LATVIJAS UNIVERSITĀTE

DATORIKAS FAKULTĀTE

METADATU VADĪTA STATISTISKO DATU APSTRĀDES SISTĒMA

BAKALAURA DARBS

Autors: **Mareks Indāns**

Studenta apliecības Nr.: mi11015

Darba vadītājs: profesors Dr. dat. Ģirts Karnītis

RĪGA 2015

ANOTĀCIJA

Darbā aprakstīta metadatu vadīta statistisko datu apstrādes sistēma, kuras specifika ir metadatu izmantošana visos pārskatu datu apstrādes procesos. Statistikas pārskatu veidlapu struktūra, saturs un datu apstrādes likumi tiek aprakstītas metadatu bāzē, lietotājiem saprotamā veidā. Šie metadati tiek izmantoti, lai automatizēti ģenerētu datu ievades formas, datu validācijas procedūras, kā arī nodrošinātu visus pārējos pārskatu datu apstrādes procesus. Sistēma ir izstrādāta un tiek izmantota Centrālajā Statistikas Pārvaldē. Darbā tiek apskatīti sistēmas mērķi, sistēmas prasības, kā arī darba autora izstrādātā sistēmas arhitektūra, fiziskie sistēmas parametri un ieguvumi no sistēmas ieviešanas.

Atslēgvārdi:

- metadati;
- pārskats;
- statistika;
- rādītājs;
- mainīgais.

ABSTRACT

Metadata driven statistical data management system.

Work describes metadata driven statistical data management system, which specifics is metadata usage in all statistical survey data processing steps. The layout, structure and validation rules of statistical questionnaires are described in the metadata base, in the way which is understandable for users. These metadata are used to generate data entry forms and data validation procedures. Metadata are used in all surveys data processing steps also. System is developed and installed in the Latvia Central Statistical Bureau. Work describes system objectives, requirements, the system architecture developed by the author, physical system parameters and benefits of system implementation.

Keywords:

- metadata;
- survey;
- questionnaire;
- statistics;
- indicator;
- variable

SATURA RĀDĪTĀJS

TERMINU UN SAĪSINĀJUMU SARAKSTS

Jēdziens	**Skaidrojums**
CSP	Centrālā Statistikas Pārvalde
Pārskats	Konkrētas statistikas tēmas pārskats vispārīgi. Atkarībā no konteksta var iet runa par konkrēta pārskata veidlapu (bez datiem), kā arī kāda respondenta iesniegtu un ar datiem aizpildītu pārskata veidlapu.
Pārskata versija	Konkrēta statistikas pārskata veidlapa, ar saturu, izskatu un struktūru, kas ir spēkā noteiktam gadam. Ja pārskats katru gadu maina saturu, tad katru gadu pārskatam tiek veidota jauna pārskata versija.
Periodiskums	Pārskata datu sniegšanas periodiskums. Katram pārskatam tiek definēts savs datu savākšanas biežums. Biznesa statistikas pārskatiem ir definēti sekojoši pārskatu periodiskumi: G – gads. Pārskatu respondenti sniedz reizi gadā P – pusgads. Pārskatu iesniegšanas biežums ik pa pusgadu C – ceturksnis. Respondentiem aizpildīti pārskati jāiesniedz katru ceturksni M – mēnesis. Pārskats tiek iesniegts katru mēnesi – 12 reizes gada laikā.
Periods	Konkrēta pārskata konkrēts periods. Runa iet par aizpildītu vienu vai vairākām pārskata veidlapām, kas attiecas uz noteiktu laika periodu. Piemēram, pārskats *„2-darbs”* par 2015.g. 1.cetukrsni.
Datu imputācija	Trūkstošu datu aizvietošana, izmantojot dažādas matemātiskas metodes. Trūkstoši dati rodas, ja respondents neiesniedz pārskatu datus par tiem apsekojumiem, kuros uzņēmums ir pievienots respondentu sarakstā.

IEVADS

Darbā tiek apskatīta metadatu vadīta statistisko datu apstrādes sistēma, kurai darba autors izstrādāja sistēmas arhitektūru, kā arī piedalījās sistēmas izstrādē. Ar sistēmas palīdzību specializētā Metadatu bāzē (bez programmēšanas) tiek aprakstīts biznesa statistikas pārskatu veidlapu saturs, struktūra un citi pārskatu datu apstrādes likumi. Pārējie sistēmas moduļi izmanto šos metadatu aprakstus, lai automatizēti ģenerētu datu ievades formas, datu validācijas procedūras, kā arī šie metadati tiek izmantoti pārējos statistikas pārskatu datu apstrādes procesos. Lietotājiem nav nepieciešamas specifiskas programmēšanas zināšanas, lai aprakstītu pārskatu struktūru metadatu bāzē.

Sistēma ir ieviesta Centrālajā Statistikas Pārvaldē un kalpo kā vienīgā biznesa statistikas pārskatu datu apstrādes sistēma, nodrošinot gandrīz visus nepieciešamos datu apstrādes procesus. Darbā tiek apskatīti mērķi, kāpēc šādu datu apstrādes sistēmu bija nepieciešams izstrādāt CSP, kā arī kādas ir galvenās prasības šai sistēmai. Sadaļā „Risinājuma izvēle" apskatīti CSP biznesa procesi, kā arī gatavu risinājumu pieejamība. Šajā sadaļā arī apskatīts profesora Bo Sundgren ieteiktais metadatu modelis statistikas sistēmu izstrādē. Sadaļā „Risinājums" apskatīta darba autora izstrādātā sistēmas arhitektūra, aprakstīti galvenie metadatu bāzes objekti. Sadaļā „Piemērs" tiek apskatīta viena konkrēta piemēra pārskata veidlapa un detalizēti apskatīts, kā šī veidlapa tiek aprakstīta metadatu bāzē. Šajā sadaļā arī apskatīti principi kā datubāzē tiek saglabāti piemēra veidlapā ievadītie pārskatu dati. Sadaļa „Fiziskie sistēmas parametri" apskata sistēmas fiziskos parametrus, kā arī reālos datubāzu apjomus uz darba izstrādes brīdi. Sadaļā „Secinājumi" aprakstītie secinājumi apskata ieguvumus (pozitīvos kā arī negatīvos) saistībā ar sistēmas ieviešanu, sistēmas novērojumus vairāku gadu garumā, kā arī darba autora ieguldījumu sistēmas izstrādē.

1. SISTĒMAS MĒRĶI

Jebkuras statistiskās organizācijas, nacionālās vai starptautiskās, pamatā ir tās informācijas sistēma (statistikas informācijas sistēma – SIS), kas sastāv no apakšsistēmām, kuras nodrošina statistiskās informācijas savākšanu, apstrādi, glabāšanu, analīzi un izplatīšanu. Minētie procesi nosaka informācijas sistēmas arhitektūru, tās infrastruktūru un tos izdala statistikas iestādes organizacionālajā struktūrā. Statistiskās informācijas lietotāju prasību pieaugums, starptautiskās sadarbības aktivitātes kāpinājums starp statistikas organizācijām informācijas apmaiņas laukā un informācijas tehnoloģiju (IT) straujā attīstība diktē augstas prasības informācijas sistēmas arhitektūras izveidei un attīstībai tuvākā un tālākā perspektīvā.

Lai palīdzētu koordinēt statistikas organizāciju darbu Apvienoto Nāciju Eiropas Ekonomiskās Komisijas (UN/ECE) un Ekonomiskās sadarbības un attīstības organizācija (OECD) paspārnē jau no 1953. gada darbojas pastāvīgs sesiju orgāns – Eiropas statistiķu konference, kurā piedalās valstu nacionālo statistikas iestāžu vadītāji. Konferences galvenie uzdevumi ir paaugstināt Eiropas oficiālo statistikas organizāciju sagatavotās informācijas kvalitāti un tās salīdzināmību starptautiskā līmenī, ievērojot Apvienoto nāciju organizācijas Statistikas komisijas rekomendācijas, veicināt ciešu Eiropas valstu sadarbību ar starptautiskajām statistikas organizācijām, apmierināt jebkuras prasības starptautiskajā sadarbībā saistībā ar pārejas un integrācijas procesiem Eiropā un citos reģionos. Konference regulāri organizē sanāksmes un darba grupas dažāda profila speciālistiem, tai skaitā IT speciālistiem. Šīs konferences darbības ietvaros 1999.gadā sanāksmē, kas veltīta statistiskās informācijas tehnoloģijas jautājumiem, tika pieņemtas vadlīnijas un rekomendācijas: "Nacionālo un starptautisko statistikas organizāciju informācijas sistēmu arhitektūra"[1].

Atbilstoši šiem norādījumiem katra statistikas organizācija veic trīs galvenās statistisko datu producēšanas funkcijas:

- (F1) – datu iegūšanas funkcija, kas tieši un/vai netieši novēro (mēra) konkrētu objektu sistēmu iezīmes un kas sagatavo un uzglabā novērojuma datus mikrodatu glabātuvē
- (F2) – datu apkopošanas (agregēšanas) funkcija, kas transformē funkcijas F1 rezultātā iegūtos mikrodatus makrodatos vai „statistikā", kas atbilstoši lietotāju vajadzībām kalpo par pamatu dažādu kopsavilkumu datu izgūšanai

- (F3) – kopsavilkumu datu izplatīšanas un piegādes funkcija, kas padara kopsavilkumu un makrodatus pieejamus lietotājiem, palīdz tos interpretēt un analizēt.

Statistisko datu vākšanas, apstrādes un izplatīšanas funkcijas veic jebkurā statistikas organizācijā, neatkarīgi no tā, vai tā ir starptautiska vai nacionāla iestāde. Šo funkciju izpilde ir saistīta ar šādiem tehnoloģiski orientētiem procesiem:

- statistisko datu vākšana (mikrodati, makrodati, metadati);
- statistisko datu kontrole;
- statistisko datu glabāšana un meklēšana;
- statistisko datu apstrāde un analīze;
- statistisko datu izplatīšana.

Latvijas statistiskās informācijas vākšanu, apkopošanu, analīzi un izplatīšanu veic Latvijas Centrālā Statistikas Pārvalde. Valsts Statistikas likums [2] nosaka, galvenās Statistikas pārvaldes funkcijas un uzdevumus. Šis likums dod tiesības Statistikas pārvaldei ar interviju vai anketu palīdzību aptaujāt dažāda veida respondentus – statistikas objektus, kuru informācija ir nepieciešama datu iegūšanai, apstrādei un statistikas datu radīšanai. Respondents var būt gan juridiska organizācija, gan arī fiziska organizācija. Likums norāda, ka respondentiem ir jādod patiesi dati, ja Statistikas pārvalde konkrētu respondentu ir iekļāvusi kādā no statistikas apsekojumiem.

Otra organizācija, kas ir statistikas datu pasūtītājs, ir Eiropas Statistika (EuroStat). Eurostat ar izdotu Eiropas regulu palīdzību precīzi nosaka kādi tieši dati katras valsts statistikas organizācijai ir jāsniedz Eiropas Statistikai [3], kopēju datu analīzei un salīdzināšanai. Katra izdotā Eiropas regula par statistikas tēmu nosaka:

- Precīzu iesniedzamo rādītāju definīcijas – precīzi aprakstīti, kādi tieši dati ir jāsniedz.
- Respondenta definīcija – precīza definīcija, kas ir respondents, konkrētu rādītāju datu sniegšanai (piemēram, uzņēmums, mājsaimniecība, persona)
- Periodiskums – sniedzamo datu periodiskums, cik regulāri dati par konkrētiem rādītājiem ir jāsniedz
- Ticamība – precīzi metodoloģiskie norādījumi un metodes, kas nosaka, ka iesniedzamie dati ir pietiekami ticami.

Katra Eiropas regula būtībā definē veselu iesniedzamo rādītāju kopu, kas attiecas uz konkrētu nozari, piemēram, tirdzniecība, darba statistika. Mainoties dažādiem apstākļiem

Eiropas attīstībā, šīs regulas regulāri tiek labotas un papildinātas, uzliekot par pienākumu valstu statistikas apstrādes organizācijām sniegt arvien jaunu rādītāju datus dažādās jomās. Tāpat katru gadu Eiropas statistikas pārvalde veic auditus, lai pārliecinātos par valsts statistikas organizāciju iesniegto datu precizitāti un ticamību.

Gan Eiropas regulas, gan arī Latvijas Statistikas likums nosaka kādi dati ir jāsniedz, no kā šie dati ir jāiegūst un kādā veidā, kā arī nosaka kādas drošības prasības ir datu apkopošanā un izplatīšanā. Bet neviens no likumiem nenorāda, kādas Informācijas Sistēmas ir jāizmanto, datu iegūšanai, apstrādei un izplatīšanai. Tas nozīmē, ka infrastruktūras veidošana un izmantotās tehnoloģijas paliek katras statistikas apstrādes iestādes ziņā.

Statistikas informācijas sistēmas apkalpo noteiktas kategorijas lietotājus un risina to tematiskos uzdevumus. Bieži vien šāds priekšmetisks dalījums, sadalot pārskatus tematiskās grupās (piemēram, rūpniecības statistika, tirdzniecības statistika, lauksaimniecības statistika, utt.) atspoguļojas arī statistikas iestādes organizacionālajā struktūrā. Arī Latvijas Statistikas Pārvaldes organizacionālā struktūra ir veidota pamatojoties uz šādu pārskatu tematisko apstrādi. Kā rezultātā, pateicoties šādai struktūrai, uzņēmumā arī tiek veidotas tematiski orientētas datu apstrādes līnijas. Praktiski tas izskatījās tā, ka katrai tematiskai pārskatu grupai tika veidotas pilnīgi savas datu apstrādes sistēmas, lietojot dažādus tehnoloģiskos risinājumus, bieži vien ļoti atšķirīgus un savstarpēji nesavienojamus. Atsevišķas datu apstrādes sistēmu rezultātā nav vienotas datubāzes, dati ir izkaisīti un lietotājiem sagādā milzīgas problēmas apstrādāt datus, kas apvieno vairāku tematisko grupu datus.

Katru gadu prasības pēc statistikas informācijas mainās, kuras galvenie pasūtītāji ir dažādas valsts institūcijas, Eurostat, gan dažādas starptautiskas organizācijas, piemēram, ANO, UNESCO. Līdz ar prasību pēc statistiskās informācijas maiņu, Centrālā Statistikas Pārvalde arī ir spiesta mainīt pārskatu veidlapu saturu, lai tajās tiktu iekļauta visa nepieciešamā informācija. Ikgadēja pārskatu veidlapu formu maiņa izsauc arī visu datu apstrādes sistēmu izmaiņas, lai nodrošinātu pārskatu datu ievades un apstrādes procedūras atbilstoši izmainītajām pārskatu veidlapu formām. Tāpēc Centrālā Statistikas pārvalde sākot jau no 90-tajiem gadiem ievērojamu daļu no attīstības budžeta tērēja nepārtrauktai visu sistēmu uzturēšanai un nepārtrauktām izmaiņu veikšanai. Lai šīs izmaksas mazinātu tika pieņemts lēmums par jaunas informācijas sistēmas izstrādi, kurai nepieciešams aptvert galvenos statistikas datu apstrādē iesaistītos procesus. Kā paši darbietilpīgākie procesi statistisko datu apstrādē ir tieši datu ievade un sākotnējā validācija. Definējot jaunās sistēmas prasības, pamatā tika izmantoti šādi principi:

- integrācija.

  Sistēmas sastāvā ir iekļauts Statistikas uzņēmumu reģistrs, kuru uztur CSP un kurš kalpo kā vienota bāze un datu avots respondentu atlasei, un nodrošina saiti starp administratīvajām vienībām un respondentu, kā pārskata datu sniedzēju. Vienlaicīgi Statistikas uzņēmumu reģistra klasifikācijas sistēma nodrošina vienotu pārskatu datu klasifikāciju pēc trim galvenajiem kritērijiem:

  - uzņēmumu īpašumtiesību klasifikācija;
  - uzņēmumu darbības sfēru klasifikācija (NACE, PRODCOM klasifikācija);
  - Ģeogrāfiskie kritēriji (ATVK klasifikācija);

  Statistikas uzņēmumu reģistra mērķis ir ar dažādām metodēm apsekot visus uzņēmumus, reģistrējot jaunos, kā arī uzturēt aktuālā stāvoklī visus esošos uzņēmumus, reģistrējot jebkādas izmaiņas šo uzņēmumu datos. Pamatinformāciju par uzņēmumiem Statistikas Uzņēmumu Reģistrs saņem no valsts uzņēmumu reģistra un VID datubāzēm. Papildus informāciju par uzņēmumiem Statistikas uzņēmumu reģistrs saņem no speciālām uzņēmumu apsekojumu anketām, kā arī no pārskatiem, kurus iesniedz respondenti. Svarīgākie uzņēmumu rādītāji, par kuru aktuālām vērtībām atbild Statistikas uzņēmumu reģistrs, ir visas klasifikācijas, kas tiek izmantotas statistisko datu apstrādes un agregācijas procesos. Statistikas uzņēmumu reģistra bāze kalpo kā galvenais avots dažādu apsekojumu izlašu kopu veidošanai. Šī prasību kopa norāda, ka visiem pārskatiem Statistikas uzņēmumu reģistrs ir jāizmanto kā galvenais avots respondentu sarakstu un to atribūtu vērtību iegūšanai. Šīs prasības mērķis ir nodrošināt mehānismus, ja uzņēmuma pamatdatu izmaiņu situācijās (piemēram, mainoties uzņēmējdarbības formai, vai nozarei), tad visas šīs izmaiņas nonāktu uz tiem pārskatiem, kuros katrs uzņēmums ir kā respondents. Lai neveidotos situācijas, kad vienā pārskatā respondents tiek apstrādāts ar vienām atribūtu vērtībām, bet citā pārskatā ar pavisam atšķirīgām atribūtu vērtībām;

- centralizācija.

  Sistēmā visi dati ir jāglabā vienotā datu noliktavā, sadalot informāciju sekojošās datu kopās (datubāzēs) atkarībā no datu objektu veidiem:

  - produkcijas metadatu bāze – aprakstošie dati, kas satur pārskatu aprakstus, respondentu avotus, rādītājus, validāciju un agregāciju nosacījumus;
  - mikrodatu bāze – attiecas uz respondentiem (datu sniedzējs) un to pārskatu datiem;

- makrodatu bāze – glabā agregētus datus, kopsavilkumus un publicējamos datus.

Šo prasību mērķis ir apvienot visu pārskatu datus vienā kopīgā datubāzē. Kā arī nodrošināt mehānismus datu atlasei no šīs kopīgās datu bāzes. Atlasot datus gan pirmdatu, gan agregēto datu līmenī ir jānodrošina divas galvenās datu atlases funkcijas:

- jābūt iespējai atlasīt datus vienlaicīgi no dažādiem pārskatiem, pat no dažādām tematiskām grupām, un apvienot tos vienotā datu masīvā, lai nodrošinātas datu analīzes prasības tai lietotāju grupai, kas analizē datus daudz plašākā datu sfērā;
- jānodrošina iespēja veidot datu laika rindas. Kas nozīmē, ka jābūt iespējai atlasīt vienam vai vairākiem rādītājiem datus par ilgstošu laika periodu, piemēram, tekošais periods un pēdējie pieci gadi. Jāņem vērā, ka šo periodu laikā pārskatu struktūra var būt mainījusies un vajadzīgais rādītājs konkrētā pārskatā norādītā laika posmā ir mainījis savu atrašanās vietu (piemēram, vienu gadu tas ir bijis tabulas pirmajā rindā, bet nākamā gadā jau tabulas trešajā rindā);

Esošajā vairāku atsevišķu sistēmu risinājumā, lai apvienotu vairāku dažādu pārskatu datus vienotā masīvā, ir jāiegulda pietiekami liels vairāku programmētāju darbs, jo dati ir izkaisīti tikai katra pārskata programmētājam vien zināmā vietā un katram savā datu struktūrā. Rezultātā apvienot dažādu sistēmu, dažādu datu struktūru datus, tas aizņem ļoti ievērojamus gan laika, gan cilvēku resursus. Arī veidojot laika rindas ir nepieciešams liels programmētāja darba ieguldījums, kas ir pa spēkam tikai konkrēta pārskata programmētājam. Jo tikai šis programmētājs zina, kur atrast vairāku gadu datus, kā tos savstarpēji apvienot, ņemot vērā pārskatu katra gada saturiskās izmaiņas;

- standartizācija.
  Vienoti un standartizēti principi jebkuru pārskatu izskata un satura aprakstiem, vienoti principi statistikas rādītāju un mainīgo veidošanai un uzturēšanai. Vienoti principi jebkura biznesa statistikas pārskata datu apstrādes procesu organizācijai, lai būtu iespējams atteikties no atsevišķām tematiskām datu apstrādes līnijām. Šis princips vairāk ir saistīts ar iestādes organizatoriskām izmaiņām. Mērķis ir definēt visus datu apstrādes principus, kas ir spēkā pilnīgi visiem statistikas pārskatiem, neatkarīgi no to sarežģītības, vai tematiskās grupas. Lai rezultātā izstrādātā sistēma nodrošinātu datu

apstrādes funkcijas, kas darbojas pēc identiskiem principiem visiem pārskatiem. Kā rezultātā visiem pārskatiem būtu vienādi datu apstrādes principi, kas veicami vienas sistēmas robežās, izmantojot vienādas ekrānformas un funkcionalitāti. Pašreizējās daudzajās tematiskajās datu apstrādes sistēmās, katrā sistēma tiek izmantota sava pieeja, secība un metodes dažādos pārskatu datu apstrādes etapos, kas bieži vien ir atšķirīgas dažādiem pārskatiem;

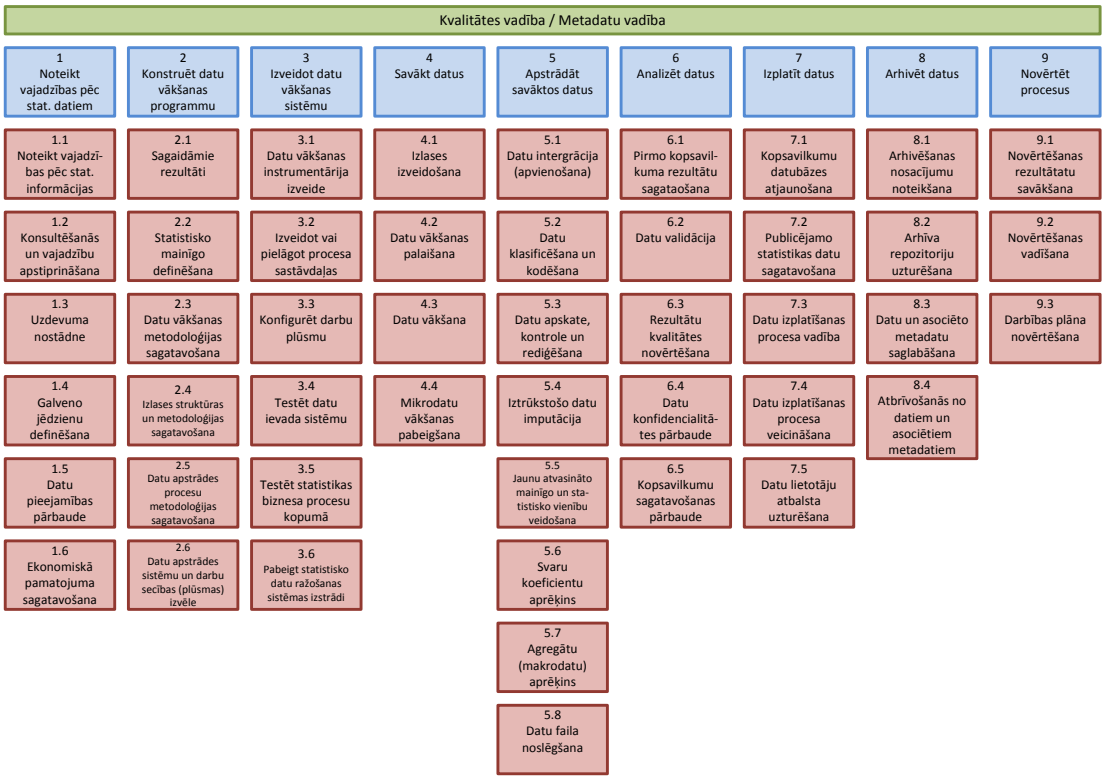
- metadatu vadīta pieeja.
  Nodrošināt statistikas datu apstrādes cilvēkus ar rīkiem, kurus izmantojot iespējams definēt un aprakstīt pārskatus, bez specifiskām programmēšanas un datu bāzu struktūru zināšanām. Tā kā iepriekšējais princips definē fiksētus datu apstrādes procesus, kas ir spēkā katram apsekojumam, tad metadatu mērķis ir aprakstīt katram pārskatam kādas manipulācijas tiek veiktas, lai nodrošinātu procesa izpildi tieši konkrētam pārskatam, veidojot metadatu vadītu pieeju datu apstrādē. Metadatu aprakstu veidošanai ir jābūt lietotājiem saprotamā veidā un valodā (bez programmēšanas). Rezultātā sistēmā ievadītie katra pārskata metadati kalpo par pamatu automatizētai datu ievades formu veidošanai, nepieciešamo pārskatu validāciju procedūru ģenerēšanai, datu agregāciju un kopsavilkumu veidošanai. Esošajā pieejā mainoties katra gada pārskatiem, programmētājam nākas pilnībā pārtaisīt esošās programmas, lai nodrošinātu datu ievadi nākamā gada pārskatiem, atbilstoši šo pārskatu izmaiņām. Šīs izmaiņas var veikt tikai programmētāji, kā rezultātā nepieciešami milzīgi programmētāju resursi, lai laicīgi tiktu sagatavotas esošās programmas nākamā gada datu apstrādei. Turklāt lietotājiem vienlaicīgi ir jāizmanto vairākas programmas (gadu mijā), lai nodrošinātu vienlaicīgu datu ievadi, gan pagājušā gada pēdējos periodos, gan arī nākamā gada jaunajos periodos. Turpretī metadatu izmaiņas var veikt ne tikai programmētāji. Metadatu izmaiņas pārskatu aprakstīšanā ir mazāk ietilpīgas un to var veikt ievērojami īsākā laikā ar mazākiem cilvēku resursiem.

2. RISINĀJUMA IZVĒLE

Sadaļā tiek apskatīts Centrālās statistikas pārvaldes biznesa procesu modelis, kas kalpo par pamatu sistēmas funkcionalitātes noteikšanai, gatavu risinājumu pieejamība uz sistēmas izstrādes brīdi. Nodaļā 2.3 tiek apskatīts metadatu jēdziena interpretācija statistikā. Nodaļā 2.4 tiek apskatīts Bo Sundgren ieteiktais metadatu modelis statistikas informāciju sistēmu izveidē.

2.1. Biznesa procesu modelis

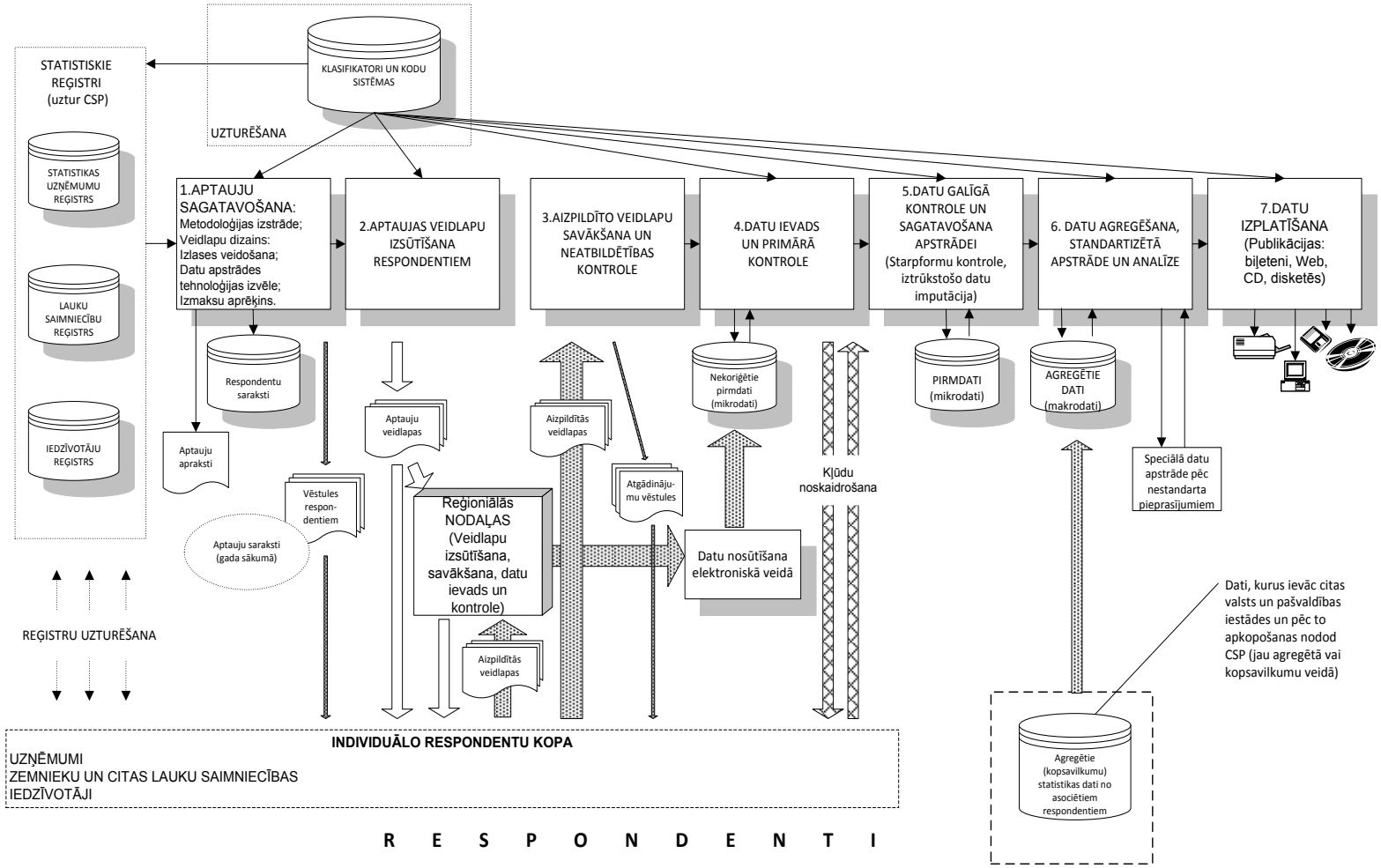
Pirms datu apstrādes sistēmas prasību formulēšanas Centrālā Statistikas Pārvalde veica visu esošo datu apstrādes sistēmu izpēti, kas tika izveidotas atsevišķi katrai tematiskai pārskatu kopai. Analizējot esošās datu plūsmas un apstrādes procedūras, tika izveidots Latvijas statistikas biznesa procesu modelis, kas pēc savas būtības ir tuvs šodien standartizētam modelim – General Statistical Business Process Model (GSBPM).



2.1. att. Vispārināts statistikas biznesa procesu modelis

Šis modelis nosaka vispārīgos procesus, kas apraksta katra apsekojuma pilnu datu apstrādes ciklu, sākot no prasību analīzes (kādi dati ir jāiegūst), beidzot ar datu izplatīšanu un novērtēšanu (vai iegūtās kopsavilkuma tabulas apmierināja sākotnēji uzstādītās prasības).

Analīzes rezultāti ļāva formulēt detalizētas prasības standartizēta pilna cikla (sākot no ievades un beidzot ar kopsavilkumu veidošanu) datu apstrādes sistēmas veidošanai, kurā katra procesa parametri aprakstīti ar speciāli strukturētu statistisko metadatu un pseidovalodas (kas nav programmēšanas valoda) palīdzību.



2.2. att. Centrālās Statistikas Pārvaldes pārskatu apstrādes datu plūsmas diagramma

Attēlā redzamas datu plūsmas un to savstarpējās saistības, kādas tās tika identificētas Centrālajā Statistikas pārvaldē, analizējot esošos datu apstrādes principus un metodes. Veidojamās sistēmas mērķis ir pilnībā aptvert procesus sākot no datu ievades, beidzot ar datu izplatīšanu – nodrošinot šo procesu vajadzībām nepieciešamo programmatūras risinājumu, kurš būtu metadatu bāzēts.

2.2. Gatavu risinājumu pieejamība

Turpinot prasību specificēšanu, Centrālās Statistikas pārvaldes speciālisti regulāri apmeklēja starptautiskās Eiropas statistiķu konferences, sanāksmes un darba grupas, kurās piedalās ne tikai valstu nacionālo iestāžu vadītāji, bet arī IT speciālisti. Mērķis šo konferenču apmeklējumiem bija apzināt, kādas pārskatu apstrādes sistēmas ir izveidotas citās valstīs, lai varētu pārņemt šo valstu pozitīvo pieredzi savas sistēmas prasību specificēšanā un veidošanā. Pārrunājot ar dažādu valstu statistikas pārvalžu vadītājiem, tika konstatēts, ka uz to brīdi principā nevienā valstī nav izveidotas statistisko datu apstrādes sistēmas, kurās tiktu izmantoti metadati, kā pārskatu veidlapu struktūras un satura apraksti, kas aktīvi tiktu izmantoti statistikas pārskatu datu apstrādes procesos. Galvenie iemesli, kāpēc citās valstīs nav metadatu bāzēta risinājuma statistisko datu apstrādē:

- daudzas attīstītās valstis plaši izmanto nacionālos reģistrus. Valstīs, kurās reģistri ir pietiekami attīstīti un plaši izmantoti, kalpo par pamatu, lai iegūtu ļoti lielu daudzumu nepieciešamās statistikās informācijas ieguvei. Pie tam šai informācijai ir pietiekami augsta ticamība, lai šos datus varētu izmantot statistikas ražošanai. Rezultātā šādās valstīs datu apstrādes procesi tiek organizēti savādāk, kā arī datu apstrādes sistēmas ir orientētas tieši uz šo reģistru datu apstrādi;
- salīdzinoši nesen tikai tika izdots apkopojošs dokuments, kurā aprakstītas vadlīnijas un rekomendācijas, nacionālo statistikas organizāciju informācijas sistēmu veidošanā [1]. Atbilstoši šīm vadlīnijām neviena valsts vēl nebija paspējusi izveidot vai modernizēt esošās statistisko datu apstrādes sistēmas;
- mazāk attīstītas valstis (līdzīgi kā Latvija) pārdzīvo reorganizācijas procesus, kuru rezultātā tiek reorganizēta gan iestādes struktūra, gan par jaunu tiek veidota iestādes infrastruktūra, ieskaitot tehnisko nodrošinājumu un programmnodrošinājumu.

Pārsvarā visās šajās valstīs situācijas bija līdzīga kā Latvijā – eksistē lokāli risinājumi, piesaistīti konkrētam pārskatu lokam;

- dažās no valstīm jau esošais risinājums ir pilnībā sevi attaisnojis un ir pietiekami labs, lai turpinātu to izmantot statistisko datu apstrādē un producēšanā. Piemēram, ir valstis, kurās datu apstrādē jau gadiem tiek izmantoti lieldatori (mainframe), un šādu sistēmu pārveidošana uz modernākām datu apstrādes tehnoloģijām izmaksu ziņā stipri pārsniedz sagaidāmos labumus;
- atsevišķās valstīs ir izveidotas nelielas lokālas metadatu sistēmas, kas paredzētas ļoti šauras jomas funkcionalitātes nodrošināšanai, piemēram, metadatu vadīta sistēma, kas veidota tikai datu izplatīšanas mērķiem;
- vienā no valstīm tika strādāts pie lielas metadatu sistēmas veidošanas, bet kura kalpotu tikai un vienīgi kā zināšanu bāze, un kuras dati nekādi nebūtu savienoti ar datu apstrādes procesiem. Šī sistēma paredzētas tikai un vienīgi, lai maksimāli detalizēti uzkrātu zināšanas par visiem procesiem, kas saistīti ar statistisko datu apstrādi.

