**1.slaids**

Labdien! Mani sauc Lauma Svilpe un mana bakalaura darba temats ir neviendabīgu integrētu datu avotu evolūcijas apstrāde. Darba vadītāja: asociētā profesore datorzinātņu doktore Darja Solodovņikova.

**2.slaids**

Sākumam vēlos ieskicēt esošo situāciju saistībā ar datu glabāšanu.

⓿Kamēr 2000.gadā visā pasaulē glabāto datu apjoms bija mērāms vien simtos petabaitu,

⓿2020.gadā tiek prognozēts, ka šis skaitlis sasniegs jau ap 35 zetabaitiem, kas šo 20 gadu laikā apstiprina eksponenciālu datu apjoma pieagumu.

⓿Pēdējo gadu laikā ieviesti vairāki jauni rīki, tehnoloģijas un ietvari, kas atbalsta šāda liela apjoma datu analītiku, tomēr tie galvenokārt risina tikai problēmas, kas saistītas ar lielo datu apjomu – vaicājumu izpildes ātrums, datu atlases ērtums, vienkāršība u.c.

Neatrisināts paliek datu un to struktūru evolūcijas jautājums.

**3.slaids**

Balstoties uz pastāvošo problēmu, tiek izvirzīts bakalaura darba mērķis - atrast risinājumu, kā apstrādāt neviendabīgu integrētu datu avotu evolūcijas rezultātā radušās izmaiņas.

Lai sasniegtu izvirzīto mērķi,

⓿ bija nepieciešams veikt literatūras analīze par datu noliktavām, lielajiem datiem un ETL procesiem

⓿ izpētīt esošā Latvijas Universitātes Datorikas fakultātē izstrādātā datu avotu evolūcijas sistēma

⓿ izstrādāt neviendabīgu integrētu datu avotu evolūcijas apstrādes mehānisms, kas paredzēts iekļaušanai esošajā sistēmā.

**4.slaids**

Pašos pirmsākumos datu glabāšanai tika izmantoti ļoti dārgi un ierobežotas ietilpības mehānismi –

⓿perfokartes, magnētiskās lentas,

⓿pēc tam ar pavisam jauniem papildinājumiem nāca klajā diskatmiņa.

⓿Drīz vien diskatmiņas tika papildinātas ar datu bāzu pārvaldības sistēmām, kuru galvenais izmantošanas ieguvums bija ļoti ātra datu ievietošana sistēmā

Līdz ar datu glabāšanas veidu attīstību, tika ieviests jēdziens “lēmumu atbalsta sistēmas”, kas apzīmē dažādu metožu un rīku kopumu, kas paredzēts stratēģisku lēmumu pieņemšanas atbalstam, iekļaujot tajā datu analīzi un apstrādi.

⓿Līdz ar šī jēdziena parādīšanos, klajā nāca dažādi OLAP jeb tiešsaistes analītiskās apstrādes rīki,

⓿kas vēlāk tika iekļauti lietojumsistēmās.

⓿Vēlāk tika ieviestas arī 4GL tehnoloģijas, kuru ideja bija padarīt programmatūras izstrādi tik vienkāršu, lai to varētu darīt jebkurš.

Kā jau noprotams, šeit nevar iezīmēt nekādu konkrētu struktūru –

⓿radās nekārtība, ko mēdz saukt par “zirnekļa tīkla” vidi. Šāds “zirnekļa tīkls” vairākās sistēmās mēdza izaugt neiedomājami sarežģīts, kā rezultātā to vairs nebija iespējams ne pienācīgi uzturēt, kur nu vēl pilnveidot. Tai ar laiku pat radies nosaukums “dabiski attīstīta arhitektūra”, jo bija maz iespēju tās attīstību ietekmēt.

**5.slaids**

Datu noliktavas būtiski izmainīja IT speciālistu domāšanu – līdz šim pastāvēja uzskats, ka datu bāze ir paradzēta, lai glabātu jebkādam nolūkam paredzētus datus. Tomēr līdz ar datu noliktavas jēdziena parādīšanos kļuva acīmredzams, ka ir nepieciešamas dažādu veidu datu bāzes,

⓿tādēļ “Zirnekļa tīkla” vide tika sadalīta divās atsevišķās daļās, lai glabātu datus pēc to nozīmes datu analītikas kontekstā.

