse staatuse tüüpide tabel „s(ervice_)o(rder)_status_type“. Tellimuse staatust peab saama vabalt (see tähendab suvalisest seisundist -> suvalisse seisundisse) tellimuse vormil muuta.	
 so_status_type	võtmeväli, sisu ei ole autonummerdud
type_name	Tellimuse staatuse tüübi nimi

3. Rakenduselt oodatav funktsionaalsus

3.1. Funktsionaalsuste loetelu



Kliendi pöördumiste registreerimine ja haldus.

Kliendi pöördumiste **[service_request]** registreerimisel eeldatakse et klient on andmebaasis (SUBJEKTID.**[customer]**) juba olemas, uue pöördumise registreerimisel otsitakse väga lihtsa otsinguvormi abil (ainult nime järgi otsing) kliendi kes on pöördumisega seotud. Kliendiks võib olla nii isik (selline klient kelle andmed on tabelis **[person]**) kui ettevõtte (selline klient kelle andmed on **tabelis [enterprise]**).

Klienti võib ka hiljem pöördumise andmetes muuta (valida teine klient kes on selle pöördumisega seotud).

Pöördumise andmete vormilt saab vajutada remondi tellimuse **[service_order]** tegemise nuppu või linki, kui andmebaasi on salvestatud kliendi pöördumisele vastav remondi tellimus **[service_order]** siis muutub vastava kliendi pöördumise **[service_request]** staatus.

Kliendi pöördumise vormil peab saama ka registreerida pöördumise tagasilükkamist (see tähendab – seda tööd ei võeta ette ja remondi tellimust tabelisse **[service_order]** ei teki), ka siis muutub vastava kliendi pöördumise staatus tabelis **[service_request]** .

Kliendi pöördumisi peaks saama ka kustutada. Milliseid kliendi pöördumisi lubada kustutada? Kas selliseid kliendi pöördumisi võiks lubada kustutada millega on juba seotud remondi tellimus tabelis **[service_order]** ? Otsustage ise kuidas oleks loogilisem ja tehke rakenduse vastavalt sellele kuidas otsustate.

Remondi tellimuste registreerimine ja haldus.

Uue remondi tellimuse registreerimine. Selle jaoks võiks olla nupp või link vastava kliendi pöördumise **[service_request]** andmete vormil. Lihtsuse mõttes võib rakenduses olla selline loogika et uut tellimust saab alati registreerida mingi välja valitud kliendi pöördumise **[service_request]** alt ja registreeritud tellimus **[service_order]** seostatakse rakenduse poolt alati vastava kliendi pöördumisega **[service_request]**

tellimuse seadmed:

Juba registreeritud ja sisestatud tellumusele peab olema võimalik lisada seadmeid olemasolevast seadmete andmebaasist (otsitakse tabelist **[device]** ja pannakse seose-kirje tabelisse **[service_device]**) aga kuskil tellimuse vormil võiks olla ka nupp või link mis lubaks seoses selle tellimusega sisestada uut seadet tabelisse **[device]** ja kuna see seadme lisamise nupu või lingi vajutamise ajal on teada kasutaja poolt välja valitud tellimus siis lisaks uuele kirjele tabelis **[device]** lisatakse kohe ka seose-kirje tabelisse **[service_device]** . Ühe tellimusega võib seostada mitu seadet (see tähendab – selle tellimuse täitmise käigus remonditakse mitu seadet).

Tellimuse alt peaks saama ka seadmeid kustutada, sellisel juhul kustutakse ära seose-kirje tabelist **[service_device]** , tabelisse **[device]** jäävad seadmed mis on sinna sisestatud alles.

Seadmeid saab tellimuse alt kustutada ainult siis kui nad ei ole selle tellimuse sees juba seostatud tellimuses kasutatud varuosadega **[service_part]** ja tellimuse töödega **[service_action]** .

Tellimuse seadmele peab olema võimalik panna ekraanivormilt seisundit „Töö selle seadmega lõpetatud” (ehk „valmis”). Kui selle tellimuse seadmega on seostatud tellimuse töid [service_action] siis peaks kehtima reegel et sellist seisundit saab panna (salvestada) ainult sellisele seadmele millega seotud tööd on kõik „valmis”-seisundis.



tellimuse varuosad:

Tellimusele peaks saama lisada varuosasid mis kulus selle tellimuse täitmiseks (remondi käigus). Varuosadel mingit liiki või tüüpi ei ole, lisamisel kirjutatakse lihtsalt ekraanivormi varuosa nimi, ühe varuosa hind, varuosa hulk tükides. Ja kui vaja siis ka seerianumber. Kui varuosal on täidetud seerianumbri väli siis peaks tükide arv olema alati 1.