Centrālā Statistikas Pārvalde meklēja arī potenciālus gatavus risinājumus ārpus Statistikas kopienām. Ar vienu no pretendentiem (Scopeland, Vācija) tika slēgts līgums par prototipa izstrādi, kurā tiktu ieviests viens no vienkāršākajiem statistikas apsekojumiem. Šis uzņēmums savu sistēmu pozicionēja kā metadatu orientētu vidi, kurā lietotājiem pašiem nekas nav jāprogrammē, bet tikai jāveido apraksti un sistēma automātiski veido datu ievades un apstrādes formas un funkcijas. Darba autors tika uzaicināts, lai novērtētu šo gatavo produktu un tā atbilstību izvirzītajām prasībām. Vairāku mēnešu garumā uzņēmuma pārstāvji mēģināja realizēt šo vienu pārskatu savā sistēmā un darba autors analizēja šī darba sasniegtos rezultātus. Rezultātā piedāvātā sistēma, likās nepieņemama, sekojošu iemeslu dēļ:

- tas ko kompānija definēja kā metadatus – tie principā bija tehniskie metadati, tādi kā tabulu nosaukumi, kolonu nosaukumi, tabulu datu tipi, ierobežojumi, relācijas;
- automātiski ģenerētās formas tiek veidotas atbilstoši identiskai fizisko tabulu struktūrai. Kas nozīmē to, ka lietotājiem pašiem ir jādefinē tabulas struktūra, kurai atbilstoši tiek veidotas ievades formas. Tāpat netika piedāvāts apmierinošs pārskata validācijas likumu realizācijas veids, kas nodrošinātu lietotājiem saprotamu veidu pārskata validāciju likumu aprakstīšanai un pārskatu datu validācijai;
- pretendents nepiedāvāja derīgu risinājumu, kā organizēt pārskatu ikgadējās izmaiņas, kā arī kā šīs izmaiņas pēc tam apvienot vienotā datu masīvā, lai būtu iespējams atlasīt laika rindas – viens rādītājs pa vairākiem laika periodiem;

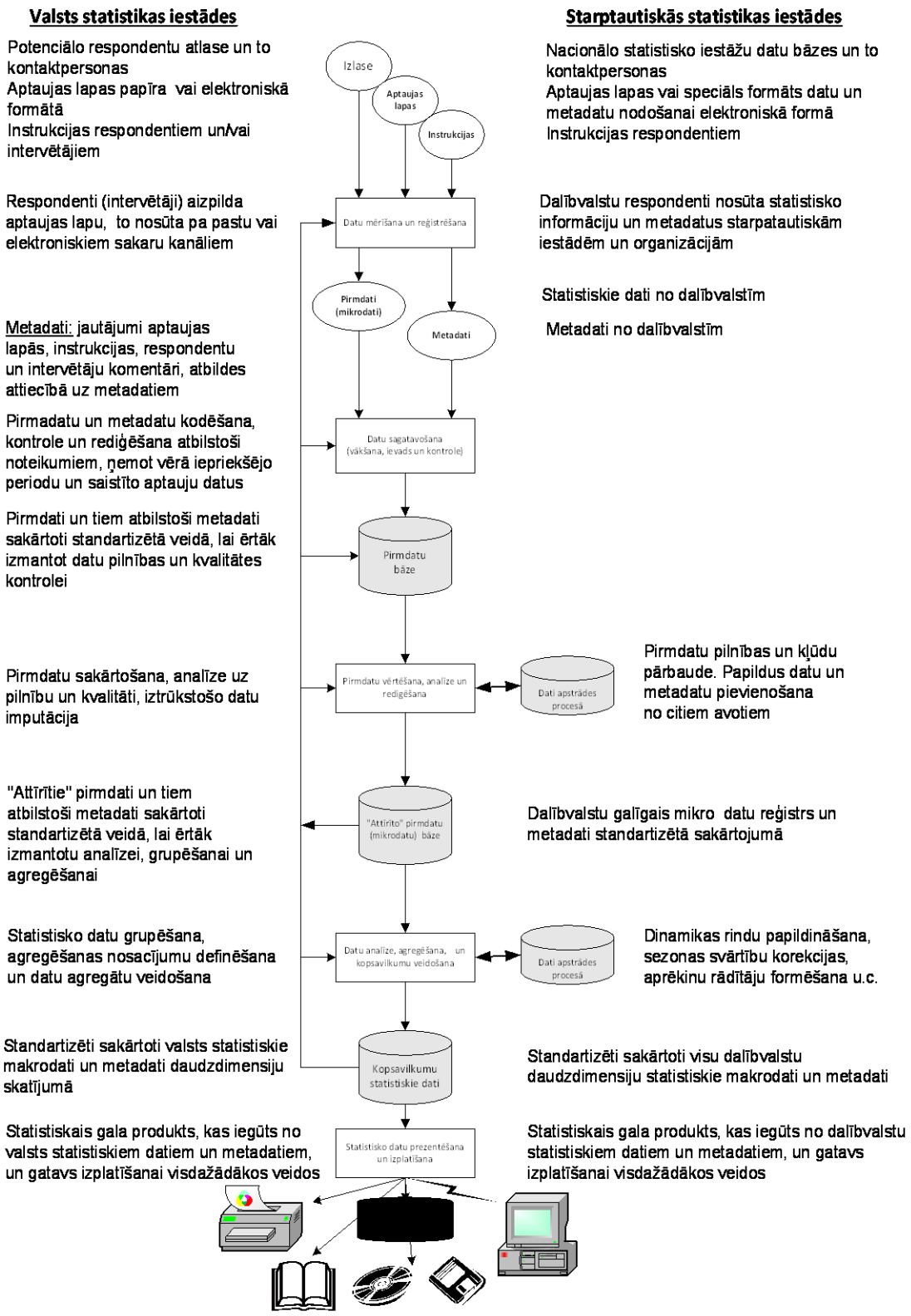
- piedāvātā vide nenodrošināja pietiekami plašas integrācijas iespējas, ar mērķi, izmantojot vides iespējas programmēt funkcijas, kuras nav iespējams realizēt ar sistēmas iebūvētām iespējām.

Kopumā piedāvātais produkts vairāk atgādināja ļoti attīstītu datu bāzes pārvaldības sistēmas rīku, kas pilnīgi noteikti neder statistikas pārskatu datu apstrādei procesu realizācijai. Rezultātā līgums ar Vācijas kompāniju tika lauzts.

Rezultātā tika pieņemts lēmums pašiem veidot nepieciešamo informācijas sistēmu un darba autoram tika uzdots uzdevums izveidot sistēmas arhitektūru, datubāzes un funkciju projektējumu kā arī iespēju robežās veikt atsevišķu funkciju programmēšanas darbus.

2.3. Metadatu interpretācijas statistikā

Vienā no statistikas konferencēm, kas veltīta statistiskās informācijas tehnoloģijas jautājumiem, tika pieņemtas vadlīnijas un rekomendācijas: “Nacionālo un starptautisko statistikas organizāciju informācijas sistēmu arhitektūra”[1]. Atbilstoši šai arhitektūrai, zemāk attēlā attēlota tipveida statistisko datu un metadatu plūsmas shēma valsts un starptautiskās statistikas iestādēs.



2.3. att. Tipveida statistisko datu un metadatu plūsmas

Definējot statistikas metadatus, balstīsimies uz sekojošiem formulējumiem par metadatiem kopumā:

1. Metadati ir metainformācijas (metazināšanu) fiziskie rādītāji - tāpat kā dati ir informācijas (zināšanu) rādītāji.
2. Metadati nodrošina informāciju par datiem, par datu producēšanas procesu un to lietošanu.
3. Metadati ir dati, kuri nepieciešami atbilstīgai datu producēšanai un lietošanai, par ko tie sniedz informāciju.
4. Tāpat kā dati nodrošina ar informāciju par objektiem "patiesajā" objektu sistēmā metadati nodrošina ar informāciju par informācijas sistēmu - metaobjektiem. Tāpat kā "patiesiem" objektiem, metaobjektiem piemīt īpašības un tie ir saistīti savā starpā ar objektu attiecībām.
5. Tāpat kā datus pārvalda informācijas sistēmas, metadatus pārvalda metainformācijas sistēmas. Tādējādi, metainformācijas sistēma izmanto un producē metadatus, sniedzot informāciju par datiem, un šī sistēma izpilda savu uzdevumu ar tādu funkciju palīdzību kā "metadatu uzkrāšana", "metadatu apstrāde", "metadatu uzglabāšana" un "metadatu izplatīšana".
6. Metainformācijas sistēma var būt aktīva vai pasīva. Aktīva metainformācijas sistēma ir fiziski integrēta ar informācijas producēšanas sistēmu, kas producē datus, par ko metadati sniedz informāciju metainformācijas sistēmā. Pasīvā metainformācijas sistēma satur tikai atsauces uz datiem nevis pašus datus.

Pirmos trīs augšminētos formulējumus var uzskatīt par trīsdimensionālās metadatu definīcijas trīs "projekcijām":

- sintaktiskā projekcija (reprezentatīvi orientēta)
- semantiskā projekcija (saturiski orientēta)
- pragmatiskā projekcija (mērķorientēta)

Ceturto augstākminēto formulējumu var uzskatīt par metainformācijas semantikas tālāku izskaidrojumu, un piektais un sestais izteikums var tikt uzskatīts par metainformācijas sintaktisko aspektu tālāku skaidrojumu.

Statistikas metadati ir dati, kas nepieciešami atbilstošai statistikas datu producēšanai un lietošanai. Tie raksturo statistikas datus un - līdz zināmai robežai - procesus un rīkus, kas tiek izmantoti statistikas datu producēšanā un lietošanā, īsāk sakot, statistikas metadati ir dati par statistikas datiem.

Statistikas metainformācijas sistēma ir sistēma, kas izmanto un producē statistikas metadatus, dodot informāciju par statistikas metadatiem, un kas veic savu uzdevumu ar tādu procedūru palīdzību, kā "statistikas metadatu uzkrāšana", "statistikas metadatu apstrāde", "statistikas metadatu uzglabāšana" un "statistikas metadatu izplatīšana". Statistikas meta informācijas sistēmas sastāvā būtu izveidojama statistikas zināšanu bāze, kas ir sava veida semantisks modelis, lai attēlotu statistiķu uzkrātās zināšanas par valsts statistikas sistēmu un procesiem tajā. Šāda zināšanu bāze varētu kalpot citām dažādu nozaru ekspertsistēmām un intelektuālām sistēmām.

Līdzīgi citām metainformācijas sistēmām, statistikas metainformācijas sistēma var būt aktīva vai pasīva, kā tas jau bija minēts iepriekš. Meta informācijas sistēma var ne tikai uzkrāt sistematizētu un strukturētu informāciju par statistikas datiem, bet arī vadīt vienu vai vairākus procesus (datu izplatīšana, datu ievads, kontrole, agregācija). Piemēram, aktīvās metainformācijas sistēmas lietotājs, kurš ir identificējis dažus potenciāli interesējošus statistikas datus, var nekavējoties turpināt izgūt tos no tās pašas sistēmas. Tādu sistēmu varam uzskatīt par integrētu informācijas/metainformācijas sistēmu. Salīdzinājumam - pasīvās metainformācijas sistēmas lietotājam, kurš ir identificējis potenciāli interesējošus statistikas datus, tie ir jāizgūst, ieejot citā sistēmā.

2.4. Bo Sundgren metadatu modelis

Šajā sadaļā tiks apskatīts zviedru profesora Bo Sundgren piedāvātais statistikas metadatu apraksta modelis. Bo Sundgren ir pasaulē atzīts eksperts gan statistikas datu savākšanas metodoloģiju izstrādes jomā, gan arī statistikas datu apstrādes sistēmu izveides jomā. Bo Sundgren ir izstrādājis Statistikas metadatu koncepciju, statistikas datiem un statistikas informācijas sistēmām. Viņš pirmo reizi terminu “statistikas metadati” definēja savā doktora disertācijā 1973.gadā [5], kurā detalizēti tika izklāstīta statistisko datu vākšanas un apstrādes metodoloģija. 1995.gadā tiek izdoti pirmie norādījumi metainformācijas sistēmu izveidošanai statistikas aptauju datu apstrādei [4]. Tie iesaka arī metadatu un datu modelēšanas veidus katram statistikas aptaujas datu apstrādes posmam atsevišķi. 1999. Gadā vienā no Statistikas sanāksmēm, kas veltīta statistiskās informācijas tehnoloģijas jautājumiem, tika pieņemtas vadlīnijas un rekomendācijas „Nacionālo un starptautisko statistikas organizāciju informācijas sistēmu arhitektūra”, kurā ir iekļauti Bo Sundgren norādījumi metadatu sistēmu izveidē statistikas aptauju datu apstrādei [1].

Zemāk tiks apskatīti Bo Sundgren ieteikumi statistikas pārskatu datu aprakstīšanai un klasificēšanai, izmantojot metadatus. Šie statistikas datu aprakstu principi tika izmantoti veidojot metadatu vadītu statistisko datu apstrādes sistēmas arhitektūru.

Statistikas dati ir statistikas metadatu nodrošinātie primārie apraksta objekti (statistikas datu vērtības). Tādējādi, lai saprastu statistikas metadatu nozīmi un saturu, mums ir jāsaprot, kas ir statistikas dati un kas tajos ir tāds, kam nepieciešams apraksts. Statistikas mikrodatus un makrodatus, var definēti šādi.

Mikrodati, dažreiz arī saukti par novērojumu datiem vai mērījumu datiem (to vērtības), ir novērojumi vai objektu iezīmju kopas (uzņēmumu un notikumu) mērījumu rezultāts. Bo Sundgren šīs objekta iezīmes formalizē kā secīgu pāri:

$$C_o =< O(t), V(t) >,$$

kur:

- C – objekta raksturlielums (īpašība);
- O – Objekta tips;
- V – Statistiskais mainīgais;
- t – laika parametrs.

Piemēram, novērojuma datus var izteikt sekojošā vārdiskā formā: *„Uzņēmumos saražotā produkcija vērtības izteiksmē 2014. gada 1.ceturksnī”*. Attiecīgi statistiskais mainīgais piemērā ir *„Saražotā produkcija vērtības izteiksmē”*, Objekta tips (O) ir uzņēmumi, laika parametrs konkrētā piemērā ir *„2014.gada 1.ceturksnis”*.

Informācijās pamatveidojošie bloki par objekta iezīmju novērojumiem un mērījumiem ir tā saucamie elementārie ziņojumi ar semantisko struktūru:

$$m_o =< o_i, p, t >,$$

vai ar alternatīvu notāciju,

$$m_o=[o_i .V(t)] = v_i,$$

kur:

- o_i ir objekta instance, kas pieder objekta tipam O;
- p ir īpašība, kas parasti izteikta kā mainīgā V vērtība v_i;

- *t* ir laika instance (brīdis vai intervāls), kurā tiek pieņemts, ka objektam ir (bijusi) īpašība *p*.

Šī notācija apraksta katru novērojuma datus veidojošo vērtības elementu. Piemēram, pamatveidojošo bloku iespējams vārdiski nosaukt šādi: *„Zemnieku saimniecības „Zaļā zeme" iepirkto graudaugu – rudzu – apjoms 2014.gada martā ir 150 tonnas."*, Kur uzņēmumu kopas instance ir konkrēts uzņēmums – piemērā, *zemnieku saimniecība „Zaļā zeme"*. Mainīgais ir – *„Iepirkto graudaugu – rudzu – apjoms"* ar mērvienību *tonnas*. Laika periods ir *2014.gada marts*. Šī pamatbloka vērtība ir *150*.

Darba autors apskata metadatu orientētu statistikas datu apstrādes sistēmu, kas paredzēta biznesa statistikas apstrādei. Attiecīgi šajā sistēmā objekta tips viennozīmīgi ir tikai un vienīgi uzņēmums, vai atsevišķās situācijās uzņēmuma struktūrvienības. Lai saprastu, kas ir statistiskais mainīgais, zemāk tabulā doti vēl dažu statistikas mainīgo piemēri:

2.1. tabula. Raksturīgākie statistisko mainīgo piemēri

Mainīgais (variable)
Saimnieciskās darbības ieņēmumi
Gada apgrozījums – kopā Gada apgrozījums sadalījumā pa darbības veidiem (NACE)
Darbinieku skaits kopā Darbinieku skaits sadalījumā pa teritorijām (ATVK)
Saražotā produkcija vērtības izteiksmē sadalījumā pa produkcijas veidiem (PRODCOM)

Novērojumu rezultātu tabulu kopa, vai attiecīga datu glabātuve, kas satur novērotos un/vai iegūtos mikrodatus, ir visnozīmīgākā katras statistiskās aptaujas sastāvdaļa, no kuras dati vēlāk tiek ņemti dažādu analītisko darbu veikšanai. Visiem kompetentiem statistiķiem būtu jānodrošina pieeja no saviem datoriem šai datu glabātuvei, lai veiktu analīzi savu datoru vidē. Mikrodatu glabātuve ir jāsasaista ar attiecīgo metadatu kopu, kas veidota datu ieguves procesā, jo veicot analītisko darbu, kas parasti notiek ievērojamu laika sprīdi pēc datu vākšanas, statistiķim nebūs iespēja atkārtoti piekļūt respondentam, kas producējis datus.

Makrodati, kas parasti tiek apzīmēti kā "statistika" ir statistisko iezīmju kopas (statistiskās koncepcijas) novērtējuma rezultāts. Novērtējums tiek veikts uz mikrodatu kopas bāzes, tas ir, objekta iezīmju kopas novērojumu kompleksa. Bo Sundgren Statistiskās iezīmes formalizē ar trim simboliem.

$$< O(t), f >$$

vai ar alternatīvu notāciju,

$$C_s = O(t).V(t).f,$$

kur:

- *O(t).V(t)* – objekta iezīmes,
- *f* – statistikas līdzeklis, tas ir, apkopošanas funkcija (skaits, summa, vidējais aritmētiskais, korelācija), rezumējot patiesās *V(t)* vērtības objektiem *O(t).*

Vienkāršotā veidā agregētus datus var vārdiski nosaukt tieši tāpat kā mikrodatus *„Uzņēmumos saražotā produkcija kopā vērtības izteiksmē 2014. gada 1.ceturksnī”*, kas papildināti ar agregācijas funkciju – dotajā gadījumā summa.

Statistikas makrodati tiek organizēti noteiktās tipveida struktūrās. Piemēram, statistikas lietotāji bieži vēlas iegūt "tās pašas" statistisko iezīmju novērtētās vērtības:

- laika perioda sērijām (nevis tikai uz vienu) - "laika sēriju dati" (time series data);
- strukturētām objektu kopām (nevis vienam objektam) - "šķērsgriezumu dati"(cross-sectional data).

Veicot uzņēmumu pārskatu datu agregāciju no uzņēmumu reģistra tiek izmantotas šādas galvenās dimensijas, kas tiek izmantotas gan datu grupēšanai, agregācijai un daudzdimensionālai analīzei:

- NACE – uzņēmuma galvenās ekonomiskās darbības veida kods (NACE klasifikators)
- ATVK – uzņēmuma reģistrētās teritorijas kods (ATVK klasifikators)
- SVTK – īpašuma un uzņēmējdarbības formas kods (SVTK klasifikators)
- APGR – uzņēmuma gada apgrozījums, vai apgrozījuma grupas klasifikators
- DARB – darbinieku skaits uzņēmumā gada beigās, vai darbinieku skaita grupas kods

Zemāk tabulā attēloti makrodatu aprakstīšanas piemēri

2.2. tabula. Makrodatu aprakstīšanas piemēri.

Statistiskā mainīgā nosaukums – interesējošās makrodatu vērtības	**Objekta vektora *O(t)* dimensijas**	**Mainīgā vektors *V(t)***	**Laika vektors *t***	**Agregācijas funkcija *f***
Saražota rūpniecības produkcija vērtības izteiksmē (Ls), pavisam Latvijā 2012.gadā	NACE-visas, ATVK-visas, IUFIK-visas, APGR-visas, DARB-visas.	Saražota rūpniecības produkcija vērtības izteiksmē- kopā, Ls	2012.gadā	Summa (SUM)
Saražotas gaļas desas naturālā izteiksmē (kg), pavisam Latvijā 2012.gada 3.ceturksnī	NACE - visas; ATVK - visas; IUFIK - visas; APGR - visas; DARB -visas	Saražota produkcija naturālā izteiksmē/ PRODCOM – 15.13.12.15.00 "Desas (izņemot aknu desas)" kg	2012.gada 3.ceturksnī	Summa (SUM)
Latvijas lielāko uzņēmumu (gada apgrozījums pārsniedz 1,0 milj. Ls) kopējie saimnieciskās darbības ieņēmumi 2012.g.	NACE-visas, ATVK-visas, IUFIK-visas; APGR >= 1 000 000 Ls; DARB-visas.	Saimnieciskās darbības ieņēmumi	2012.gadā	Summa (SUM)
Latvijas lielāko uzņēmumu (ar darbinieku skaitu virs 250) kopējie saimnieciskās darbības ieņēmumi 2012.g.	NACE-visas, ATVK-visas, IUFIK-visas; APGR- visas DARB > 250.	Saimnieciskās darbības ieņēmumi	2012.gadā	Summa (SUM)
Vidējais darbinieku skaits Latvijas kokapstrādes nozares uzņēmumos 2012.gada beigās	NACE – 20.10 "Koku zāģēšana, ēvelēšana un impregnēšana"; ATVK-visas, IUFIK-visas, APGR-visas, DARB –visas.	Darbinieku skaits gada beigās	2012.gads	Vidējais (MEAN)

Veidojot statistikas metadatu bāzi, kas orientēta uz profesionālu statistikas gala lietotāju, kad tiek pieprasīta plaša spektra statistiskā kopsavilkumu informācija par dažādām

statistikas nozarēm, ieskaitot dažāda veida atvasinātos (aprēķina rādītājus), Bo Sundgrem piedāvā sekojošu vispārīgu formātu, kas aptver lielāko daļu statistiskās metainformācijas struktūras:

$O(t_a)$ ***(with)*** p_a ***(by*** $V_g(t_g))$. $V_b(t_b)$. f ***,***

kur

- $O(t_a)$ ir objekta(u) kopu sērijas, kas eksistē laikā t_a;
- p_a ir īpašība, ***alfa*** (α) īpašība, izvēloties $O(t_a)$ apakškopu;
- $V_g(t_g)$ ir mainīgo vektors, ***gamma*** (γ) mainīgais, kas šķērsklasificē $O(t_a)$ kopumu, un t_g ir laika parametru vektors, kas atbilst mainīgo vektoram; laika perioda sēriju gadījumā katrs laika parametrs būs mainīgs katrā vērtību posmā, nosakot laika parametra t_a vērtību posma saskanību;
- $V(t_b)$ ir mainīgo vektors, ***beta*** (β) mainīgais, kura summārā vērtība tiek novērtēta, un t_b ir laika parametru vektors, kas atbilst mainīgo vektoram; laika perioda sēriju gadījumā katrs laika parametrs būs mainīgs katrā vērtību posmā, nosakot laika parametra t_b vērtību posma saskanību;
- f ir agregācijas funkcija.

Statistikas makrodatu vispārējā struktūra, ko lieto datu izplatīšanai, dažreiz tiek apzīmēta kā kārbu struktūra vai ***alfa-beta-gamma-tau*** $(\alpha\beta\gamma\tau)$ struktūra, kur "*alfa*" apzīmē selekcijas īpašību, "*beta*" - summētos mainīgos, "*gamma*" - šķērsklasificētos mainīgos, un "*tau*" - laika parametrus, kā jau aprakstīts iepriekš. Šī struktūra ir ļoti noderīga sistemātiskām analīzēm un statistikas makrodatu aprakstam.

Piemēram, atbilstoši šai notācijai iespējams aprakstīt sekojošus agregētos datus: „*Uzņēmumu saimnieciskās darbības ieņēmumi sadalījumā par teritorijām un nozarēm laikā no 2010 līdz 2015 gadam*" Tad objektu kopa ir uzņēmumi katrā apsekojuma periodā no *2010 līdz 2015 gadam*. Novērojamais mainīgais ***beta*** (β) *– „Saimnieciskās darbības ieņēmumi"*, ***alfa*** (α) īpašības ir griezumi *teritoriju un nozaru* griezumos. Agregācijas funkcija f tiek izmantota *summa*. Šajā gadījumā netiek izmantots ***gamma*** (γ) mainīgais, kas nodrošina papildus grupēšanu pēc cita mainīgā. Apzīmējums ***tau*** (τ) ir laika griezums, jeb laika rinda *no 2010. Līdz 2015.gadam*.

3. RISINĀJUMS

Pirms tika izstrādāta sistēmas kopējā arhitektūra, tika veikta statistikas pārskatu analīze. Analīzes mērķis bija saprast šo pārskatu struktūru un uzbūvi, kā arī konstatēt galvenās iezīmes, kas ir kopīgas visiem biznesa statistikas pārskatiem. [1. Pielikumā] attēlots viens no statistikas pārskatiem, kas pilnībā dod ieskatu statistikas pārskatu uzbūvē. Galvenās pārskata īpašības, kas tieši attiecas uz pārskatu datu ievadi un apstrādi ir sekojošas:

- Katrs pārskats tiek identificēts ar unikālu indeksu. Pielikumā 1 minētajam pārskatam indekss ir „2-darbs". Indekss ir neatņemama sastāvdaļa pārskatu datu apstrādē. Visi lietotāji, jebkuru pārskatu atpazīst pēc indeksa. Indekss paliek nemainīgs, pat ja mainās pārskata saturs.
- Pārskata ievads apraksta respondentu – fiksētu atribūtu kopums, par konkrēto respondentu.
- Katram pārskatam ir noteikts periodiskums, kas norāda konkrētā pārskata iesniegšanas reižu biežumu gada laikā. Iespējamie pārskata periodiskumi ir – Gads, Pusgads, Ceturksnis, Mēnesis. Piemēram, ja pārskats ir ar mēneša periodiskumu, tad respondentiem dati par konkrēto pārskatu ir jāsniedz par katru mēnesi.
- Pārskata saturs nav fiksēts – tas var mainīties. Bet jebkurš izmainītais pārskats ir spēkā vismaz vienu gadu. Tas nozīmē, ka gada vidū pārskata saturs un izskats netiek mainīts.
- Pārskats sastāv no vienas vai vairākām divdimensiju tabulām - sadaļām. Tabulas ir galvenās vienības, kuras satur ievadāmus statistikas datus
- Katra sadaļa (divdimensiju tabula) sastāv no rindām un kolonām. Katras rindas galvenie aprakstošie lielumi ir rindas kods un nosaukums. Rindas kods ir unikāls pārskata ietvaros. Statistiķi, kas vada un apstrādā pārskata datus, darbojas tikai rindu kodu līmenī – pēc rindas koda, lietotāji ļoti labi atpazīst konkrēta skaitļa nozīmi, bet papildus aprakstiem. Rindas kods ir neatņemams lielums pārskatu datu apstrādē. Tieši tāda pati situācija ir ar kolonām. Arī katru kolonu raksturo kods un nosaukums, bet kolonu kodi, atšķirībā no rindu kodiem, nav unikāli. Katrā jaunā sadaļā kolonu kodi var atkārtoties.
- Katra pārskata sadaļa var būt sekojoša veida:
    - Rindu un kolonu skaits ir fiksēts
    - Sadaļa ar mainīgu rindu skaitu. Šādās situācijās rindām ir piesaistīts kāds konkrēts klasifikators. Respondents atkarībā no uzņēmuma aktivitātēm, izvēlas

konkrētu vienu vai vairāku klasifikatora ierakstus un ievada datus par katru no klasifikatora ierakstiem.

- Sadaļa ar mainīgu kolonu skaitu. Tieši tāpat kā rindās, arī kolonas var būt saistītas ar konkrētu klasifikatoru. Datu ievadē katrs respondents ievada tos klasifikatora kodus, ar kuru saistīta konkrētā respondenta uzņēmējdarbība.

- Pārskata tabulā ievadītās vērtības var būt ne tikai skaitliskas (decimālskaitļi), bet arī teksts, datums, klasifikatoru kodi, loģiskās vērtības (jā/nē).

Nākamais analīzes etaps bija vairāku esošo sistēmu analīze, ar mērķi pārņemt labās īpašības no esošajām pārskatu apstrādes sistēmām. Kā bija iepriekš minēts, visas esošās pārskatu apstrādes sistēmas bija tematiski orientētas, kas nozīmē, ka katra sistēma prata apstrādāt vienu vai nedaudzus konkrētus pārskatus. Katra Informācijas sistēma bija speciāli veidota konkrētam pārskatam. Analizējot visas sistēmas, to arhitektūra bija veidota pēc viena principa. Pārskatu datu ievadei un datu glabāšanai tika izmantotas datu struktūras, kuru vienkāršots piemērs attēlots zemāk tabulā:

3.1. tabula. Tipveida datu struktūra pārskatu datu apstrādes sistēmās

Respondenta kods	Periods	Rindas kods	Kolona1	Kolona2	Kolona3	Kolona4
12345678	2010.g.janvāris	100	34	45		67
12345678	2010.g.janvāris	110	56	67	78	89
12345678	2010.g.janvāris	120			99	58

Tātad šī datu glabāšanas struktūra tika veidota atbilstoši katra konkrētā pārskata izskatam un saturam. Programmatūra nodrošina, ka uzsākot jauna respondenta datu ievadi, datu tabulās tiek izveidoti ieraksti, katrai pārskata datu rindai. Laika periods ir nepieciešams, ja sistēma apstrādā vairāku periodu datus vienā datu tabulā. Atkarībā no pārskatu izmēriem, dati katrai sadaļai var tik glabāti dažādās tabulās, bet tikpat labi var tikt glabāti arī vienā tabulā. Šāda struktūra ir ļoti pateicīga, lai veidotu pārskatu datu ievades formas un arī pati datu struktūra ir ļoti skaidra. Bet izmantojot šādu struktūru nav iespējams panākt īpašības, kas tika definētas sistēmas prasībās:

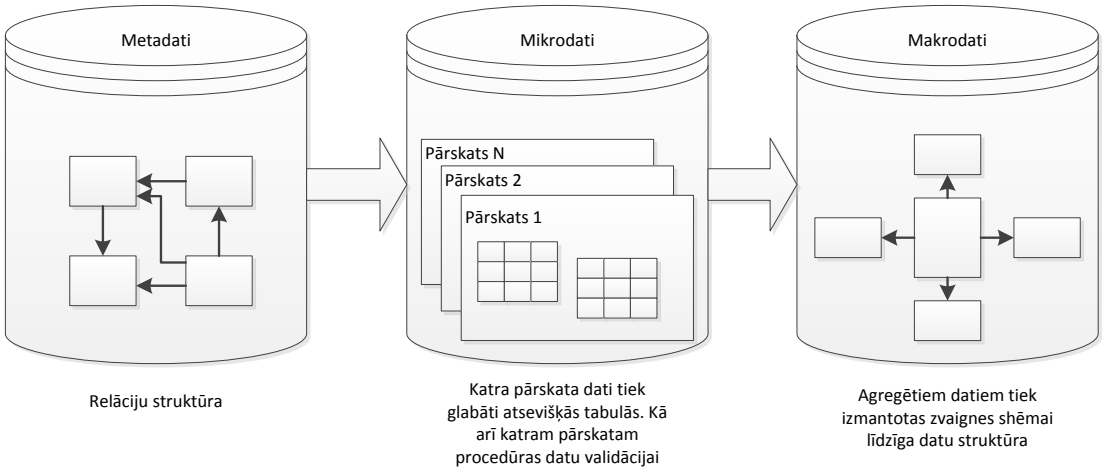
- Šī struktūra nodrošina tikai viena datu tipa informācijas ievadi. Tā kā biznesa statistikā 99% informācijas ir tieši skaitliska, tad šajā gadījumā tā nav lielākā problēma, lai gan,

piemēram, ja rindā 110 un kolonā2 būtu nepieciešams glabāt tekstuālu vērtību, tad būtu nepieciešams sistēmā veidot specifiskus izņēmumus, lai šādu informāciju saglabātu.

- Sistēma paredzēta konkrēta pārskata izskatam. Vadot datus par konkrētu respondentu, sistēma tabulā ievieto programmā iestrādātus fiksētus ierakstus ar rindu kodiem, kas atbilst konkrētam pārskatam. Ja pārskatam nākamajā gadā mainās struktūra – gan rindu kodi, gan saturs, tad nepieciešams pārveidot programmas, lai tiktu veidotas datu rindas ar pareizajiem kodiem
- Mainoties pārskata struktūrai, nekādi netiek glabāta vēsture, jeb sasaiste starp rindu kodiem un tajā vadītai informācijai dažādos gados. Rezultātā nav iespējams konkrētam vienam statistikas rādītājam veidot laika rindas – vērtības vairāku periodu garumā.
- Šī datu struktūra tāpat nav ērta datu analīzei, kad lietotāji paši izvēlas sev nepieciešamos rādītājus un atlasa tos. Piemēram, ar vienkāršu pieprasījumu nav iespējams izveidot datu masīvu, kurā būtu atlasīti respondentu sniegtie dati par 100. Rindas 2.kolonu, 110 rindas 1.kolonu un 120 rindas 2.kolonu.
- Sistēma neuztur pilnīgi nekādu informāciju par pašiem datiem, kādi rādītāji tiek ievadīti, piemēram, 100. Rindā. Katras nozares statistiķi ļoti labi pārzina pārskatus tikai pēc rindu kodiem, tāpēc statistiķiem apstrādājot un analizējot savas nozares pārskatu datus, nav nepieciešama papildus informācija par pārskatu rādītājiem. Bet ja statistiķiem nepieciešams veikt datu analīzi izmantojot vairāku pārskatu datus, tai skaitā citu nozaru pārskatus, tad tas rada problēmas, jo reti kurš statistiķis atpazīst citu statistikas nozaru pārskatu uzbūvi un saturu tikai pēc rindu kodiem.

Summējot esošo sistēmu labās un sliktās puses darba autors atzina, ka šādai struktūrai ir vairāk negatīvo īpašību nekā pozitīvo. Tāpēc sistēmas uzbūvē šādu pieeju nevar izmantot.

