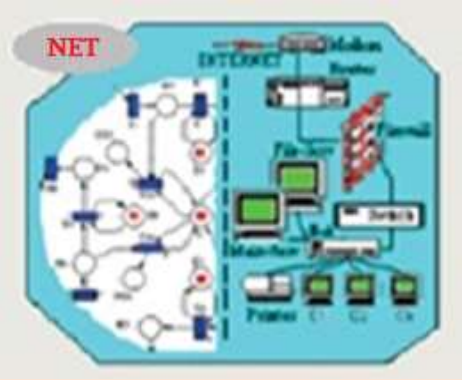
საქართველოს ტექნიკური უნივერსიტეტი
1922 წლიდან

გია სურგულაძე, ლილი პეტრიაშვილი

მონაცემთა საცავის დაპროექტება და აგება ინტერნეტული ბიზნესისათვის



„სტუ-ს IT კონსალტინგის ცენტრი"

საქართველოს ტექნიკური უნივერსიტეტი

გია სურგულაძე, ლილი პეტრიაშვილი

მონაცემთა საცავის დაპროექტება და აგება ინტერნეტული ბიზნესისათვის

დამტკიცებულია:

სტუ-ს „IT კონსალტინგის სამეცნიერო ცენტრის“ სარედაქციო კოლეგიის მიერ ოქმი N4, 10.06.2021

თბილისი 2022

უაკ 004.5

განხილულია მონაცემთა საცავების ობიექტ-ორიენტირებული ანალიზის, დაპროექტების და პროგრამული რეალიზაციის მეთოდები და ინსტრუმენტული საშუალებები ბიზნესის განაწილებული მართვის სისტემებისათვის, მათი კლასიფიკაციის, სტრუქტურული ანალიზის და ღრუბლოვანი ტექნოლოგიების ბაზაზე. გადმოცემულია მონაცემთა მოპოვების (Data Mining) და მონაცემთა ონლაინ დამუშავების (OLAP) თეორიული საფუძვლები და პრაქტიკული გამოყენების საკითხები; ინფოსისტემების მახასიათებლების კვლევა რიგების თეორიის მოდელების მრავალარხიანი ღია და ჩაკეტილი სისტემებით. გამოკვლეულია მოგების ოპტიმიზაციის საკითხები ბიზნესის და კომერციის ობიექტებზე კლიენტ-სერვერული არქიტექტურისათვის. შემუშავებულია სისტემების რესურსების ეფექტური მართვის დინამიკური მოდელი სტოქასტიკური პეტრის ქსელის გრაფო-ანალიზური ინსტრუმენტით; მონაცემთა სტრუქტურების კლასტერიზაციის, კონვერტირებისა და აგრეგაციის ალგორითმული სქემები. მონოგრაფია განკუთვნილია ინფორმატიკის სფეროს დოქტორანტ-მაგისტრანტების, სპეციალისტებისა და სტუდენტებისათვის.

რეცენზენტები:

- *ეკატერინე თურქია* (ტ.მ.კ. პროფესორი, საქ.ეროვნული ბანკი),
- დავით *გულუა* (თსუ, ტ.მ.კ., პროფესორი)

რედკოლეგია:

ა. ფრანგიშვილი (თავმჯდომარე), მ. ახობაძე, გ. გოგიჩაიშვილი, ზ. ბოსიკაშვილი, ე. თურქია, რ. კაკუბავა, თ. ლომინაძე, ნ. ლომინაძე, ვ. კვარაცხელია, ჰ. მელაძე, თ. ობგაძე, გ. სურგულაძე (რეადაქტორი), გ. ჩაჩანიძე, ა. ცინცაძე, ზ. წვერაიძე, ო. შონია

ISBN 978-9941-8-0623-0

Georgian Technical University

Gia Surguladze, Lili Petriashvili

DESIGN AND BUILD OF DATA WAREHOUSE FOR INTERNET BUSINESS

DEDICATED
TO THE 100th FOUNDATION ANNIVERSARY OF THE
GEORGIAN TECHNICAL UNIVERSITY
(1922-2022)

ISBN 978-9941-8-0623-0

ავტორთა შესახებ:

გია სურგულაძე – სტუ-ს „მართვის ავტომატიზებული სისტემების (პროგრამული ინჟინერიის) დეპარტამენტის უფროსი, პროფესორი, ტექნიკის მეცნიერებათა დოქტორი, *გაერო*სთან არსებული „ინფორმატიზაციის საერთაშორისო აკადემიის (IIA)" აკადემიკოსი, სტუ-ს „IT-კონსალტინგის სამეცნიერო ცენტრის" და „ინფორმატიკის" სადოქტორო პროგრამის ხელმძღვანელი, გერმანიის DAAD-ის გრანტის მრავალგზის მფლობელი, ბერლინის ჰუმბოლდტის, ნიურნბერგ-ერლანგენის და სხვა უნივერსიტეტების მიწვეული პროფესორი (1974-2019). 420 სამეცნიერო ნაშრომის ავტორი, მათ შორის 100 წიგნი (25 მონოგრ., 20 სახელმძღ. 55 დამხმ.სახ.). 75 ელ-წიგნი (gtu.ge და პარლამენტის ციფრული ბიბლიოთეკა „ივერიელი") გამოყენებითი პროგრამული ინჟინერიის სფეროში. მისი ხელმძღვანელობით დაცულია 40 სადოქტორო დისერტაცია და 80 სამაგისტრო ნაშრომი. მისია – ინოვაციური ICT ტექნოლოგიების კვლევა, განვითარება და მათი სწავლების პოპულარიზაცია საქართველოში, შემეცნებითი აზროვნების სპეციალისტთა აღზრდა.