**6.slaids**

Datu noliktavu kontekstā svarīgs jēdziens ir arī lielie dati, jo visbiežāk to glabāšanai un analīzei tiek izmantotas tieši datu noliktavas.

⓿Lielos datus raksturo to apjoms – tie var būt terabaitus lieli datu vaili, dažādi ieraksti, datu bāzes tabulas. Ātrums, kādā dati ienāk noliktavā – sērijveidā, reāllaikā vai kā nepārtraukta datu straume. Kā arī datu dažādība – tie sākotnēji ir strukturēti, nestrukturēti vai jaukti dati.

**7.slaids**

Lai neapstrādātus lielos datus ievietotu datu noliktavās, tiek izmantoti ETL procesi. Atšifrējums latviešu valodā saīsinājumam ETL nozīmē – iegūšana, transformācija, ielādēšana.

⓿ Datu iegūšanas procesā tie tiek savākti no dažādiem ārējiem datu avotiem,

⓿kur tie tiek validēti, attīrīti un filtrēti.

⓿Transformācijas procesā dažādu struktūru dati tiek pārveidoti vienā vienotā struktūrā.

⓿Kad tas izdarīts, datus var ielādēt datu noliktavā.

**8.slaids**

Latvijas Universitātes Datorikas fakultātē piedāvāts datu noliktavas risinājums lielo datu analīzei, kurā iekļauti algoritmi dažādu evolūcijas rezultātā radušos izmaiņu atklāšanai. Risinājumā iekļauta lielo datu analīzei paredzēta arhitektūra, kas ļauj izpildīt OLAP operācijas un cita veida analīzi datu noliktavā integrētiem datu avotiem, kā arī spēj atklāt dažādas izmaiņas strukturētos, daļēji strukturētos un nestrukturētos datos, balstoties uz tos aprakstošajiem metadatiem.

Datu avotu evolūcijas sistēmas arhitektūras pamatkomponentes ir datu avoti, datu maģistrāle, metadatu glabātuve un adaptācijas komponente.

⓿Datu avotu līmenī neviendabīgi jeb dažādu formātu dati tiek iegūti no dažādiem avotiem un ielādēti sistēmā tālākai to apstrādei.

⓿Datu maģistrāle sastāv no vairākiem līmeņiem. Pirmajā līmenī tiek glabāti neapstrādāti dati *Raw* formātā, kas iegūti pa tiešo no datu avotiem. Katra nākamā maģistrāles līmeņa dati tiek iegūti no iepriekšējā līmeņa ar ETL procesu palīdzību.

⓿Sistēmas arhitektūras darbība pamatā balstīta uz metadatu glabātuvē esošajiem datiem. Izmantojot metadatu pārvaldības rīku un definējot dažādus metadatus, izstrādātājs nosaka, kā darbosies sistēma.

⓿Lielo datu noliktavas pamatelements, kas atbild par datu avotu un informācijas prasību izmaiņu apstrādi, ir adaptācijas komponente. Tās galvenais uzdevums ir atpazīt notikušās vai ģenerēt potenciālās izmaiņas un pēc tam ļaut izstrādātājam izvēlēties vispiemērotākās izmaiņas, kas jāievieš.

Bakalaura darba ietvaros adaptācijas komponente papildināta ar mehānismu, kas datu avotu evolūcijas rezultātā radušās izmaiņas adaptē sistēmas metadatos.

**9.slaids**

Lai praktiski pierādītu izstrādātās lielo datu analīzes sistēmas arhitektūras metadatu modeļa darbību, tā izmantota publikāciju lielo datu sistēmā. Publikāciju sistēmas mērķis ir no vairākiem neviendabīgiem datu avotiem integrēt datus par Latvijas Universitātes darbinieku un studentu publikācijām un nodrošināt šo datu analīzi datu noliktavā.