Tāpēc sistēmas arhitektūra tika veidota pilnīgi no jauna, neizmantojot esošo sistēmu arhitektūru. Veidojot jaunās sistēmas arhitektūru, visi dati tika sadalīti 3 lielās datu bāzēs, kā attēlots zemāk attēlā:



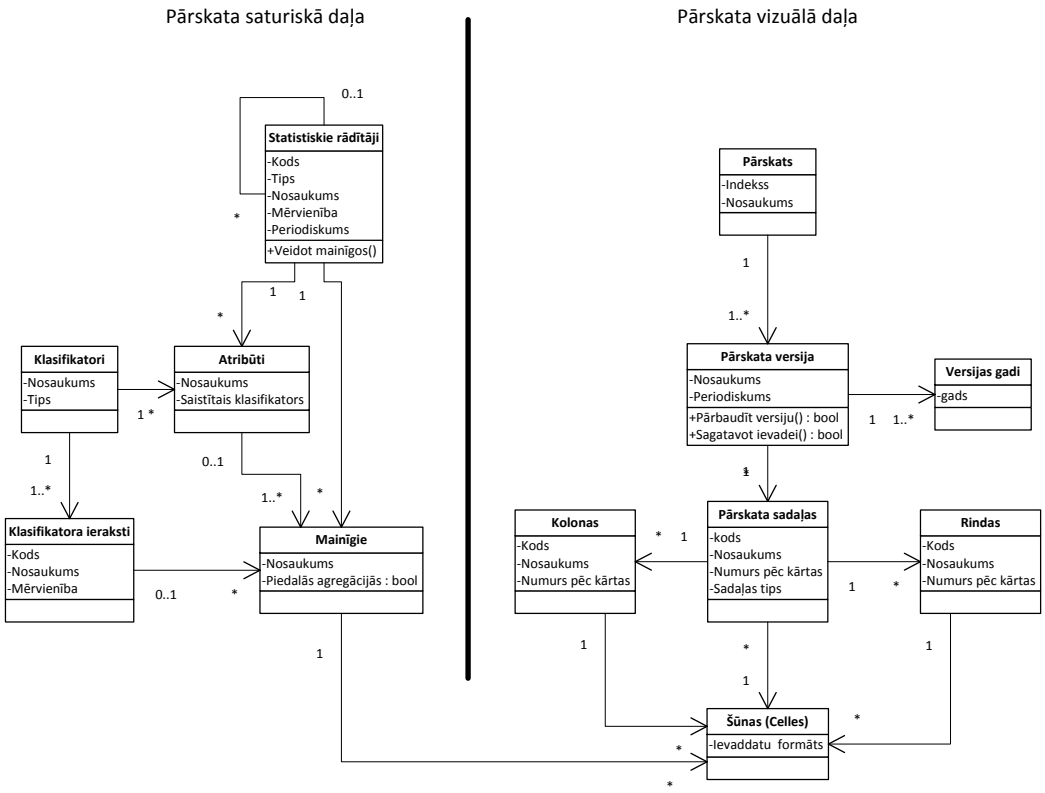
3.1. att. Pārskatu datu apstrādes sistēmas datu organizācija

- Metadatu bāze – visi detalizēti pārskatu apraksti – gan pārskatu saturs, gan izskats, gan struktūra un visa pārējā pārskatus aprakstošā informācija, kas nepieciešama pilnvērtīgai datu ievades un apstrādes formu ģenerācijai. Tāpat metadatu bāze paredzēta, lai glabātu pārskatu validāciju nosacījumus, agregāciju un kopsavilkumu aprakstus. Metadatu bāzē arī tiek glabāti tie dati, kas ir kopīgi visiem pārskatiem – piemēram, klasifikatori.
- Mikrodatu bāze – pirmdatu bāze. Šajā bāzē glabājas visu pārskatu respondentu saraksti, kā arī visu pārskatu ievadītie pirmdati. Papildus šajā bāzē tiek glabāta informācija, kas saistīta ar šo pirmdatu apstrādi, piemēram, validācijas kļūdu protokoli.
- Makrodatu bāze – agregēto datu bāze. Šajā bāzē glabājas visu pārskatu agregētie dati, kā arī aprēķinātie kopsavilkumi. Datu glabāšanai tika izvēlēta datu struktūra, kas ir ļoti līdzīga zvaigznes shēmai.

3.1. Metadatu modulis – pārskatu apraksts

Šajā sadaļā tiks aprakstīta metadatu bāzes struktūra. Metadatu bāzē tiek aprakstītas tikai un vienīgi pārskatu veidlapas. Šajā sadaļā netiek apskatītas jau aizpildītas pārskatu veidlapas un to dati – tikai pārskatu veidlapu apraksti.

Veidojot sistēmas arhitektūru, katrs statistikas pārskata veidlapa tiek sadalīts atsevišķās vienībās. Sīkākā vienība ir katras divdimensiju tabulas katra datus saturošā šūna, jeb celle. Arhitektūras veidošanā tika ņemtas vērā Latvijas Statistikas pārskatu veidlapu īpatnības, kā arī tika izmantotas Bo Sundgren ieteiktās metodes statistisko datu aprakstīšanai – piesaistot katrai pārskata veidlapas šūnai statistiskos mainīgos. Tieši galvenais uzsvars tika likts uz mainīgo veidošanu un izmantošanu. Mainīgais sistēmā tika veidots kā neatkarīgs no pārskata struktūras un izskata lielums, kas tieši nekādi nav saistīts ar pārskata konkrētu struktūru, bet tai pat laikā katrs statistiskais mainīgais pilnībā apraksta katru pārskata veidlapas vērtību – tās būtību, periodiskumu, mērvienības. Sistēmā statistiskie mainīgie tiek izmantoti kā pamatobjekti (vērtības, identifikatori), kam tiks piekārtotas visas reālās respondentu sniegtās pārskatu datu vērtības – gan pirmdatu, gan arī agregēto datu līmenī. Darba autors rezultātā izveidoja metadatu struktūru, kura attēlota zemāk attēlā.



3.2. att. Pārskata metadatu apraksta klašu diagramma

Kā redzams attēlā – pārskata struktūra tiek veidota no divām pusēm. No vienas puses tiek aprakstīta pārskata vizuālā daļa – pārskats tiek sadalīts sadaļās, kur katra sadaļa ir

divdimensiju datu tabula. Katrai sadaļai definē rindas un kolonas. Kā arī katrai sadaļai veido šūnas (celles), kas ir rindas un kolonas krustpunkts, un kas kalpo kā datu ievades vienība. No otras puses tiek veidota saturiskā daļa – tiek definēti statistiskie rādītāji, tiem tiek piesaistīti (ne obligāti) atribūti (klasifikācijas). Apvienojot rādītāju ar atribūtu, no šīs kombinācijas tiek veidoti mainīgie.

Šī struktūra pilnībā ir pietiekoša, lai aprakstītu pārskata saturu un izskatu metadatu bāzē. Šī informācija ir pietiekama, lai to izmantojot, varētu veidot datu ievades formas, kas atbilst katra pārskata struktūrai. Zemāk aprakstīts katrs attēlā iekļautais objekts sīkāk.

3.1.1. Pārskats

Šis objekts vispārīgi definē katru pārskatu, kas ir ievadīts un aprakstīts sistēmā. Pārskats, tas ir konkrētas tēmas apsekojums, kuram tālāk tiks pakārtota veidlapas struktūra (katru gadu sava, ja pārskats no gada uz gadu mainīsies). Atbilstoši aprakstītajai veidlapas struktūras, pārskatam tiek ievadīti respondentu sniegtie pārskata dati un tiek veikta datu analīze. Galvenais pārskatu aprakstošs atribūts ir indekss. Pārskata indekss ir nemainīgs pārskata identifikators, neatkarīgi no tā, vai pārskata struktūra katru gadu mainās, vai nē. Pēc pārskata indeksa statistiķi viennozīmīgi atpazīst pārskatu. Piemēri pārskatu indeksiem ir „2-darbs", „1-tirdzniecība", „1-gada". Tāpat arī lietotāju tiesības darbam ar sistēmas funkcijām tiek definētas pārskata griezumā, kas norāda, kurš lietotājs kādas darbības ar kādiem pārskatiem var veikt.

3.1.2. Pārskata versija

Pārskata versija ir objekts, kam piekārtota konkrēta izskata un satura pārskats. Ar pārskata versiju attiecīgi saprot konkrētā pārskata kāda konkrēta gada veidlapu. Pārskata veidlapas saturs var mainīties, bet katra pārskata veidlapas saturs ir fiksēts vismaz vienu gadu. Ja pārskata veidlapa nemainās, tad šī pati pārskata versija tiek izmantota vairākus gadus, turpretī, ja nākamajā gadā pārskatam ir ievērojamas saturiskās vai vizuālās izmaiņas, tad nepieciešams veidot jaunu pārskata versiju.

Tā kā pārskata versija – tas jau ir konkrēta pārskata pilnībā aprakstīta veidlapa katram konkrētam gadam Metadatu bāzē, tad šim objektam ir nepieciešamas vairākas svarīgas

metodes, kas nepieciešamas katras pārskata versijas pārvaldībai. Svarīgākās no metodēm ir aprakstītas zemāk.

Metode – *pārbaudīt pārskatu* – nodrošina visu metadatu aprakstu savstarpēju pārbaudi vienas pārskata versijas ietvaros. Šīs pārbaudes nepieciešamas, lai pārliecinātos vai pārskatā sagatavē metadatu bāzē ir pareizi izveidotas sadaļas, vai sadaļās ir definētas rindas/kolonas. Šī metode arī pārbauda izveidoto pārskata ievades vienības – šūnu (ceļļu) pareizību, kā arī kontrolē validācijas likumu sintaksi un pareizību..

Metode – *sagatavot pārskatu ievadei* – nodrošina visu nepieciešamo objektu sākotnēju izveidi un aizpildi mikrodatu bāzē. Kā tika minēts iepriekš, mikrodatu bāzē tiek glabāti katra pārskata visi respondentu saraksti, kā arī respondentu sniegtās un ievadītās reālās pārskatu vērtības. Katram pārskatam respondentu iesniegtie dati sastāda ievērojamu datu apjomu, kurš atkarībā no pārskata periodiskuma katru gadu papildinās, tad sistēmas arhitektūrā tika paredzēts, ka katra pārskata respondentu sniegtie pārskatu dati tiek glabāti atsevišķās tabulās. Piemēram, pārskata „2-darbs" respondentu iesniegtie pārskatu dati glabājas atsevišķi no pārskata „1-gada" datiem. Tādejādi tiek uzlabota gan ātrdarbība, gan arī vienlaicīgu lietotāju darbības ar pārskatu datiem. Toties kā mīnuss šādai datu sadalīšanai atsevišķās tabulās ir tas, ka visi SQL pieprasījumi pārskatu datu apstrādei ir jāģenerē dinamiski.

3.1.3. Versijas gadi

Katra pārskata versija – kas ir noteiktā pārskata veidlapas apraksts, ir spēkā vismaz gadu. Šī klase definē tos gadus, kad konkrēta versija tieši ir spēkā. Attiecīgi sistēma ļauj apstrādāt tikai tos pārskatu datus, kuriem ir definēti gadi, kad katra konkrēta datu versija ir bijusi spēkā. Ja pārskatam nākamajā gadā nav nekādu izmaiņu, tad versijai vienkārši definē jaunu gadu, un nekādi papildus apraksti nav nepieciešami un sistēma ir gatava apstrādāt konkrētā pārskata nākamā gada datus.

3.1.4. Pārskata sadaļas

Jebkurš statistikas pārskats sastāv no vienas vai vairākām datu tabulām, kā tas redzams piemēra pārskatā, skatīt [1. pielikums]. Pārskats tiek sadalīts atsevišķās tabulās, lai nodalītu atsevišķas tēmas, par kurām respondentam jāsniedz dati. Katra tabula – tā ir divdimensiju struktūra, kas satur rindas un kolonas. Metadatu bāzē katru šādu tabulu sauc par sadaļu.

Metadatu bāzē katru šo sadaļu atsevišķi apraksta. Katrai sadaļai ir savs izmērs (rindu, kolonu skaits), kā arī sadaļām var būt dažādi tipi. Sistēmā iespējams sadaļām definēt 3 tipus, kas pilnībā ļauj definēt jebkura statistikas pārskata veidus. Papildus katrai sadaļai tiek piešķirts kods un nosaukums. Papildus sadaļām tiek piešķirts kārtas numurs, kas nosaka sadaļu secīgu izkārtošanu. Sadaļām iespējami sekojoši tipi:

- fiksēts izmērs – sadaļa, kurai ir fiksēts rindu un kolonu skaits;
- sadaļa ar mainīgu rindu skaitu. Šādās sadaļās sākas (ne obligāti) ar fiksētām rindām, kurām ir piešķirts rindas kods. Pēc šīm fiksēta skaita rindām (ne obligātām) seko mainīgs rindu skaits, kas vienmēr ir saistīts ar konkrētu klasifikatoru. Katrs respondents aizpilda datus tikai par tiem klasifikatora ierakstiem, kas atbilst konkrētā respondenta darbībai. Zemāk attēlā attēlota sadaļa ar mainīgu rindu skaitu.

4. Apkalpoto cilvēku skaits pēc dzīvesvietas valsts

4000. rindas 2. aile ir vienāda ar 3000. rindas 4. aili

Pavadītā nakts ir katra nakts, kuru viesis pavada (vai ir reģistrējies) kādā tūristu mītnē, to summa veido kopējo pavadīto nakšu skaitu.

Piemēri aizpildīšanai:

Ja 5 cilvēki no Latvijas pavadījuši katrs 1 nakti, tad 4000. rindas 1. ailē ir rakstāms „5" un 4000. rindas 2. ailē ir rakstāms „5" (5 x 1 = 5 naktis).

Ja 3 cilvēki no Igaunijas pavadījuši katrs 2 naktis, tad 4000. rindas 1. ailē ir rakstāms „3" un 4000. rindas 2. ailē ir rakstāms „6" (3 x 2 = 6 naktis).

Ja 2 cilvēki no Lietuvas ir pavadījuši katrs 4 naktis un vēl 1 cilvēks no Lietuvas ir pavadījis 6 naktis, tad 4000. rindas 1. ailē ir rakstāms „3" (2 + 1 = = 3 cilvēki) un 4000. rindas 2. ailē ir rakstāms „14" (2 x 4+ 1 x 6 = 14 naktis).

Valstu saraksts	Valsts kods	Apkalpoto cilvēku skaits pārskata mēnesī	Pavadītās naktis pārskata mēnesī *(visiem apkalpotajiem cilvēkiem)*
A	B	1	2
PAVISAM	4000		
tai skaitā *(norādiet)*			

3.3. att. Sadaļas piemērs ar mainīgām rindām

Šajā sadaļā mainīgām rindām tiek piekārtots valstu klasifikators. Attiecīgi respondents vada datus tikai par tām valstīm, par kurām iespējams sniegt informāciju.

- Sadaļa ar mainīgu kolonu skaitu. Šādās sadaļās princips ir tieši tāds pats kā iepriekšējā situācijā ar mainīgu rindu skaitu. Tikai šajā gadījumā klasifikators tiek piekārtots kolonām. Respondents katrai pievienotai kolonai izvēlas atbilstošo klasifikatora ierakstu, par kuru rindās tiek aizpildīta nepieciešamā informācija. Zemāk attēlā piemērs sadaļai ar mainīgu kolonu skaitu.

	Rindas kods	Kopā rūpnieciskajā ražošanā (2., 3., 4. ailes summa), ***euro***	tai skaitā pa rūpnieciskās darbības veidiem		
A	C	1	2	3	4
Rūpniecības produkcijas izlaide *Pārskata mēnesī saražotās produkcijas un izpildīto rūpnieciska rakstura darbu apjoms**	110				
Rūpniecības apgrozījums, bez pievienotās vērtības un akcīzes nodokļa	111				
vietējā tirgū	116				
eksports (ārpus valsts robežām)	117				
tai skaitā uz eirozonas valstīm *Austrija, Beļģija, Francija, Grieķija, Igaunija, Itālija, Īrija, Kipra, Lietuva, Luksemburga, Malta, Nīderlande, Portugāle, Slovākija, Slovēnija, Somija, Spānija, Vācija*	118				

3.4. att. Piemērs pārskata sadaļai ar mainīgu rindu skaitu

Šajā piemērā, sākot ar otro kolonu, respondents norāda savus darbības veidu kodus un sniedz datus par katru darbības veidu. Attiecīgi šai sadaļai kolonās ir piekārtots NACE (darbības veidu) klasifikators.

3.1.5. Rindas, kolonas

Katrai sadaļai ir savs izmērs un to nosaka rindu/kolonu skaits. Šie objektu veidi definē katras pārskata sadaļas rindas un kolonas. Katru rindu un kolonu identificē kods no nosaukums. Papildus īpašība ir kārtas numurs, kas norāda kādās secībā rindas un kolonas tiek sakārtotas, jo ne vienmēr rindu vai kolonu kodi nosaka pareizo secību. Svarīga pārskatu īpašība, kas tiek kontrolēta ir tā, ka rindu kodi pārskatā neatkārtojas un tie ir unikāli. Tātad, ja vienā sadaļā ir rinda ar kodu 100, tad citās sadaļās rindas ar šādu kodu vairs nav iespējamas. Rindu kodu ir unikāli arī tāpēc, ka datu apstrādē statistiķi ļoti labi orientējas pārskata struktūrā tieši pēc rindu kodiem. Kā arī visās esošajās datu apstrādes sistēmās rindu kodi vienmēr tiek izmantoti attēlojot sadaļas datu ievades formās. Kolonu kodiem turpretī šādas unikāla koda īpašības nav. Tie ir unikāli tikai vienas sadaļas ietvaros. Tātad katrā pārskata sadaļā var būt kolona ar kodu 1. Ja aprakstāmā sadaļa ir ar mainīgu rindu skaitu, tad šīs rindas arī nepieciešams aprakstīt kopā ar pārējām sadaļas rindām. Vienīgi nav nepieciešams aprakstīt visas mainīgās rindas, jo to daudzums jau nav zināms. Visas mainīgās rindas definē metadatos kā vienu konkrētu rindu, tieši tāpat kā visas pārējās, piešķirot kodu un nosaukumu, kā arī kārtas numuru. Nekādas papildus īpašības rindām netiek definētas. Jo pašai sadaļai jau tika

definēta īpašība, vai tā ir fiksēta izmēra sadaļa, vai arī ar mainīgu rindu/kolonu skaitu. Tātad ja sadaļai ir definēts mainīgs rindu skaits, tad sistēma automātiski pēdējo rindu (ar lielāko kārtas numuru) uzskatīs par mainīgo rindu un datu ievades formas automātiski spēs šīs rindas pievienot atkarībā no katra respondenta iesniegtajiem pārskata datiem. Analoģiska situācija ir arī ar sadaļām ar mainīgo kolonu skaitu. Metadatos tiek aprakstīta tikai viena šī mainīgā kolona, kurai ir jābūt pēdējai.

Ja sadaļai ir aprakstītas rindas un kolonas, tad tas jau nosaka katras sadaļas matricas izmēru – cik rindas reiz cik kolonas. Bet šie rindu/kolonu apraksti nosaka tikai matricas izmēru, bet ne katra matricas elementa (ko sauc arī par šūnu, celli) īpašības. Pie tam mainīgu rindu/kolonu gadījumā arī ir zināms kura ir tā pēdējā rinda vai kolona, kas datu ievades formai jāprot pavairot tik eksemplāros, cik datu respondents ir sniedzis. Katras matricas elementa (šūnas vai celles) īpašības tiks aprakstītas ar citu objektu palīdzību.

Līdz šim aprakstīto objektu kopums definē pārskata vizuālo izskatu, cik sadaļu ir pārskata veidlapā, kādas katrai sadaļai ir rindas un kolonas. Tālāk sekos objekti, kas apraksta tieši pārskata saturisko pusi, kādus datus katra pārskata veidlapa saturēs.

3.1.6. Klasifikatori

Pārskatu datu apstrādē klasifikatori tiek plaši izmantoti. Pie tam klasifikatorus izmanto arī Statistiskas Uzņēmumu Reģistrs. Sistēmas arhitektūra neparedz pilnvērtīgu klasifikatoru uzturēšanas funkciju, bet tikai minimāli nepieciešamās informācijas saglabāšanu, kas nepieciešama gan pārskatu veidlapu aprakstīšanai, gan pārskatu datu ievadei un datu apstrādei. Katru klasifikatoru sistēmā apraksta nosaukums un tips. Iespējami sekojoši klasifikatoru tipi:

- Lokālais – šāds tips tiek piešķirts parasti neliela izmēra klasifikatoriem. Šie klasifikatori tiek izmantoti tikai pārskata veidlapu aprakstīšanai un statistisko mainīgo veidošanai. Piemēri šādiem klasifikatoriem varētu būt „*Ēku sienu materiāli*”, ko izmanto pārskata 1-būvniecība veidlapas aprakstos, vai arī „*Zivju sugas*”, ko izmanto pārskata veidlapas 1-zvejniecība aprakstīšanai
- Starptautiskais, nacionālais – šāds tips tiek piešķirts visiem liela izmēra klasifikatoriem (ierakstu skaits sākot no apmēram 50 un vairāk). Šie klasifikatori tiek izmantoti arī Statistikas Uzņēmumu Reģistrā, definējot dažādus uzņēmumu atribūtus, kas ir saistīti ar klasifikatoriem. Kā arī šos klasifikatorus izmanto pārskatu veidlapu

aprakstīšanā, sadaļās, kurās ir mainīgs rindu/kolonu skaits. Piemēri starptautiskiem un nacionāliem klasifikatoriem ir – teritoriālais klasifikators ATVK, starptautisks nozaru klasifikators NACE, starptautisks produktu veidu klasifikators PRODCOM.

- Intervālu klasifikatori – šādu klasifikatoru pielietojums ir tikai un vienīgi datu agregācijas procesos. Pārskatu veidlapu aprakstiem, kā arī pārskatu datu ievadē šādi klasifikatori netiek izmantoti, tāpēc šāda tipa klasifikatori turpmāk netiek apskatīti.

3.1.7. Klasifikatora ieraksti

Šī objektu klase definē klasifikatoru ierakstus. Kā bija minēts iepriekš, klasifikatori plaši tiek izmantoti ne tikai pārskatu veidlapu aprakstos un pārskatu datu ievades un apstrādes procesos, bet arī Statistikas Uzņēmumu Reģistrs izmanto metadatos definētos klasifikatorus dažādu uzņēmumu atribūtu sasaistei ar klasifikatoru kodiem. Visi izmantotie klasifikatori sistēmā tika standartizēti – tas ir katru klasifikatora ierakstu raksturo vieni un tie paši atribūtu, neatkarīgi no klasifikatora veida un nosaukuma. Katru ierakstu raksturo sekojoši atribūti – kods, nosaukums, mērvienība, ieraksta spēkā esamības datumu intervāls. Šis atribūtu kopums ir pilnībā pietiekams, lai nodrošinātu visu ar pārskatu datu apstrādi saistīto procesu veikšanu.

3.1.8. Rādītāji

Šis objektu tips apraksta statistiskos rādītājus. Katrs rādītājs tā ir konkrēta noteikta ekonomiskā aktivitāte, kas ir izmērāma, un par kuru respondents sniedz datus. Rādītāji paši par sevi nav saistīti ar pārskatu, tāpēc šī objektu kopa ietver sevī visu pārskatu visus rādītājus, kas ir ievadīti sistēmā. Aprakstot katru jaunu pārskatu, tiek veidoti jauni rādītāji, vai arī tiek izmantots kāds esošais. Esošie rādītāji tiek izmantoti situācijās, ja vairākos pārskatos respondentam tiek jautāti vieni un tie paši dati. Šī kopējā rādītāju datu bāze tieši palīdz analizēt pārskatu saturu un izvairīties no šādām situācijām, kad dažādos pārskatos tiek jautāts viens un tas pats. Tipiski rādītāju piemēri:

- Rūpniecības produkcijas izlaide
- Rūpniecības apgrozījums bez pievienotās vērtības un akcīzes nodokļa
- Bruto darba samaksa natūrā
- Darba ņēmēju skaits pamatdarbā (ar algas nodokļa grāmatiņām)

Katru rādītāju raksturo sekojoši lielumi:

- Nosaukums – precīzs rādītāja nosaukums, kas viennozīmīgi apraksta konkrēto aktivitāti
- Kods – rādītāja kods. Tiek pielietoti, lai identificētu svarīgākos rādītājus (piemēram, kuru dati jāiesniedz Eirostat birojam)
- Periodiskums – katrs rādītājs identificē kādu aktivitāti par noteiktu laika periodu. Periodiskums norāda to, ar kādu biežumu konkrētā rādītāja dati tiek iegūti no respondenta. Pārsvarā gadījumu rādītāja un pārskata periodiskumi sakrīt, bet atsevišķās situācijās tā nav. Tā kā periodiskums ir neatņemama rādītāja sastāvdaļa, tad var būt situācijas, kad viens un tas pats rādītājs tiek ievākts dažāda periodiskuma pārskatos. Tādās situācijās nepieciešams veidot atsevišķus rādītājus – katru ar savu periodiskumu
- Mērvienība. Raksturo kādās mērvienības konkrētā rādītāja dati no respondentiem tiek iegūti
- Rādītāja tips. Nosaka konkrētā rādītāja vērtības tipu. Biznesa statistikā lielākoties visi rādītāji ir ar tipu skaitlis. Bet atsevišķās situācijās nepieciešami rādītāji arī ar citiem tipiem. Rādītāju tipi var būt sekojoši:
    - Skaitlis. Lielākā daļa visu biznesa statistikas rādītāju ar šādu tipu. Tieši šādu rādītāju dati tiek analizēti, apkopoti un izplatīti.
    - Datums
    - Klasifikators. Šāda veida rādītāji parasti detalizē sīkāk kāda cita rādītāja informāciju. Šāda veida rādītāji parasti kļūst par agregētu datu dimensiju pārskata datu agregācijas procesā. Piemēram, rādītājs ar klasifikatora tipu ir sekojošs: Mājsaimniecības, biroja vai pamatdarbības vienības adreses ATVK kods.
    - Teksts – šie pārsvarā ir sava veida palīgrādītāji, kas tiek izmantoti, lai respondents brīvā tekstā raksturotu kādu konkrētu aktivitātes aspektu.
- Piezīmes – nepieciešams, lai sarežģītākiem rādītājiem papildus ieviestu skaidrojumu, kas pilnībā izklāsta konkrētā rādītāja jēgu, ja rādītāja nosaukums nesniedz pilnu ieskatu.

Tāpat arī paši rādītāji veido savstarpēju hierarhiju – jo eksistē vispārīgi un globāli rādītāji, kā arī rādītāji, kas ir detalizētāki un sniedz informāciju par kādu konkrētāku ekonomiskās aktivitātes aspektu. Piemēram, galvenais rādītājs ir „Kopējais veikto būvniecības darbu apjoms pasūtītājiem". Turpretī kā pakārtotais rādītājs ir „Ārpus republikas veikto būvniecības darbu apjoms".

3.1.9. Atribūti

Atribūtu klase detalizē rādītājus. Katrs rādītāja atribūts sniedz informāciju par to, ka konkrēta rādītāja informācija ir pieejama sīkākā detalizācijas pakāpē jeb griezumā. Atribūts – tas vienmēr ir klasifikators. Piemēram, ja ir rādītājs „Saražotā rūpniecības produkcija" un tam piekārto atribūtu, jeb klasifikatoru – produkcijas veidi, tas nozīmē, ka respondents sniegs informāciju par saražoto rūpniecības produkciju sadalījumā pa produkcijas veidiem. Atribūtiem ir liela nozīme mainīgo veidošanā, kā arī tie vienmēr tiek izmantoti, lai aprakstītu pārskatu sadaļas ar mainīgām rindām vai kolonām. Rādītājiem atribūti var arī nebūt un tas nozīmē, ka dati par konkrētu rādītāju netiek vākti kādā klasificējamā sadalījumā. Katru atribūtu raksturo piesaistītais klasifikators, kas norāda, kādā griezumā dati tiks iegūti, kā arī atribūta nosaukums, kas veidojas no rādītāja nosaukuma savienojot to ar klasifikatora nosaukumu.

3.1.10. Mainīgie

Mainīgie (statistiskie mainīgie) ir pārskatu veidlapu aprakstu galvenais pamatelements metadatu bāzē. Katra pārskata veidlapas šūna vienmēr ir piesaistīta kādam konkrētam mainīgajam. Tātad mainīgais ir sīkākā pārskata veidlapā ievadāmo datu vienība metadatu bāzē un katrs mainīgais pilnīgi precīzi apraksta kādu konkrētu uzņēmuma ekonomisko aktivitāti no saturiskā viedokļa. Varētu teikt, ka mainīgais – tā ir pārskata veidlapas šūna, kurai nav piešķirta konkrēta koordināte, jo mainīgais nav piesaistīts nevienai pārskata sadaļas šūnai (rindas/kolonas krustpunktam). Piemēram, ja pārskata veidlapa sastāv no vienas sadaļas, kurai ir 10 rindas un 10 kolonas, tad nepieciešams izveidot un aprakstīt 100 mainīgos, kur katrs mainīgais apraksta kādu no sadaļās matricas šūnām. Mainīgie tiek veidoti, kā rādītāju un atribūtu kombinācija, bet tikpat labi mainīgais var tikt veidots tikai no rādītāja, bez atribūta. Ja rādītāji metadatos tiek aprakstīti nesaistīti ar kādu konkrētu pārskatu, tad mainīgie tiek veidoti jau konkrētam pārskatam. Katrs mainīgais apraksta vienu konkrētu pārskata veidlapas vērtību. Mainīgā nosaukums – tas ir galvenais lielums, kas pilnībā apraksta katru konkrētu pārskata veidlapas šūnu. Komplektā ar piesaistītā rādītāja īpašībām (periodiskums, tips, mērvienība) tas pilnībā skaidro katru pārskata veidlapas vērtību. Pievienojot mainīgo pārskata sadaļas šūnai – mainīgais iegūst vizuālu koordināti sadaļas tabulā, kurā tiek ievadīti un attēloti dati. Bo Sungren teorijā arī pieminēti mainīgie, kā datu pamatvienības. Mainīgos iespējams izveidot 3 veidos:

1) Rādītājs bez atribūta – tiek izveidots viens mainīgais. Šāda mainīgā nosaukums ir tieši tāds pats kā rādītājam un principā šāds mainīgais apraksta konkrēto rādītāja aktivitāti kopā vai pavisam. Piemēram, ja ir rādītājs „Saražota rūpniecības produkcija", tad izveidojot mainīgo (bez atribūta) – tiek izveidots mainīgais ar tādu pašu nosaukumu. Attiecīgi pārskata vērtība, kuru sniedzis respondents par konkrētu mainīgo tieši arī identificē konkrēta respondenta (uzņēmuma) saražoto rūpniecības produkciju kopā.
2) Rādītājs ar atribūtu, kuram piesaistīts klasifikators ar tipu „lokālais vai vietējais". Šādi klasifikatori ir nelielu ierakstu daudzumu, un tiek veidoti tikai un vienīgi pārskatu specifikas dēļ (lai vieglāk būtu aprakstīt pārskatu, kā arī vieglāk būtu analizēt to iegūtās un ievadītās vērtības). Veidojot mainīgos no šādas kombinācijas tiek izveidoti tik daudz mainīgo, cik ierakstu ir piesaistītajā klasifikatorā. Tātad tiek izveidots viens mainīgais katram klasifikatora ieraktam. Piemēram, ja ir rādītājs „*iepirkto graudu daudzums*" un atribūtam ir piesaistīts klasifikators „*graudaugu kultūras*" ar 4 ierakstiem – kvieši, mieži, rudzi, auzas. Tad no šādas kombinācijas tiek izveidoti 4 mainīgie. Izveidotajiem mainīgajiem būs sekojoši nosaukumi:
    - iepirkto graudaugu kultūras – kvieši;
    - iepirkto graudaugu kultūras – mieži;
    - iepirkto graudaugu kultūras – rudzi;
    - iepirkto graudaugu kultūras – auzas;

    Attiecīgi katrs mainīgais raksturo rādītāju bet par konkrētu tēmu.

3) Rādītājs ar atribūtu, kuram piesaistīts klasifikators ar tipu „starptautiskais vai nacionālais". Šādi klasifikatori satur ļoti daudz ierakstus, sākot no daži desmiti līdz tūkstoši. Veidojot mainīgos no šādas kombinācijas tiek izveidots tikai viens mainīgais. Piemēram, ja ir rādītājs „*Strādājošu skaits*" un atribūtam pievienos klasifikators „*Nozares – NACE*", tad no šādas kombinācijas tiks izveidots mainīgais ar nosaukumu „*Strādājošo skaits sadalījumā pa – Nozares – NACE*". Šādu mainīgo pielietojums ir specifisks. Šādi mainīgie tiek izmantoti tikai lai aprakstītu pārskatu veidlapu tās sadaļās, kurās ir mainīgs rindu vai kolonu skaits. Kad respondents sniedz informāciju par strādājošo skaitu, tad uzrāda konkrētas uzņēmuma nozares un katrai nozarei ievada atbilstošu strādājošo skaitu. Tādejādi datu līmenī konkrētajam mainīgajam piesaistītie dati iegūst arī konkrētu klasifikatora vērtību, kādu to norādījis respondents. Bet metadatu aprakstu līmenī mainīgajam tiek piesaistīts tikai klasifikators.

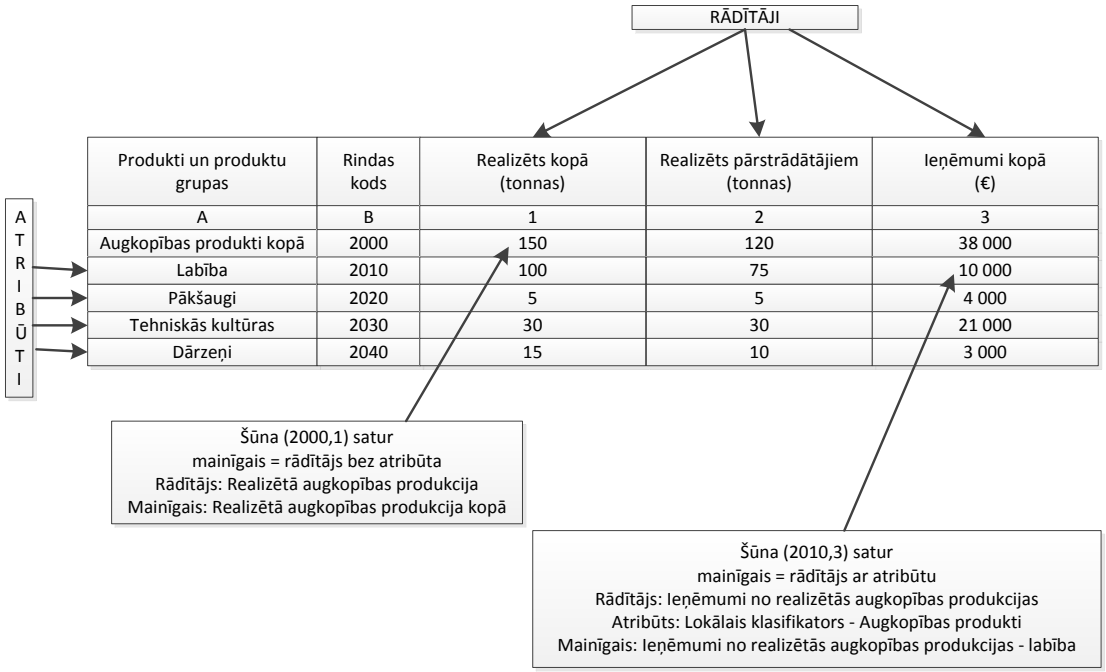
Tātad īsi summējot – mainīgie – tie ir galvenie statistikas pārskatu veidlapu aprakstošie elementi. Jebkurā pārskata ievadītā vērtība vienmēr tiek piesaistīta mainīgajam. Šāda piesaiste nodrošina to, ka neatkarīgi kā mainās pārskata izskats – viena vieda informācija, kas tiek ievadīta no pārskatiem vairāku gadu garumā tiek piesaistīta vienmēr vienam un tam pašam mainīgajam, kas dod iespēju atlasīt laika rindas – konkrēta rādītāja vērtības par vairākiem periodiem (pat ja laika gaitā ir mainījusies pārskata struktūra). Jebkuras darbības ar pārskatu datiem, vai tā būtu datu ievade, validācija, agregācija, analīze – vienmēr tiek veiktas ar atbilstošu mainīgo palīdzību.

3.1.11. Šūnas (Celles)

Kad ir definēti visi mainīgie, kas apraksta katru pārskata veidlapas datu vienību no saturiskā viedokļa, tad beidzamais solis pārskata veidlapas aprakstā ir definēt visas pārskata veidlapas šūnas.