ლილი პეტრიაშვილი – საქართველოს განათლების მეცნიერებათა აკადემიის წევრი, საინჟინრო აკადემიის მრჩეველი, სტუ-ის „კომპიუტერული მეცნიერების" საბაკალავრო და „ინფორმატიკის" სადოქტორო პროგრამების ხელმძღვანელი, „ტექნიკის და ტექნოლოგიის მსოფლიო აკადემიის" სამეცნიერო რედკომიტეტის წევრი (https://waset.org/committees). Volkswagen-ის ფონდისა და DAAD-ის სტიპენდიანტი. ერლანგენის, კაიზერსლაუტერნის, შტუტგარტის და სხვ.უნივერსიტეტის მიწვეული პროფესორი (1994-2019). 100-ზე მეტი სამეცნიერო ნაშრომი (5 მონოგრ., 25 სახელმძღვ.). კითხულობს ლექციებს საქართველოს წამყვან უნივერსიტეტებში, მონაცემთა ბაზების, ანალიტიკის, პროექტირების, ლოგისტიკის მენეჯმენტის, ვიზუალური დაპროგრამების, ტელემატიკის და ERP კურსებში ქართულ და გერმანულ ენებზე. მისი ხელმძღვანელობით დაცულია 5 სადოქტორო და 20 ზე მეტი სამაგისტრო ნაშრომი. არის სამეცნიერო კონფერენციების ორგკომიტეტის აქტიური წევრი.

ეძღვნება:

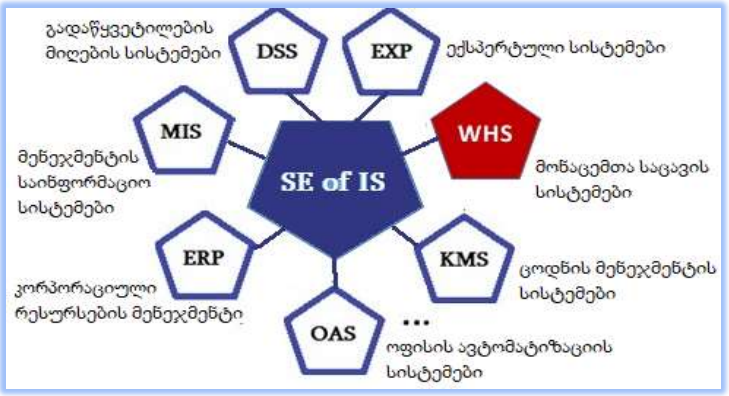
საქართველოს ტექნიკური უნივერსიტეტის დაარსების 100 წლისთავს (1922-2022)

სარჩევი

შესავალი

21-ე საუკუნის ოციანი წლების დასაწყისის ერთ-ერთ მნიშვნელოვან გლობალურ გამოწვევად შეიძლება ჩათვალოს საინფორმაციო საზოგადოების (Information Society) სწრაფი ფორმირების პროცესი, რომლის მთავარი კატალიზატორიც Covid_19-ის მსოფლიო პანდემიის მოვლენა გახდა. მისგან ინიცირებული რეალობა კი მოსახლეობის მიერ კომპიუტერული და თანამედროვე მობილური სისტემების დაუფლება გახდა (მიუხედავად ასაკის, ეროვნების, სქესის, რელიგიის, პოლიტიკის, პროფესიისა და დასაქმების სფეროს მიკუთვნებისა) [1,2].

ინფორმაციული და კომუნიკაციური ტექნოლოგიების განვითარებამ, ინტერნეტისა და ზოგადად, ქსელური ინდუსტრიის სწრაფმა პროგრესმა ბოლო ათწლეულში მნიშვნელოვან შედეგებს მიაღწია. თითქმის მთლიანად შეიცვალა განსხვავებული ტიპების საინფორმაციო სისტემების (ნახ.1) შექმნის (დაპროექტების, დეველოპმენტის და იმპლემენტაციის) ეფექტური და საიმედო ტექნოლოგიები, პლატფორმები, ფრეიმვორკები. გაჩნდა ახალი ცნებები და ტერმინები, რომლებიც თანამედროვე კონცეფციებს, პროექტებს და სისტემებს ედება საფუძვლად [3].



ნახ.1. ინფორმაციული სისტემების (IS) პროგრამული ინჟინერია (SE)

ორგანიზაციული სისტემების მართვის პრობლემების და ამოცანების გადასაწყვეტად საჭირო ხდება ახალი კომპიუტერული და ინფორმაციული ტექნოლოგიების გამოყენება, რომელთა შორის ერთ-ერთი აქტუალური და მნიშვნელოვანი მიმართულებაა მონაცემთა საცავების (Data Warehouse ან Data Repository) აგება. ბოლო პერიოდში იგი ითვლება, განსაკუთრებით პერსპექტიულ და დინამიკურ მიმართულებად საინფორმაცო სისტემების დაპროექტებაში, იგი უკავშირდება მრავალდონიანი განაწილებული საინფორმაციო სისტემის შექმნის პრობლემების გადაწყვეტას.

მონაცემთა საცავი, კონკრეტულად, დიდი ორგანიზაციის (მაგალითად, კორპორაციის) განკუთვნილი სპეციალური სუპერბაზაა, სადაც მიმდინარე ოპერატიული სამუშაოს შესრულებისას თავს იყრის ქრონოლოგიურ ინფორმაციათა მთელი სპექტრი. მისი დანიშნულებაა მომხმარებლისთვის ინტერნეტ გვერდებზე მიზნობრივად განლაგებული ტექსტური, გრაფიკული და აუდიო-ვიზუალური საინფორმაციო ბლოკების მიწოდება. ეს კი, საბოლოო ჯამში, გამოიყენება მონაცემთა ინტელექტუალური ანალიზისა და ანგარიშგებისათვის. იგი შეიცავს ინტეგრირებულ მონაცემებს სხვადასხვა წყაროდან და ხელსაყრელია ბიზნესის ანალიტიკური ინფორმაციის მოსამზადებლად.

თავისუფალ საბაზრო ეკონომიკის პირობებში ბიზნესის მართვის სისტემებისათვის გადაწყვეტლებათა მიღების მხარდამჭერი ეფექტური მექანიზმების დამუშავება და მათი კვლევა თანამედროვე ქსელური საინფორმაციო ტექნოლოგიების ინტეგრირებული გამოყენებით, ერთ-ერთი მნიშვნელოვანი მიმართულებაა.