⓿Strukturēti dati tiek iegūti no Latvijas Universitātes informācijas sistēmas *LUIS*, taču daļēji strukturēti dati - no bibliotēkas datu pārvaldības sistēmas *Aleph*, kā arī citātu indeksācijas sistēmām *Scopus* un *Web of Science*.

⓿Sistēmas arhitektūra sastāv no trīs datu maģistrāles līmeņiem, kur pēdējā no tiem dati tiek glabāti datu noliktavā, kas realizēta, balstoties uz *Apache Hive* tehnoloģiju. Pārejas starp datu maģistrāles līmeņiem nodrošina kartēšanas metadatos aprakstītie ETL procesi.

**10.slaids**

Datu avotu evolūcijas rezultātā iespējamas izmaiņas datu avotos. Lai būtu iespējams turpināt datu ielasīšanu datu noliktavā no evolucionējušiem avotiem, nepieciešams veikt izmaiņu adaptāciju. Daži no bakalaura darba ietvaros izstrādātajā mehānismā iekļautie izmaiņu veidi kā konkrēti publikāciju lielo datu sistēmas piemēri aprakstīti ekrānā redzamajā tabulā.

|  |  |
| --- | --- |
| **Izmaiņa** | **Izmaiņas apstrāde** |
| Datu kopai *Scopus\_metrics* pievienota datu vienība *citeScoreYearInfoList.* | Jaunā datu vienība (XML elements) sastāv no vairākiem apakšelementiem, kuri neeksistēja arī iepriekšējās datu kopās, tomēr metadatos tiek izveidota tikai viena izmaiņa, kas attiecināta tikai uz augšējo apakšelementu. |
| No datu kopas *Scopus\_metrics* izdzēsta datu vienība *IPP.* | Šī izmaiņa ietekmē datu ielādēšanas procesu. Tā kā izdzēsto datu vienību nav iespējams aizstāt ar nevienu no vienībām, kas atrodas citos datu avotos, metadatos tiek reģistrēta kartēšanas dzēšana. |
| Pievienots jauns datu avots *DSpace*. | Saskaņā ar jaunajām analīzes prasībām, sistēma ir jāpapildina ar datiem, kas satur pirmsdrukas vai publicētus darbus pilnos tekstus. Jaunais datu avots satur nestrukturētus datus (pilna teksta failus) un ar tiem saistītos metadatus kā birkas (angļu val. - “*tags*” [1]). |
| Atjaunināta datu kopas *Scopus\_metrics* metadatu īpašības *API request* vērtība. | Šī izmaiņa tiek atklāta datu iegūšanas laikā no *API.* Šādu izmaiņu nepieciešams apstrādāt manuāli, jo nav iespējams automātiski izmainīt *API* pieprasījumu, balstoties uz jauno metadatu īpašību. |

**11.slaids**

Lai apstrādātu datu avotu evolūcijas rezultātā radušās izmaiņas un veiksmīgi adaptētu tās sistēmas metadatos, bakalaura darba ietvaros izstrādāts mehānisms, kurš darbojas, balstoties uz dažādiem metadatiem.

Tāpat kā jau esošās sistēmas metadatu glabāšanas mehānisms, arī evolūcijas apstrādes mehānisms izstrādāts, izmantojot Oracle SQL relāciju datu bāzi. Darba gaitā izstrādāti adaptācijas scenāriji tikai reāli publikāciju lielo datu sistēmā notikušajiem izmaiņu veidiem. Izstrādātā adaptācijas funkcionalitāte pārbaudīta uz publikāciju lielo datu sistēmas metadatiem.

⓿Izmaiņu adaptācijas operācijas ir darbības jeb soļi, kas jāveic, lai adaptētu izmaiņas sistēmā. Tās ir iespējami īsas un universālas. Katra no darbībām tiek glabāta kā ieraksts tabulā ***ChangeAdaptationOperation****.* Darbībai ir tips, kas norāda uz to, vai tā ir manuāli veicama vai automātiski izpildāma. Vairākas dažādu tipu izmaiņu adaptāciju veikšanai izmantotās darbības pārklājas, tāpēc tās tiek glabātas atsevišķā tabulā, lai novērstu informācijas dublēšanos datu bāzē.