Šūnas ir tie beigu objekti, kuri izveido pārskata vizuālo izskatu, kā arī nodrošina datu vienību (datu ievadei, apskatei, analīzei) veidošanu. Katra šūna tiek veidota kā konkrētas rindas un kolonas kombinācija, kurai pievieno saturiski atbilstošo mainīgo. Tādejādi katrai šūnai ir koordināte, kur šī šūna atrodas, kā arī pievienotais mainīgais saturiski nosaka, kāda informācijas vienība tiek ievadīta vai attēlota konkrētajā pārskata šūnā. Veidojot šūnas, uzmanība ir jāpievērš sadaļām, kurām ir mainīgs rindu vai kolonu skaits. Augstāk bija minēts, ka katrai sadaļai tiek definēts sadaļas tips (vai tā ir fiksēta izmēra sadaļa, vai arī sadaļa ar mainīgu rindu vai kolonu skaitu). Šādām sadaļām metadatos tiek arī aprakstīta viena rinda vai kolona, kura paredzēta šo mainīgo rindu aprakstīšanai. Veidojot šūnas (celles), ir svarīgi pievienot pareizus mainīgos tām rindām, kuras pārskatu datu ievades procesā var būt nezināms daudzums. Piemēram, ja sadaļu definē ar mainīgu rindu skaitu, tad šādā sadaļā pēdējās rindas šūnām ir atļauts pievienot tikai tādus mainīgos, kas ir veidoti kā kombinācija no rādītāja un atribūta, kuram piesaistīts klasifikators ar tipu – starptautisks, nacionālais. Cita veida mainīgos mainīgās rindas šūnām pievienot nedrīkst. Ja mainīgajā rindā ir jādefinē vairākas šūnas (sadaļas matrica satur vairākas kolonas), tad visām šīm šūnām ir jābūt definētām, izmantojot atribūtu, kuram piesaistīts viens un tas pats klasifikators ar tipu „starptautisks, nacionālais”. Tāds pats princips tiek izmantots aprakstot sadaļu ar mainīgu kolonu skaitu šūnas. Augstāk, aprakstot pārskata versijas metodes, tika minēta specifiska metode „Pārbaudīt pārskatu”. Šī metode tieši pārbauda vai sadaļās ar mainīgu rindu un kolonu skaitu tiek piesaistīti pareizi mainīgie, ar pareiziem klasifikatoriem.

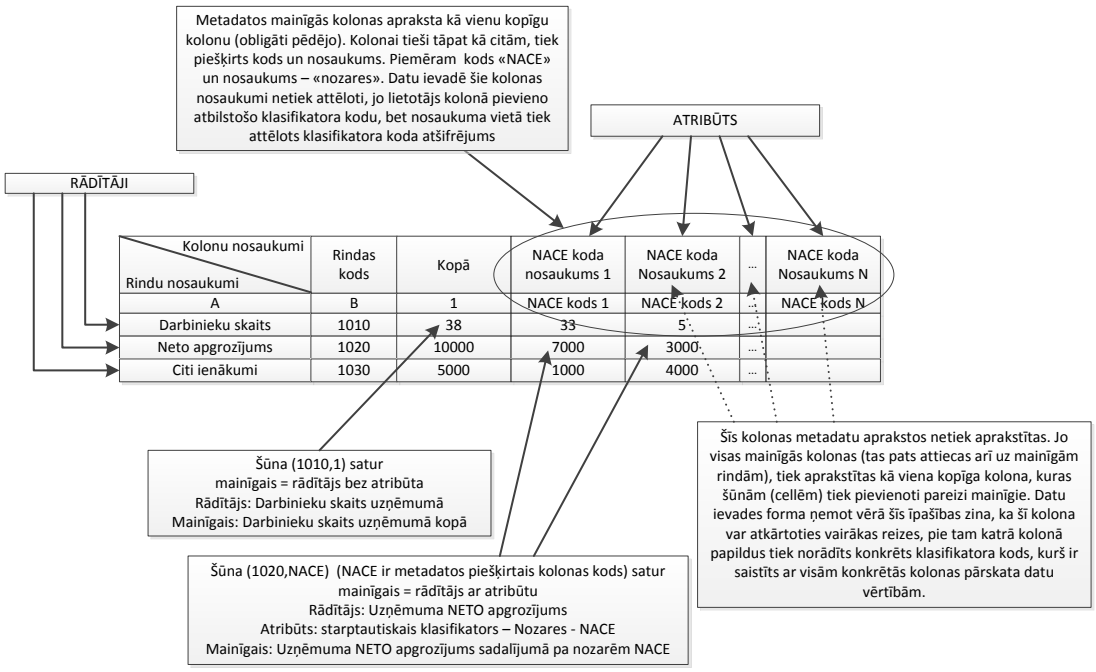
Zemāk attēlā attēlota vienkārša sadaļa – ar fiksētu rindu un kolonu skaitu.



3.5. att. Šūnu definēšana fiksēta izmēra pārskata veidlapas sadaļai

Lai pilnībā aprakstītu katru attēlā attēloto sadaļas šūnu nepieciešams izveidot 15 mainīgos (sadaļā ir 5 rindas un 3 kolonas). Rezultātā nepieciešams izveidot 3 rādītājus (katra kolona savs rādītājs) un katram rādītājam tiek definēts atribūts – piesaistīts lokālā tipa klasifikators „Augkopības produkti". Katram rādītājam tiek izveidoti 5 mainīgie, nu kuriem viens mainīgais tiek veidots no kombinācijas rādītājs bez atribūta. Pārējie 4 mainīgie tiek izveidoti kā kombinācija no rādītāja ar atribūtu (kuram piesaistīts lokālais klasifikators – augkopības produkti). No šādas rādītāja-atribūta kombinācijas katram klasifikatora ierakstam tiek izveidots savs mainīgais.

Nākamais attēls attēlo principus, kā tiek aprakstīta pārskata sadaļa ar mainīgu kolonu skaitu.



3.6. att. Šūnu definēšana sadaļā ar mainīgu kolonu skaitu

Šādas sadaļas aprakstīšanai metadatos tiek definētas 3 rindas un 2 kolonas. Pirmā kolona nepieciešama, lai aprakstītu aili kopā un otrā kolona apraksta visas mainīgās kolonas, kā vienu kopīgu. Lai definētu visas sadaļas šūnas, nepieciešami 6 mainīgie (3 rindas x 2 kolonas). Rezultātā atbilstoši sadaļas saturam nepieciešams izveidot 3 rādītājus (katrai sadaļas rindai). Katram rādītājam nepieciešams izveidot 2 mainīgos. Viens no mainīgajiem tiek veidots izmantojot kombināciju – rādītājs bez atribūta. Šo mainīgo piesaista 1 kolonas šūnām. Bet otrs mainīgais tiek izveidots kombinācijai rādītājs plus atribūts (kuram piesaistīts starptautiskais, nacionālais klasifikators – Nozares NACE). Šo mainīgo piesaista otrās kolonas šūnām. Kā bija minēts iepriekš – veidojot mainīgos, izmantojot starptautiskos klasifikatorus, tiek izveidots tikai viens mainīgais, nevis vairāki (katram klasifikatora ierakstam savs).

Sadaļā [4.] aprakstīts piemērs, kā pārskats tiek sadalīts atsevišķos objektos un tiek aprakstīts metadatu bāzē.

3.2. Metadatu modulis – datu validācija

Sadaļā tiek apskatīti principi un metodes, kā metadatu bāzē tiek aprakstīti pārskatu veidlapu validāciju nosacījumi.

3.2.1. Validāciju pseidovaloda

Iepriekšējā sadaļā tika aprakstīts, kā pārskata saturs un vizuālais izskats tiek aprakstīts metadatu bāzē. Šī informācija ir pietiekama, lai varētu ģenerēt datu ievades formas. Nākamais datu apstrādes etaps pēc datu ievades ir datu kontroles, jeb validācijas. Datu validācija tas ir vairāku nosacījumu komplekts, kuru mērķis ir pārbaudīt vai respondenta iesniegtie un ievadītie pārskata dati ir gan aritmētiski, gan loģiski pareizi. Ļoti daudzas aritmētiskās sakarības jau tiek iekļautas pašos apsekojumu aprakstos.

1. DATI PAR DARBA ŅĒMĒJIEM, PAR KURIEM JĀVEIC DARBA LAIKA UZSKAITE

	Mēr-vienība	Rindas kods	Pavisam *(2.+3.aile)*	tai skaitā strādāja	
				normālo darba laiku	nepilnu darba laiku
A	B	C	1	2	3
1.1. DARBA ŅĒMĒJU SKAITS DARBA ATTIECĪBĀS (IESKAITOT NEREZIDENTUS)					
Darba ņēmēju skaits ceturkšņa pirmajā kalendārajā darba dienā	cilvēki	1110			
Darba ņēmēju skaits ceturkšņa pēdējā kalendārajā darba dienā *(1140.rinda = 1141.rinda + 1142.rinda)*	cilvēki	1140			
tai skaitā: darba ņēmēju skaits pamatdarbā (ar algas nodokļa grāmatiņām)	cilvēki	1141			
darba ņēmēju skaits blakus darbā (bez algas nodokļa grāmatiņām)	cilvēki	1142			

3.7. att. Pārskata fragments ar iekļautiem validāciju nosacījumiem

Attēlā redzams, ka 1140 rindā ievadītajiem datiem jāsakrīt ar 1141 un 1142 rindās ievadītajiem datiem kopā. Tāpat jebkuras rindas pirmajā ailē ievadītajiem datiem ir jābūt kā 2. un 3. ailes summai. Tā kā rindas kodi ir neatņemama jebkura pārskata sastāvdaļa, tad arī visi validāciju nosacījumi tiek pierakstīti un statistiķiem labi saprotami tieši rindu, kolonu pieraksta veidā. Tāpēc izstrādājot sistēmas funkcionalitāti pārskatu datu validācijai, tika ņemts vērā tieši šis rindu, kolonu pieraksta veids. Kopumā analizējot validāciju nosacījumus, tiem tika identificētas sekojošas īpašības:

- datu validācijas process tikai pārbauda datus un nekad neveic nekādas korekcijas tajos;
- validāciju likumos tiek pārbaudītas loģiskas sakarības dažādu šūnu starpā;
- katra validācijas sakarība apraksta pareizus datus. Attiecīgi, ja sakarība neizpildās, tiek identificēta validācijas kļūda;
- katrs validācijas nosacījums sastāv no divām daļām:
    - ja daļa – loģisks priekšnosacījums, kurš nosaka, kad ir jāpārbauda konkrēta sakarība. Nav obligāts visiem likumiem;
    - tad daļa – pati loģiskā sakarība kas tiek pārbaudīta;
- validāciju likumos tiek izmantotas ne tikai pārskata datu šūnas, bet arī respondenta atribūti, tādi kā ATVK kods, darbības nozares NACE kods, utt.

Rezultātā, lai lietotājiem būtu ērtāk aprakstīt visus validāciju nosacījumus, dokumenta autors izveidoja speciālu pseidovalodu, ar kuru lietotāji var ērti un saprotami aprakstīt visus validāciju nosacījumus. Pseidovalodas galvenais elements – šūna, kas identificē pārskata konkrētas šūnas vērtību. Šajā pseidovalodā šūnai tiek izmantots sekojošs pieraksta veids:

[R, K], kur

R – rindas kods, un attiecīgi K – kolonas kods.

Atbilstoši šim pieraksta veidam augstāk attēlā minētā pārskata fragmenta validāciju likums 1140 rindai izskatīsies sekojoši:

JA daļa – nav, jo likumam nav priekšnosacījumu, kad tas ir jāpārbauda

TAD daļa – **[11401,1] = [1141,1] + [1142,1]**

Likumos drīkst izmantot gan aritmētiskas operācijas, tādas kā +, -, *, / gan arī salīdzināšanas operācijas, kā =, <, >, <=, >=, <>. Tāpat likumos ir atļauts izmantot gan iekavas, gan arī loģiskos operatorus AND un OR.

Tā kā validācijās atsevišķās situācijās ir nepieciešams izmantot arī respondenta atribūtus, tad šiem atribūtiem tika izveidots speciāls pieraksta veids:

$respondenta_atribūts

Respondenta atribūti – tie ir fiksēti respondentu aprakstoši lauki, kuri ir definēti visiem apsekojumiem. Standarta atribūti ir piemēram, ATVK kods, darbības nozares kods NACE, darbinieku skaits, apgrozījums, uzņēmējdarbības formas kods SVTK, utt. Arī šos respondenta atribūtus ir atļauts izmantot validāciju pseidovalodā. Piemēram, ja augstāk minētais piemērs par validāciju 1140 rindai būtu jāpārbauda tikai Rīgas uzņēmumiem, tad likums veidotos šādi:

JA daļa - **$ATVK = 01** – *Pārbaudam, vai respondentam ATVK kods sākas ar 01 (Rīgas uzņēmumi)*

TAD daļa - **[11401,1] = [1141,1] + [1142,1]**

Ja respondentam ATVK kods sākas ar „01”, tad tiek pārbaudīta arī likuma TAD daļa. Ja šī daļa neizpildās, tad respondentam tiek reģistrēta konkrētā validācijas likuma kļūda.

Šie divi standarta elementi ļauj aprakstīt vairāk kā 50% no katra pārskata validāciju nosacījumiem. Lielākoties tās visas ir aritmētiskās pārbaudes, kas pārbauda vai visi ievadītie

dati atbilst pamata aritmētiskiem nosacījumiem, kad pārskatā ir nepieciešams ievadīt kādu kopējo summu un tās sadalījumu pa atsevišķām komponentēm. Nākamā pārbaužu grupa, kas saistīta ar validācijām ir salīdzināšana ar vēsturiskiem periodiem. Katram pārskatam tiek definēts periodiskums. Periodiskums norāda, cik reizes un ar kādu intervālu respondentam ir jāiesniedz konkrēts statistikas pārskats. Piemēram, ja pārskatam ir periodiskums „M" – mēneša, tas nozīmē, ka respondentiem konkrētais pārskats ir jāsniedz par katru mēnesi visa gada garumā. Datu validācijas procedūra tiek veidota tā, ka tā spēj validēt jebkura perioda pārskatu datus. Statistiķi darbojoties ar pārskatiem, vienmēr darbojas ar vienu konkrētu pārskata periodu, kas dotajā brīdī ir aktīvais. Atsevišķa validāciju grupa sastāv no pārbaudēm, kas salīdzina pašreizējā (apstrādājamā) perioda datus ar iepriekšējā periodā respondenta iesniegtajiem pārskatu datiem. Šādas pārbaudes vairs netiek iekļautas pārskatu aprakstos un šīs pārbaudes jau veic statistiķi dziļāk analizējot un pārbaudot datus. Mērķis šādām validācijām ir kontrolēt, vai respondents nav ievadījis vērtības, kurām ir neadekvāts palielinājums vai samazinājums salīdzinot ar iepriekšējo periodu. Lai validāciju pseidovalodā varētu vērsties pie cita perioda datiem, standarta šūnas pierakstam tika izmantots trešais parametrs un pseidovalodā vēršanās pie šūnas vērtības ir sekojoša:

[R,K,P], kur

R – rindas kods

K – kolonas kods, kas kopā ar rindas kodu identificē konkrētu pārskata šūnu

P – pārskata periodu skaits, cik seni vēsturiskie dati par konkrēto respondentu ir jāatlasa un jāpārbauda salīdzinot ar apstrādājamo periodu. Piemēram, ja statistiķis dotajā brīdī apstrādā pārskatu, kuram ir ceturkšņa periodiskums un apstrādā kāda konkrēta respondenta 3.ceturkšņa pārskata datus. Salīdzinot pašreizējā perioda datus ar viena perioda iepriekš datiem – tas nozīmē salīdzināt respondenta 3.ceturkšņa datus ar otrajā ceturksnī ievadītajiem pārskata datiem. Periodu skaitam ir iesējami vairāki pieraksta veidi, kas tika identificēti analizējot dažādus iespējamos pārskatu validāciju nosacījumus:

- skaitlis (1,2,3), ja nepieciešams norādīt periodu skaitu atpakaļ, salīdzinot ar pārbaudāmo pārskata periodu
- skaitlis + burts norāda konkrētu periodu skaitu atpakaļ, kur burts identificē konkrētu periodiskumu M (mēnesis), C (ceturksnis), P (pusgads), G (gads). Attiecīgi norādot 2C, tiek meklēti dati 2 ceturkšņus atpakaļ pret tekošo pārskata periodu

- burts + skaitlis, izmanto ja nepieciešams norādīt konkrētu periodu, piemēram, C2 – tiek izmantoti tieši 2.ceturkšņa dati.
- Gads + burts + skaitlis – izmanto ja nepieciešams norādīt pilnīgi konkrēta perioda datus. Piemēram, 2014M08 – nepieciešams apskatīt celles datus par 2014.gada augustu.

Piemēram likums:

Ja daļa – **[12150,1]>[12150,1,1]** – *pārbauda vai 12150 rindā ievadītā vērtība ir lielāka par iepriekšējā periodā ievadīto vērtību*

TAD daļa – **[12150,1]<[12150,1,1]+[12150,1,1]*0.5** –*pārbauda vai šajā periodā rindā 12150 ievadītās vērtības pieaugums salīdzinot ar iepriekšējo periodu nepārsniedz 150%*

Nākamās validācijas pārbaudes, kuras arī veic statistiķi pēc visu pārskatu datu ievades ir starpformu pārbaudes. Šīs pārbaudes salīdzina datus starp vairākiem pārskatiem. Bieži vien šīs pārbaudes tiek izmantotas vienāda temata pārskatos, kad vienas un tās pašas vērtības tiek ievadītas dažādos pārskatos un dažādos sadalījumos. Tad šīs starpformu validācijas kontrolē šīs vienādās vai līdzīgās vērtības starp pārskatiem. Lai validāciju likumos varētu vērsties pie cita pārskata datiem, tiek izmantots sekojoši pseidoelementi:

„pārskata indekss"[R,K,P], kur

„pārskata indekss" – cita pārskata indekss, kas tiek definēts katram pārskatam. Pēc indeksa viennozīmīgi tiek identificēts pārskats neatkarīgi no tā izmaiņām gadu laikā. Indekss paliek vienmēr nemainīgs.

R – rindas kods, K – kolonas kods, kas identificē šūnu citā pārskatā

P – periodu skaits, ja no cita pārskata nepieciešams izmantot kādu konkrēta perioda, vai relatīva perioda vēsturisko vērtību.

Piemēram likums:

TAD daļa – **[3240,1]="2-investīcijas"[1821,1]+"2-investīcijas"[1821,2]**

pārbauda, vai konkrētā pārbaudāmā pārskata vērtība šūnā [3240,1] sakrīt ar cita pārskata „2-investīcijas" divu šūnu summu par to pašu periodu.

Kombinējot visus šos pseidovalodas elementus kopā, iespējams definēt ļoti sarežģītus validāciju likumus, kas ir jāpārbauda. Piemēram validāciju likuma TAD daļa var izskatīties sekojoši (skatīt attēlu zemāk):

```
(([410,1] + [420,1] + [430,1] + [440,1] + [450,1]) / ("1,2-
ceturkšņa_2014"[1470,1,C1] + "1,2-ceturkšņa_2014"[2110,1,C1] + "1,2-
ceturkšņa_2014"[2120,1,C1] + "1,2-ceturkšņa_2014"[2130,1,C1] + "1,2-
ceturkšņa_2014"[310,1,C1] + "1,2-ceturkšņa_2014"[320,1,C1] + "1,2-
ceturkšņa_2014"[330,1,C1] + "1,2-ceturkšņa_2014"[340,1,C1] + "1,2-
ceturkšņa_2014"[350,1,C1] + "1,2-ceturkšņa_2014"[360,1,C1] + "1,2-
ceturkšņa_2014"[1470,1,C2] + "1,2-ceturkšņa_2014"[2110,1,C2] + "1,2-
ceturkšņa_2014"[2120,1,C2] + "1,2-ceturkšņa_2014"[2130,1,C2] + "1,2-
ceturkšņa_2014"[310,1,C2] + "1,2-ceturkšņa_2014"[320,1,C2] + "1,2-
ceturkšņa_2014"[330,1,C2] + "1,2-ceturkšņa_2014"[340,1,C2] + "1,2-
ceturkšņa_2014"[350,1,C2] + "1,2-ceturkšņa_2014"[360,1,C2] + "1,2-
ceturkšņa_2014"[1470,1,1] + "1,2-ceturkšņa_2014"[2110,1,1] + "1,2-
ceturkšņa_2014"[2120,1,1] + "1,2-ceturkšņa_2014"[2130,1,1] + "1,2-
ceturkšņa_2014"[310,1,1] + "1,2-ceturkšņa_2014"[320,1,1] + "1,2-
ceturkšņa_2014"[330,1,1] + "1,2-ceturkšņa_2014"[340,1,1] + "1,2-
ceturkšņa_2014"[350,1,1] + "1,2-ceturkšņa_2014"[360,1,1] + "1,2-
ceturkšņa_2014"[1470,1] + "1,2-ceturkšņa_2014"[2110,1] + "1,2-
ceturkšņa_2014"[2120,1] + "1,2-ceturkšņa_2014"[2130,1] + "1,2-
ceturkšņa_2014"[310,1] + "1,2-ceturkšņa_2014"[320,1] + "1,2-ceturkšņa_2014"[330,1]
+ "1,2-ceturkšņa_2014"[340,1] + "1,2-ceturkšņa_2014"[350,1] + "1,2-
ceturkšņa_2014"[360,1])) * 100 <= 1
```

3.8. att. Piemērs starpformu validācijas nosacījumam

Attēlā attēlotais validācijas likums ir definēts ceturkšņa periodiskuma pārskatam, kuram šūnā [410,1], [420,1], [430,1], [440,1], [450,1] tiek ievadīta informācija par visu gadu. Šīs šūnas tiek aizpildītas tikai pārskata 4.ceturksnī. Tāpēc šis validācijas likums tiek pārbaudīts tikai 4.ceturkšņa datiem. Šis validācijas likums pārbauda, vai konkrētajā pārskatā norādītās uzņēmuma darba devēja izmaksas (par visu gadu), kas nepieciešamas ražošanas procesam nepārsniedz kopējās uzņēmuma darbaspēka izmaksas, par kuru informācija tiek iegūta citā ceturkšņa periodiskuma pārskatā, kurā dati vadīti katru ceturksni.

3.2.2. Datu validācijas process

Datu validāciju likumi katram pārskatam tiek aprakstīti saraksta veidā, kur katrs ieraksts atbilst vienam validācijas nosacījumam. Validāciju nosacījumus var aprakstīt tikai tad, kad

pārskatam ir aprakstīts izskats, definētas rindas, kolonas un šūnas. Tas ir nepieciešams lai pārliecinātos, ka katrā validācijas likumā iekļautās šūnas, kas tiek aprakstītas ar pseidovalodas elementiem [R,K] notācijā tiešām ir definētas konkrētajā apsekojumā. Kad likumi ir aprakstīti, no visiem likumiem tiek ģenerēta viena konkrēta validācijas procedūra ar sekojošām īpašībām:

- Validācijas procedūra tiek ģenerēta konkrētam pārskatam, konkrētam gadam un tā darbojas ar šī konkrētā gada visu periodu datiem
- Validācijas procedūra ir izveidota TSQL valodā un pilnībā izpildās SQL serverī
- Validācijas procedūra prot apstrādāt viena respondenta, vai padotas respondentu kopas datus
- Validācijas izpildes laikā visas konstatētās kļūdas katram respondentam tiek reģistrētas validāciju žurnālā

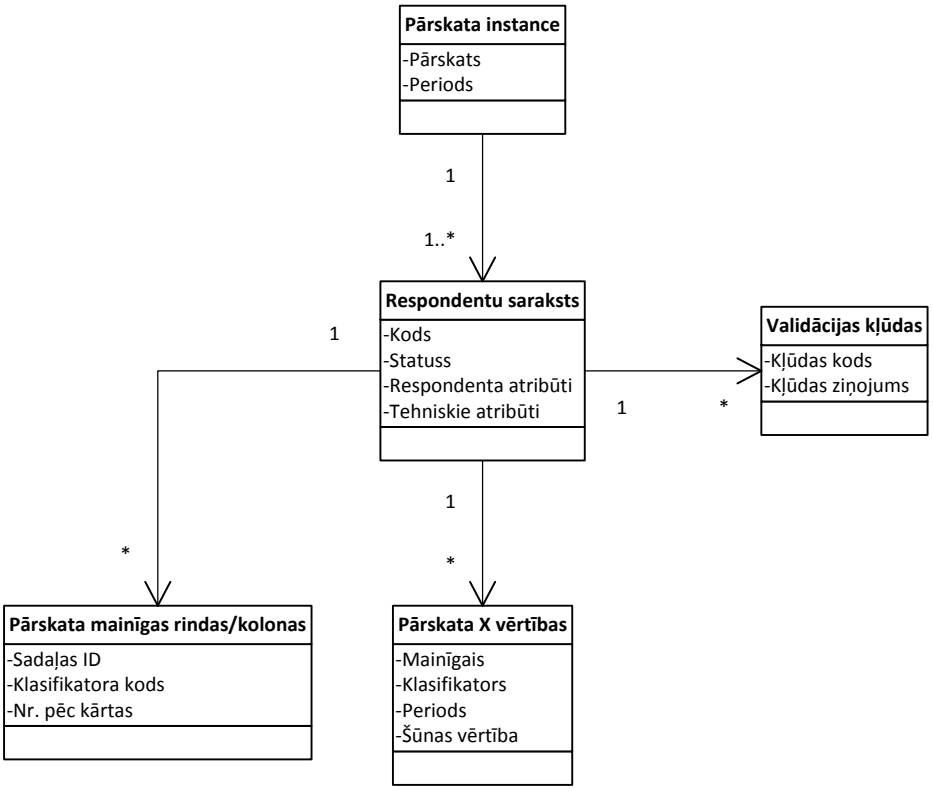
Pati validācijas procedūra tiek veidota pēc sekojoša principa:

- Pārskata visi validācijas likumi tiek sadalīti sīkākos šūnu elementos
- Katram šūnas elementam tiek izveidots savs SQL mainīgais.
- Visos mainīgajos tiek ielasīti konkrētā respondenta pārskata dati. Datu atlase tiek veikta pēc mainīgo identifikatoriem. Procedūras veidošanas laikā visas šūnas, kuras lietotājs pieraksta ar rindas/kolonas notāciju tiek pārveidotas uz mainīgo identifikatoriem.
- Kad mainīgajos dati ir ielasīti, tiek pārbaudīti visi validācijas nosacījumi, kur pseidovalodā katrs šūnas elements tiek aizstāts ar ģenerētu SQL mainīgo

Laika gaitā sistēmai attīstoties tiek pievienoti arvien jauni validāciju elementi. Tā piemēram, validācijas pseidovalodā ir ieviesta funkcija ABS(), kas atgriež izteiksmes absolūto vērtību, kā arī elements „_0”, lai pārbaudītu konkrētas šūnas vērību, vai tā nav NULL (nav ievadīta). Tāpat ir izveidoti pseidovalodas elementi, kas speciāli paredzēti darbībām ar pārskatu sadaļām, kurās ir mainīgs rindu vai kolonu skaits. Pielikumā [3.pielikums] attēlota viena validācijas procedūra.

3.3. Mikrodatu bāze

Mikrodatu bāzē tiek glabātas respondentu iesniegto un ievadīto pārskatu datu vērtības. Sarežģītākā lieta sistēmas arhitektūrā bija izveidot struktūru, kā glabāt pašu pārskatu vērtības tā, lai tās nebūtu saistītas ar konkrēta pārskata izskatu, bet tajā pašā laikā būtu iespējams vērtības atlasīt un attēlot matricas veidā pareizās koordinātēs. Rezultātā mikrodatu bāzei pārskatu datu glabāšanai tika izveidota arhitektūra, kas attēlota zemāk attēlā:



3.9. att. Mikrodatu bāzes arhitektūra pārskatu datu glabāšanai

Tā kā Mikrodatu bāzē tiek glabāts liels daudzums datu – visu pārskatus visu periodu dati, tad lai uzlabotu pārskata datu ievades, apstrādes, analīzes veiktspēju, katra pārskata vērtības tiek glabātas atsevišķā fiziskā datubāzes tabulā. Dotajā kontekstā ar pārskatu nav domāta viena konkrēta respondenta aizpildīta pārskata veidlapa. Ar pārskatu tiek domāts konkrēts statistikas pārskats (piemēram, pārskats 1-rūpniecība ar ceturkšņa periodiskumu) vairāku gadu garumā, pat ja katru gadu pārskatam tiek metadatos aprakstītas atsevišķas pārskata versijas. Tas nozīmē, ka visu respondentu iesniegtās konkrētā pārskata aizpildītās

veidlapas par jebkādu laika periodu (gan šā gada, gan 10 gadus vecas šī paša pārskata) tiek ievadītas sistēmā un fiziski glabājas vienā tabulā. Turpretī cita pārskata visas respondentu iesniegto aizpildīto un ievadīto pārskatu veidlapu dati glabājas fiziski citā tabulā. Visas pārskatu vērtību tabulas ir identiskas pēc struktūras. Tas uzlabo datu piekļuves un apstrādes ātrumu, bet negatīvā puse ir tā, ka visus pārskata datu pieprasījumus nepieciešams sistēmā ģenerēt, jo pārskatu tabulu nosaukumi katram pārskatam ir citi. Zemāk katra mikrodatu bāzes klase aprakstīta sīkāk.

3.3.1. Pārskata instances

Šī objektu klase identificē katru pārskata apstrādes periodu. Katram periodam ir pakārtots atbilstošs respondentu saraksts un tālāk šo respondentu sniegtās pārskatu vērtības. Instances tiek veidotas brīdī, kad katrs pārskats pirmo reizi tiek sagatavots datu ievadei. Šajā brīdī tiek izveidotas konkrētā pārskata instances (ja nav) visiem pārskata norādītā gada periodiem. Piemēram, ja tiek sagatavots apstrādei pārskats ar mēneša periodiskumu, tad konkrētā gadā tiek izveidotas 12 instances – katram mēnesim sava instance.

3.3.2. Respondentu saraksts

Šī klase paredzēta respondentu sarakstu uzkrāšanai. Katrai pārskata instancei tiek pilnībā glabāts pilns respondentu saraksts. Tie ir respondenti, kuri izvēlēti kā konkrēta pārskata, katra perioda datu sniedzēji. Piemēram, ja kādam konkrētam uzņēmuma ir jāsniedz mēneša periodiskuma pārskata dati visa gada garumā, tad šis respondents tiks pievienots respondentu sarakstā 12 reizes – atbilstoši pie katras instances. Katru respondentu viennozīmīgi identificē kods, kurš katram uzņēmuma tiek piešķirts Statistikas uzņēmumu reģistrā. Šis kods viennozīmīgi identificē respondentu, kas ir ļoti svarīgi lai analizētu respondentu datus gan dažādu pārskatu, gan dažādu periodu griezumos – tieši kods apvieno šo respondentu vairāku pārskatu respondentu sarakstos. Atribūts statuss norāda ievadītā pārskata statusu, kas dod informāciju, vai pārskats ir ievadīts, vai nav. Kā arī vai pārskata dati ir validēti vai nē. Respondenta atribūti – tas ir vairāku lauku kopums kuru vērtības katrā pārskata apstrādes brīdī tiek aktualizētas no Statistikas uzņēmumu reģistra. Šīs vērtības var tikt izmantotas gan datu validācijai, bet visbiežāk šīs vērtības tiek izmantotas pārskatu datu agregāciju procesos, kad katra pārskata perioda dati ir ievadīti un validēti. Standarta respondenta atribūtu vērtības ir ATVK kods, uzņēmējdarbības formas kods, uzņēmuma

darbības nozares kods, apgrozījums, strādājošo skaits. Papildus katru respondentu respondentu sarakstā raksturo dažādi tehniskie atribūti, kā pārskata ievades laiks un lietotājs un līdzīgi datu apstrādes procesu atribūti

3.3.3. Pārskata „X" vērtības

Šī klase nodrošina pārskatu vērtību glabāšanu. Kā tika minēts iepriekš – katram statistikas pārskatam tiek veidota sava pārskata vērtību tabula, ar mērķi samazināt datu apstrādes slodzi, kad lietotāji vienlaicīgi apstrādā vairāku pārskatu datus. Datu struktūrā tika izmantots Bo Sungrem ieteiktais statistisko datu apraksta mehānisms. Kā redzams katrs vērtības objekts tā ir viena pārskata vērtība, kura nekādi tieši nav saistīta ar pārskata konkrētu šūnu. Objektā katru vērtību identificē šādu identifikatoru kopums:

- Norāde uz respondentu, kas dod informāciju par to, kuram respondentam, kāda pārskata un perioda datu vienība ir saglabāta
- Norāde uz mainīgo – identificē precīzi kāda statistiskā vērtība tieši šī konkrētā ir. (Pēc metadatu struktūras pēc mainīgā un pārskata ir iespējams atrast kādā koordinētā un kurā pārskata sadaļā šī vērtība atrodas pārskatā)
- Periods – raksturo precīzi šīs vērtības periodu
- Vērtība – pati vērtība kas tika ievadīta un saglabāta datu bāzē.
- Klasifikators – papildus vērtība, kas tiek izmantota un aizpildīta situācijās, kad tiek ievadīti dati no sadaļām ar mainīgu rindu un kolonu skaitu. Kā bija minēts šo mainīgu rindu/kolonu situācijā – respondents pats norāda klasifikatora vērtību par kuru sniedz detalizētus datus.

Tieši šāda pirmdatu glabāšanas struktūra nodrošina to, ka katra pārskata vērtība tiek glabāta neatkarīgi no tās atrašanās vietas matricā. Bet tai pašā laikā metadatu bāzē ir pietiekama informācija lai katram konkrētam mainīgajam identificētu tā vizuālo atrašanās vietu sadaļā. Bet tai pat laikā šī struktūra nodrošina ērtu datu atlasīšanu konkrētam mainīgajam par iepriekšējiem datu periodiem (veidojot laika rindas), kā arī ir pietiekami vienkāršs mehānisms, kas nodrošina lietotājiem iespēju atlasīt datus no vairākiem pārskatiem un apvienot tos vienā datu masīvā tālākai datu analīzei.

3.3.4. Pārskata mainīgās rindas/kolonas

Šī ir specifiska klase, kas tiek izmantota tikai un vienīgi saglabājot datus par pārskata sadaļām ar mainīgām rindām/kolonām. Katrs šīs klases objekts identificē vienu konkrētu pārskata mainīgo rindu vai kolonu. Katrs objekts satur datus par respondentu, kura dati tiek vadīti, pārskata sadaļu – uz kuru attiecas mainīgā rinda vai kolona, kā arī norāde uz klasifikatora kodu, kas atbilst pašai mainīgajai rindai. Numurs pēc kārtas ir nepieciešams, lai attēlotu datus sadaļās ar mainīgām rindām/kolonām tieši tādā kārtībā – kā tās respondents tika iesniedzis.

3.3.5. Validācijas kļūdas

Šī klase nodrošina konstatēto validācijas kļūdu uzkrāšanu katram respondentam, veicot pārskata datu validāciju. Katru validācijas kļūdu raksturo sekojoši atribūti – respondents, kuram kļūda konstatēta, kļūdas kods – norāde uz konkrētu validāciju kļūdas aprakstu metadatu bāzē – kļūdas ziņojums, skaidrojoša informācija par konkrētu validācijas kļūdu.

Šī universālā datu struktūra, kas ir identiska visiem pārskatiem, nodrošina arī to, ka sistēmā ir izveidotas pirmdatu importa/eksporta funkcijas, kas nodrošina jebkura pārskata datu importu un eksportu vienādā universālā datu struktūrā, kas nekādi nav saistīts ar konkrētu pārskatu. Šī pieeja arī palīdz CSP programmētājiem veikt datu analīzi un datu apstrādi ārpus sistēmas, veicot datu apstrādes funkcijas, kas nav realizējamas ar sistēmas līdzekļiem.