ელექტრონული ბიზნესისა და კომერციის მომხმარებელს მიეცემა საშუალება ინტერაქტიულ-დიალოგური კრიტერიუმებით განსაზღვროს მისთვის საჭირო ინფორმაცია. ამ მიზნით ანალიზური პროცესების სისტემა ეყრდნობა მრავალგანზომილებიან

მონაცემთა სტრუქტურებს. საინფორმაციო წყაროებიდან მიღებული მონაცემები ტრანსფორმაციის, კონვერტაციის და ინტეგრირების შემდეგ, თავსდება საცავის ობიექტ-ორიენტირებულ მონაცემთა ბაზებში (მაგალითად, Oracle, Ms SQL Server, MongoDB ან სხვ.), Windows- ან Web-აპლიკაციების ბაზაზე.

წინამდებარე ნაშრომში შემოთავაზებულია ავტორთა მიერ საქართველოს ტექნიკურ უნივერსიტეტსა და გერმანიის უნივერსიტეტებში (ბერლინის ჰუმბოლდტისა და ნიურნბერგ- ერლანგენის ინფორმატიკის ინსტიტუტებში) მიღებული თეორიული და ექსპერიმენტული სამუშაოების სამეცნიერო შედეგები. გერმანელ კოლეგებთან ერთად 20 წლის განმავლობაში ხდებოდა ამ მიმართულებით სამეცნიერო და აკადემიური თანამშრომლობა, რაც ასახულია ერთობლივად გამოქვეყნებულ წიგნებში, სამეცნიერო კონფერენციებსა და სტატიებში [4-8]. ერთ-ერთი პირველი მონოგრაფია *მონაცემთა საცავების* თემატიკაზე გამოიცა 2005 წელს სტუ-ში ჩვენი ავტორობით, რომელიც დოქტორანტის, ლილი პეტრიაშვილის სადისერტაციო ნაშრომის (ხელმძღვანელი - პროფ. გ. სურგულაძე) გაფართოებით იყო შექმნილი (ნახ.2) [9].

წინამდებარე წიგნის ***შესავალში*** განხილულია კორპორაციული მენეჯმენტის საინფორმაციო სისტემების მონაცემთა საცავის აგების თანამედროვე ტექნოლოგიების და ინსტრუმენტულ საშუალებათა დამუშავების საკითხების აქტუალობა, მათი მნიშვნელობა. ჩამოყალიბებულია ნაშრომის ძირითადი მიზნები, ამოცანები, მათი გადაწყვეტის გზები.

პირველ თავში გადმოცემულია მონაცემთა საცავების (Data Warehouses) ტიპის ინფორმაციული სისტემის ძირითადი ცნებები და ტერმინები, მათი დანიშნულება და როლი კორპორაციული მენეჯმენტის ბიზნესპროცესების მართვის საინფორმაციო სისტემებისათვის. წარმოდგენილია მონაცემთა საცავების არქიტექტურა და ძირითადი კომპონენტები, საცავის ETL (Extract, Transform, Load) და

ELT (Extract, Load, Transform) ტიპები [10]. აგრეთვე რელაციური და NoSQL ბაზების საფუძველზე საცავების აგების ტექნოლოგიები, კლასიკური და ღრუბლოვანი არქიტექტურებით.

ნახ.2. პირველი ქართული მონოგრაფია მონაცემთა საცავების და ბიზნეს-ანალიზის ტექნოლოგიაზე, სტუ, 2005 [9]

როგორც ცნობილია, მონაცემთა საცავში ინახება კომპანიების მუშაობის ისტორიის მთელი საინფორმაციო სპექტრი. ისტორიული მონაცემების შენახვა არის ერთ-ერთი აუცილებელი პირობა და იგი მონაცემთა საცავის მთავარი ღირსებაა.

მომხმარებლებს საშუალება ეძლევა ინტერნეტ გვერდებზე არსებული ინფორმაციის დახმარებით ეფექტურად და მიზნობრივად იმოქმედონ ბიზნესის სფეროში. საჭირო ხდება ენერგიის, დროის და, რა თქმა უნდა, ფინანსური რესურსების დაზოგვის ღონისძიებების გატარება, რისი უნიკალური საშუალებაცაა ელექტრონული კომერციის (E-Commerce) გამოყენება. აქ აღწერილია *მისი* ინფრასტრუქტურული სისტემა. წარმოდგენილია ეკონომიკური მოვლენის განმსაზღვრელი ცვალებადი ფაქტორები.

გაანალიზებულია დამოკიდებულება პროდუქციის მიწოდებას, მოთხოვნასა და ფასს შორის თავისუფალ საბაზარო ეკონომიკის პირობებში.

პირველი თავის ბოლოს განხილულია მონაცეთა საცავების აგების საკითხები ახალი ტექნოლოგიების საფუძველზე. კერძოდ წარმოდგენილია ღრუბელზე დაფუძნებული მონაცემთა საცავების (Cloud-based Data Warehouse) კონცეფციის განვითარების საკითხები. მათი უპირატესობები და ნაკლოვანებანი კორპორაციებში გამოყენების თვალსაზრისით. აქვე განიხილება მონაცემთა საცავების აგების შესაძლებლობები NoSQL ტიპის ბაზების საფუძველზე, კერძოდ, MongoDB Compass და Atlas პაკეტებისთვის.

მეორე თავში წარმოდგენილია მართვის განაწილებული სისტემების მონაცემთა საცავების დაპროექტების ეტაპების, მისი ძირითადი კომპონენტების, ფუნქციებისა და კლასიფიცირებულ მონაცემთა მეტაინფორმაციის ეფექტური ორგანიზაციის, გარდაქმნისა და ძებნის მეთოდების და ალგორითმების აღწერა.

მონაცემთა საცავი მუდმივად ივსება საგნობრივი სფეროს შესაბამისი ინფირმაციით და ორგანიზებულია მონაცემთა ოპერატიული ბაზებიდან შემოსული სტრუქტურულად გადამუშავებულ მონაცემთა ქვესიმრავლეების საფუძველზე (ETL საცავი). ინფორმაციის წყარო სხვადასხვა ორგანიზაციის აპლიკაციებია, რომლებიც გამოიყენებს განსხვავებულ პროგრამულ პლატფორმებს და უკავშირდება ოპერატიულ მონაცემთა ბაზას ინტერნეტის საშუალებით.