Varuosi peaks saama tellimuse alt ka kustutada. Iga eraldi kirje tellimusega seotud varuosade tabelis annab sisuliselt ühe tellimuse rea mis on samas ka vastav arve rida. Varuosa hinna ja koguse muutmisel (selle muudatuse salvestamisel) tuleb ümber arvutada ja salvestada ka tellimuse kogusumma tabelis [service_order] .

Tellimuse varuosa peaks saama ekraanivormil ka seostada tellimuse seadmega, siis tekib andmebaasi info selle kohta millist varuosa kasutati millise tellimuse seadme remondiks. Aga ei pea seostama, võimalus peaks olema. Kui mingi tellimuse varuosa on seostatud tellimuse seadmega tabelis [service_device] siis ei peaks seda tellimuse seadet tabelist [service_device] saama kustutada.

tellimuse tööd:

Tellimusele peab saama lisada töid (mida selle tellimuse käigus tehakse), need erinevad tööd salvestatakse tabelisse [service_action] . Tööl on hind ja ühik milles seda tööd mõõdetakse („tund”, „tööoperatsioon”, „minut”), need on kirjas tabelis [service_type] . [service_type] tabel on põhimõtteliselt hinnakiri selles süsteemi jaoks. Töö lisamisel tellimusele tuleb valida ka selle töö liik [service_type] ja sellest [service_type] tabelist saab teada töö ühiku ja selle ühiku vaikimisi (default) hinna mida kuvada ekraanivormil. Töö lisamisel või muutmisel võib kasutaja konkeetset tellimuse töö kirjes ühiku hinda muuta (see tähendab muudab [service_type] tabelist saadud „default”-hinna ära, kallimaks või odavamaks).

Tellimuse töö ühiku hinda ja kohust peab saama ekraanivormil muuta. Tellimuse tööd peab saama kustutada.

Tellimuse tööle peab oelma võimalik panna ekraanivormilt seisundit „valmis” .

Tellimuse seisundi muutmine „hinnastatud” seisundisse (et saaks arvet teha):

Kui kõik tellimusega seotud tööd on seisundis „valmis” ja kõigi tellimusega seotud seadmete seisundiks on „töö seadmega lõpetatud” siis peaks saama tellimuse seisundiks panna (salvestada) „Hinnastatud” - see tähendab et tellimuse hinnad on paigas ja tabelites

[service_action] ja [service_part] enam hindu ei muudeta (ehk reaalses elus – keegi töötaja kelle ülesandeks on tellimuste maksumuste kinnitamine on hinnad paika pannud).

Kui remondi tellimuse seisundiks on „hinnastatud” (ehk – „hind kinnitatud”) siis peaks saama sellele tellimusele arvet teha. „Hinnastatud” tellimuses ei peaks saama ekraanivormil enam hindu tabelites [service_action] ja [service_part] muuta.

tellimuse kustutamine:

Peaks oelma võimalik tellimust ka kustutada. Millist tellimust peaks saama kustutada? Millist tellimust ei tohiks olla võimalik kustutada? Kas tellimust millele on juba arve tehtud peaks saama kustutada? Millised kirjed teistest tabelitest tuleks kustutada kui kustutakse tellimuse kirje (sellise tellimuse puhul mida on võimalik kustutada)? Otsustage ise ja tehke rakendus vastavalt sellele mida otsustasite.

Arve tegemine.

NB! Need kes teevad seda ülesannet üksi ei pea arvetega tegelema.

Kui tellimuse seisundiks on „hinnastatud” siis peaks sellisele tellimusele olema võimalik teha arvet (arve tegemise nupp võiks olla siis vastava tellimuse andmete muutmise vormil nähtav).

Arve tegemisel peaks arve vormil oelma võimalik otsida arvele klienti ja sisestada arve maksmise kuupäeva (due_date). Muud andmed peaksid arvele ([invoice]) ja selle arev ridadesse ([invoice_row]) tulema tellimuse andmetest (tabelid [service_order] , [service_part] , [service_action]..)

Kliendi sisselogimine süsteemi ja oma tellimuste vaatamine. Märkuste (teadete) sisestamine tellimustele kliendi poolt. Teadete sisestamine tellimustele töötaja poolt.

NB! Need kes teevad seda ülesannet üksi ei pea seda punkti tegema.

SUBJEKTIDE andmebaasi registreeritud klient kes on isik ([person]) peaks saama oma kasutajanime ja parooliga süsteemi sisse logida ja peaks nägema nimestikuna oma tellimusi (väga lihtne ekraanivorm – tellimuse kood, seisund, sisestamise kuupäev, tellimusega seotud märkused tabelist [service_note] , võib olla veel midagi. Otsinguvormi pole vaja, kõiki tellimusi nädiatakse kohe).