4. PIEMĒRS

Šajā sadaļā tiek apskatīta viena reāls Centrālās Statistikas Pārvaldes pārskata veidlapa. Sadaļā aprakstīts, kā šī veidlapa tiek aprakstīta metadatos, kā arī tiek attēlots kā mikrodatu bāzē tiek saglabāti vienas aizpildītas veidlapas dati.

Pārskata veidlapa ar aizpildītiem datiem attēlota [2.pielikums]. Piemēra dati veidlapas šūnās attēloti treknrakstā (bold).

Aprakstot pārskata veidlapu metadatu bāzē, katram objektam tiks pievienota papildus identifikācijas kolona - ID (skaitlis), lai saprotamāk attēlotu metadatu objektu savstarpējās relācijas.

Aprakstot objektus, slīprakstā (*italic*) pie atsevišķu objektu nosaukumiem tiek pievienots komentārs, kas detalizē konkrētā objekta specifiskas īpašības.

4.1. Pārskata veidlapas apraksts Metadatu bāzē

Konkrētā pārskata veidlapa satur trīs sadaļas, no kurām pirmā sadaļa ir ar mainīgu kolonu skaitu, sākot no otrās kolonas. Ar otro kolonu tiek saprasta kolona, kurās tiek vadītas pārskata vērtības. Fiziski pirmās divas kolonas jebkurā pārskata veidlapas sadaļā ir rindas kods un rindas nosaukums. Šīs pirmās divas kolonas netiek ņemtas vērā aprakstot pārskata veidlapas struktūru metadatu bāzē. Bet otrā un trešā sadaļa – tās ir vienkāršas fiksēta izmēra sadaļas.

4.1.1. Pārskats

4.1. tabula. Pārskata vispārīgs apraksts metadatu bāzē

Pārskata ID	*Indekss*	*Nosaukums*
1	2-Gada	Kompleksais pārskats par iestādes (organizācijas) darbību

4.1.2. Pārskata versija

4.2. tabula. Pārskata versijas apraksts metadatu bāzē

Versijas ID	*Pārskata ID (norāde uz pārskatu)*	*Nosaukums*	*Periodiskums*
1	1	Kompleksais pārskats par iestādes (organizācijas) darbību	G (*gads*)

4.1.3. Versijas gadi

4.3. tabula. Pārskata versijas gadu apraksts metadatu bāzē

Versijas ID (norāde uz pārskata versiju)	*Gads*
1	2013

4.1.4. Pārskata sadaļas

4.4. tabula. Pārskata sadaļu apraksts metadatu bāzē

Sadaļas ID	*Versijas ID (norāde uz pārskata versiju)*	*Kods*	*Nosaukums*	*Nr. pēc kārtas*	*Sadaļas tips*
1	1	1.	Atsevišķi iestādes darbības rādītāji	1	Mainīgas kolonas
2	1	2.	Izejvielu, materiālu, mazvērtīgā intervāla iegāde un krājumi	2	Konstanta sadaļa
3	1	Laiks	Pārskata aizpildīšanai patērētais laiks	3	Konstanta sadaļa

4.1.5. Rindas

4.5. tabula. Sadaļas rindu apraksts metadatu bāzē

Rindas ID	*Sadaļas ID (norāde uz pārskata sadaļām)*	*Kods*	*Nosaukums*	*Nr. pēc kārtas*
1	1	1110	Vidējais darbinieku skaits darba attiecībās pārskata gadā, par kuriem nodokļus maksā darba devējs	1
2	1	1111	no 1110.rindas - pamatdarbā	2
3	1	1130	Aprēķinātā bruto darba samaksa visiem darbiniekiem, kuri bija darba attiecībās pārskata gadā un par kuriem nodokļus maksāja darba devējs	3

4	1	1210	Ieņēmumi no sniegtajiem maksas pakalpojumiem pārskata gadā	4
5	1	1310	Kārtējie izdevumi (pavisam)	5
6	1	1311	no 1310.rindas - pakalpojumu apmaksa	6
7	1	1312	no 1311. rindas – maksa par apkuri, ūdeni, elektroenerģiju un gāzi	7
8	2	600	Izejvielas, materiāli, mazvērtīgais inventārs un citas preces iestādes vajadzībām – kopā	1
9	3	Laiks	Pārskata 2-gada aizpildīšanai patērētais laiks	1

4.1.6. Kolonas

Pārskatā pirmā sadaļa ir sadaļa ar mainīgu kolonu skaitu. Mainīgās kolonas sākas ar otro kolonu, kur katrs respondents norāda tik nozares, cik konkrētajā organizācijā ir un par katru nozari sniedz atbilstošos datus. Metadatos mainīgās kolonas apraksta kā vienu kolonu. Pārskata datu ievades funkcija prot interpretēt šīs kolonas un dod iespēju datu ievades laikā pievienot šīs mainīgās kolonas tik, cik ir nepieciešams katra konkrētā respondenta iesniegtā pārskata datu ievadei.

4.6. tabula. Sadaļas kolonu apraksts metadatu bāzē

Kolonas ID	*Sadaļas ID (norāde uz pārskata sadaļām)*	*Kods*	*Nosaukums*	*Nr. pēc kārtas*
1	1	1	Pārskata gadā (kopā iestādē)	1
2	1	2	Sadalījumā pa nozarēm *(Šī ir mainīgā kolona, kas metadatos tiek aprakstīta kā viena kopīga. Nekāda papildus pazīme šai kolonai netiek piešķirta, jo sadaļai jau ir piešķirts tips – mainīgas kolonas. Attiecīgi datu ievades forma analizē sadaļas pēdējo kolonu un to veido kā speciālu kolonu, kuru var pievienot tik reizes, cik nepieciešamas konkrētas aizpildītas pārskata veidlapas datu ievadei)*	2
3	2	1	Krājumi pārskata gada sākumā	1
4	2	2	Iegādāts pārskata gadā	2
5	2	3	Izlietots pārskata gadā	3
6	2	4	Krājumi pārskata gada beigās	4
7	3	1	stundas	1
8	3	2	minūtes	2

4.1.7. Klasifikatori

Saistībā ar konkrētā pārskata veidlapu, tiek izmantoti tikai divi klasifikatori, kuri tad arī tiek aprakstīti. Atsevišķos pārskatos var būt izmantoti pat 5 un vairāk dažādi klasifikatori.

4.7. tabula. Klasifikatoru apraksts metadatu bāzē

Klasifikatora ID	*Nosaukums*	*Kods (tiek izmantots validācijās)*	*Tips*
1	NACE 2.red. klasifikators	NACE7	starptautiskais/nacionālais
2	Gada sākums, beigas		lokālais

4.1.8. Klasifikatora ieraksti

Tā kā NACE 2.red ir starptautiskais klasifikators, tas satur ļoti daudz ierakstus. Piemēra pēc, tabulā tiks pārskaitīti tikai daži ieraksti (tai skaitā ieraksti, kuri ir iekļauti aizpildītajā veidlapā)

4.8. tabula. Klasifikatora ierakstu struktūra metadatu bāzē

Ieraksta ID	*Klasifikatora ID (norāde uz klasifikatoru)*	*Kods*	*Nosaukums*	*Mērvienība (šim klasifikatoram mērvienības netiek izmantotas)*
1	1	0111	Graudaugu (izņemot rīsu), pākšaugu un eļļas augu sēklu audzēšana	
2	1	0220	Mežizstrāde	
3	1	4110	Būvniecības projektu izstrādāšana	
4	1	7220	Pētījumu un eksperimentālo izstrāžu veikšana sociālajās un humanitārajās zinātnēs	
5	1	8411	Vispārējo valsts dienestu darbība	
101	2	1	Pārskata gada sākumā	
102	2	2	Pārskata gada beigās	

4.1.9. Rādītāji

Konkrētajā piemērā, aprakstot statistiskos rādītājus, atribūts „kods” nevienam rādītājam netiek piešķirts, tāpēc zemāk tabulā šo atribūtu neattēloju.

4.9. tabula. Rādītāju apraksts metadatu bāzē

Rādītāja ID	*Tips*	*Mērvienība*	*Periodiskums*	*Nosaukums*
1	Skaitlis	cilvēki	G	Vidējais darbinieku skaits, par kuriem nodokļus maksā darba devējs
2	Skaitlis	cilvēki	G	Vidējais darbinieku skaits, par kuriem nodokļus maksā darba devējs, kuriem tas ir pamatdarbs
3	Skaitlis	Ls	G	Bruto darba samaksa pilnu darba laiku un nepilnu darba laiku strādājošiem darbiniekiem
4	Skaitlis	Ls	G	Budžeta iestādes ieņēmumi - maksas pakalpojumi un citi pašu ieņēmumi
5	Skaitlis	Ls	G	Budžeta iestādes un organizācijas kārtējie izdevumi
6	Skaitlis	Ls	G	Budžeta iestādes izmantotie pirktie pakalpojumi
7	Skaitlis	Ls	G	Budžeta iestādes izmantotie pirktie pakalpojumi - maksa par apkuri, ūdeni, elektroenerģiju un gāzi
8	Skaitlis	Ls	G	Izejvielu, materiālu, preču krājumi
9	Skaitlis	Ls	G	Izejvielas, materiāli, preces, iegādātas (saņemtas) pārskata gada laikā
10	Skaitlis	Ls	G	Izejvielas, materiāli, mazvērtīgais inventārs u.c. - izlietots pārskata gada laikā
11	Skaitlis	stundas	G	Pārskata aizpildīšanai nepieciešamais laiks (stundas)
12	Skaitlis	minūtes	G	Pārskata aizpildīšanai nepieciešamais laiks (minūtes)

4.1.10. Atribūti

4.10. tabula. Atribūtu apraksts metadatu bāzē

Atribūta ID	*Rādītāja ID (norāde uz rādītāju)*	*Klasifikatora ID (norāde uz klasifikatoru)*	*Nosaukums*
1	1	1	Vidējais darbinieku skaits, par kuriem nodokļus maksā darba devējs/NACE 2.red. klasifikators
2	2	1	Vidējais darbinieku skaits, par kuriem nodokļus maksā darba devējs, kuriem tas ir pamatdarbs/NACE 2.red. klasifikators
3	3	1	Bruto darba samaksa pilnu darba laiku un nepilnu

Atribūta ID	*Rādītāja ID (norāde uz rādītāju)*	*Klasifikatora ID (norāde uz klasifikatoru)*	*Nosaukums*
			darba laiku strādājošiem darbiniekiem/NACE 2.red. klasifikators
4	4	1	Budžeta iestādes ieņēmumi - maksas pakalpojumi un citi pašu ieņēmumi/NACE 2.red. klasifikators
5	5	1	Budžeta iestādes un organizācijas kārtējie izdevumi/NACE 2.red. klasifikators
6	8	2	Izejvielu, materiālu, preču krājumi /Gada sākums, beigas

4.1.11. Mainīgie

Mainīgie tiek veidoti kā rādītāja un (ar vai bez) atribūta kombinācija. Katrai pārskata veidlapas šūnai nepieciešams izveidot savu mainīgo. Rādītāja un atribūta kombinācija nosaka katra mainīgā izmantošanu pārskata šūnās (celles), vai tas ir saistīts ar fiksēta izmēra sadaļu, vai arī saistīts ar sadaļām, kurās ir mainīgs rindu/kolonu skaits.

4.11. tabula. Mainīgo apraksts metadatu bāzē

Mainīgā ID	*Rādītāja ID (norāde uz rādītāju)*	*Atribūta ID (norāde uz atribūtu)*	*Ieraksta ID (norāde uz klasifikatora ierakstu)*	*Nosaukums*
1	1			Vidējais darbinieku skaits, par kuriem nodokļus maksā darba devējs, kopā
2	1	1		Vidējais darbinieku skaits, par kuriem nodokļus maksā darba devējs/NACE 2.red. klasifikators
3	2			Vidējais darbinieku skaits, par kuriem nodokļus maksā darba devējs, kuriem tas ir pamatdarbs, kopā
4	2	2		Vidējais darbinieku skaits, par kuriem nodokļus maksā darba devējs, kuriem tas ir pamatdarbs/NACE 2.red. klasifikators
5	3			Bruto darba samaksa pilnu darba laiku un nepilnu darba laiku strādājošiem darbiniekiem, kopā
6	3	3		Bruto darba samaksa pilnu darba laiku un nepilnu darba laiku strādājošiem darbiniekiem/NACE 2.red. klasifikators

Mainīgā ID	*Rādītāja ID (norāde uz rādītāju)*	*Atribūta ID (norāde uz atribūtu)*	*Ieraksta ID (norāde uz klasifikatora ierakstu)*	*Nosaukums*
7	4			Budžeta iestādes ieņēmumi - maksas pakalpojumi un citi pašu ieņēmumi, kopā
8	4	4		Budžeta iestādes ieņēmumi - maksas pakalpojumi un citi pašu ieņēmumi/NACE 2.red. klasifikators
9	5			Budžeta iestādes un organizācijas kārtējie izdevumi, kopā
10	5	5		Budžeta iestādes un organizācijas kārtējie izdevumi/NACE 2.red. klasifikators
11	6			Izmantotie pirktie pakalpojumi, kopā
12	7			Izmantotie pirktie pakalpojumi - maksa par apkuri, ūdeni, elektroenerģiju un gāzi
13	8	6	101	Izejvielu, materiālu, preču krājumi pavisam /pārskata gada sākumā
14	8	6	102	Izejvielu, materiālu, preču krājumi pavisam /pārskata gada beigās
15	9			Izejvielas, materiāli, preces, iegādātas (saņemtas) pārskata gada laikā, pavisam
16	10			Izejvielas, materiāli, mazvērtīgais inventārs u.c. – izlietots pārskata gada laikā, pavisam
17	11			Pārskata aizpildīšanai nepieciešamais laiks (stundas)
18	12			Pārskata aizpildīšanai nepieciešamais laiks (minūtes)

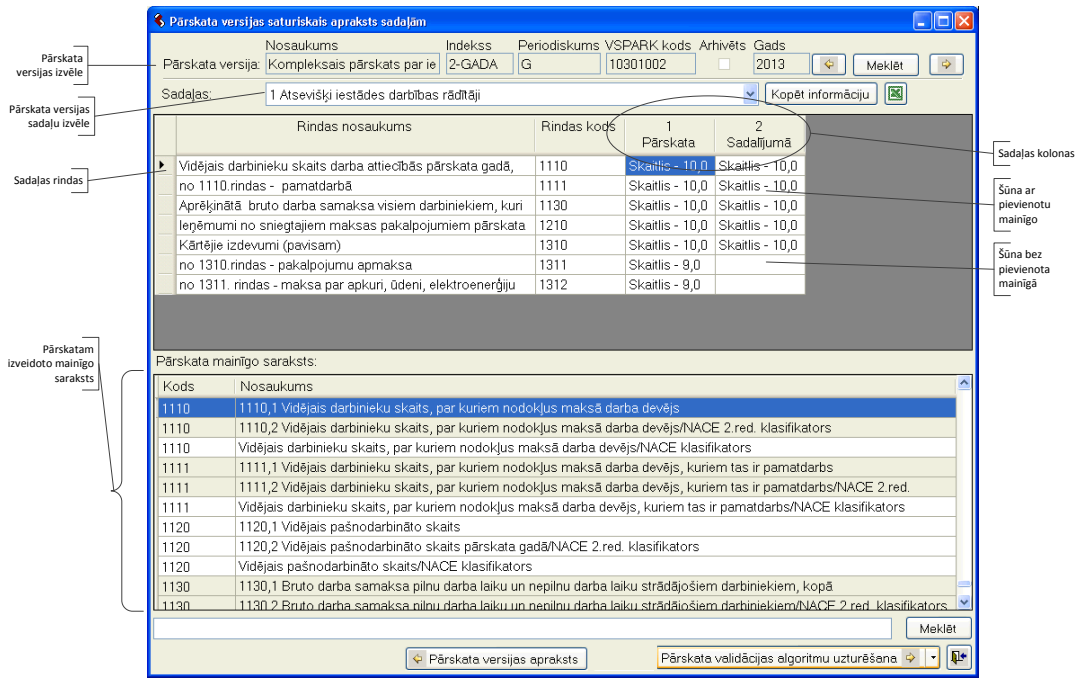
4.1.12. Šūnas (celles)

Pārskata veidlapas vizuālā un saturiskā apraksta pēdējais solis – mainīgo pievienošana konkrētas sadaļas rindas un kolonas krustpunktam, veidojot pārskata šūnas. Ievaddatu formāts skaitliskām šūnām norāda, cik skaitļus pirms un pēc komata atļauts ievadīt konkrētā šūnā.

4.12. tabula. Šūnu (ceļļu) apraksts metadatu bāzē

Sadaļas ID (norāde uz pārskata sadaļu)	*Rindas kods (norāde uz pārskata sadaļas rindu)*	*Kolonas kods (norāde uz pārskata sadaļas kolonu)*	*Mainīgā ID (norāde uz mainīgo)*	*Ievaddatu formāts*
1	1110	1	1	9,0
1	1110	2	2	9,0
1	1111	1	3	9,0
1	1111	2	4	9,0
1	1130	1	5	9,0
1	1130	2	6	9,0
1	1210	1	7	9,0
1	1210	2	8	9,0
1	1310	1	9	9,0
1	1310	2	10	9,0
1	1311	1	11	9,0
1	1312	1	12	9,0
2	600	1	13	9,0
2	600	2	15	9,0
2	600	3	16	9,0
2	600	4	14	9,0
3	Laiks	1	17	2,0
3	Laiks	2	18	2,0

Zemāk attēlota programmatūras forma, kas nodrošina pārskata veidlapas satura izveidi un aprakstīšanu metadatu bāzē.



4.1. att. Pārskata versijas saturiskā apraksta formas izskats

Šī forma nodrošina gan rindu un kolonu veidošanu (izmantojot atbilstošās izvēlnes, kas tiek piedāvātas lietotājiem atbilstošās formas vietās veidot peles labo klikšķi), gan šūnu veidošanu – pievienojot mainīgo atbilstošās rindas/kolonas krustpunktam. Šūnas veidošana notiek izvēloties no saraksta atbilstošo mainīgo un veicot „vilkt un nomest" (drag and drop) metodi, pievieno mainīgo atbilstošai šūnai. Forma nodrošina dažādus kopsavilkumus par pārskata metadatu aprakstiem, kas lietotājiem palīdz kontrolēt un pārbaudīt pārskata aprakstu pareizību Metadatu bāzē.

4.1.13. Validācijas nosacījumi

Konkrētajai pārskata veidlapai zemāk pārskaitīti tikai daži no validācijas nosacījumiem, kas ir savdabīgi un atšķirīgi savā starpā.

4.13. tabula. Validācijas nosacījumu apraksts metadatu bāzē

Valid. nosacījuma kods	*JA daļa*	*TAD daļa*	*Skaidrojums*
1-00		{[:NACE7] <> _0}	Validācijas likums izmantojot specifiskus pseidoelementus pārbauda, vai sadaļā ar mainīgām rindām/kolonām ir norādīta kaut viena kolona. Tiek pārbaudītas tās mainīgās rindas/kolonas, kuras saistītas ar klasifikatoru ar kodu „NACE7"
1-01		[1110,1] + [1111,1] + [1130,1] + [1210,1] + [1310,1] + [1311,1] + [1312,1] > 0	Pārbaude, vai pirmajā sadaļā kaut vienā pirmās ailes šūnā ir ievadīta kāda vērtība.
1-02-1110		[1110,1]=[1110,2]	Likums pārbauda vai pirmajā ailē ievadītā vērtība (kopā) ir vienāda ar pārējās (mainīgajās nezināma skaita ailēs) ievadīto vērtību summu. Analoģiski likumi tiek definēti katrai pārskata rindai, lai katra rindai pārbaudītu vērtību kopā un to sadalījumu pa atsevišķām nozarēm (mainīgās ailes)
1-02-1111		[1111,1]=[1111,2]	
1-03		[1110,1] > 0	Obligāti jābūt datiem norādītā šūnā

Valid. nosacījuma kods	*JA daļa*	*TAD daļa*	*Skaidrojums*
1-05		[1110,1] >= [1111,1]	Tā kā 1111 rindā tiek ievadītas vērtības, kas ir daļa no 1110 rindā norādītās vērtības, tad atbilstoši 1111 rindā ievadītajai vērtībai ir jābūt mazākai par 1110 rindā ievadīto vērtību
1-05-2		[1110,2] >= [1111,2]	Analoģiska pārbaude tiek veikta arī katrai mainīgajai kolonai. Katram aizpildītam pārskatam tiek analizēta atsevišķi katra mainīgā kolona un tiek pārbaudīts vai 1111 rindā ievadītā vērtība ir mazāka par 1110 rindā ievadīto vērtību
1-07	[1110,1] > 5 AND [1110,1] <= 100 AND "2-DARBS"[1110,1,C1] > 0 AND "2-DARBS"[1110,1,C2] > 0 AND "2-DARBS"[1110,1,C3] > 0 AND "2-DARBS"[1110,1,C4] > 0	[1110,1] * 0.8 < (("2-DARBS"[1110,1] - "2-DARBS"[1144,1] + "2-DARBS"[1140,1] - "2-DARBS"[1144,1]) / 8 + ("2-DARBS"[2120,1]) / 12) AND (("2-DARBS"[1110,1] - "2-DARBS"[1144,1] + "2-DARBS"[1140,1] - "2-DARBS"[1144,1]) / 8 + ("2-DARBS"[2120,1]) / 12) < 1.2 * [1110,1]	Likums pārbauda vai uzņēmumam, kuram norādīts vidējais darbinieku skaits robežās no 5 līdz 100 salīdzinot ar darbinieku skaitu, kas norādīts citā ceturkšņa pārskatā „2-darbs”, šo skaitļu atšķirības nepārsniedz 20%
1-20	[1110,1] > 0 AND [1130,1] > 0	[1130,1] / [1110,1] / 12 >= 200	Pārbauda vai vidēji alga vienam cilvēkam nav mazāka par 200Ls
2-01		[600,4] = [600,1] + [600,2] + [600,3]	Otrās sadaļas ceturtās ailes skaitlis ir vienāds ar pirmo trīs aiļu summu
Laiks		[Laiks,1] > 0 OR [Laiks,2] > 0	Jābūt norādītiem datiem par pārskata aizpildīšanas ilgumu.

Atbilstoši norādītajiem validācijas nosacījumiem automātiski ģenerētā SQL procedūra ir attēlota [3.pielikums]. Procedūra satur ne tikai augstākminēto validācijas likumu pārbaudi, bet visus validācijas nosacījumus, kas aprakstīti konkrētajam pārskatam (bet nav attēloti augstākminētajā tabulā).

4.2. Pārskata ievadīto datu apraksts Mikrodatu bāzē

Augstāk bija minēts, ka Mikrodatu bāze satur jau reālos respondentu iesniegto pārskatu ievadītos datus. Bet pārskata datu vērtībām, mikrodatubāzē arī tiek uzkrāti katra pārskata par katru periodu respondentu saraksti, kā arī validācijas kļūdu protokoli.

4.2.1. Pārskata instance

Šīs tabulas ieraksti tiek veidoti brīdī, kad pārskats tiek sagatavots datu ievadei. Pārskata instanču tabulā tiek veidots viens ieraksts katram pārskata periodam norādītajā gadā. Konkrētajā piemērā pārskatam ir gada periodiskums, tāpēc tiek veidota tikai viena instance, kas nozīmē, ka pārskats gada laikā no respondentiem tiek saņemts tikai vienu reizi, kurā ir dati par visu gadu. Piemēram, ja pārskatam būtu ceturkšņa periodiskums, tad instanču tabulā tiktu izveidoti 4 ieraksti – par katru gada ceturksni. Attiecīgi respondenti sniedz pārskatus 4 reizes gadā – par katru gada ceturksni.

4.14. tabula. Pārskata instanču apraksts mikrodatu bāzē

Instances ID	*Pārskata ID (Norāde uz pārskatu Metadatu bāzē)*	*Periods*
1	1	2013

4.2.2. Respondentu saraksts

Katram pārskata periodam (instancei) tiek glabāts pilns respondentu saraksts. Pat ja pārskats ir mēneša, kas nozīmē, ka gada laikā respondentam ir jāsniedz pārskats 12 reizes, viens un tas pats respondents sarakstā parādīsies 12 reizes, pie katras instances. Tas ir tāpēc, ka gada laikā respondentam datos var būt dažādas izmaiņas, piemēram, mainīties nosaukums vai adrese, kā arī uzņēmuma darbības nozare vai uzņēmējdarbības forma. Tāpēc respondents sarakstā tiek glabāts katram periodam, tādejādi saglabājot respondenta stāvokli konkrētā periodā. Katram pārskatam katrā periodā respondentu saraksta izmērs var būt no dažiem simtiem līdz vairākiem tūkstošiem. Darba autors piemēra respondentu sarakstā uzskaitīs tikai vienu respondentu, no kura ir saņemta aizpildīta pārskata veidlapa. Arī no respondenta atribūtiem piemērā ir pārskatīti tikai daži, lai gan praktiski bez šiem atribūtiem vēl tiek uzkrāti

daudzi tehniski atribūti, piemēram, pārskata ievades datums un lietotājs, pārskata neatbildētības kods, un tamlīdzīgi.

4.15. tabula. Instances respondentu saraksts mikrodatu bāzē

Respon-denta ID	*Instances ID*	*Kods*	*Nosaukums*	*Statuss*	*NACE kods*	*ATVK*	*SVTK kods*
1	1	12345678	„Gudrinieku akadēmija”	Ievadīts, validēts	8411	010000	1001

4.2.3. Pārskata vērtības

Katram Metadatos aprakstītam pārskatam, kuram gadu laikā atkarībā no izmaiņām tiek aprakstītas vairākas versijas, tiek veidota sava pārskata vērtību tabula. Šajā tabulā tiek uzkrātas visu respondentu sniegtās konkrētā pārskata vērtības par visiem laika periodiem, sākot ar metadatu bāzē aprakstīto pirmo pārskata versiju. Tā kā piemērā minētajam pārskatam metadatos tika piešķirts identifikators 1, tad arī Mikrodatu bāzē tiek izveidota tabula „Vērtības_1”, kurā tiks vadītas visu respondentu konkrētā pārskata vērtības. Neatkarīgi no pārskata uzbūves un struktūras, visām vērtību tabulām ir identiska struktūra.

4.16. tabula. Pārskatu vērtības mikrodatu bāzē

Vērtības ID	*Respondenta ID (norāde uz respondentu konkrētā instancē)*	*Rādītāja ID (norāde uz rādītāju metadatu bāzē)*	*Mainīgā ID (norāde uz mainīgo metadatu bāzē)*	*Klas. ieraksta ID (norāde uz klasifikatora ierakstu metadatu bāzē)*	*Periods*	*Vērtība*
1	1	1	1		2013	130
2	1	1	2	4	2013	75
3	1	1	2	5	2013	55
4	1	2	3		2013	58
5	1	2	4	4	2013	17
6	1	2	4	5	2013	41
7	1	3	5		2013	452206
8	1	3	6	4	2013	262011
9	1	3	6	5	2013	190195
10	1	4	7		2013	710196
11	1	4	8	4	2013	346564
12	1	4	8	5	2013	363632
13	1	5	9		2013	1072298
14	1	5	10	4	2013	741225
15	1	5	10	5	2013	331072

Vērtības ID	*Respondenta ID (norāde uz respondentu konkrētā instancē)*	*Rādītāja ID (norāde uz rādītāju metadatu bāzē)*	*Mainīgā ID (norāde uz mainīgo metadatu bāzē)*	*Klas. ieraksta ID (norāde uz klasifikatora ierakstu metadatu bāzē)*	*Periods*	*Vērtība*
16	1	6	11		2013	349800
17	1	7	12		2013	88573
18	1	8	13	101	2013	3410
19	1	9	15		2013	32770
20	1	10	16		2013	34505
21	1	8	14	102	2013	1675
22	1	11	17		2013	1

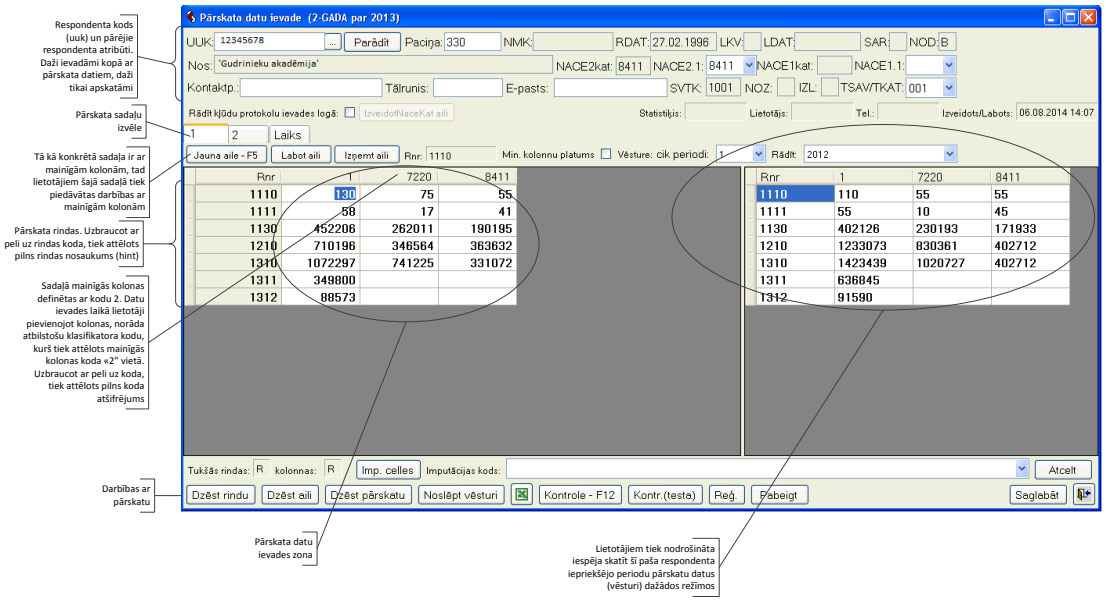
4.2.4. Pārskata mainīgas rindas/kolonas

Tabula satur secību, kādā ievadītas pārskatos ar mainīgām rindām/kolonām šī mainīgās rindas katram respondenta iesniegtajam pārskatam.

4.17. tabula. Pārskatu mainīgās rindas un kolonas mikrodatu bāzē

Respondenta ID (norāde uz respondentu konkrētā instancē)	*Sadaļas ID (norāde uz pārskata veidlapas sadaļu metadatu bāzē)*	*Klas. Ieraksta ID (norāde uz klasifikatora ierakstu metadatu bāzē)*	*Nr. pēc kārtas*
1	1	4	1
1	1	5	2

Zemāk attēlota datu ievades forma, kas nodrošina respondentu iesniegto pārskatu datu ievadi datubāzē. Datu ievades forma katram pārskatam tiek veidota atbilstoši izvēlētā pārskata metadatu aprakstiem metadatu bāzē.



4.2. att. Pārskatu datu ievades forma

Datu ievades forma nodrošina visu nepieciešamo funkcionalitāti, kas nepieciešama pārskatu datu ievades laikā. Formas funkcionalitāte, sadarbībā ar CSP speciālistiem veidota tā, ka pārskata datus iespējams vadīt tikai izmantojot klaviatūru, bez peles palīdzības. Visa nepieciešamā funkcionalitāte ir pieejama izmantojot karstos taustiņus, vai arī tiek izpildīta automātiski pēc noteiktu darbību veikšanas. Piemēram, veicot datu validāciju – taustiņš F12, dati automātiski tiek saglabāti. Ja nav validācijas kļūdu, kursors automātiski pozicionējas respondenta koda laukā, lai varētu turpināt nākamā pārskata ievadi. Svarīga funkcija datu ievades formā ir respondenta vēsturisko pārskata datu attēlošana. Lietotājiem ir iespējams izvēlēties kura iepriekšējā perioda datus nepieciešams apskatīt. Vēsturisko datu apskate ir pieejama 2 režīmos:

- Konkrēta perioda dati – tiek attēloti pašreizējās sadaļas lietotāju izvēlētā vēsturiskā perioda dati. Visbiežāk lietotāji izmanto apskatīt iepriekšējā perioda datus. Tas ir noderīgi situācijās, ja datu validācijas likumos ievadāmā perioda dati tiek salīdzināti ar iepriekšējo periodu datiem. Kļūdu gadījumā statistiķi vienmēr var vizuāli novērtēt gan pašreizējā, gan iepriekšējā perioda datus un pieņemt nepieciešamos lēmumus
- Tekošās rindiņas vēsture. Šis vēstures attēlošanas veids attēlo pašreizējās rindas (kur tiek veikta datu ievade) iepriekšējo periodu datus. Pie tam vēsturisko periodu skaitu var izvēlēties lietotājs. Piemēram, ja tiek vadīts mēneša periodiskuma pārskats par 2015. Gada aprīli, un lietotājs ir izvēlējies skatīt rindiņas vēsturi par iepriekšējiem trim periodiem, tad attiecīgi tiek attēloti pašreizējās rindiņas dati par 2015.gada martu,

februāri un janvāri. Pozicionējoties uz citu rindiņu, vēsturiskie dati tiek attēloti par nākamo izvēlēto datu rindu.

Vēsturisko datu attēlošanu lietotājs var arī atslēgt, ja pašreizējā pārskata ievadē šī funkcija nav nepieciešama – tādejādi vairāk ekrāna vietas izbrīvējot pārskata datu ievadei, kas ir ļoti svarīgi vadot pārskatus, kuru sadaļās ir vairāk par desmit kolonām. Pie tam sistēma atceras lietotāja uzstādījumu datu ievades formā katram pārskatam atsevišķi. Tādejādi lietotāji var nodefinēt katram pārskatam nepieciešamos datu ievades formas uzstādījumus vienu reizi un turpināt vadīt dažādu pārskatu datus, bez nepieciešamības katram pārskatam no jauna mainīt uzstādījumus atbilstoši pārskata specifikai.

5. FIZISKIE SISTĒMAS PARAMETRI

Sistēmas lietotāji ir tikai CSP darbinieki. Metadatu vadīta statistiko datu apstrādes sistēma veidota kā Windows programmatūra, kas ir jāuzstāda uz katra darbinieka datora. Programmatūras pēdējās versijas veidotas izmantojot Microsoft Visual Studio izstrādes vidi. Programmatūra veidota, izmantojot C# programmēšanas valodu un Microsoft .NET 4.0 ietvarprogrammatūru. Lietotāju darbstacijām jābūt operētājsistēmas versijai Microsoft Windows Vista vai jaunāka. Lielākā daļa programmatūras formu veidotas atbilstoši ekrāna izšķirtspējai 1028x768 punkti, kas vēl joprojām ir izplatīta CSP lietotāju datoriem.

Programmatūras arhitektūrā izmantota klienta – servera pieeja.