მესამე თავი ეხება მონაცემთა საცავის ინტელექტუალური ანალიზის შესაძლებლობების აღწერას მონაცემთა მოპოვების (Data Mining) პროცესის საფუძველზე. განიხილება მისი კავშირი და როლი დიდ მონაცემთა სისტემებში, აგრეთვე დამოკიდებულება მონაცემთა მეცნიერების (Data science) მიმართულებასთან. წარმოდგენილია გამოყენების სფეროები, სპეციალური ალგორითმები და პროგრამები.

მეოთხე თავში გადმოცემულია განაწილებული ბიზნეს-ობიექტების სისტემების მონაცემთა საცავების საინფორმაციო ბლოკებისა და მათი დამუშავების მეთოდების პროგრამული რეალიზაციის საკითხები. განიხილება გლობალური და ლოკალური ქსელის ტექნიკური უზრუნველყოფის საკითხები და მასში ინფორმაციის დაცვისა და აღდგენის საშუალებანი.

მეხუთე თავი ეხება მონაცემთა საცავების ბაზაზე მომუშავე კორპორაციის მენეჯმენტის ბიზნეს-პროცესების იმიტაციური მოდელირებისა და ანალიზის ალტერნატიული გრაფო-ანალიზური ინსტრუმენტების გამოყენებას მისი საინფორმაციო სისტემის რესურსების ეფექტიანობის განსაზღვრის მიზნით.

წარმოდგენილია რიგების თეორიის და პეტრის ქსელების პროგრამული პროდუქტები და შესაბამისი სიმულაციური პროცესების შედეგები. კერძოდ, აგებულია განაწილებული სისტემის რესურსების მართვის პროცესის მასობრივი მომსახურების მოდელი სტაციონარული რეჟიმისათვის, პეტრის ქსელის გრაფული მოდელი პროცესების კვლევის დინამიკური რეჟიმისთვის.

ბოლოს წარმოდგენილია ***დასკვნები***, რომლებშიც ასახულია ნაშრომში მიღებული ორიგინალური შედეგები და მათი გამოყენების რეკომენდაციები.

წიგნის მეორე გამოცემა განახლებული და გამდიდრებულია ახალი მასალით. ავტორები გულისხმიერებით მოეკიდებიან მკითხველთა შენიშვნებს და რეკომენდაციებს შესაბამისი საკითხების შემდგომი სრულყოფისა და განვითარების მიზნით.

ავტორები, 20.01.2022
g.surguladze@gtu.ge, l.petriashvili@gtu.ge

I თავი

მონაცემთა საცავი ბიზნესისათვის (მიმოხილვა): დანიშნულება, არქიტექტურა და ძირითადი ცნებები

მონაცემთა საცავი (DWH – Data Warehouse, Repository, Data Storage) ან ორგანიზაციის (კორპორაციის) „მონაცემთა საწყობი“ (Enterprise DW) არის გადაწყვეტილების მიღების მხარდამჭერი სისტემა. იგი გამოიყენება კომპანიის მონაცემთა ინტელექტუალური ანალიზისა და ანგარიშგებისათვის, ითვლება ბიზნეს ინტელექტის ძირითად კომპონენტად, რომლის ფორმირება ხორციელდება რამდენიმე განსხვავებული მონაცემთა წყაროს ინტეგრირების საფუძველზე, როგორც ერთიანი ცენტრალური საცავი [11].

კორპორაციულ მონაცემთა საცავის შესაქმნელად გამოიყენება ორი ძირითადი მიდგომა ETL (Extract, Transform, Load) და ELT (Extract, Load, Transform) [10].

ETL არქიტექტურით აგებული მონაცემთა საცავი შედგება:

- *შუალედური დონის* (წყაროებიდან შემოსული დაუმუშავებელi მონაცემთა ბაზა);
- *მონაცემთა ინტეგრაციის დონის* (შუალედური დონის დანაწევრებული მონაცემების გაერთიანება გარდაქმნის საფუძველზე);
- *წვდომის დონისგან* (ოპერაციული მონაცემების და აგრეგირებული ფაქტების ბაზა მომხმარებლებისთვის).

ამგვარად, გაწმენდილი, ტრანსფორმირებული და კატალოგიზირებული მონაცემები ხელმისაწვდომი ხდება მენეჯერებისა და ბიზნეს-მომხმარებლებისთვის ინტელექტუალური ანალიზის ჩატარების და გადაწყვეტილების მიღების მხარდასაჭერად.

ELT არქიტექტურის საფუძველზე აგებული საცავი გამორიცხავს მონაცემთა ტრანსფორმაციის ინსტრუმენტის არსებობას. არაერთგვაროვანი წყაროებიდან ამოღებული მონაცემები გარდაქმნების გარეშე აიტვირთება უშუალოდ მონაცემთა საცავში. შემდეგ აქვე ხდება მათი სათანადო დამუშავება და მომზადება.

კორპორაციული მონაცემთა საცავის გამოყენებას აქვს შემდეგი უპირატესობები [12]:

- ***მონაცემთა სტანდარტიზაცია:*** ნებისმიერი საწარმოს ბიზნეს მონაცემები ინახება სხვადასხვა ონლაინ ტრანზაქციის დამუშავების (OLTP) სისტემებსა და მონაცემთა ბაზაში. თითოეული სისტემა ან მონაცემთა ბაზა შეინახავს მონაცემებს სხვადასხვა ფორმატში. ნებისმიერი სასარგებლო ანალიზის ჩასატარებლად, მონაცემები ჯერ უნდა იყოს სტანდარტიზებული ისე, რომ ყველა მონაცემი იყოს იმავე ფორმატში. ამ მონაცემთა ბაზებიდან მონაცემები იტვირთება მონაცემთა საწყობში ETL-ის (ამოღება, ტრანსფორმაცია, ატვირთვა) გამოყენებით. იგი შეიძლება გამოყენებულ იქნას მონაცემების სტანდარტულ ფორმატში გადასაყვანად და მონაცემთა საცავში შესანახად. ეს აადვილებს მონაცემთა ანალიზს და აუმჯობესებს საცავში არსებული მონაცემების გაგებას;
- ***მოქნილობა:*** ბიზნესის ზრდასთან ერთად, შენახული მონაცემების რაოდენობაც სწრაფად იზრდება. მოცულობის მატებასთან ერთად, შეიძლება იყოს სიტუაციები, როდესაც მონაცემთა მოდელი საჭიროებს განახლებას, რათა განთავსდეს სხვა ატრიბუტები, რომლებიც უნდა ჩაიწეროს. კორპორაციული მონაცემთა საცავები (EDW) ცნობილია თავისი მოქნილობით და ადვილად აკმაყოფილებს მოთხოვნებს ნებისმიერ ცვლილებას, როგორიცაა შენახული მონაცემების რაოდენობა ან მონაცემთა მოდელის ცვლილება, სისტემის სრული აღდგენის გარეშე;
- **ბიზნეს-ანალიზის ინსტრუმენტებთან დაკავშირების შესაძლებლობა.** კორპორაციულ მონაცემთა საცავების უმეტესობა (EDW) აადვილებს დაკავშირებას არჩეულ ბიზნეს-ანალიზის ინსტრუმენტთან. ეს უწყვეტი კავშირი საშუალებას აძლევს ბიზნესს ადვილად დაადგინოს და თვალყური ადევნოს ეფექტიანობის სხვადასხვა

ძირითად ინდიკატორს (KPI – Key Performance Indicators), რომლებიც შეიძლება გამოყენებულ იქნას იმის გასაზომად, თუ რამდენად კარგად აღწევს ბიზნესი თავის ძირითად მიზნებს.

1.1. მონაცემთა საცავების ზოგადი მიმოხილვა

მონაცემთა საცავი (Data Warehouse) არის საგნობრივ სფეროზე ორიენტირებული, ინტეგრირებული, ქრონოლოგიურად დალაგებული, ხანგრძლივი დროის სპექტრში მიღებული არასტაბილურ მონაცემთა ერთობლიობა, რომელიც გამოიყენება ოპტიმალური გადაწყვეტილებების მისაღებად მენეჯმენტის პროცესების სწორი ორგანიზაციისა და საოპერაციო მონაცემთა საფუძველზე.

მონაცემთა საცავი მხარს უჭერს მონაცემთა ონლაინ ანალიზს OLAP (Online Analytical Processing) ინსტრუმენტის გამოყენებით, რომლის ფუნქციონირება და მოთხოვნის შესრულების ხარისხი საკმაოდ მაღალია სხვადასხვა ონლაინ პროცესის დროს წარმოქმნილი ტრანზაქციებისას [9]. როგორც აღინიშნა, მონაცემთა საცავი გადაწყვეტილების მიღების მხარდამჭერი ტექნოლოგიაა, რომლის მიზანია სწრაფად გამოიმუშავოს ოპტიმალური ცოდნა. იგი სერვისია, როგორც მონაცემთა მოდელების ფიზიკური იმპლემენტაციისათვის, ასევე იმ ინფორმაციის შესანახად, რომელიც საჭიროა საქმიანი პროცესების ეფექტურად წარმართვის და სტრატეგიული გადაწყვეტილებების მიღებისა და მართვისათვის [14].

საცავების ტექნლოგია მოიცავს მონაცემთა მოწესრიგების, ინტეგრაციის და ონლაინ ანალიზის პროცესს. მონაცემთა ონლაინ ანალიზის ტექნოლოგია OLAP მონაცემთა ანალიზის ისეთი ტექნიკური საშუალებაა, რომელიც უზრუნველყოფს მონაცემთა კონსოლიდაციას, აგრეგაციას და შემაჯამებელი მნიშვნელობის გამოტანას, რაც უზრუნველყოფს ინფორმაციის სხვადასხვა ანალიზის ჭრილში მიღებას.

მონაცემთა ბაზაში ინახება განსხვავებული ტიპის მონაცემები. საცავი არის მრავალჯერადი ჰეტეროგენული მონაცემთა წყაროს რეპოზიტორი, რომელიც ორგანიზებულია ერთიანი სქემის ქვეშ და ხელს უწყობს მმართველი გადაწყვეტილების მიღებას.

გასული საუკუნის ბოლო ათწლეულში მონაცემთა საცავების კონცეფცია ნაკლებად გამოიყენებოდა. დღეს კი თითქმის ყველა დიდი ბიზნესკომპანია მნიშვნელოვან მხარდაჭერას იღებს მისგან.

განსაკუთრებით გამოიკვეთა მათი მნიშვნელობა საბანკო ინდუსტრიაშიც [13]. ისტორიული ფაქტია, რომ ამერიკულმა ერთერთმა პრესტიჟულმა კორპორაციამ (რეგიონალური ბანკი – FAC), რომელიც მდებარეობდა ქვეყნის სამხრეთ-აღმოსავლეთში, 1990 წელს დაკარგა 60 მილიონი დოლარი, რადგან საბანკო რისკების შეფასება ხდებოდა მხოლოდ ადგილობრივი რისკების შეფასების სამსახურიდან. კრიზისის შემდეგ მენეჯმენტის ჯგუფმა შეიმუშავა მომხმარებელთან ურთიერთობის ახალი სტრატეგია, სადაც ძირითად დასაყრდენად მონაცემთა საცავები გამოიყენეს. მათი დახმარებით შეძლეს მომხმარებელთა მოთხოვნების და პოტენციალის განსაზღვრა. პროგრამისტების ჯგუფის დახმარებით კი შეიქმნა მომხმარებელზე ორიენტირებული ახალი პროდუქტები და სერვისები, რამაც კომპანია გადააქცია ინოვაციურ ლიდერად, ფინანსურ და მომსახურების ინდუსტრიაში.

სწორედ ეს ფაქტი გახდა ამერიკელი მეცნიერის, ვილიამ ინმონის შთაგონების წყარო, რომ უკვე 1992 წელს შეექმნა უნიკალური, მრავალფუნქციური მონაცემთა საცავების კონცეფციაზე ორიენტირებული ტექნოლოგია [15].