⓿Izmaiņu adaptācijas scenārijs ir secīgu darbību virkne, kas tiek veikta, lai veiksmīgi adaptētu izmaiņu sistēmā. Katram no izmaiņu veidiem ir atšķirīgs izmaiņu adaptācijas scenārijs. Katra scenārija soļi glabājas tabulā ***ChangeAdaptationScenario****,* kas tiek aizpildīta manuāli pirms izmaiņas notikšanas.

⓿Lai sekotu līdzi izmaiņu adaptācijas procesam un glabātu informāciju par katras darbības izpildi, nepieciešama tabula, kurā glabājas katrai reāli notikušai izmaiņai atbilstošais adaptācijas scenārijs. Tas tiek glabāts tabulā **ChangeAdaptationProcess**.

⓿Neskatoties uz to, ka izmaiņas tips ir nosakāms jau pie tās rašanās, tas negarantē viennozīmīga izmaiņas adaptācijas scenārija esamību. Ir dažādi nosacījumi, kuru izpildes rezultātā scenārijs var zaroties. Tie var būt gan automātiski izpildāmi, gan manuāli apstrādājami. Šie nosacījumi tiek glabāti tabulā **ChangeAdaptationCondition**.

⓿Lai nodrošinātu konkrēto nosacījumu izpildi un atbilstību katram scenārija solim, paredzētas tabulas **CA\_ManualCOnditionFulfillment** un **CA\_ConditionMapping.**

⓿Izmaiņu adaptācijas procesa laikā no izstrādātāja var tikt prasīti dažādi papildus dati, kas būs nepieciešami tālāko scenāriju darbību izpildei un nosacījumu pārbaudei. Šādi dati tiek glabāti tabulā **ChangeAdaptionAdditionalData**.

⓿Izstrādātais risinājums ar esošo sistēmu saistīts pamatā izmantojot tabulu **Change**, kas glabā identifikatoru uz konkrēto izmaiņu, kā arī **Tipu** un **Autoru** klasifikatorus.

**12.slaids**

Veicot pirmreizējo izmaiņas apstrādi (skat. att. 3.2.), tiek apstrādātas tikai izmaiņas ar statusu “Jauns”. Nosakot izmaiņas tipu (izmaiņu tipus skatīt nodaļā 3.3.), pareizajā secībā tiek atlasīti visi izmaiņas tipam atbilstošie scenārija soļi, pēc kā izveidoti konkrētajai izmaiņai atbilstošie procesa soļi. Pēc tam tiek atlasīti konkrētajam procesa solim atbilstošie scenārija soļu manuālie nosacījumi un saglabāti ar statusu “Nav izpildīts”. Līdz ar konkrētās izmaiņas procesu izveidošanu, izmaiņas statuss tiek nomainīts uz “Procesā”, līdz ar to šīs izmaiņas adaptāciju var uzskatīt par sākušos.

**13.slaids**

Katras izmaiņas adaptācijai sistēmā atbilst scenārijs jeb veicamo darbību kopums, kas nepieciešams izmaiņas veiksmīgai apstrādei. Darba ietvaros netiek apskatīti visu iespējamo izmaiņu apstrādes scenāriji – aprakstīti septiņi reāli publikāciju lielo datu sistēmā notikušo izmaiņu adaptācijas scenāriji. Katra izstrādātā scenārija mērķis ir panākt, lai tas būtu pēc iespējas automātiski izpildāms (lai būtu nepieciešama pēc iespējas mazāka izstrādātāja iejaukšanās izmaiņu adaptācijas procesā), tomēr konceptuālu lēmumu pieņemšana par datu izmantošanu sistēmā nav iespējama bez cilvēka iejaukšanās.

**14.slaids**

A screenshot of a cell phone

Description automatically generated

**+15.slaids**

A close up of a map

Description automatically generated

**16.slaids**

Darba gaitā izpildīti visi sākotnēji izvirzītie darba uzdevumi. Vispirms izanalizēta pieejamā teorētiskā literatūra par datu noliktavām, to arhitektūru un veidiem, kā arī tajās izmantotajiem ETL procesiem. Datu noliktavu pielietojuma kontekstā izpētīta un apkopota informācija par lielajiem datiem un to svarīgākajām raksturiezīmēm.