Kā datubāzes serveris kalpo divi servera datori, kas saslēgti klāsterī, izmantojot klāstera shēmu „Active-Active". Serveriem ir instalēta Microsoft Windows Server 2003 operētājsistēma. Datubāzes vadībai tiek izmantota Microsoft SQL Server 2005 Enterprise Edition programmatūras versija. Servera fiziskie parametri:

Procesors: 4 x 4 kodolu Intel Xeon 2.9Ghz

Operatīvā atmiņa – 16Gb, no kuriem SQL Serverim ir izdalīti 6Gb

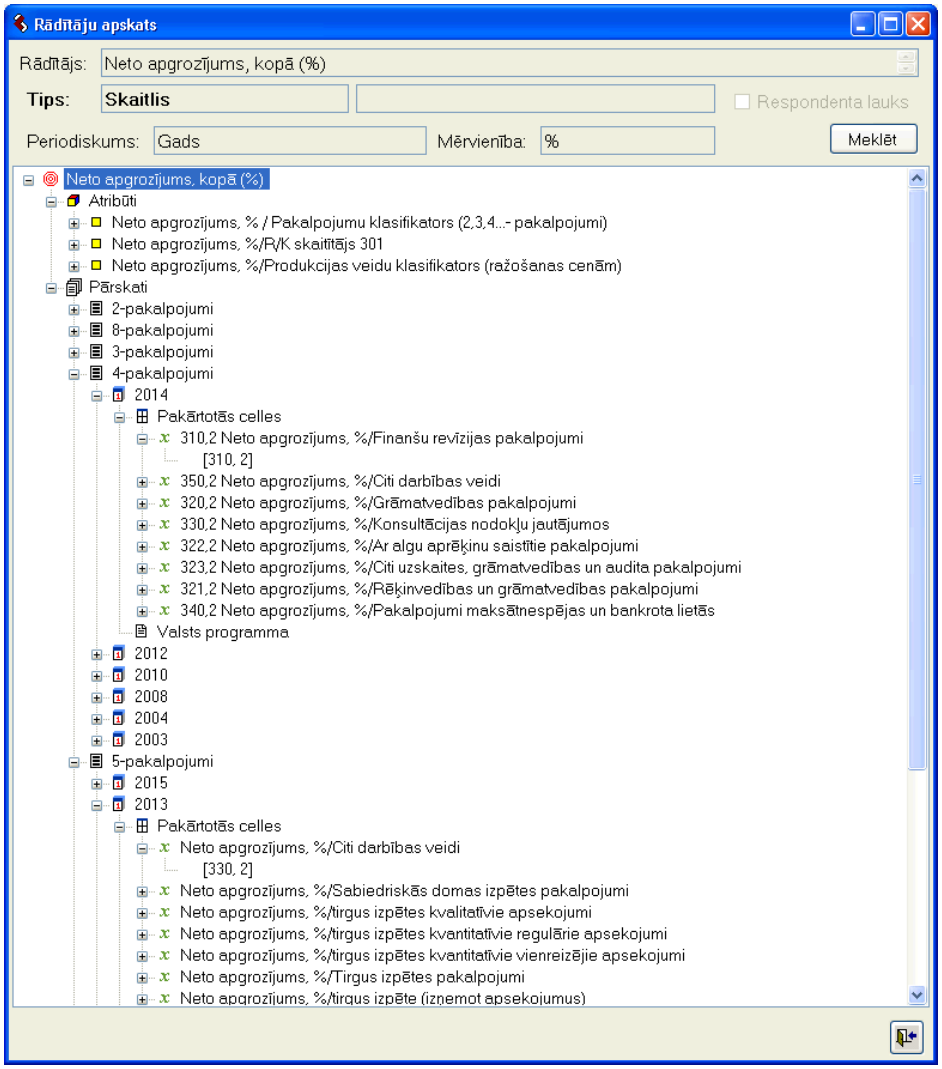
Cietais disks – Ārējais disku masīvs (SAN), kurš sadalīts starp klāstera serveriem

Metadatu vadītas statistiko datu apstrādes sistēmas datubāzu parametri:

- 400 reģistrētu sistēmas lietotāju
- Sistēmā reģistrēti ~200 statistikas pārskati
- Metadatu bāzes fiziskais izmērs – 370Mb
- Mikrodatubāzē uzkrāti pārskatu dati sākot ar 2000. Gadu.
- Mikrodatubāzes datu tabulu fiziskais izmērs (65 Gb, neņemot vērā vietu, kas nepieciešama tabulu indeksu un log failu glabāšanai)
- Tabulas, kur tiek glabāti pārskatu dati (katram metadatos reģistrētam pārskatam, dati glabājas atsevišķā tabulā) satur 30 000 līdz 50 000 000 ierakstu, atkarībā no katra pārskata specifikas un respondentu skaita.
- Makrodatubāzes datu tabulu fiziskais izmērs (41 Gb, neņemot vērā vietu, kas nepieciešama tabulu indeksu un log failu glabāšanai)

- Makrodatubāzes vērtību (faktu) tabula satur 100 000 000 ierakstu, klasifikāciju (dimensiju) tabula satur 370 000 000 ierakstu.

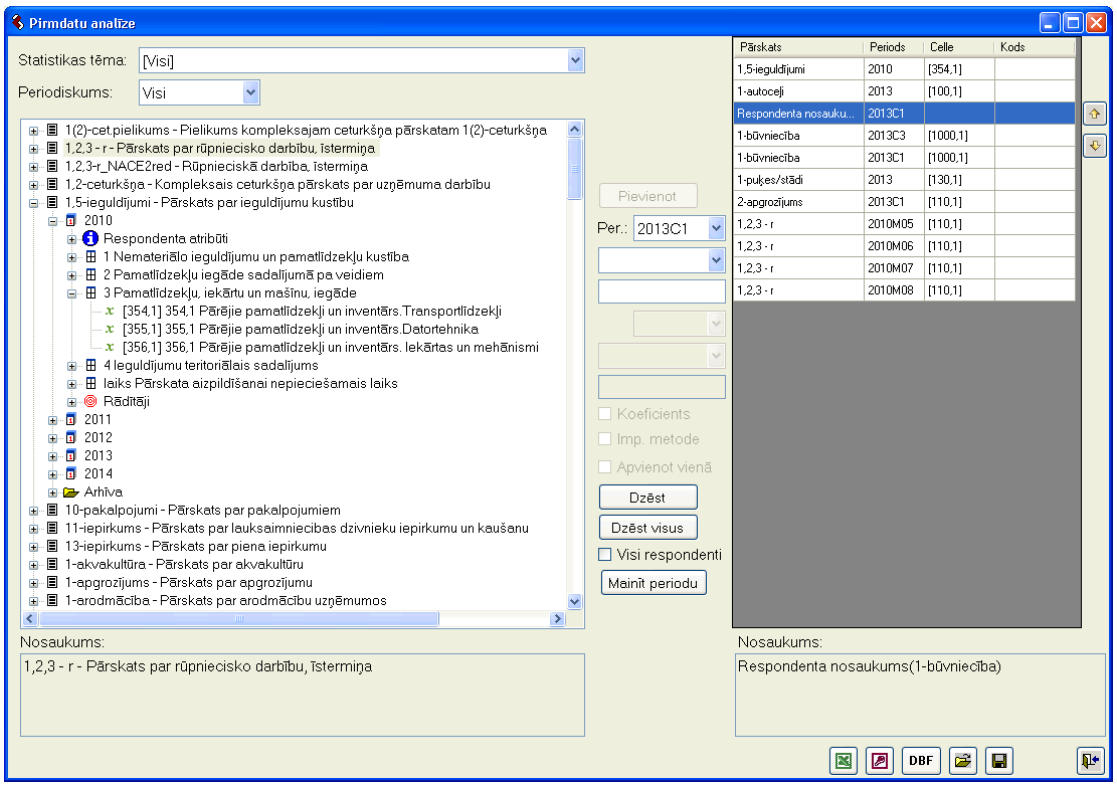
Zemāk attēlota sistēmas ekrānforma, kas paredzēta rādītāja informācijas apskatei metadatu bāzē.



5.1. att. Forma rādītāju informācijas apskatei

Formā hierarhiskā veidā attēlota rādītāja izmantošana dažādos pārskatos. Forma attēlo kādi, atribūti ir izveidoti rādītājam, kuros pārskatos, kuru gadu versijās un kurās šūnās pievienoti konkrētā rādītāja izveidoti mainīgie.

Nākamā forma attēlo mikrodatubāzes analīzes iespējas:



5.2. att. Pārskatu pirmdatu analīzes forma

Lietotājiem ir iespējams atlasīt jebkurus datus no jebkura pārskata par jebkuru laika periodu, apvienot tos vienā kopīgā datu masīvā. Izveidotā datu pieprasījuma rezultātu iespējams eksportēt uz Ms Excel, Ms Access vai DBF datu formātiem. Datu atlase lietotājiem organizēta hierarhiskā kokveida struktūrā. Augstākajā līmenī izvietoti visi pārskati. Nākamais līmenis attēlo pieejamo pārskatu versiju gadus. Nākamajā līmenī tiek attēlotas pārskata versijas sadaļas, kā arī atsevišķā mezgls izveidots respondentu atribūtu izvēlei, un pārskatā izmantoto rādītāju izvēlei. Lietotājs var izvēlēties jebkuras interesējošās šūnas no jebkuras sadaļas. Ja lietotājs nepārzin konkrēto pārskatu un nezin, kurā sadaļas šūnā ir nepieciešamā informācija, lietotājam ir pieejams konkrētajā pārskata izmantoto rādītāju saraksts, kur lietotājs var izvēlēties nepieciešamo rādītāju no konkrētās pārskata versijas. Jebkuram atlasītajam datu elementam lietotājam ir iespējams uzstādīt papildus filtrēšanas kritērijus, lai ierobežoto atlasāmo datu kopu. Forma nodrošina lietotāju izveidoto datu pieprasījumu saglabāšanu ārējā failā un tā ielādi, kas ir sevišķi ērti, ja lietotājs ir izveidojis sarežģītu datu pieprasījumu, kuru bieži nākas izmantot.

SECINĀJUMI

Sistēma ir izstrādāta un tiek izmantota jau vairāk nekā 10 gadus. Sistēma regulāri tiek papildināta ar jaunu funkcionalitāti. CSP tā ir vienīgā biznesa statistikas pārskatu datu apstrādes sistēma. Salīdzinot ar stāvokli pirms sistēmas ieviešanas, panākti sekojoši ieguvumi:

- CSP rīcībā ir vienota sistēma jebkuru biznesa statistikas pārskatu datu apstrādei
- Vienota datu noliktava, kurā uzkrāti visu biznesa statistikas pārskatu pirmdati un agregētie dati.
- Jaunu pārskatu veidošana un esošo pārskatu modificēšana (izmaiņu gadījumā) tiek veikta aizpildot vai labojot sistēmā atbilstošus pārskata metadatus – bez programmēšanas. Šie metadati tiek izmantoti, lai ģenerētu pārskatu datu ievades formas, kā arī datu validācijas un agregācijas procedūras.
- Vienota pieeja jebkura pārskata datu apstrādes procesiem. Lietotājiem, kas veica vairāku pārskatu datu ievadi, vairs nav nepieciešams izmantot vairākas un dažādas, katram pārskatam atsevišķi veidot programmatūru.

Tai pat laikā salīdzinot jauno sistēmu ar iepriekšējiem, atsevišķiem pārskatiem izveidotajiem IT risinājumiem, iespējams identificēt dažas negatīvas iezīmes:

- Sistēma, kas ir izveidota kopīgi jebkādam biznesa statistikas pārskatam, vizuāli nav tik pievilcīga kā konkrētam pārskatam specifiski izveidota programmatūra
- Sistēmā nav iespējams pievienot funkcionalitāti, kas ir specifiska tikai kādam konkrētam pārskatam. Sistēmas pamatā ir procesu standartizācija, kas nozīmē, ka jebkura sistēmas funkcija vienmēr ir pieejama un strādā jebkuram metadatos aprakstītam biznesa statistikas pārskatam.
- Atsevišķiem pārskatiem ar pseidovalodas palīdzību nav iespējams aprakstīt ļoti specifiskus validācijas nosacījumus. Šādi specifiski validācijas likumi ir jāveido CSP programmētājiem atsevišķi.
- Sistēmas līdzekļi nenodrošina sarežģītu un specifisku kopsavilkumu izveidi, piemēram, cenu indeksus. Šādi kopsavilkumi ir jāveido CSP programmētājiem.
- Lietotājiem sākotnēji bija grūti uztvert pārskatu aprakstīšanas principus metadatu bāzē, īpaši rādītāju, atribūtu, mainīgo veidošanas un izmantošanas principus.

Pirms sistēmas ieviešanas CSP organizacionālā struktūra bija orientēta uz pārskatu tematiskām grupām. Katram pārskatam, vai grupai tika veidota sava nodaļa, kurā bija cilvēki, kas veic pilnu pārskatu datu apstrādes procesu šai pārskatu kopai. Šādu nodaļu tipisks sastāvs:

- nodaļas vadītājs, kas atbild par katra pārskata vai tematiskas pārskatu grupas datu apstrādi
- programmētāji, kas izstrādāja un uzturēja konkrēta pārskata datu ievades un apstrādes programmatūru. Veic kopsavilkumu, vai jebkādu citu datu izgūšanu no pašu izstrādātajām datubāzēm, atbilstoši statistiķu vai nodaļas vadītāju vajadzībām.
- operatori, kas veic respondentu iesniegto pārskata veidlapu datu ievadi
- statistiķi, kas veica datu validāciju un datu analīzi.

Līdz ar sistēmas ieviešanu, tika mainīta arī CSP organizacionālā struktūra no pārskatu orientētas datu apstrādes struktūras uz procesu orientētu datu apstrādes struktūru, reorganizējot esošās tematiskās pārskatu nodaļas uz sekojošām:

- metadatu nodaļa – atbild par pārskatu metadatu ievadi, labošanu;
- datu ievades operatori – centra un reģionālās nodaļas, kas veic jebkuru biznesa statistikas pārskatu datu ievadi sistēmā;
- programmētāji – veic dažādas datu apstrādes procedūras, kuras nav iespējams veikt ar sistēmas līdzekļiem (piemēram, specifisku datu validāciju veikšana vai sarežģītu kopsavilkumu iegūšana)
- statistiķi – veic datu analīzi (pārsvarā bez programmētāju palīdzības)

Līdz ar strukturālām iestādes izmaiņām, tika veiktas izmaiņas arī pārskatu datu apstrādes metodoloģijā – tika unificēti un standartizēti pārskatu apstrādes procesi, lai tie būtu kopīgi un vienādi pilnīgi jebkuram biznesa statistikas pārskatam.

Darba autors analizējot sistēmas lietojumu vairāku gadu garumā nonācis pie sekojoša slēdziena:

- izveidotā metadatu vadīta statistisko pārskatu apstrādes sistēma ir inovatīvs un moderns risinājums, kurš ir pielietojams ne tikai statistisko pārskatu datu apstrādei, bet derīgs jebkurā citā nozarē, kur nepieciešams ievadīt un apstrādāt lielu daudzumu dažādu pārskatu (veidlapu) datus, ņemot vērā, ka pārskata veidlapas saturs un struktūra mainās laika gaitā;

- sistēma tiešām ar metadatu vadīta sistēma, jo visi pārskatu aprakstošie metadati aktīvi tiek izmantoti visos pārskata datu apstrādes procesos.
- sistēmas pamatā izmantotais rādītāju, atribūtu, mainīgo veidošanas princips un to piesaiste pārskata sadaļas šūnām (sadaļas rindas un kolonas krustojums) ir veiksmīgs risinājums, un sistēma tiek regulāri papildināta ar jaunu funkcionalitāti, bez nepieciešamības mainīt šo pamat principu
- laika gaitā pieaugot sistēmā uzkrāto datu daudzumam, galveno funkciju (datu ievade, validācija) ātrdarbība būtiski nav pasliktinājusies, kas nozīmē, ka sistēmas arhitektūra un programmatūra ir rūpīgi izstrādāta
- iespējams, sistēmas arhitektūru varēja veidot savādāk saistībā ar klasifikatoriem, kuri sistēmā ir nodalīti divās daļās (starptautiskie/nacionālie un lokālie). Katra veida klasifikatoram ir savs pielietojums mainīgo veidošanā un pārskatu veidlapu struktūras aprakstīšanā. Viena veida klasifikatori izmantojami pārskatos ar fiksētu rindu/kolonu skaitu, turpretī otra veida klasifikatori ir pielietojami tikai pārskatu sadaļās ar mainīgu rindu/kolonu skaitu. Bet ir identificēts CSP pārskats, kurā viens un tas pats klasifikators tiek izmantots gan fiksēta izmēra sadaļās, gan arī sadaļās ar mainīgu rindu/kolonu skaitu. Pašreizējā sistēmas realizācijā šādus klasifikatorus nepieciešams metadatos ievadīt divas reizes – katru reizi ar savu tipu.
- Sistēmas funkciju programmēšana ir sarežģītāka, nekā citās sistēmās (kas nav metadatu vadītas). Sarežģītās lietas ir dinamisku pieprasījumu veidošana, jo katra pārskata (visu respondentu) dati tiek glabāti fiziski atsevišķās tabulās. Kā arī vairāku programmatūras funkciju realizācija ir saistīta ar sarežģītu SQL pieprasījumu veidošanu, kas saistīti ar datu atlasi no pārskatu vērtību tabulām un datu transformācijām uz matricas veida struktūru. Visas šādi sarežģītie pieprasījumi ir rūpīgi jātestē un jāpielāgo, lai nodrošinātu apmierinošu pieprasījumu ātrdarbību ar liela apjoma datu tabulām.

CSP pārstāvji vairakkārt metadatu vadītu statistiko datu apstrādes sistēmu ir demonstrējuši dažādās starptautiskās statistikas konferencēs. Dažādu valstu eksperti izveidoto sistēmu ir atzinīgi novērtējuši, norādot, ka sistēma ir progresīva, bet ambicioza.

Bakalaura darbā autors aprakstījis metadatu vadītu statistisko datu apstrādes sistēmu, tās vajadzības, prasības, kā arī izstrādātās sistēmas arhitektūru. Zemāk pārskaitītas svarīgākās no aktivitātēm, kuras darba autors veicis sistēmas izstrādes laikā.

- Pārrunas ar dažādu tematisko statistikas nodaļu lietotājiem, lai novērtētu esošo sistēmu funkcionalitāti un definētu funkcionalitātes prasības jaunajai sistēmai, tieši datu ievades, validācijas funkcijām;
- Sistēmas arhitektūras izveide;
- Datubāzu struktūru izveide;
- Programmatūras projektējuma izstrāde;
- Datubāzu programmēšanas darbi – trigeri, funkcijas, procedūras;
- Datu validācijas procedūras sagataves izveide, kas tiek izmantota par pamatu automātiski ģenerētām datu validācijas procedūrām;
- Pieprasījumu sagatavju izveide, kas tiek izmantota pirmdatu analīzes formā, veidojot datu pieprasījumus, kas atlasa jebkura pārskata, jebkura perioda datus;
- Daudz laika tika veltīts pieprasījumu optimizācijai;
- Dažu programmatūras formu un funkciju programmēšana (piemēram, datu agregācija);

IZMANTOTĀ LITERATŪRA UN AVOTI

1. Information systems architecture for national and international statistical organisations. Guidelines and Recommendations. Statistical Standards and Studies No. 51. UN/ECE. Geneva,1999. [tiešsaiste] [skatīts 10.05.2015] Pieejams: http://www.unece.org/stats/documents/information_systems_architecture/1.e.pdf
2. Valsts statistikas likums. [tiešsaiste] [skatīts 10.05.2015] Pieejams: http://likumi.lv/doc.php?id=45932
3. ES tiesību akti par statistiku. [tiešsaiste] [skatīts 20.05.2015] Pieejams: http://eur-lex.europa.eu/content/legis/legis-statistiques.html
4. Guidelines for the modeling of Statistical Data and Metadata. UN/ECE. Geneva, 1995
5. Sundgren, B. An Infological Approach to Data Bases. Stockholm 1973. PhD dissertation. ISBN 91-38-01750-4. 478p.

1. PIELIKUMS. STATISTIKAS PĀRSKATA VEIDLAPAS PIEMĒRS

CENTRĀLĀ STATISTIKAS PĀRVALDE

Mūsu adrese:

Lāčplēša iela 1, Rīga, LV-1301,
fakss 67366658, www.csb.gov.lv

Elektroniskā pārskata iesniegšana:
https://eparskats.csb.gov.lv

Konsultācijas par pārskata aizpildīšanu:

tālr. ***67366840, 67366917***

06.11.2006. Ministru kabineta noteikumu
Nr.922 pielikums Nr.30
VSPARK 10515017

2-darbs

ceturkšņa

Pārskats par darbu

Iesniedz ***līdz 15. datumam*** *pēc pārskata ceturkšņa*

2015. gada pārskata ceturksnis *(lūdzu, atzīmējiet atbilstošo)*: I | II | III | IV

RESPONDENTS

Nosaukums		
Pasta adrese		
Mājaslapas adrese		
Biroja vai pamatdarbības vienības adrese		
Tālrunis		fakss
E-pasta adrese		

Nodokļu maksātāja reģistrācijas numurs

VEIDLAPAS AIZPILDĪTĀJS

Vārds, Uzvārds

Tālrunis

e-pasta adrese

- Apsekojuma mērķis ir iegūt statistisko informāciju par darba tirgus rādītājiem - aizņemto un brīvo darbvietu skaitu, atalgojumu un pārējām darbaspēka izmaksām, nostrādātajām stundām, streikiem sadalījumā pa visām tautsaimniecības nozarēm un sektoriem.
- Ar šī apsekojuma rezultātiem var iepazīties CSP mājaslapas datubāzes sadaļā „Iedzīvotāji un sociālie procesi", ikceturkšņa informatīvajos apskatos, Latvijas statistikas ikmēneša biļetenā un gadagrāmatā, kā arī preses izlaidumos.
- Veidlapu ieteicams aizpildīt uzņēmuma grāmatvedim; par brīvajām darbvietām - uzņēmuma vai personāldaļas vadītājam.
- Veidlapas aizpildīšanas norādījumi atrodami Centrālās statistikas pārvaldes (CSP) mājaslapā www.csb.gov.lv (Respondentiem/Veidlapas/2-darbs/Kā aizpildīt).

Centrālā statistikas pārvalde saskaņā ar Valsts statistikas likumu garantē sniegtās informācijas konfidencialitāti

1. DATI PAR DARBA ŅĒMĒJIEM, PAR KURIEM JĀVEIC DARBA LAIKA UZSKAITE

	Mēr-vienība	Rindas kods	Pavisam *(2. + 3. aile)*	tai skaitā strādāja	
				normālo darba laiku	nepilnu darba laiku
A	B	C	1	2	3
1.1. DARBA ŅĒMĒJU SKAITS DARBA TIESISKAJĀS ATTIECĪBĀS (IESKAITOT NEREZIDENTUS)					
Darba ņēmēju skaits ceturkšņa pirmajā kalendārajā darba dienā	cilvēki	1110			
Darba ņēmēju skaits ceturkšņa pēdējā kalendārajā darba dienā *(1140. rinda = 1141. rinda + 1142. rinda)*	cilvēki	1140			
tai skaitā:					

darba ņēmēju skaits pamatdarbā (ar algas nodokļa grāmatiņām)		cilvēki	1141			
darba ņēmēju skaits blakus darbā (bez algas nodokļa grāmatiņām)		cilvēki	1142			
no 1140. rindas – bērna kopšanas atvaļinājumā		cilvēki	1144		X	X
no 1140. rindas – nerezidenti (ārvalstnieki, kuriem ir darba atļauja un kuri strādā Latvijā mazāk par vienu gadu)		cilvēki	1145		X	X
1.3. NOSTRĀDĀTAIS UN APMAKSĀTAIS LAIKS						
Nostrādātās stundas	ceturkšņa **1.** mēnesī	stundas	1310			
	ceturkšņa **2.** mēnesī	stundas	1320			
	ceturkšņa **3.** mēnesī	stundas	1330			
Nenostrādātās, bet apmaksātās stundas (ikgadējie atvaļinājumi, darbnespējas lapas A u.c.)	ceturkšņa **1.** mēnesī	stundas	1340			
	ceturkšņa **2.** mēnesī	stundas	1350			
	ceturkšņa **3.** mēnesī	stundas	1360			
1.4. DARBA SAMAKSA						
Darba ņēmēju skaits, kam aprēķināta darba samaksa	par ceturkšņa **1.** mēnesi	cilvēki	1410			
	par ceturkšņa **2.** mēnesi	cilvēki	1420			
	par ceturkšņa **3.** mēnesi	cilvēki	1430			
Aprēķinātā bruto darba samaksa	par ceturkšņa **1.** mēnesi	*euro*	1440			
	par ceturkšņa **2.** mēnesi	*euro*	1450			

	par ceturkšņa **3.** mēnesi	*euro*	1460			
Aprēķinātā bruto darba samaksa ceturksnī *(1470. rinda = 1440. līdz 1460. rindas summa =*		*euro*	1470			
tai skaitā: regulārā darba samaksa, ko aprēķina katru mēnesi		*euro*	1471		X	X
samaksa par ikgadējo atvaļinājumu un papildatvaļinājumu; par dīkstāvēm; par citām dienām, kurās nestrādā (mācību atvaļinājumi, kāzas, bēres, donoru dienas)		*euro*	1473		X	X
neregulārā darba samaksa, ko neaprēķina katru mēnesi (ceturkšņa, pusgada, gada prēmijas, prēmijas svētkos, 13. alga, naudas balva u.c.); piemaksas pie atvaļinājuma, atvaļinājuma pabalsts; kompensācija par neizmantoto atvaļinājumu		*euro*	1474		X	X
darbnespējas lapu A apmaksa		*euro*	1477		X	X
no 1470. rindas – aprēķinātā neto darba samaksa		*euro*	1480		X	X
no 1470. rindas – nerezidentiem aprēķinātā bruto darba samaksa		*euro*	1490		X	X
no 1470. rindas – darba samaksas subsīdijas		*euro*	1499		X	X

(turpinājums)

	Mēr-vienība	Rindas kods	Pavisam *(2. + 3. aile)*	tai skaitā strādāja	
				normālo darba laiku	nepilnu darba laiku
A	B	C	1	2	3
1.5 DATI PAR DARBA ŅĒMĒJIEM, KURI NAV BIJUŠI DARBA ATTIECĪBĀS PILNU MĒNESI ***(no 1.3. un 1.4. sadaļas – nav nostrādājuši pilnu mēnesi sakarā ar pieņemšanu darbā vai atlaišanu no darba)***					
Attiecīgi no 1410. līdz 1430. rindai – darba ņēmēju skaits, kam aprēķināta darba samaksa, bet	ceturkšņa **1.** mēnesī	cilvēki	1510	X	X
	ceturkšņa **2.** mēnesī	cilvēki	1520	X	X

kas nebija darba attiecībās pilnu mēnesi	ceturkšņa **3.** mēnesī	cilvēki	1530	X		X
Attiecīgi no 1310. līdz 1360. rindai – nostrādātās un nenostrādātās, bet apmaksātās stundas darba ņēmējiem, kuri uzrādīti attiecīgi no 1510. līdz 1530. rindai	ceturkšņa **1.** mēnesī	stundas	1540	X		X
	ceturkšņa **2.** mēnesī	stundas	1550	X		X
	ceturkšņa **3.** mēnesī	stundas	1560	X		X

1.6. DATI PAR SIEVIETĒM (aizpilda tikai par 1. ceturksni)						
Nostrādātās stundas sievietēm kopā 1. ceturksnī *(no 1310. līdz 1330. rindai)*		stundas	1610			
Nenostrādātās, bet apmaksātās stundas sievietēm kopā 1. ceturksnī *(no 1340. līdz 1360. rindai)*		stundas	1620			
Sieviešu skaits, kam aprēķināta darba samaksa	par janvāri *(no 1410. rindas)*	cilvēki	1630			
	par februāri *(no 1420. rindas)*	cilvēki	1640			
	par martu *(no 1430. rindas)*	cilvēki	1650			
Aprēķinātā bruto darba samaksa sievietēm kopā par 1. ceturksni *(no 1470. rindas)*		*euro*	1660			

2. DATI PAR PĀRĒJIEM NODARBINĀTAJIEM (neietver 1. sadaļā uzrādītos darba ņēmējus)

2.1. NODARBINĀTIE AR APRĒĶINĀTU DARBA SAMAKSU

Katru nodarbināto uzrāda tikai vienā no rindām: 2110., 2120. vai 2130.	Rindas kods	Nodarbināto skaits ceturkšņa			Aprēķinātā bruto darba samaksa ceturksnī, *euro*
		1. mēnesī	2. mēnesī	3. mēnesī	
A	C	1	2	3	4
Nodarbinātie steidzamos, īslaicīgos, vienreizējos darbos, par kuriem norēķinās uzreiz pēc darba izpildes	2110				
Nodarbinātie ar darba vai uzņēmuma līgumu, par kura darbu izpildi nav iespējams noteikt faktiski nostrādātās stundas (nav reģistrējušies kā pašnodarbinātās personas)	2120				
Pašnodarbinātie (valsts sociālās apdrošināšanas iemaksas kārto patstāvīgi)	2130				

2.2. NODARBINĀTIE BEZ APRĒĶINĀTAS DARBA SAMAKSAS

	Rindas kods	Personu skaits, kas ceturksnī strādāja vismaz vienu dienu
A	C	1
Darba devēji; strādājošie ģimenes (mājsaimniecības) locekļi; personas, kuras veic uzņēmējdarbību un nenodarbina citus	2210	

3. CITAS DARBA DEVĒJA IZMAKSAS

	Rindas kods	*Euro*
A	C	1
Bruto darba samaksa natūrā ceturksnī (dāvanas un mantiskās balvas; preces, pakalpojumi personīgai lietošanai; izdevumi nodarbināto nodrošināšanai ar dzīvojamo platību; darba devēja izdevumi automašīnu lietošanai personīgajām vajadzībām; ēdināšanas taloni; izmaksas bērnudārza nodrošināšanai; transporta izdevumi no mājām uz darbu, mobilā tālruņa, ko lieto darba un personīgām vajadzībām, rēķinu apmaksa u.c. – pilnu aprakstu skatīt veidlapas aizpildīšanas norādījumos)	310	
Darba devēja valsts sociālās apdrošināšanas obligātās iemaksas, kas aprēķinātas 1. sadaļā ietvertajiem darba ņēmējiem ceturksnī	320	
Darba devēja brīvprātīgās sociālās apdrošināšanas iemaksas, kas samaksātas par nodarbinātajiem ceturksnī (papildu pensiju apdrošināšana, pensijas uzkrājuma veidošana, veselības, dzīvības apdrošināšana, dzīvības apdrošināšana ar uzkrājuma veidošanu, nelaimes gadījumu apdrošināšana u.c.)	330	

Darba devēja pabalsti un kompensācijas, kas izmaksātas esošajiem, kā arī bijušajiem nodarbinātajiem ceturksnī (izmaksas jubileju, kāzu, bērna dzimšanas, apbedīšanas gadījumā u.c. materiālie pabalsti; pabalsti veselības aprūpei; briļļu iegādes apmaksa; atlīdzība par kaitējumu veselībai; stipendijas, mācību maksa u.c.)	340	
Aprēķinātais bruto atlaišanas pabalsts ceturksnī	350	
tai skaitā, kas tiek aprēķināts, ja darba devējs samazina darbinieku skaitu vai darba devējs tiek likvidēts (saskaņā ar Darba likuma 101. panta pirmās daļas 9. un 10. punktu)	351	

4. AIZŅEMTO UN BRĪVO DARBVIETU SKAITS PA PROFESIJU PAMATGRUPĀM

Profesiju klasifikatora pamatgrupas [1]	Rindas kods	Aizņemto darbvietu skaits *(1140. rindas 1. aile - 1144. rindas 1. aile + 2120. rindas 3. aile + 2130. rindas 3. aile)*	Brīvo darbvietu skaits ceturkšņa pēdējā kalendārajā darba dienā (ieskaitot tās, par kurām nav jāveic darba laika uzskaite, kā arī pašnodarbināto vietas) [2]
A	C	1	2
Pavisam *(401.– 410. rindas summ*	400		
1. Vadītāji	401		
2. Vecākie speciālisti	402		
3. Speciālisti	403		
4. Kalpotāji	404		
5. Pakalpojumu un tirdzniecības darbinieki	405		
6. Kvalificēti lauksaimniecības, mežsaim- niecības un zivsaimniecības darbinieki	406		
7. Kvalificēti strādnieki un amatnieki	407		
8. Iekārtu un mašīnu operatori un izstrādājumu montieri	408		
9. Vienkāršās profesijas	409		
10. Nacionālo bruņoto spēku profesij	410		

[1)] 401.–409. rindas A ailē uzrādītas Profesiju klasifikatora pamatgrupas. Tās kārtas numurs atbilst profesijas (aroda, amata, specialitātes) koda pirmajai zīmei. Lai pareizi aizpildītu šo sadaļu, katram amatam nosaka profesijas kodu saskaņā ar Profesiju klasifikatoru atbilstoši Ministru kabineta 2010. gada 18. maija noteikumiem Nr. 461 „Noteikumi par Profesiju klasifikatoru, profesijai atbilstošajiem pamatuzdevumiem un kvalifikācijas pamatprasībām un Profesiju klasifikatora lietošanas un aktualizēšanas kārtību".

Profesiju klasifikators atrodams Labklājības ministrijas mājaslapā www.lm.gov.lv sadaļā *Darba devējiem / Profesiju klasifikators*.

[2)] Par brīvu darbvietu uzskata tādu darbvietu, par kuru nav noslēgts līgums un pretendents nav izvēlēts, darba devējs veic aktīvus pasākumus, lai atrastu darbvietai piemērotu pretendentu, izņemot pretendentu no saviem nodarbinātajiem, un darba devējs gatavojas to aizpildīt nekavējoties vai tuvākajā laikā.

Brīvo darbvietu skaitā neiekļauj darbvietas, kurām netiek meklēts pretendents (piemēram, darbu pilda esošie nodarbinātie, darbvieta pagaidām nav nepieciešama, paredzēts to likvidēt).

5. STREIKI

	Mērvienība	Rindas kods	Pārskata ceturksnī
A	B	C	1
Streiku skaits	skaits	510	
Nodarbināto skaits, kas piedalījās streikos	cilvēki	520	
Nodarbināto, kuri piedalījās streikos,	cilvēkdienas	530	

Lūdzu, norādiet pārskata aizpildīšanai patērēto laiku

stundas minūtes

201__. gada _____. _____ Vadītājs _____

/Vārds, uzvārds, paraksts/

Paldies par veltīto laiku!