მონაცემთა საცავების გამოყენება წარმატებით მოხდა მრავალ ინდუსტრიულ, კომერციულ თუ კორპორაციულ ობიექტზე, წარმოებაში (მაგალითად, შეკვეთა, გადაზიდვა და მომხმარებელთა მომსახურება და სხვ.), საცალო ვაჭრობაში (მომხმარებელთა პროფილის და მოთხოვნების მენეჯმენტი), ფინანსური მომსახურება (რისკების ანალიზი, საკრედიტო ბარათების ანალიზი, თაღლითობის გამოვლენა და ა.შ.), ტრანსპორტირება („ფლოტის“

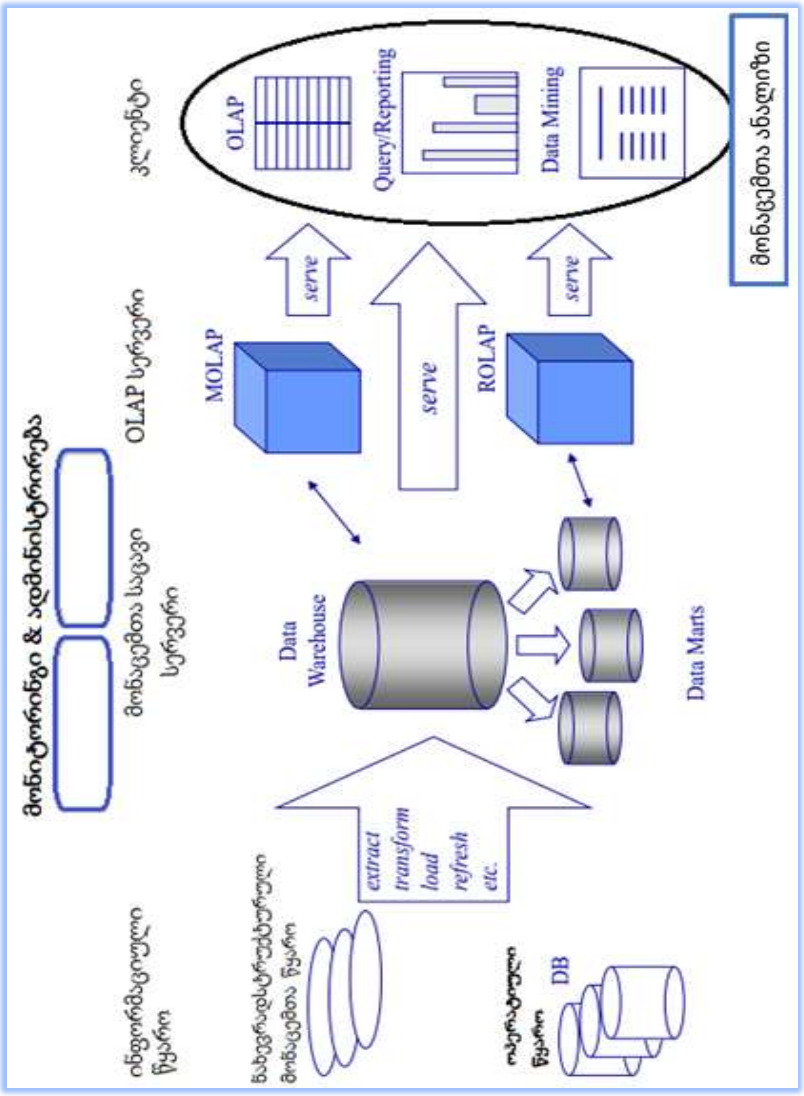
მენეჯმენტი), ტელეკომუნიკაცია (შემომავალი და გამავალი ზარების ანალიზი), ჯანდაცვა (შედეგების ანალიზი) და ა.შ [16].

მონაცემთა საცავი არის მონაცემთა სპეციალური სანახი, რომელიც უზრუნველყოფს გადაწყვეტილების მიღების ხელშეწყობას. საცავში მონაცემების მიღება ხდება ოპერატიული სისტემიდან და გარე წყაროებიდან, მიღების პროცესში პარალერულად ხდება მონაცემთა მოწესრიგება (შეცდომების გამოვლენა, გასწორება) და გარდაქმნა (ექცევა საგნობრივ ჯგუფებში და ერთიანდება ლოგიკურად). მონაცემებს საცავში აქვს შემდეგი მახასიათებლები:

- *საგნობრივ ორიენტირებული* – მონაცემები ლოგიკურად ორგანიზებულია ძირითადი საგნის გარშემო, მაგალითად, მომხმარებლების, გაყიდვების ან პროდუქციის შესაბამისად;
- *ინტეგრირებული* – საგნის შესახებ არსებული ყველა მონაცემი კომბინირებულია, რომელთა ანალიზი ხდება ერთდროულად;
- *პერიოდულობა* – ისტორიული მონაცემები (ქრონოლოგიურად) ინახება დეტალიზებული ფორმით;
- *მდგრადობა* – მონაცემები მხოლოდ წაკითხვადია, არ ხდება განახლება ან წაშლა მომხმარებლის მიერ.

ოპერატიულ სისტემას აქვს თავის საკუთარი მონაცემთა ბაზა, რომელსაც იყენებს მონაცემთა ტრანზაქციის პროცესისთვის. მონაცემთა საცავის შექმნისთანავე მისი მომხმარებლები (ანალიტიკოსები, მენეჯერები) ახდენენ მონაცემებზე წვდომას SQL-ის საფუძველზე, სადაც მოთხოვნების ან სხვა აპლიკაციის გამოყენებით შესაძლებელი ხდება გადაწყვეტილების მიღების მხარდამჭერი საინფორმაციო სისტემის გამოყენება.

1.1 ნახაზზე მოცემულია მონაცემთა საცავის ზოგადი არქიტექტურა [19]. წარმოდგენილი სტრუქტურის მიხედვით საცავში ბოლო ეტაპზე ხდება მულტიფუნქციური ოპერატიული მონაცემთა ბაზებიდან ინფორმაციის მიღება და მრავალგანზომილებიანი ონლაინ-ანალიზი, მას შემდეგ, რაც მოხდება პერიოდულად განახლებული მონაცემთა „გასუფთავება“, ტრანსფომაცია და ინტეგრაცია.