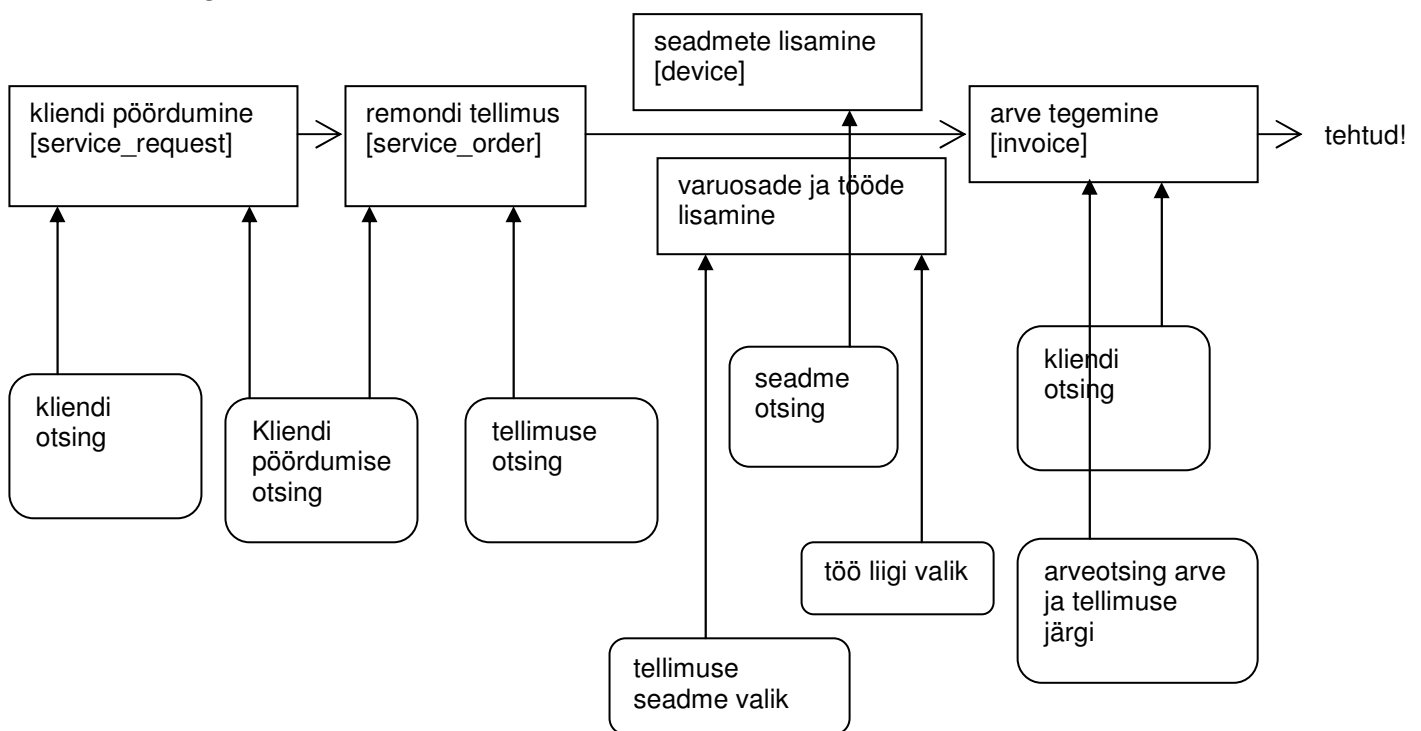
Klient peaks saama sisestada tellimustele omapoolseid märkusi (tabelisse [service_note]) ja nägema selle tellimusega seotud märkusi (teateid) mida on sisestanud töötajad.

Kuidas peaks toimima kliendi autentimine kui ta vajutab „logi sisse” nuppu ja millistest tabelitest võetakse andmed – seda näitab järgmine SQL päring:

```
SELECT C.customer,UA.user_account, P.first_name, P.last_name FROM customer C INNER
JOIN user_account UA ON C.customer = UA.subject_fk
INNER JOIN person P ON C.subject_fk = P.person
WHERE C.subject_type_fk = 1 AND UA.subject_type_fk = 4 AND UA.username='kaarel' AND
UA.passw='kmmm89';
```

Rakendusse sisse loginud töötaja peaks saama panna tellimuste külge omapoolseid teateid. Kes teeb seda ülesannet üksi siis tema rakenduses ei pea töötaja saama tellimusele teateid lisada/vaadata.

Otsingud.



Milliseid otsinguvorme teha ja kuhu rakenduses neid teha – seda otsustate ise. Arvestama peaksite paari asjaoluga.