2. PIELIKUMS. PIEMĒRS AIZPILDĪTAI PĀRSKATA VEIDLAPAI

LATVIJAS REPUBLIKAS

CENTRĀLĀ STATISTIKAS PĀRVALDE

2-gada

gada

Kompleksais pārskats par darbību 2013.gadā

*Iesniedz **līdz 2014.gada 10.maijam***

Mūsu adrese:

Lāčplēša iela 1, Rīga, LV-1301,
fakss 67366658, www.csb.gov.lv

Elektroniskā pārskata iesniegšana:
https://eparskats.csb.gov.lv

Konsultācijas par pārskata aizpildīšanu:

*tālr. **67366840, 67366917, 67366838***

06.11.2006. Ministru kabineta noteikumu Nr.922 pielikums Nr.98
VSPARK 10301002

1. ATSEVIŠĶI IESTĀDES DARBĪBAS RĀDĪTĀJI

*Konsultācijas par sadaļas aizpildīšanu: tālr. **67366840, 67366917***	Rindas kods	Pārskata gadā (pavisam iestādē)	1.nozare: Kods1: 7220	2.nozare: Kods2: 8411	3.nozare:	4.nozare:
A	B	1	2	3	4	5
Vidējais darbinieku skaits darba attiecībās pārskata gadā, par kuriem nodokļus maksā darba devējs (neiekļauj darbiniekus, kuri bija bērna kopšanas atvaļinājumā)	1110	**130**	**75**	**55**		
no 1110.rindas – **pamatdarbā** (ar algas nodokļa grāmatiņām)	1111	**58**	**17**	**41**		
Aprēķinātā bruto darba samaksa visiem darbiniekiem, kuri bija darba attiecībās pārskata gadā un par kuriem nodokļus maksā darba devējs (latos)	1130	**452206**	**262011**	**190195**		
Ieņēmumi no sniegtajiem maksas pakalpojumiem pārskata gadā **(izpilde pēc uzkrāšanas principa)**	1210	**710196**	**346564**	**363632**		
Kārtējie izdevumi **(izpilde pēc uzkrāšanas principa), saskaņā ar budžetu izdevumu klasifikāciju atbilstoši ekonomiskajām kategorijām**	1310	**1072297**	**741225**	**331072**		

no 1310.rindas – **pakalpojumu apmaksa** (piemēram, remonts (izņemot kapitālos izdevumus), tehniskā apkope, transporta pakalpojumi, noliktavas nomas maksa, banku pakalpojumi, pasts, sakari, apsardze, reklāma, apdrošināšana, automobiļu tehniskā apskate, telpu tīrīšana, komunālie pakalpojumi (izdevumi par apkuri, ūdeni, elektroenerģiju, gāzi un pārējiem komunālajiem pakalpojumiem), grāmatvedības, juridiskie, individuālie u.c. pakalpojumi, kas apmaksāti pārskata gadā; neieskaita nodokļus)	1311	**349800**	**X**	**X**	**X**	**X**
no 1311.rindas – maksa par apkuri, ūdeni, elektroenerģiju un gāzi	1312	**88573**	**X**	**X**	**X**	**X**

2. IZEJVIELU, MATERIĀLU, MAZVĒRTĪGĀ INVENTĀRA IEGĀDE UN KRĀJUMI

(latos)

Konsultācijas par sadaļas aizpildīšanu: *tālr.* ***67366838***	Rindas kods	Krājumi pārskata gada sākumā	Iegādāts pārskata gadā	Izlietots pārskata gadā	Krājumi pārskata gada beigās *(1.+2.-3.aile)*
A	B	1	2	3	4
Izejvielas, materiāli, mazvērtīgais inventārs un citas preces iestādes vajadzībām	600	**3410**	**32770**	**34505**	**1675**

Lūdzu, norādiet pārskata aizpildīšanai patērēto laiku **1** stundas ____ minūtes

2014.gada _____. _____ Vadītājs _____

/Vārds, uzvārds, paraksts/

Paldies par veltīto laiku!