ნახ.1.1. მონაცემთა საცავის ზოგადი ინფრასტრუქტურა

მონაცემთა საცავში, ინფორმაციის განახლების ყველა ეტაპზე,

ხდება მონაცემთა არქივაცია [18].

მონაცემთა საცავის დაპროექტება კომპლექსური პროცესია, რომელიც მოიცავს შემდეგ საკითხებს:

– მონაცემთა საცავის არქიტექტურის, მონაცემთა შენახვის სერვისების, მონაცემთა წყაროს, დაგეგმვის მოცულობის და OLAP სერვერის მოცულობის განსაზღვრა;

– სერვერების, სამომხმარებლო ინსტრუმენტების და სანახების ინტეგრირება;

– მონაცემთა საცავის დიზაინის და ხედების სქემატური განსაზღვრა;

– მონაცემთა საცავის ფიზიკური სტრუქტურის დადგენა, მონაცემთა ორგანიზება და შესაბამისი მეთოდების განსაზღვრა;

– მონაცემთა წყაროსთან დაკავშირება შესაბამისი დრაივერების საშუალებებით;

– მონაცემთა ტრანსფორმაციის, განახლების და ჩამოტვირთვის საჭირო დიზაინის და სკრიპტების იმპლემენტაცია;

– რეპოზიტორის შევსება სქემების, სკრიპტების და სხვა მეტამონაცემებით;

– სამომხმარებლო აპლიკაციის და დიზაინის იმპლემენტაცია;

– მონაცემთა საცავის და შესაბამისი აპლიკაციის ფართოდ გამოყენება.

1.2. OLTP და OLAP სისტემები

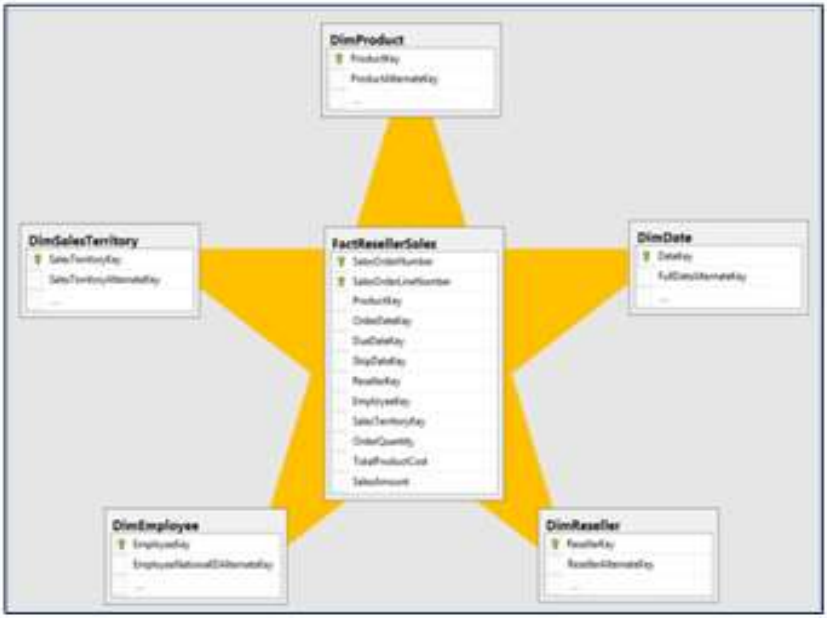
ონლაინ ტრანზაქციის დამუშავება (OLTP - Online transaction processing) იღებს, ინახავს და ამუშავებს ტრანზაქციების მონაცემებს რეალურ დროში. ონლაინ ანალიტიკური დამუშავება (OLAP - Online Analytical Processing) კი იყენებს რთულ მოთხოვნებს OLTP სისტემებიდან აგრეგირებული ისტორიული მონაცემების გასაანალიზებლად.

მონაცემთა საცავის დაპროექტებისას ერთ-ერთი მნიშვნელოვანი კომპონენტია მონაცემთა ონლაინ-ანალიზის ტექნოლოგია.

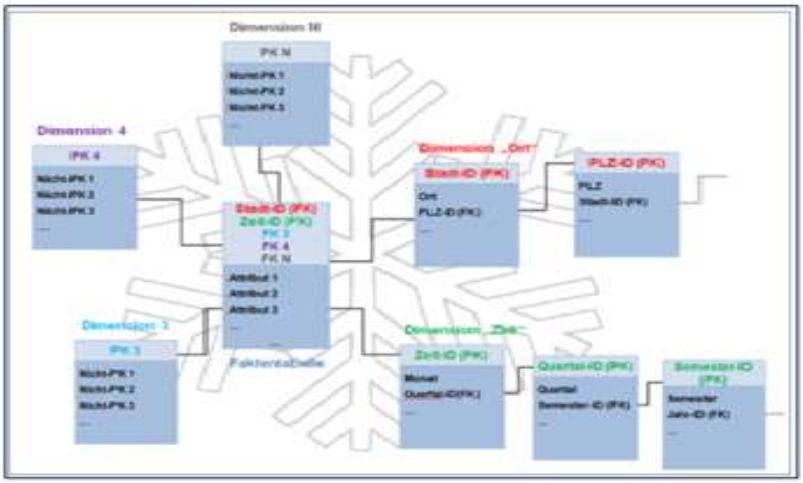
ონლაინ ოპერატიული სისტემების დანიშნულება ადრეულ პერიოდში იყო მხოლოდ ტრანზაქციათა და მოთხოვნათა დამუშავება (OLTP), თანამედროვე მონაცემთა საცავებში კი უზრუნველყოფენ მონაცემთა ანალიზსა და ცოდნაზე დამყარებულ გადაწყვეტილების მიღებას. OLAP სისტემები ორგანზებას და პრეზენტირებას უწევს სხვადასხვა მომხმარებლის მიერ განსხვავებული ფორმატის მონაცემების წარმოდგენას და დამუშავებას.