Ühe remondiga seotud tööprotsessi ei läbita rakenduses korraga ja andmeid reaalses süsteemis on palju:

Kuigi põhimõtteliselt on kõiki ühe remondiga seotud tegevusi võimalik teha järjest, ühel vormilt järgmisele liikudes (registreerime kliendi pöördumise, sealt läheme kohe vastava remondi tellimuse registreerimise/muutmise vormile, tellimuse vormilt läheme vastava arve tegemise vormile) siis reaalses elus asjad nii ilmselt selles remondiettevottes ei käi. Kliendi pöördumise registreerimisest kuni remondi tellimuse registreerimiseni võib minna palju aeg – mis tähendab et mingile kliendi pöördumisele tellimuse tegemiseks on vaja see kleidni pöördumine otsinguvormi abil süsteemist jälle üles leida. Mis tähendab et tellimuse andmete muutmiseks tuleb see otsinguvormi abil süsteemist üles leida. Mis tähendab et arve tegemiseks tuleb vastav tellimus otsinguvormi abil süsteemist jälle üles leida.

Rakenduse tegemisel tuleks ka arvestada kui palju reaalselt töötavas süsteemis mingis andmetabelis on andmeid. Kui teete rakenduse mis näiteks seadmete valiku tarbeks (seadmete lisamine tellimusele) kuvab ekraanile kõik kirjed tabelist [device] siis see vist ei ole töötava rakenduse puhul töötav lahendus – kui andmebaasis on registreeritud tuhandeid seadmeid tuleb seadme valikuks tellimuse külge teha otsing.

Sama jutt käib ka näiteks tellimuste kohta – selleks et teie rakenduses saaks tellimust valida (muutmisvormil lahti teha) tuleb teha tellimuste otsing mitte näidata kõiki andmebaasis registreeritud tellimusi veebilehel sest reaalses rakenduse kus on tuhandeid tellimusi selline lahendus ei tööta.

Valik või otsing?:

Nii tulebki teil teha mõistlik otsus ekraanivormidel valiku või otsingu vahel – otsus sõltub sellest kui suurest hulgast tuleb valida. Näiteks: tellimuse vormil peaks varuosi ja töid olema võimalik seostada selle tellimuse seadmetega. Kuna reaalses elus on tellimuses seadmeid ilmselt vähe (1,2..10) siis võib seadmete valikus teha vabalt nn. „combo-box”-i.

Kui aga tahate seadmete andmebaasist (tabel [device]) otsida tellise „külge” seadet siis peate tegema otsingu – sest reaalses süsteemis on seadmeid registreeritud palju.

vali tellimuse seade: külmkapp KM788
külmkapp KM788
elekripliit PL9

või

SEADME OTSING

nimi: pesumasin

model: WK7

seerianumber:

seadme tüüp: ..pesumasinad

kliendi nimi:

Otsi

leitud:

Pesumasin WP768 WK768
Pesumasin Whirlpool WK778

[lsa seade tellimusse](#)
[lsa seade tellimusse](#)

?

Otsiguparameetrid otsinguvormidel?:

Mille järgi peaks saama süsteemis otsida tellimusi? Arveid? Kliendi pöördumisi? Seda otsustate ise – üldiselt peaks aga otsinguvormides (eriti tellimuse ja arve otsingutes) olema võimalik otsida 4-5 parametri järgi. Mõelge mille järgi sellise süsteemi kasutajad andmeid otsiksid?

Kindlasti peaks tellimusi saama otsida ka tellimustega seotud seadmete [device] andmete järgi (see tähendab – otsitakse tellimusi mis on seotud mingite seadmetega).

Mõned otsiguparameetrid võiksid sisaldada ka mingit vahemikku: otsing mingist kuupäevade vahemikust, otsing mingist summade vahemikust.

Kui aega jääb väheks ja vaatate et tööd on veel teha palju – siis jätke otsinguvormid ja otsingud lihtsamaks. Kui jääb aega üle – tehke otsinguvormidesse võimalisi juurde – nii on kindlam et saate parema hinde.

Töötaja 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 kasutajanimemega 'marten' logib systeemi 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. Mõned reeglid on selle praktikatöö juhendis erinevates kohtades ka juba toodud, neid võib ka 10 hulka arvestada.

Näiteks:

- millised väljad ei tohi olla tühjad
- milliseid andmeid ei tohi kustutada kuna nendele andmetele viidatakse teistes tabelites
- millised summad või kuupäevad peavad jääma teatud piiridesse
- millal võib tellimusele teha arvet
- millal võib tellimust kustutada
- ja nii edasi, ja nii edasi.

seeria nr.: RG6362611 arv: 2 Toote seeria number täidetud. Toote arv ei saa olla suurem kui 1 !

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 .