3. PIELIKUMS. PIEMĒRS AUTOMĀTISKI ĢENERĒTAI VALIDĀCIJAS PROCEDŪRAI

```
ALTER Procedure [dbo].[mi_validation_917_2013]
(
	@period varchar(7),
	@resp_quest_id integer,
	@SPID integer = 0,
	@retval integer = 0 OUTPUT
)
AS set nocount on

--K 2-GADA
set @resp_quest_id = isnull(@resp_quest_id, 0)
set @SPID = isnull(@SPID, 0)

declare @SetOption varchar(100)
declare @Value varchar(100)
declare @arithignore bit
declare @arithabort bit
declare @ansi_warnings bit
set @arithignore = 0
set @arithabort = 0
set @ansi_warnings = 0

CREATE TABLE #useroptions (
	SetOption varchar(100),
	Value varchar(100))
INSERT INTO #useroptions
   EXEC ('dbcc useroptions')

DECLARE [useroptions] CURSOR
LOCAL FAST_FORWARD READ_ONLY
FOR SELECT * FROM #useroptions
	WHERE SetOption IN ('arithignore', 'arithabort', 'ansi_warnings')

OPEN [useroptions]
FETCH NEXT FROM [useroptions] INTO @SetOption, @Value
While (@@FETCH_STATUS = 0)
BEGIN
	if @SetOption = 'arithignore'
		set @arithignore = 1
	if @SetOption = 'arithabort'
		set @arithabort = 1
	if @SetOption = 'ansi_warnings'
		set @ansi_warnings = 1
	FETCH NEXT FROM [useroptions] INTO @SetOption, @Value
END

CLOSE [useroptions]
DEALLOCATE [useroptions]
drop table #useroptions

Set arithabort off
Set arithignore on
Set ansi_warnings off

declare @RESPCODE varchar(30)
declare @imp_code char(2)
declare @avvoid_error int
declare @AGR varchar(3)
declare @apgrozijums int
declare @ATVK char(7)
declare @contact_name varchar(200)
declare @contact_tel varchar(20)
declare @darbskaits int
```

```
declare @dlkv datetime
declare @email varchar(300)
declare @importance_rank varchar(1)
declare @IUFIK char(4)
declare @lkv char(1)
declare @lkvkat char(1)
declare @Nace_kat char(4)
declare @NACE1 char(4)
declare @NACE2 char(4)
declare @NACE3 char(4)
declare @Nace71 char(4)
declare @Nace72 char(4)
declare @Nace73 char(4)
declare @Nace7kat char(4)
declare @nod char(1)
declare @outlayer varchar(1)
declare @rdt datetime
declare @sar char(1)
declare @spec1 char(4)
declare @spec2 char(4)
declare @strata int
declare @Tkat char(7)
declare @Trakst char(1)
declare @Tsav char(3)
declare @instance_id integer
declare @state tinyint
declare @state_new tinyint
declare @one_resp tinyint
declare @error_variable_1 varchar(100)
declare @error_variable_2 varchar(100)
declare @error_variable_3 varchar(100)
declare @record_id integer
declare @resp_code varchar(130)
declare @koef float
declare @message varchar(500)

declare      @validation_id integer
declare @ex_flag bit
declare      @periodicity char(1)
declare @priority tinyint
declare @error_message varchar(100)
declare @code varchar(100)



declare @917_781_1110_1_G__0 numeric(16,0)
declare @917_781_1110_1_G_C1_0 numeric(16,0)
declare @917_781_1110_1_G_C2_0 numeric(16,0)
declare @917_781_1110_1_G_C3_0 numeric(16,0)
declare @917_781_1110_1_G_C4_0 numeric(16,0)
declare @917_781_1140_1_G__0 numeric(16,0)
declare @917_781_1141_1_G__0 numeric(16,0)
declare @917_781_1141_1_G_C1_0 numeric(16,0)
declare @917_781_1141_1_G_C2_0 numeric(16,0)
declare @917_781_1141_1_G_C3_0 numeric(16,0)
declare @917_781_1141_1_G_C4_0 numeric(16,0)
declare @917_781_1144_1_G__0 numeric(15,0)
declare @917_781_1470_1_G__0 numeric(19,0)
declare @917_781_2120_1_G__0 numeric(16,0)
declare @917_781_2120_4_G__0 numeric(19,0)
declare @917_917_1110_1_G__0 numeric(19,0)
declare @917_917_1110_2_G__0 numeric(19,0)
declare @917_917_1111_1_G__0 numeric(19,0)
declare @917_917_1111_2_G__0 numeric(19,0)
declare @917_917_1130_1_G__0 numeric(19,0)
declare @917_917_1130_2_G__0 numeric(19,0)
declare @917_917_1210_1_G__0 numeric(19,0)
declare @917_917_1210_2_G__0 numeric(19,0)
declare @917_917_1310_1_G__0 numeric(19,0)
declare @917_917_1310_2_G__0 numeric(19,0)
```

```
declare @917_917_1311_1_G__0 numeric(18,0)
declare @917_917_1312_1_G__0 numeric(18,0)
declare @917_917_600_1_G__0 numeric(18,0)
declare @917_917_600_2_G__0 numeric(18,0)
declare @917_917_600_3_G__0 numeric(18,0)
declare @917_917_600_4_G__0 numeric(18,0)
declare @917_917_Laiks_1_G__0 numeric(11,0)
declare @917_917_Laiks_2_G__0 numeric(11,0)
DECLARE @resp_quests_x TABLE (
       resp_quest_id int,
       quest_id int,
       vers_id int,
       period varchar(7),
       periodicity char(1))
DECLARE @resp_quests TABLE (
       resp_quest_id int,
       quest_id int,
       vers_id int,
       period varchar(7),
       periodicity char(1))
DECLARE @resp_codes TABLE (
       resp_quest_id int,
       table_id int,
       code varchar(11))
DECLARE @errors TABLE (
       validation_id integer,
       row_col_id integer,
       record_id bigint,
       period varchar(20),
       periodicity char(1),
       resp_quest_id integer,
       message varchar(500),
       error_value_1 varchar(100),
       error_value_2 varchar(100),
       error_value_3 varchar(100))
DECLARE @validations TABLE (
       validation_id int,
       ex_flag bit,
       error_message varchar(100),
       priority tinyint)

       INSERT INTO @validations (validation_id, ex_flag, error_message, priority)
       SELECT validation_id, ex_flag, error_message, priority
       FROM [MetaDataBase].[dbo].[me_validations]
       WHERE vers_id = 917 and
              quest_year = 2013 and
              gen_flag = 1

if @resp_quest_id = 0
       BEGIN
              if @SPID = 0
                     DECLARE RespList CURSOR
                     LOCAL FAST_FORWARD READ_ONLY
                     FOR SELECT DISTINCT RQ.[resp_quest_id], RQ.[state]
                            FROM [MicroDataBase].[dbo].[mi_resp_quests] as RQ,
                                   [MicroDataBase].[dbo].[mi_survey_instances] as SI
                            WHERE (RQ.[state] % 10) <> 0 and
                                   SI.[version_id] = 917 and
                                   SI.[period] = @period and
                                   SI.[instance_id] = RQ.[instance_id]
              else
                     DECLARE RespList CURSOR
                     LOCAL FAST_FORWARD READ_ONLY
                     FOR SELECT DISTINCT RQ.[resp_quest_id], RQ.[state]
                            FROM [MicroDataBase].[dbo].[mi_resp_quests] as RQ,
                                   [MicroDataBase].[dbo].[mi_uuk_filtred] as F
                            WHERE F.[SPID] = @SPID and
                                   (RQ.[state] % 10) <> 0 and
                                   RQ.[resp_quest_id] = F.[resp_quest_id]
              set @one_resp = 0
```

```
	END
else
	BEGIN
		DECLARE RespList CURSOR
		LOCAL FAST_FORWARD READ_ONLY
		FOR SELECT DISTINCT [resp_quest_id], [state]
			FROM [MicroDataBase].[dbo].[mi_resp_quests]
			WHERE [resp_quest_id] = @resp_quest_id
		set @one_resp = 1
	END
OPEN RespList
FETCH NEXT FROM RespList INTO @resp_quest_id, @state
While (@@FETCH_STATUS = 0)
BEGIN

	if @one_resp = 0
		BEGIN
			SELECT @state_new = RQ.[state]
			FROM [MicroDataBase].[dbo].[mi_resp_quests] as RQ
			WHERE RQ.[resp_quest_id] = @resp_quest_id

			if (isnull(@state_new, 0) % 10)  = 0
				GOTO Naakamais
		END

	SELECT @RESPCODE = RQ.[code], @imp_code = RQ.[imp_code], @instance_id =
RQ.[instance_id], @koef = RQ.koef
	FROM [MicroDataBase].[dbo].[mi_resp_quests] as RQ
	WHERE RQ.[resp_quest_id] = @resp_quest_id

	SELECT @resp_quest_id = RQX.[resp_quest_id], @AGR = RQX.[AGR], @apgrozijums =
RQX.[apgrozijums], @ATVK = RQX.[ATVK], @contact_name = RQX.[contact_name],
@contact_tel = RQX.[contact_tel], @darbskaits = RQX.[darbskaits], @dlkv =
RQX.[dlkv], @email = RQX.[email], @importance_rank = RQX.[importance_rank], @IUFIK
= RQX.[IUFIK], @lkv = RQX.[lkv], @lkvkat = RQX.[lkvkat], @Nace_kat =
RQX.[Nace_kat], @NACE1 = RQX.[NACE1], @NACE2 = RQX.[NACE2], @NACE3 = RQX.[NACE3],
@Nace71 = RQX.[Nace71], @Nace72 = RQX.[Nace72], @Nace73 = RQX.[Nace73], @Nace7kat =
RQX.[Nace7kat], @nod = RQX.[nod], @outlayer = RQX.[outlayer], @rdt = RQX.[rdt],
@sar = RQX.[sar], @spec1 = RQX.[spec1], @spec2 = RQX.[spec2], @strata =
RQX.[strata], @Tkat = RQX.[Tkat], @Trakst = RQX.[Trakst], @Tsav = RQX.[Tsav]
	FROM [MicroDataBase].[dbo].[mi_resp_quests_BR] as RQX
	WHERE RQX.[resp_quest_id] = @resp_quest_id

	DELETE FROM [MicroDataBase].[dbo].[mi_valid_errors] WHERE [resp_quest_id] =
@resp_quest_id

	INSERT INTO @resp_quests ([resp_quest_id], [quest_id], [vers_id], [period],
[periodicity])
	SELECT distinct RQ1.[resp_quest_id], QV1.[quest_id], SI1.[version_id],
SI1.[period], Q.[periodicity]
	FROM [mi_resp_quests] as RQ1,
		[mi_resp_quests] as RQ2,
		[mi_survey_instances] as SI1,
		[mi_survey_instances] as SI2,
		[MetaDataBase].[dbo].[me_quest_versions] as QV1,
		[MetaDataBase].[dbo].[me_quest_versions] as QV2,
		[MetaDataBase].[dbo].[me_questionnaires] as Q
	WHERE RQ2.resp_quest_id = @resp_quest_id and
		RQ1.code = RQ2.code and
		RQ1.[instance_id] = SI1.[instance_id] and
		RQ2.[instance_id] = SI2.[instance_id] and
		SI1.[version_id] = QV1.[vers_id] and
		SI2.[version_id] = QV2.[vers_id] and
		QV1.[quest_id] = Q.[quest_id] and
		QV1.[quest_id] = QV2.[quest_id]
	UNION
	SELECT distinct RQ1.[resp_quest_id], QV1.[quest_id], SI1.[version_id],
V917.[period], Q.[periodicity]
	FROM [mi_resp_quests] as RQ1,
		[mi_resp_quests] as RQ2,
```

```
              [mi_values_61] as V917,
              [mi_survey_instances] as SI1,
              [mi_survey_instances] as SI2,
              [MetaDataBase].[dbo].[me_quest_versions] as QV1,
              [MetaDataBase].[dbo].[me_quest_versions] as QV2,
              [MetaDataBase].[dbo].[me_questionnaires] as Q
       WHERE RQ2.resp_quest_id = @resp_quest_id and
              RQ1.code = RQ2.code and
              RQ1.[instance_id] = SI1.[instance_id] and
              RQ2.[instance_id] = SI2.[instance_id] and
              SI1.[version_id] = QV1.[vers_id] and
              SI2.[version_id] = QV2.[vers_id] and
              QV1.[quest_id] = Q.[quest_id] and
              QV1.[quest_id] = QV2.[quest_id] and
              RQ1.[resp_quest_id] = V917.[resp_quest_id]
       UNION
       SELECT distinct RQ1.[resp_quest_id], QV1.[quest_id], SI1.[version_id],
V781.[period], Q.[periodicity]
       FROM [mi_resp_quests] as RQ1,
              [mi_resp_quests] as RQ2,
              [mi_values_50] as V781,
              [mi_survey_instances] as SI1,
              [MetaDataBase].[dbo].[me_quest_versions] as QV1,
              [MetaDataBase].[dbo].[me_questionnaires] as Q
       WHERE RQ2.resp_quest_id = @resp_quest_id and
              RQ1.code = RQ2.code and
              RQ1.[instance_id] = SI1.[instance_id] and
              SI1.[version_id] = QV1.[vers_id] and
              QV1.[quest_id] = Q.[quest_id] and
              RQ1.[resp_quest_id] = V781.[resp_quest_id]

       INSERT INTO @resp_codes ([resp_quest_id], [table_id], [code])
       SELECT distinct RC.[resp_quest_id], RC.[table_id], RC.[code]
       FROM [mi_resp_codes] as RC,
              @resp_quests as RQ
       WHERE RC.resp_quest_id = RQ.resp_quest_id

       if (dbo.sf_periodtrans('G', @period) = dbo.sf_fullperiodiflastpart('G',
@period))
       if exists(SELECT * FROM @resp_quests WHERE [quest_id] = 61 and
(dbo.sf_periodtrans_extend('G', [period]) = dbo.sf_fullperiodiflastpart_extend('G',
@period)))
       BEGIN
              --aizbaaznis ja gadās ka BEGIN - END ir tukšs
              SELECT 1

              DELETE FROM @resp_quests_x

              INSERT INTO @resp_quests_x
              SELECT distinct RQ1.* FROM @resp_quests as RQ1,
                     @resp_quests as RQ2
              WHERE RQ1.[quest_id] = RQ2.[quest_id]
                     and RQ2.vers_id = 781 and (dbo.sf_periodtrans_extend('G',
RQ1.[period]) = dbo.sf_fullperiodiflastpart_extend('G', @period)
                     or dbo.sf_periodtrans_extend('C', RQ1.[period]) =
dbo.sf_fullperiodiflastpart_extend('C', dbo.sf_previousperiod_extend_new('C',
@period, '', 'C1'))
                     or dbo.sf_periodtrans_extend('C', RQ1.[period]) =
dbo.sf_fullperiodiflastpart_extend('C', dbo.sf_previousperiod_extend_new('C',
@period, '', 'C2'))
                     or dbo.sf_periodtrans_extend('C', RQ1.[period]) =
dbo.sf_fullperiodiflastpart_extend('C', dbo.sf_previousperiod_extend_new('C',
@period, '', 'C3'))
                     or dbo.sf_periodtrans_extend('C', RQ1.[period]) =
dbo.sf_fullperiodiflastpart_extend('C', dbo.sf_previousperiod_extend_new('C',
@period, '', 'C4')))
              SELECT
              @917_781_1110_1_G__0 = sum(case when V.[variable_id] = 21434 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
```

```
      ,      @917_781_1110_1_G_C1_0 = sum(case when V.[variable_id] = 21434 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C1'))) then V.[cvalue] else
null end)
      ,      @917_781_1110_1_G_C2_0 = sum(case when V.[variable_id] = 21434 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C2'))) then V.[cvalue] else
null end)
      ,      @917_781_1110_1_G_C3_0 = sum(case when V.[variable_id] = 21434 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C3'))) then V.[cvalue] else
null end)
      ,      @917_781_1110_1_G_C4_0 = sum(case when V.[variable_id] = 21434 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C4'))) then V.[cvalue] else
null end)
      ,      @917_781_1140_1_G__0 = sum(case when V.[variable_id] = 21433 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
      ,      @917_781_1141_1_G__0 = sum(case when V.[variable_id] = 22102 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
      ,      @917_781_1141_1_G_C1_0 = sum(case when V.[variable_id] = 22102 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C1'))) then V.[cvalue] else
null end)
      ,      @917_781_1141_1_G_C2_0 = sum(case when V.[variable_id] = 22102 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C2'))) then V.[cvalue] else
null end)
      ,      @917_781_1141_1_G_C3_0 = sum(case when V.[variable_id] = 22102 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C3'))) then V.[cvalue] else
null end)
      ,      @917_781_1141_1_G_C4_0 = sum(case when V.[variable_id] = 22102 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C4'))) then V.[cvalue] else
null end)
      ,      @917_781_1144_1_G__0 = sum(case when V.[variable_id] = 21517 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
      ,      @917_781_1470_1_G__0 = sum(case when V.[variable_id] = 21216 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
      ,      @917_781_2120_1_G__0 = sum(case when V.[variable_id] = 22091 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
      ,      @917_781_2120_4_G__0 = sum(case when V.[variable_id] = 21308 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
      FROM [MicroDataBase].[dbo].[mi_values_50] as V WITH( INDEX(
IX_resp_var_per_rec_mark))
            LEFT JOIN [MetaDataBase].[dbo].[me_attrib_classif_codes] as C ON
(V.[record_dim_id] = C.[record_id])
      WHERE [resp_quest_id] in (SELECT resp_quest_id FROM @resp_quests_x)
      SELECT
            @917_917_1110_1_G__0 = sum(case when V.[variable_id] = 34052 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
      ,      @917_917_1110_2_G__0 = sum(case when V.[variable_id] = 79454 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
      ,      @917_917_1111_1_G__0 = sum(case when V.[variable_id] = 34061 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
      ,      @917_917_1111_2_G__0 = sum(case when V.[variable_id] = 79455 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
```

```
	,	@917_917_1130_1_G__0 = sum(case when V.[variable_id] = 4851 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_1130_2_G__0 = sum(case when V.[variable_id] = 79453 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_1210_1_G__0 = sum(case when V.[variable_id] = 10639 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_1210_2_G__0 = sum(case when V.[variable_id] = 79457 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_1310_1_G__0 = sum(case when V.[variable_id] = 1421 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_1310_2_G__0 = sum(case when V.[variable_id] = 79458 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_1311_1_G__0 = sum(case when V.[variable_id] = 1546 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_1312_1_G__0 = sum(case when V.[variable_id] = 79459 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_600_1_G__0 = sum(case when V.[variable_id] = 1496 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_600_2_G__0 = sum(case when V.[variable_id] = 1446 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_600_3_G__0 = sum(case when V.[variable_id] = 54048 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_600_4_G__0 = sum(case when V.[variable_id] = 1471 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_Laiks_1_G__0 = sum(case when V.[variable_id] = 110439 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	,	@917_917_Laiks_2_G__0 = sum(case when V.[variable_id] = 63453 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
	FROM [MicroDataBase].[dbo].[mi_values_61] as V WITH( INDEX(
IX_resp_var_per_rec_mark))
		LEFT JOIN [MetaDataBase].[dbo].[me_attrib_classif_codes] as C ON
(V.[record_dim_id] = C.[record_id])
	WHERE [resp_quest_id] = @resp_quest_id
			BEGIN
	-- likums nr 1-00
			if NOT (exists(SELECT 1 FROM @resp_codes as RC, @resp_quests as
RQ WHERE RC.[table_id] = 4253 and RQ.[vers_id] = 917 and
dbo.sf_periodtrans_extend(dbo.sf_max_periodicity(RQ.[periodicity], 'G'),
RQ.[period]) = dbo.sf_periodtrans_extend(dbo.sf_max_periodicity(RQ.[periodicity],
'G'), @period) and RQ.[resp_quest_id] = RC.[resp_quest_id] and (RC.[code] is not
null)))
				BEGIN
					set @error_variable_1 = null
					set @error_variable_2 = null
					set @error_variable_3 = null
					if isnumeric(@error_variable_1) = 1
						set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
					if isnumeric(@error_variable_2) = 1
						set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
					if isnumeric(@error_variable_3) = 1
						set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
					INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
```

```
                                    VALUES (94221, @period, 'G', @resp_quest_id,
'1.iedaļā nav norādīts NACE2red kods vai sadalījums pa NACE2red.',
@error_variable_1, @error_variable_2, @error_variable_3)
                              END
            END

            if (exists(SELECT * FROM @resp_quests WHERE [quest_id] = 50 and
(dbo.sf_periodtrans_extend('G', [period]) = dbo.sf_fullperiodiflastpart_extend('G',
@period)))
                  or exists(SELECT * FROM @resp_quests WHERE [quest_id] = 50 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C1'))))
                  or exists(SELECT * FROM @resp_quests WHERE [quest_id] = 50 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C2'))))
                  or exists(SELECT * FROM @resp_quests WHERE [quest_id] = 50 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C3'))))
                  or exists(SELECT * FROM @resp_quests WHERE [quest_id] = 50 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C4'))))
)           BEGIN
      -- likums nr 1-07
                  if (isnull(@917_917_1110_1_G__0, 0) > 5 AND
isnull(@917_917_1110_1_G__0, 0) <= 100 AND isnull(@917_781_1110_1_G_C1_0, 0) > 0
AND isnull(@917_781_1110_1_G_C2_0, 0) > 0 AND isnull(@917_781_1110_1_G_C3_0, 0) > 0
AND isnull(@917_781_1110_1_G_C4_0, 0) > 0)
                        if NOT (isnull(@917_917_1110_1_G__0, 0) * 0.8 <
((isnull(@917_781_1110_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0) +
isnull(@917_781_1140_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 8 +
(isnull(@917_781_2120_1_G__0, 0)) / 12) AND ((isnull(@917_781_1110_1_G__0, 0) -
isnull(@917_781_1144_1_G__0, 0) + isnull(@917_781_1140_1_G__0, 0) -
isnull(@917_781_1144_1_G__0, 0)) / 8 + (isnull(@917_781_2120_1_G__0, 0)) / 12) <
1.2 * isnull(@917_917_1110_1_G__0, 0))
                        BEGIN
                              set @error_variable_1 =
round(isnull(@917_917_1110_1_G__0, 0), 3)
                              set @error_variable_2 =
round(((isnull(@917_781_1110_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0) +
isnull(@917_781_1140_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 8 +
(isnull(@917_781_2120_1_G__0, 0)) / 12), 3)
                              set @error_variable_3 = null
                              if isnumeric(@error_variable_1) = 1
                                    set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                              if isnumeric(@error_variable_2) = 1
                                    set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                              if isnumeric(@error_variable_3) = 1
                                    set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                              INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                              VALUES (94229, @period, 'G', @resp_quest_id,
'1110.rinda atšķiras no 2-darbs vairāk kā par 20%', @error_variable_1,
@error_variable_2, @error_variable_3)
                        END
            -- likums nr 1-08
                  if (isnull(@917_917_1110_1_G__0, 0) > 100 AND
isnull(@917_917_1110_1_G__0, 0) <= 250 AND isnull(@917_781_1110_1_G_C1_0, 0) > 0
AND isnull(@917_781_1110_1_G_C2_0, 0) > 0 AND isnull(@917_781_1110_1_G_C3_0, 0) > 0
AND isnull(@917_781_1110_1_G_C4_0, 0) > 0)
                        if NOT (isnull(@917_917_1110_1_G__0, 0) * 0.9 <
((isnull(@917_781_1110_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0) +
isnull(@917_781_1140_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 8 +
(isnull(@917_781_2120_1_G__0, 0)) / 12) AND ((isnull(@917_781_1110_1_G__0, 0) -
isnull(@917_781_1144_1_G__0, 0) + isnull(@917_781_1140_1_G__0, 0) -
isnull(@917_781_1144_1_G__0, 0)) / 8 + (isnull(@917_781_2120_1_G__0, 0)) / 12) <
1.1 * isnull(@917_917_1110_1_G__0, 0))
                        BEGIN
```

```
                                set @error_variable_1 =
round(isnull(@917_917_1110_1_G__0, 0), 3)
                                set @error_variable_2 =
round((isnull(@917_781_1110_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0) +
isnull(@917_781_1140_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 8 +
(isnull(@917_781_2120_1_G__0, 0)) / 12, 3)
                                set @error_variable_3 = null
                                if isnumeric(@error_variable_1) = 1
                                       set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_2) = 1
                                       set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_3) = 1
                                       set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                VALUES (94230, @period, 'G', @resp_quest_id,
'1110.rinda atšķiras no 2-darbs vairāk kā par 10%', @error_variable_1,
@error_variable_2, @error_variable_3)
                           END
             -- likums nr 1-09
                    if (isnull(@917_917_1110_1_G__0, 0) > 250 AND
isnull(@917_781_1110_1_G_C1_0, 0) > 0 AND isnull(@917_781_1110_1_G_C2_0, 0) > 0 AND
isnull(@917_781_1110_1_G_C3_0, 0) > 0 AND isnull(@917_781_1110_1_G_C4_0, 0) > 0)
                           if NOT (isnull(@917_917_1110_1_G__0, 0) * 0.95 <
((isnull(@917_781_1110_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0) +
isnull(@917_781_1140_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 8 +
(isnull(@917_781_2120_1_G__0, 0)) / 12) AND ((isnull(@917_781_1110_1_G__0, 0) -
isnull(@917_781_1144_1_G__0, 0) + isnull(@917_781_1140_1_G__0, 0) -
isnull(@917_781_1144_1_G__0, 0)) / 8 + (isnull(@917_781_2120_1_G__0, 0)) / 12) <
1.05 * isnull(@917_917_1110_1_G__0, 0))
                           BEGIN
                                set @error_variable_1 =
round(isnull(@917_917_1110_1_G__0, 0), 3)
                                set @error_variable_2 =
round(((isnull(@917_781_1110_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0) +
isnull(@917_781_1140_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 8 +
(isnull(@917_781_2120_1_G__0, 0)) / 12), 3)
                                set @error_variable_3 = null
                                if isnumeric(@error_variable_1) = 1
                                       set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_2) = 1
                                       set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_3) = 1
                                       set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                VALUES (94231, @period, 'G', @resp_quest_id,
'1110.rinda atšķiras no 2-darbs vairāk kā par 5%', @error_variable_1,
@error_variable_2, @error_variable_3)
                           END
             -- likums nr 1-11
                    if (isnull(@917_917_1130_1_G__0, 0) > 0 AND
isnull(@917_917_1110_1_G__0, 0) > 0 AND isnull(@917_917_1110_1_G__0, 0) <= 100 AND
isnull(@917_781_1110_1_G_C1_0, 0) > 0 AND isnull(@917_781_1110_1_G_C2_0, 0) > 0 AND
isnull(@917_781_1110_1_G_C3_0, 0) > 0 AND isnull(@917_781_1110_1_G_C4_0, 0) > 0)
                           if NOT (isnull(@917_917_1130_1_G__0, 0) * 0.8 <
(isnull(@917_781_1470_1_G__0, 0) + isnull(@917_781_2120_4_G__0, 0)) AND
(isnull(@917_781_1470_1_G__0, 0) + isnull(@917_781_2120_4_G__0, 0)) < 1.2 *
isnull(@917_917_1130_1_G__0, 0))
                           BEGIN
                                set @error_variable_1 =
round(isnull(@917_917_1130_1_G__0, 0), 3)
                                set @error_variable_2 =
round(isnull(@917_781_1470_1_G__0, 0) + isnull(@917_781_2120_4_G__0, 0), 3)
```

```
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
                                          set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                          set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
                                          set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                    VALUES (94232, @period, 'G', @resp_quest_id,
'1130.rinda atšķiras no 2-darbs vairāk kā par 20%', @error_variable_1,
@error_variable_2, @error_variable_3)
                              END
                  -- likums nr 1-12
                        if (isnull(@917_917_1130_1_G__0, 0) > 0 AND
isnull(@917_917_1110_1_G__0, 0) > 100 AND isnull(@917_917_1110_1_G__0, 0) <= 250
AND isnull(@917_781_1110_1_G_C1_0, 0) > 0 AND isnull(@917_781_1110_1_G_C2_0, 0) > 0
AND isnull(@917_781_1110_1_G_C3_0, 0) > 0 AND isnull(@917_781_1110_1_G_C4_0, 0) >
0)
                              if NOT (isnull(@917_917_1130_1_G__0, 0) * 0.9 <
(isnull(@917_781_1470_1_G__0, 0) + isnull(@917_781_2120_4_G__0, 0)) AND
(isnull(@917_781_1470_1_G__0, 0) + isnull(@917_781_2120_4_G__0, 0)) < 1.1 *
isnull(@917_917_1130_1_G__0, 0))
                              BEGIN
                                    set @error_variable_1 =
round(isnull(@917_917_1130_1_G__0, 0), 3)
                                    set @error_variable_2 =
round(isnull(@917_781_1470_1_G__0, 0) + isnull(@917_781_2120_4_G__0, 0), 3)
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
                                          set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                          set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
                                          set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                    VALUES (94233, @period, 'G', @resp_quest_id,
'1130.rinda atšķiras no 2-darbs vairāk kā par 10%', @error_variable_1,
@error_variable_2, @error_variable_3)
                              END
                  -- likums nr 1-13
                        if (isnull(@917_917_1130_1_G__0, 0) > 0 AND
isnull(@917_917_1110_1_G__0, 0) > 250 AND isnull(@917_781_1110_1_G_C1_0, 0) > 0 AND
isnull(@917_781_1110_1_G_C2_0, 0) > 0 AND isnull(@917_781_1110_1_G_C3_0, 0) > 0 AND
isnull(@917_781_1110_1_G_C4_0, 0) > 0)
                              if NOT (isnull(@917_917_1130_1_G__0, 0) * 0.95 <
(isnull(@917_781_1470_1_G__0, 0) + isnull(@917_781_2120_4_G__0, 0)) AND
(isnull(@917_781_1470_1_G__0, 0) + isnull(@917_781_2120_4_G__0, 0)) < 1.05 *
isnull(@917_917_1130_1_G__0, 0))
                              BEGIN
                                    set @error_variable_1 =
round(isnull(@917_917_1130_1_G__0, 0), 3)
                                    set @error_variable_2 =
round(isnull(@917_781_1470_1_G__0, 0) + isnull(@917_781_2120_4_G__0, 0), 3)
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
                                          set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                          set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
```

```
                                        set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                    VALUES (94234, @period, 'G', @resp_quest_id,
'1130.rinda atšķiras no 2-darbs vairāk kā par 5%', @error_variable_1,
@error_variable_2, @error_variable_3)
                            END
             -- likums nr 1-14
                    if (isnull(@917_917_1110_1_G__0, 0) > 0 AND
isnull(@917_917_1110_1_G__0, 0) <= 5 AND isnull(@917_781_1110_1_G_C1_0, 0) > 0 AND
isnull(@917_781_1110_1_G_C2_0, 0) > 0 AND isnull(@917_781_1110_1_G_C3_0, 0) > 0 AND
isnull(@917_781_1110_1_G_C4_0, 0) > 0)
                            if NOT (isnull(@917_917_1110_1_G__0, 0) - 5 <
((isnull(@917_781_1110_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0) +
isnull(@917_781_1140_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 8 +
(isnull(@917_781_2120_1_G__0, 0)) / 12) AND ((isnull(@917_781_1110_1_G__0, 0) -
isnull(@917_781_1144_1_G__0, 0) + isnull(@917_781_1140_1_G__0, 0) -
isnull(@917_781_1144_1_G__0, 0)) / 8 + (isnull(@917_781_2120_1_G__0, 0)) / 12) <
isnull(@917_917_1110_1_G__0, 0) + 5)
                            BEGIN
                                    set @error_variable_1 =
round(isnull(@917_917_1110_1_G__0, 0), 3)
                                    set @error_variable_2 =
round((isnull(@917_781_1110_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0) +
isnull(@917_781_1140_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 8 +
(isnull(@917_781_2120_1_G__0, 0)) / 12, 3)
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
                                            set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                            set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
                                            set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                    VALUES (94235, @period, 'G', @resp_quest_id,
'1110.rinda atšķiras no 2-darbs vairāk kā par ± 5 cilvēkiem', @error_variable_1,
@error_variable_2, @error_variable_3)
                            END
             -- likums nr 1-16
                    if (isnull(@917_917_1111_1_G__0, 0) > 5 AND
isnull(@917_917_1111_1_G__0, 0) <= 100 AND isnull(@917_781_1110_1_G_C1_0, 0) > 0
AND isnull(@917_781_1110_1_G_C2_0, 0) > 0 AND isnull(@917_781_1110_1_G_C3_0, 0) > 0
AND isnull(@917_781_1110_1_G_C4_0, 0) > 0)
                            if NOT (isnull(@917_917_1111_1_G__0, 0) * 0.8 <
((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4) AND
((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4) < 1.2 *
isnull(@917_917_1111_1_G__0, 0))
                            BEGIN
                                    set @error_variable_1 =
round(isnull(@917_917_1111_1_G__0, 0), 3)
                                    set @error_variable_2 =
round((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4, 3)
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
                                            set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                            set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
                                            set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
```

```
                                VALUES (94236, @period, 'G', @resp_quest_id,
'1111.rinda atšķiras no 2-darbs vairāk kā par 20%', @error_variable_1,
@error_variable_2, @error_variable_3)
                        END
            -- likums nr 1-17
                  if (isnull(@917_917_1111_1_G__0, 0) > 100 AND
isnull(@917_917_1111_1_G__0, 0) <= 250 AND isnull(@917_781_1110_1_G_C1_0, 0) > 0
AND isnull(@917_781_1110_1_G_C2_0, 0) > 0 AND isnull(@917_781_1110_1_G_C3_0, 0) > 0
AND isnull(@917_781_1110_1_G_C4_0, 0) > 0)
                        if NOT (isnull(@917_917_1111_1_G__0, 0) * 0.9 <
((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4) AND
((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4) < 1.1 *
isnull(@917_917_1111_1_G__0, 0))
                        BEGIN
                                set @error_variable_1 =
round(isnull(@917_917_1111_1_G__0, 0), 3)
                                set @error_variable_2 =
round((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4, 3)
                                set @error_variable_3 = null
                                if isnumeric(@error_variable_1) = 1
                                        set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_2) = 1
                                        set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_3) = 1
                                        set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                VALUES (94237, @period, 'G', @resp_quest_id,
'1111.rinda atšķiras no 2-darbs vairāk kā par 10%', @error_variable_1,
@error_variable_2, @error_variable_3)
                        END
            -- likums nr 1-18
                  if (isnull(@917_917_1111_1_G__0, 0) > 250 AND
isnull(@917_781_1110_1_G_C1_0, 0) > 0 AND isnull(@917_781_1110_1_G_C2_0, 0) > 0 AND
isnull(@917_781_1110_1_G_C3_0, 0) > 0 AND isnull(@917_781_1110_1_G_C4_0, 0) > 0)
                        if NOT (isnull(@917_917_1111_1_G__0, 0) * 0.95 <
((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4) AND
((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4) < 1.05 *
isnull(@917_917_1111_1_G__0, 0))
                        BEGIN
                                set @error_variable_1 =
round(isnull(@917_917_1111_1_G__0, 0), 3)
                                set @error_variable_2 =
round((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4, 3)
                                set @error_variable_3 = null
                                if isnumeric(@error_variable_1) = 1
                                        set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_2) = 1
                                        set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_3) = 1
                                        set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                VALUES (94238, @period, 'G', @resp_quest_id,
'1111.rinda atšķiras no 2-darbs vairāk kā par 5%', @error_variable_1,
@error_variable_2, @error_variable_3)
                        END
            -- likums nr 1-19
                  if (isnull(@917_917_1111_1_G__0, 0) > 0 AND
isnull(@917_917_1111_1_G__0, 0) <= 5 AND isnull(@917_781_1110_1_G_C1_0, 0) > 0 AND
isnull(@917_781_1110_1_G_C2_0, 0) > 0 AND isnull(@917_781_1110_1_G_C3_0, 0) > 0 AND
isnull(@917_781_1110_1_G_C4_0, 0) > 0)
                        if NOT (isnull(@917_917_1111_1_G__0, 0) - 5 <
((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4) AND
```

```
((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4) <
isnull(@917_917_1111_1_G__0, 0) + 5)
                        BEGIN
                                set @error_variable_1 =
round(isnull(@917_917_1111_1_G__0, 0), 3)
                                set @error_variable_2 =
round((isnull(@917_781_1141_1_G__0, 0) - isnull(@917_781_1144_1_G__0, 0)) / 4, 3)
                                set @error_variable_3 = null
                                if isnumeric(@error_variable_1) = 1
                                        set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_2) = 1
                                        set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_3) = 1
                                        set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                VALUES (94247, @period, 'G', @resp_quest_id,
'1111.rinda atšķiras no 2-darbs vairāk kā par ± 5 cilvēkiem', @error_variable_1,
@error_variable_2, @error_variable_3)
                        END
                END
        ELSE
        BEGIN
                set @avvoid_error = 1
                END

                if (exists(SELECT * FROM @resp_quests WHERE [quest_id] = 50 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C1'))))
                        or exists(SELECT * FROM @resp_quests WHERE [quest_id] = 50 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C2'))))
                        or exists(SELECT * FROM @resp_quests WHERE [quest_id] = 50 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C3'))))
                        or exists(SELECT * FROM @resp_quests WHERE [quest_id] = 50 and
(dbo.sf_periodtrans_extend('C', [period]) = dbo.sf_fullperiodiflastpart_extend('C',
dbo.sf_previousperiod_extend_new('C', @period, '', 'C4'))))
)               BEGIN
        -- likums nr 1-30
                        if (isnull(@917_917_1110_1_G__0, 0) > 0 AND
(isnull(@917_781_1141_1_G_C1_0, 0) > 0 OR isnull(@917_781_1141_1_G_C2_0, 0) > 0 OR
isnull(@917_781_1141_1_G_C3_0, 0) > 0 OR isnull(@917_781_1141_1_G_C4_0, 0) > 0))
                                if NOT (isnull(@917_917_1111_1_G__0, 0) > 0)
                                BEGIN
                                        set @error_variable_1 =
round((isnull(@917_781_1141_1_G_C1_0, 0) + isnull(@917_781_1141_1_G_C2_0, 0) +
isnull(@917_781_1141_1_G_C3_0, 0) + isnull(@917_781_1141_1_G_C4_0, 0)) / 4, 3)
                                        set @error_variable_2 = null
                                        set @error_variable_3 = null
                                        if isnumeric(@error_variable_1) = 1
                                                set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                        if isnumeric(@error_variable_2) = 1
                                                set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                        if isnumeric(@error_variable_3) = 1
                                                set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                        INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                        VALUES (94250, @period, 'G', @resp_quest_id,
'Pārbaudīt vai 1111.1 tiešām nav datu.(V1 - "2-darbs" 1141 vidējā vērtība)',
@error_variable_1, @error_variable_2, @error_variable_3)
                                END
                -- likums nr 1-32
```

```
                  if ((isnull(@917_781_1141_1_G_C1_0, 0) +
isnull(@917_781_1141_1_G_C2_0, 0) + isnull(@917_781_1141_1_G_C3_0, 0) +
isnull(@917_781_1141_1_G_C4_0, 0)) > 0)
                        if NOT (isnull(@917_917_1111_1_G__0, 0) > 0)
                        BEGIN
                              set @error_variable_1 =
round((isnull(@917_781_1141_1_G_C1_0, 0) + isnull(@917_781_1141_1_G_C2_0, 0) +
isnull(@917_781_1141_1_G_C3_0, 0) + isnull(@917_781_1141_1_G_C4_0, 0)) / 4, 3)
                              set @error_variable_2 = null
                              set @error_variable_3 = null
                              if isnumeric(@error_variable_1) = 1
                                    set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                              if isnumeric(@error_variable_2) = 1
                                    set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                              if isnumeric(@error_variable_3) = 1
                                    set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                              INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                              VALUES (94253, @period, 'G', @resp_quest_id,
'Pārbaudīt vai 1111.1 tiešām nav datu.(V1 - "2-darbs" 1141.1 vidējā vērtība)',
@error_variable_1, @error_variable_2, @error_variable_3)
                        END
            END
      ELSE
      BEGIN
            set @avvoid_error = 1
            END
                        BEGIN
      -- likums nr 1-01
                  if NOT (isnull(@917_917_1110_1_G__0, 0) +
isnull(@917_917_1111_1_G__0, 0) + isnull(@917_917_1130_1_G__0, 0) +
isnull(@917_917_1210_1_G__0, 0) + isnull(@917_917_1310_1_G__0, 0) +
isnull(@917_917_1311_1_G__0, 0) + isnull(@917_917_1312_1_G__0, 0) > 0)
                        BEGIN
                              set @error_variable_1 = null
                              set @error_variable_2 = null
                              set @error_variable_3 = null
                              if isnumeric(@error_variable_1) = 1
                                    set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                              if isnumeric(@error_variable_2) = 1
                                    set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                              if isnumeric(@error_variable_3) = 1
                                    set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                              INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                              VALUES (94222, @period, 'G', @resp_quest_id, 'Nav
aizpildīta 1.iedaļa', @error_variable_1, @error_variable_2, @error_variable_3)
                        END
            -- likums nr 1-02
            if NOT (isnull(@917_917_1110_1_G__0, 0) = isnull(@917_917_1110_2_G__0,
0))
                        BEGIN
                              set @error_variable_1 =
round(isnull(@917_917_1110_1_G__0, 0), 3)
                              set @error_variable_2 =
round(isnull(@917_917_1110_2_G__0, 0), 3)
                              set @error_variable_3 = null
                              if isnumeric(@error_variable_1) = 1
                                    set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                              if isnumeric(@error_variable_2) = 1
                                    set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                              if isnumeric(@error_variable_3) = 1
```

```
                                    set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                INSERT INTO @errors (validation_id, row_col_id,
period, periodicity, resp_quest_id, message, error_value_1, error_value_2,
error_value_3)
                                VALUES (94223, 63544, @period, 'G',
@resp_quest_id, '1110.1.aile nav pārējo aiļu summa', @error_variable_1,
@error_variable_2, @error_variable_3)
                            END
            if NOT (isnull(@917_917_1111_1_G__0, 0) = isnull(@917_917_1111_2_G__0,
0))
                            BEGIN
                                set @error_variable_1 =
round(isnull(@917_917_1111_1_G__0, 0), 3)
                                set @error_variable_2 =
round(isnull(@917_917_1111_2_G__0, 0), 3)
                                set @error_variable_3 = null
                                if isnumeric(@error_variable_1) = 1
                                    set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_2) = 1
                                    set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_3) = 1
                                    set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                INSERT INTO @errors (validation_id, row_col_id,
period, periodicity, resp_quest_id, message, error_value_1, error_value_2,
error_value_3)
                                VALUES (94223, 63546, @period, 'G',
@resp_quest_id, '1111.1.aile nav pārējo aiļu summa', @error_variable_1,
@error_variable_2, @error_variable_3)
                            END
            if NOT (isnull(@917_917_1130_1_G__0, 0) = isnull(@917_917_1130_2_G__0,
0))
                            BEGIN
                                set @error_variable_1 =
round(isnull(@917_917_1130_1_G__0, 0), 3)
                                set @error_variable_2 =
round(isnull(@917_917_1130_2_G__0, 0), 3)
                                set @error_variable_3 = null
                                if isnumeric(@error_variable_1) = 1
                                    set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_2) = 1
                                    set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_3) = 1
                                    set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                INSERT INTO @errors (validation_id, row_col_id,
period, periodicity, resp_quest_id, message, error_value_1, error_value_2,
error_value_3)
                                VALUES (94223, 63547, @period, 'G',
@resp_quest_id, '1130.1.aile nav pārējo aiļu summa', @error_variable_1,
@error_variable_2, @error_variable_3)
                            END
            if NOT (isnull(@917_917_1210_1_G__0, 0) = isnull(@917_917_1210_2_G__0,
0))
                            BEGIN
                                set @error_variable_1 =
round(isnull(@917_917_1210_1_G__0, 0), 3)
                                set @error_variable_2 =
round(isnull(@917_917_1210_2_G__0, 0), 3)
                                set @error_variable_3 = null
                                if isnumeric(@error_variable_1) = 1
                                    set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_2) = 1
```

```
                                        set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_3) = 1
                                        set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                  INSERT INTO @errors (validation_id, row_col_id,
period, periodicity, resp_quest_id, message, error_value_1, error_value_2,
error_value_3)
                                  VALUES (94223, 63548, @period, 'G',
@resp_quest_id, '1210.1.aile nav pārējo aiļu summa', @error_variable_1,
@error_variable_2, @error_variable_3)
                            END
            if NOT (isnull(@917_917_1310_1_G__0, 0) = isnull(@917_917_1310_2_G__0,
0))
                            BEGIN
                                  set @error_variable_1 =
round(isnull(@917_917_1310_1_G__0, 0), 3)
                                  set @error_variable_2 =
round(isnull(@917_917_1310_2_G__0, 0), 3)
                                  set @error_variable_3 = null
                                  if isnumeric(@error_variable_1) = 1
                                        set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_2) = 1
                                        set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_3) = 1
                                        set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                  INSERT INTO @errors (validation_id, row_col_id,
period, periodicity, resp_quest_id, message, error_value_1, error_value_2,
error_value_3)
                                  VALUES (94223, 63549, @period, 'G',
@resp_quest_id, '1310.1.aile nav pārējo aiļu summa', @error_variable_1,
@error_variable_2, @error_variable_3)
                            END
            -- likums nr 1-03
                  if NOT (isnull(@917_917_1110_1_G__0, 0) > 0)
                            BEGIN
                                  set @error_variable_1 = null
                                  set @error_variable_2 = null
                                  set @error_variable_3 = null
                                  if isnumeric(@error_variable_1) = 1
                                        set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_2) = 1
                                        set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_3) = 1
                                        set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                  INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                  VALUES (94224, @period, 'G', @resp_quest_id,
'Jābūt datiem 1110,1', @error_variable_1, @error_variable_2, @error_variable_3)
                            END
            -- likums nr 1-05
                  if NOT (isnull(@917_917_1110_1_G__0, 0) >=
isnull(@917_917_1111_1_G__0, 0))
                            BEGIN
                                  set @error_variable_1 = null
                                  set @error_variable_2 = null
                                  set @error_variable_3 = null
                                  if isnumeric(@error_variable_1) = 1
                                        set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_2) = 1
                                        set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_3) = 1
```

```
                                        set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                    VALUES (94225, @period, 'G', @resp_quest_id,
'1111.1 rinda nevar būt lielāka par 1110.1 rindas aili', @error_variable_1,
@error_variable_2, @error_variable_3)
                             END
              -- likums nr 1-06
                    if (isnull(@917_917_1110_1_G__0, 0) >= 1)
                           if NOT (isnull(@917_917_1130_1_G__0, 0) > 0)
                           BEGIN
                                    set @error_variable_1 = null
                                    set @error_variable_2 = null
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
                                           set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                           set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
                                           set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                    VALUES (94227, @period, 'G', @resp_quest_id, 'Nav
datu 1130,1', @error_variable_1, @error_variable_2, @error_variable_3)
                             END
              -- likums nr 1-20
                    if (isnull(@917_917_1110_1_G__0, 0) > 0 AND
isnull(@917_917_1130_1_G__0, 0) > 0)
                             if NOT (isnull(@917_917_1130_1_G__0, 0) /
isnull(@917_917_1110_1_G__0, 0) / 12 >= 200)
                             BEGIN
                                    set @error_variable_1 =
round(isnull(@917_917_1130_1_G__0, 0) / isnull(@917_917_1110_1_G__0, 0) / 12, 3)
                                    set @error_variable_2 = null
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
                                           set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                           set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
                                           set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                    VALUES (94239, @period, 'G', @resp_quest_id,
'1130.r.uz 1 cilvēku mēnesī nav >=200Ls', @error_variable_1, @error_variable_2,
@error_variable_3)
                             END
              -- likums nr 1-21
                    if NOT (isnull(@917_917_1210_1_G__0, 0) > 0)
                             BEGIN
                                    set @error_variable_1 = null
                                    set @error_variable_2 = null
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
                                           set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                           set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
                                           set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
```

```
                                INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                VALUES (94240, @period, 'G', @resp_quest_id, 'Nav
datu 1210.1', @error_variable_1, @error_variable_2, @error_variable_3)
                          END
             -- likums nr 1-23
                    if NOT (isnull(@917_917_1310_1_G__0, 0) > 0)
                          BEGIN
                                set @error_variable_1 = null
                                set @error_variable_2 = null
                                set @error_variable_3 = null
                                if isnumeric(@error_variable_1) = 1
                                      set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_2) = 1
                                      set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_3) = 1
                                      set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                VALUES (94242, @period, 'G', @resp_quest_id, 'Nav
datu 1310.1', @error_variable_1, @error_variable_2, @error_variable_3)
                          END
             -- likums nr 1-25
                    if NOT (isnull(@917_917_1311_1_G__0, 0) > 0)
                          BEGIN
                                set @error_variable_1 = null
                                set @error_variable_2 = null
                                set @error_variable_3 = null
                                if isnumeric(@error_variable_1) = 1
                                      set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_2) = 1
                                      set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_3) = 1
                                      set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                VALUES (94244, @period, 'G', @resp_quest_id, 'Nav
datu 1311.1', @error_variable_1, @error_variable_2, @error_variable_3)
                          END
             -- likums nr 1-26
                    if NOT (isnull(@917_917_1311_1_G__0, 0) <
isnull(@917_917_1310_1_G__0, 0))
                          BEGIN
                                set @error_variable_1 = null
                                set @error_variable_2 = null
                                set @error_variable_3 = null
                                if isnumeric(@error_variable_1) = 1
                                      set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_2) = 1
                                      set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                if isnumeric(@error_variable_3) = 1
                                      set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                VALUES (94245, @period, 'G', @resp_quest_id,
'1311.1 nevar būt >= 1310.1', @error_variable_1, @error_variable_2,
@error_variable_3)
                          END
             -- likums nr 1-27
                    if NOT (isnull(@917_917_1312_1_G__0, 0) >= 1)
                          BEGIN
```

```
                                    set @error_variable_1 = null
                                    set @error_variable_2 = null
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
                                          set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                          set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
                                          set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                    VALUES (94246, @period, 'G', @resp_quest_id, 'Nav
datu 1312 ', @error_variable_1, @error_variable_2, @error_variable_3)
                              END
                     -- likums nr 1-28
                           if NOT (isnull(@917_917_1312_1_G__0, 0) <
isnull(@917_917_1311_1_G__0, 0))
                              BEGIN
                                    set @error_variable_1 = null
                                    set @error_variable_2 = null
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
                                          set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                          set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
                                          set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                    VALUES (94248, @period, 'G', @resp_quest_id, '1312
nevar būt >= 1311.1', @error_variable_1, @error_variable_2, @error_variable_3)
                              END
                     -- likums nr 1-29
                           if NOT (isnull(@917_917_1130_1_G__0, 0) <=
isnull(@917_917_1310_1_G__0, 0))
                              BEGIN
                                    set @error_variable_1 =
round(isnull(@917_917_1130_1_G__0, 0), 3)
                                    set @error_variable_2 =
round(isnull(@917_917_1310_1_G__0, 0), 3)
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
                                          set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                          set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
                                          set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                    VALUES (94249, @period, 'G', @resp_quest_id,
'1130.1 nevar būt > 1310.1', @error_variable_1, @error_variable_2,
@error_variable_3)
                              END
                     -- likums nr 1-31
                           if (isnull(@917_917_1110_1_G__0, 0) > 0)
                              if NOT (isnull(@917_917_1111_1_G__0, 0) > 0)
                              BEGIN
                                    set @error_variable_1 = null
                                    set @error_variable_2 = null
                                    set @error_variable_3 = null
                                    if isnumeric(@error_variable_1) = 1
```

```
                                        set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_2) = 1
                                        set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_3) = 1
                                        set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                  INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                  VALUES (94252, @period, 'G', @resp_quest_id,
'Pārbaudīt vai 1111.1 tiešām nav datu. ', @error_variable_1, @error_variable_2,
@error_variable_3)
                            END
                -- likums nr 2-01
                      if NOT (isnull(@917_917_600_4_G__0, 0) =
isnull(@917_917_600_1_G__0, 0) + isnull(@917_917_600_2_G__0, 0) -
isnull(@917_917_600_3_G__0, 0))
                            BEGIN
                                  set @error_variable_1 =
round(isnull(@917_917_600_4_G__0, 0), 3)
                                  set @error_variable_2 =
round(isnull(@917_917_600_1_G__0, 0) + isnull(@917_917_600_2_G__0, 0) -
isnull(@917_917_600_3_G__0, 0), 3)
                                  set @error_variable_3 = null
                                  if isnumeric(@error_variable_1) = 1
                                        set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_2) = 1
                                        set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_3) = 1
                                        set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                  INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                  VALUES (94254, @period, 'G', @resp_quest_id,
'600.4 jābūt = 600.1 + 600.2 - 600.3', @error_variable_1, @error_variable_2,
@error_variable_3)
                            END
                -- likums nr 2-02
                      if (isnull(@917_917_1110_1_G__0, 0) <> 0)
                            if NOT (isnull(@917_917_600_1_G__0, 0) +
isnull(@917_917_600_2_G__0, 0) + isnull(@917_917_600_3_G__0, 0) +
isnull(@917_917_600_4_G__0, 0) <> 0)
                            BEGIN
                                  set @error_variable_1 = null
                                  set @error_variable_2 = null
                                  set @error_variable_3 = null
                                  if isnumeric(@error_variable_1) = 1
                                        set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_2) = 1
                                        set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                  if isnumeric(@error_variable_3) = 1
                                        set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                  INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                  VALUES (94255, @period, 'G', @resp_quest_id,
'Jābūt datiem 600.rindā', @error_variable_1, @error_variable_2, @error_variable_3)
                            END
                -- likums nr Laiks
                      if NOT (isnull(@917_917_Laiks_1_G__0, 0) > 0 OR
isnull(@917_917_Laiks_2_G__0, 0) > 0)
                            BEGIN
                                  set @error_variable_1 = null
                                  set @error_variable_2 = null
                                  set @error_variable_3 = null
```

```
                                    if isnumeric(@error_variable_1) = 1
                                          set @error_variable_1 =
round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_2) = 1
                                          set @error_variable_2 =
round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                    if isnumeric(@error_variable_3) = 1
                                          set @error_variable_3 =
round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                    INSERT INTO @errors (validation_id, period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
                                    VALUES (94256, @period, 'G', @resp_quest_id,
'Lūdzu norādiet pāskata aizpildīšanai patērēto laiku', @error_variable_1,
@error_variable_2, @error_variable_3)
                              END
             END
                    DECLARE Records CURSOR
             LOCAL FAST_FORWARD READ_ONLY
             FOR SELECT distinct [record_id], [code]
                    FROM [mi_resp_codes]
                    WHERE [resp_quest_id] = @resp_quest_id and
                           [table_id] = 4253

             OPEN Records

             FETCH NEXT FROM Records INTO @record_id, @resp_code
             While (@@FETCH_STATUS = 0)
                    BEGIN
                           SELECT
             @917_917_1110_2_G__0 = sum(case when V.[variable_id] = 79454 and
(dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
                    ,      @917_917_1111_2_G__0 = sum(case when V.[variable_id] =
79455 and (dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
                    ,      @917_917_1130_2_G__0 = sum(case when V.[variable_id] =
79453 and (dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
                    ,      @917_917_1210_2_G__0 = sum(case when V.[variable_id] =
79457 and (dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
                    ,      @917_917_1310_2_G__0 = sum(case when V.[variable_id] =
79458 and (dbo.sf_periodtrans_extend('G', V.[period]) =
dbo.sf_fullperiodiflastpart_extend('G', @period)) then V.[cvalue] else null end)
                    FROM [MicroDataBase].[dbo].[mi_values_61] as V WITH( INDEX(
IX_resp_var_per_rec_mark))
                    LEFT JOIN [MetaDataBase].[dbo].[me_attrib_classif_codes] as C
ON (V.[record_dim_id] = C.[record_id])
             WHERE V.[resp_quest_id] = @resp_quest_id  and
                    V.[record_id] = @record_id
             BEGIN
             -- likums nr 1-05
                                    if NOT (isnull(@917_917_1110_2_G__0, 0) >=
isnull(@917_917_1111_2_G__0, 0))
             BEGIN
                                                                  set @error_variable_1 =
null
                                                                  set @error_variable_2 =
null
                                                                  set @error_variable_3 =
null
                                                                  if
isnumeric(@error_variable_1) = 1
                                                                        set
@error_variable_1 = round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                                                  if
isnumeric(@error_variable_2) = 1
                                                                        set
@error_variable_2 = round(cast(@error_variable_2 as decimal(20, 6)), 3)
```

```
                                                        if
isnumeric(@error_variable_3) = 1
                                                            set
@error_variable_3 = round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                                        set @message = ''
                                                        set @message = '#' +
@resp_code + ' - 1111 rinda nevar būt lielāka par 1110 rindas aili'
                                                        INSERT INTO @errors
(validation_id, record_id, period, periodicity, resp_quest_id, message,
error_value_1, error_value_2, error_value_3)
                                                        VALUES (94226,
@record_id, @period, 'G', @resp_quest_id, @message, @error_variable_1,
@error_variable_2, @error_variable_3)
                                                    END
            -- likums nr 1-06
                                if (isnull(@917_917_1110_2_G__0, 0) >= 1)
                                       if NOT (isnull(@917_917_1130_2_G__0, 0) >
0)
            BEGIN
                                                        set @error_variable_1 =
null
                                                        set @error_variable_2 =
null
                                                        set @error_variable_3 =
null
                                                        if
isnumeric(@error_variable_1) = 1
                                                            set
@error_variable_1 = round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                                        if
isnumeric(@error_variable_2) = 1
                                                            set
@error_variable_2 = round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                                        if
isnumeric(@error_variable_3) = 1
                                                            set
@error_variable_3 = round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                                        set @message = ''
                                                        set @message = '#' +
@resp_code + ' - Nav datu 1130,2'
                                                        INSERT INTO @errors
(validation_id, record_id, period, periodicity, resp_quest_id, message,
error_value_1, error_value_2, error_value_3)
                                                        VALUES (94228,
@record_id, @period, 'G', @resp_quest_id, @message, @error_variable_1,
@error_variable_2, @error_variable_3)
                                                    END
            -- likums nr 1-22
                                if NOT (isnull(@917_917_1210_2_G__0, 0) > 0)
            BEGIN
                                                        set @error_variable_1 =
null
                                                        set @error_variable_2 =
null
                                                        set @error_variable_3 =
null
                                                        if
isnumeric(@error_variable_1) = 1
                                                            set
@error_variable_1 = round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                                        if
isnumeric(@error_variable_2) = 1
                                                            set
@error_variable_2 = round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                                        if
isnumeric(@error_variable_3) = 1
                                                            set
@error_variable_3 = round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                                        set @message = ''
```

```
                                                        set @message = '#' +
@resp_code + ' - Nav datu 1210'
                                                        INSERT INTO @errors
(validation_id, record_id, period, periodicity, resp_quest_id, message,
error_value_1, error_value_2, error_value_3)
                                                        VALUES (94241,
@record_id, @period, 'G', @resp_quest_id, @message, @error_variable_1,
@error_variable_2, @error_variable_3)
                                                    END
            -- likums nr 1-24
                                if NOT (isnull(@917_917_1310_2_G__0, 0) > 0)
            BEGIN
                                                        set @error_variable_1 =
null
                                                        set @error_variable_2 =
null
                                                        set @error_variable_3 =
null
                                                        if
isnumeric(@error_variable_1) = 1
                                                            set
@error_variable_1 = round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                                        if
isnumeric(@error_variable_2) = 1
                                                            set
@error_variable_2 = round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                                        if
isnumeric(@error_variable_3) = 1
                                                            set
@error_variable_3 = round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                                        set @message = ''
                                                        set @message = '#' +
@resp_code + ' - Nav datu 1310'
                                                        INSERT INTO @errors
(validation_id, record_id, period, periodicity, resp_quest_id, message,
error_value_1, error_value_2, error_value_3)
                                                        VALUES (94243,
@record_id, @period, 'G', @resp_quest_id, @message, @error_variable_1,
@error_variable_2, @error_variable_3)
                                                    END
            -- likums nr 1-31
                                if (isnull(@917_917_1110_2_G__0, 0) > 0)
                                       if NOT (isnull(@917_917_1111_2_G__0, 0) >
0)
            BEGIN
                                                        set @error_variable_1 =
null
                                                        set @error_variable_2 =
null
                                                        set @error_variable_3 =
null
                                                        if
isnumeric(@error_variable_1) = 1
                                                            set
@error_variable_1 = round(cast(@error_variable_1 as decimal(20, 6)), 3)
                                                        if
isnumeric(@error_variable_2) = 1
                                                            set
@error_variable_2 = round(cast(@error_variable_2 as decimal(20, 6)), 3)
                                                        if
isnumeric(@error_variable_3) = 1
                                                            set
@error_variable_3 = round(cast(@error_variable_3 as decimal(20, 6)), 3)
                                                        set @message = ''
                                                        set @message = '#' +
@resp_code + ' - Pārbaudīt vai 1111 tiešām nav datu. '
                                                        INSERT INTO @errors
(validation_id, record_id, period, periodicity, resp_quest_id, message,
error_value_1, error_value_2, error_value_3)
```

```
                                                                VALUES (94251,
@record_id, @period, 'G', @resp_quest_id, @message, @error_variable_1,
@error_variable_2, @error_variable_3)
                                                        END
            END
                                FETCH NEXT FROM Records INTO @record_id,
@resp_code
                  END
            CLOSE Records
            DEALLOCATE Records


      END

--[R, K] EXISTS [R, K] validācijas

      declare @exist_resp_quest_id1 varchar(30)
      declare @exist_resp_quest_id2 varchar(30)

-- klasifikatoru valid?cija

      INSERT INTO @errors (validation_id, [row_col_id], [record_id], period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
      SELECT 94220, v.variable_id, v.record_id, @period, 'G', @resp_quest_id,
            'Klasifikatora v?rt?ba: - ' + clf.code + ' nav aktu?la. Ieraksta
der?guma termi?? ir l?dz ' + convert(varchar, CLF.valid_to,104),
            case when S.is_variable_row = 1 Then CR.code else S.row_code end,
            case when S.is_variable_col = 1 Then CR.code else S.col_code end,
            CLF.code
      FROM mi_values_61 as V
            JOIN mi_resp_quests as RQ ON (V.resp_quest_id = RQ.resp_quest_id)
            JOIN mi_inst_struct as S ON (RQ.instance_id = S.inst_id and
V.variable_id = S.variable_id)
            JOIN Metadatabase..me_attrib_classif_codes as CLF ON (CLF.record_id =
V.record_dim_id)
            LEFT JOIN Metadatabase..me_attrib_classif_codes as CR ON (CR.record_id
= V.record_id)
      WHERE V.resp_quest_id = @resp_quest_id
            and s.is_classif_cell = 1
            and (Metadatabase.dbo.mesfGetPeriodStartDate(@period)>clf.valid_to
                  OR
                  Metadatabase.dbo.mesfGetPeriodEndDate(@period)<clf.valid_from)

      INSERT INTO @errors (validation_id, [row_col_id], [record_id], period,
periodicity, resp_quest_id, message, error_value_1, error_value_2, error_value_3)
      SELECT 94220, v.variable_id, v.record_id, @period, 'G', @resp_quest_id,
            'Vērtība: - ' + isnull(v.tvalue,'') + ' neatbilst klasifikatoram!',
            case when S.is_variable_row = 1 Then CR.code else S.row_code end,
            case when S.is_variable_col = 1 Then CR.code else S.col_code end,
            isnull(v.tvalue,'')
      FROM mi_values_61 as V
            JOIN mi_resp_quests as RQ ON (V.resp_quest_id = RQ.resp_quest_id)
            JOIN mi_inst_struct as S ON (RQ.instance_id = S.inst_id and
V.variable_id = S.variable_id)
            LEFT JOIN Metadatabase..me_attrib_classif_codes as CR ON (CR.record_id
= V.record_id)
      WHERE V.resp_quest_id = @resp_quest_id
            and s.is_classif_cell = 1
            and v.record_dim_id is null



-- izmet likumus, kuri izdz?sti, vai nav iek?auti proced?r?
      DELETE FROM @errors
      WHERE (validation_id) not in (SELECT V.validation_id
            FROM @errors as E,
                  @validations as V
            WHERE E.validation_id = V.validation_id)

-- izmet likumus, kuriem ir iz??mums dotaj? period?
```

```
	DELETE FROM @errors
	WHERE (validation_id) in (SELECT X.validation_id
		FROM [MetaDataBase].[dbo].[me_valid_exceptions] as X,
			[MetaDataBase].[dbo].[me_validations] as V,
			@errors as E
		WHERE X.validation_id = E.validation_id and
			V.validation_id = E.validation_id and
			X.period = SUBSTRING(dbo.sf_fullperiodiflastpart(V.periodicity,
@period), 6, 20))

-- izmet likumus, kuri attiecas uz nor?d?tajiem respondentiem
	DELETE FROM @errors
	WHERE (validation_id) in (SELECT X.validation_id
		FROM [MicroDataBase].[dbo].[mi_valid_exceptions] as X,
			@validations as V,
			@errors as E
		WHERE V.validation_id = E.validation_id and
			V.ex_flag = 1 and
			X.validation_id = E.validation_id and
			X.[code] = @RESPCODE)

-- izmet likumus, kuri attiecas uz p?r?jiem respondentiem
	DELETE FROM @errors
	WHERE (validation_id) in (SELECT X.validation_id
	FROM [MicroDataBase].[dbo].[mi_valid_exceptions] as X,
		@validations as V,
		@errors as E
	WHERE V.validation_id = E.validation_id and
		V.ex_flag = 0 and
		X.validation_id = E.validation_id and
		X.[code] <> @RESPCODE and
		V.validation_id not in (SELECT X.validation_id
			FROM [MicroDataBase].[dbo].[mi_valid_exceptions] as X,
				@validations as V,
				@errors as E
			WHERE V.validation_id = E.validation_id and
				V.ex_flag = 0 and
				X.validation_id = E.validation_id and
				X.[code] = @RESPCODE))

	INSERT INTO [MicroDataBase].[dbo].[mi_valid_errors] ([validation_id],
[row_col_id], [record_id], [resp_quest_id], [message], [error_value], [priority],
[error_value_2], [error_value_3])
	SELECT E.[validation_id], isnull(E.[row_col_id],0), isnull(E.[record_id],0),
@resp_quest_id, E.[message], E.[error_value_1], V.[priority], E.[error_value_2],
E.[error_value_3]
	FROM @errors as E,
		@validations as V
	WHERE V.validation_id = E.validation_id

	if EXISTS(SELECT * FROM MicroDataBase.dbo.sysobjects WHERE
[name]='mi_validation_917_2013_x' and [xtype] = 'P')
		BEGIN TRY
			EXEC [MicroDataBase].[dbo].[mi_validation_917_2013_x] @period,
@resp_quest_id
		END TRY
		BEGIN CATCH
			DECLARE @errormessage NVARCHAR(4000);
			DECLARE @errorseverity INT;
			DECLARE @errorstate INT;

			SELECT
				@errormessage = ERROR_MESSAGE(),
				@errorseverity = ERROR_SEVERITY(),
				@errorstate = ERROR_STATE();

		Set @errormessage = 'CSP speciālistu izveidotās validācijas
papildprocedūras kļūda: ' + @errormessage
		RAISERROR (@errormessage,
				@errorseverity,
```

```
                                        @errorstate
                                        );
            END CATCH
      DELETE FROM @resp_quests
      DELETE FROM @resp_codes
      DELETE FROM @errors
      UPDATE [MicroDataBase].[dbo].[mi_resp_quests]
      SET [state] = @state - (@state % 10)
      WHERE [resp_quest_id] = @resp_quest_id

Naakamais:

      FETCH NEXT FROM RespList INTO @resp_quest_id, @state
END
CLOSE RespList
DEALLOCATE RespList

if @arithignore = 1
      Set arithignore on
if @arithabort = 0
      Set arithabort off
if @ansi_warnings = 1
      Set ansi_warnings on

set @retval = 1
```

Bakalaura darbs „Metadatu vadīta statistisko datu apstrādes sistēma” izstrādāts LU Datorikas fakultātē.

Ar savu parakstu apliecinu, ka darbs izstrādāts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: **Mareks Indāns**

Rekomendēju darbu aizstāvēšanai

Vadītājs: profesors Dr. dat. **Ģirts Karnītis** __.__.2015.

Recenzente: docente Dr. dat. **Darja Solodovņikova**

Darbs iesniegts __.__.2015.

Dekāna pilnvarotā persona:

Darbs aizstāvēts bakalaura gala pārbaudījuma komisijas sēdē

__.__.2015. prot. Nr. _____

Komisijas sekretārs (-e): _____