Datu avotu evolūcijas kontekstā pētīta un analizēta esoša sistēma, kurā iekļauti algoritmi dažādu evolūcijas rezultātā radušos izmaiņu atklāšanai. Darbā aprakstīta sistēmas arhitektūra, īpašu uzmanību pievēršot tajā iekļautajai adaptācijas komponentei, kas izvirzīta par pamatu neviendabīgu integrētu datu avotu evolūcijas apstrādes mehānisma izstrādei.

Balstoties uz esošās sistēmas datu bāzes struktūras, kā arī pastāvošo metadatu glabāšanas mehānismu, izstrādāts risinājums, kas paredzēts datu avotu evolūcijas rezultātā radušos izmaiņu adaptācijai sistēmas metadatos. Izstrādāta datu bāzes struktūra izmaiņu adaptācijas procesā radušos metadatu glabāšanai, kā arī datu atlases vaicājumi un procedūras pašas izmaiņu adaptācijas veikšanai. Reāli notikušām datu avotu izmaiņām sastādīti pilni izmaiņu adaptācijas scenāriji, kas sastāv gan no automātiski izpildāmas funkcionalitātes (procedūru formā), gan manuāli veicamiem norādījumiem (veicamās darbības apraksta formā). Izstrādātais risinājums ir universāls, tāpēc tas izmantojams arī nākotnē jaunu izmaiņu veidu adaptācijas scenāriju glabāšanai un adaptēšanai sistēmas metadatos.

**17.slaids**

⓿Datu noliktavu jēdziens attīstījies līdz ar lielu datu plūsmu parādīšanos tehnoloģiju vidē. Šie dati tika uzkrāti, bet arvien vairāk tika novērotas dažādas "zirnekļa tīkla" vides, kuras bija grūti pārvaldīt. Lai strukturizētu iegūto informāciju, stratēģisku lēmumu pieņemšanai nepieciešamā informācija tika glabāta atsevišķā datu noliktavā, kuras aizpildīšanai izmantoti ETL procesi. Drīz vien, līdz ar esošo iespēju efektīvi un ātri uzkrāt un pārvaldīt datus, tika ieviests lielo datu jēdziens. Taču, ņemot vērā lielo datu apjomu, dažādību un ātrumu, izkristalizējušās neviendabīgu integrētu datu avotu evolūcijas problēmas.

⓿Bakalaura darba ietvaros izpētīta esoša datu avotu evolūcijas sistēma. Šī sistēma iekļauj datu ieguvi no dažādiem avotiem, kā arī ETL procesus datu pārveidošanai vienotā struktūrā, lai tos būtu iespējams ievietot datu noliktavā. Sistēmas darbība balstīta uz metadatiem, kas tiek glabāti par katru saņemto informācijas vienību un tās transformāciju integrācijai datu noliktavā.

⓿Lai nodrošinātu neviendabīgu integrētu datu avotu izmaiņu adaptāciju sistēmas metadatos, izstrādāts risinājums, kas papildina esošo metadatu glabāšanas shēmu ar papildus datu bāzes struktūru un funkcionalitāti. Minētā struktūra nodrošina adaptācijas scenāriju glabāšanu, kā arī adaptācijas procesa monitoringu, taču funkcionalitāte - adaptācijas soļu izpildi. Izmaiņu adaptācijas scenāriji izstrādāti reāli notikušām izmaiņām.

Darbā izvirzītais mērķis ir sasniegts - atrasts risinājums, kā apstrādāt neviendabīgu integrētu datu avotu evolūcijas rezultātā radušās izmaiņas un adaptēt tās sistēmā. Mērķis sasniegts, veicot izvirzīto uzdevumu izpildi.

Darbā sasniegtais rezultāts izmantojams gan kā esošās sistēmas papildinājums, gan kā universāls konceptuāls piemērs citu līdzīgu sistēmu papildināšanai ar šādu izmaiņu adaptācijas komponenti. Turpmāk plānots attīstīt sistēmu, iekļaujot visu iespējmo datu avotu izmaiņu veidu adaptācijas scenāriju realizāciju.