მნიშვნელოვანია ყურადღება გამახვილებულ იქნას იმ ძირითად განმასხვავებელ ნიშნებზე, რომლებიც OLTP და OLAP-ს შორისაა [17]:

- **სისტემას და მომხმარებელს შორის დამოკიდებულება:** OLTP არის *მომხმარებელზე ორიენტირებული* სისტემა, ტრანზაქციათა და მოთხოვნათა პროცესებისთვის, რომელსაც აყენებს ინფორმაციული ტექნოლოგიების სპეციალისტები და „კლიენტები“, ხოლო OLAP არის *ბაზარზე ორიენტირებული* სისტემა და გამოიყენება მონაცემთა ანალიზისთვის, რომელსაც სისტემას უყენებს მენეჯერები და ანალიტიკოსები;
- **მონაცემთა კონტენტი:** OLTP სისტემა მონაცემებს მართავს დეტალიზებული ფორმატით, ხოლო OLAP სისტემა ახდენს ფართო დიაპაზონით მართვას, სადაც გათვალისწინებულია, როგორც მონაცემთა ისტორიული შინაარსი, ასევე მათი ჯამური შდეგი და აგრეგაცია. გარდა ამისა, ინფორმაცია ინახება და იმართება ინკაფსულაციის განსხვავებულ დონეებზე, რაც აიოლებს მონაცემთა გამოყენებას საინფორმაციო გადაწყვეტილების მიღებაში;
- **მონაცემთა ბაზის დიზაინი:** OLTP სისტემის ზოგადი მოდელი ორიენტირებულია მონაცემთა მოდელსა და აპლიკაციაზე, მონაცემთა ბაზის დიზაინზე. OLTP სისტემის საგანზე ორიენტირებულ მონაცემთა ბაზის დიზაინს უმეტეს შემთხვევაში აქვს ვარსკვლავის ან ფიფქის მოდელის სტრუქტურა (ნახ.1.2-ა,ბ);



ნახ.1.2-ა. სქემა „ვარსკვლავი“



ნახ.1.2-ბ. სქემა „ფიფქი“

- **ხედი:** OLTP სისტემა ძირითადად ორიენტირებულია მიმდინარე მონაცემებზე მათი ისტორიული ცნობების გარეშე ან სხვადასხვა ორგანიზაციიდან მიღებულ მონაცემებზე. აღნიშნული სისტემისგან განსხვავებით OLAP სისტემა მოიცავს მონაცემთა ბაზის სქემის მრავალ ვერსიას, ორგანიზაციის ევოლუციის პროცესების შესაბამისად, თუმცა მისი დიდი მოცულობის გამო OLAP მონაცემები გაზიარებულია მრავალ სხვადასხვა სანახში;
- **წვდომის შაბლონები:** OLTP სისტემის წვდომის შაბლონის პლატფორმა ძირითადად შდგება მოკლე „ატომური“ ტრანზაქციებისგან. ასეთი სისტემისთვის აუცილებებლია მუდმივი ზედამხედველობა და აღდგენის მექანიზმის ჩართულობა, ხოლო OLAP სისტემაზე წვდომა უმეტეს შემთხვევაში მოიცავს მხოლოდ წაკითხვის ოპერაციებს, თუმცა ბევრი მათგანი შესაძლოა იყოს კომპლექსური მოთხოვნები.

1.3. მონაცემთა საცავი და OLAP სისტემა

ერთ-ერთ ყველაზე აქტუალურ მიმართულებად, მიუხედავად ოპერატიული მონაცემთა ბაზების არსებობისა, დღემდე რჩება მონაცემთა საცავი, რადგან, ოპერატიული მონაცემთა ბაზა დაპროექტებული და დამუშავებულია ცოდნაზე დაფუძნებულ ამოცანებზე, ისეთზე როგორიცაა ინდექსაცია, ძირითადი გასაღების გამოყენებით. კონკრეტული ჩანაწერების ძებნა და „შაბლონური ჩანაწერების“ ოპტიმიზაცია.

მონაცემთა საცავებში მოთხოვნები ხშირად კომლექსურია, რაც ითვალისწინებს მონაცემთა დიდი ჯგუფების შემაჯამებელ დონეზე გამოთვლებს, ასევე შეიძლება გამოყენებულ იყოს მონაცემთა ორგანიზების სპეციალური მოთხოვნა, რომელიც ახდენს მონაცემთა მრავალგანზომილებიან წარმოდგენას. OLAP მოთხოვნათა დამუშავება ოპერატიულ მონაცემთა ბაზაში მნიშვნელოვნად ამცირებს ოპერატიულ დავალებებს.

ოპერატიული მონაცემთა ბაზა ასევე მხარს უჭერს მრავალი ტრანზაქციის ერთდროულ დამუშავებას. მიმდინარე პროცესების მეთვალყურეობა და მდგრადობა გარემო ფაქტორების მიმართ მნიშვნელოვანია, რომლის უზრუნველსაყოფად პერიოდულად ხდება ტრანზაქციათა დაბლოკვა, მიუხედავად იმისა, რომ OLAP მოთხოვნას ხშირად სჭირდება მონაცემთა ჩანაწერების მხოლოდ წაკითხვადი წვდომა, რათა მოხდეს მათი შეჯამება და აგრეგაცია, ხოლო რაც შეეხება კონტროლისა და აღდგენის მექანიზმებს, მათი გამოყენება თუ მოხდება OLAP ოპერაციებისათვის, მაშინ არსებობს საფრთხე იმისა, რომ ვერ მოხერხდეს ერთდროული ტრანზაქციების შესრულებას.

გადაწყვეტილების მიღების მხარდაჭერა მოითხოვს ისტორიული ცნობების გათვალისწინებას, ხოლო ოპერატიული მონაცემთა ბაზები, როგორც წესი არ საჭიროებს ისტორიულ ცნობებზე დაყრდნობას. ამიტომ, მიუხედავად იმისა, რომ ოპერატიულ მონაცემთა ბაზაში საკმაოდ ბევრი ინფორმაციაა გადაწყვეტილების მისაღებად, ის მაინც შორს არის სრულყოფილებისგან.

გადაწყვეტილების მიღების მხარდაჭერა ჰეტეროგენულ წყაროებიდან საჭიროებს მონაცემთა კონსოლიდაციას (როგორიცაა აგრეგაცია და შეჯამება), სადაც ოპერატიული მონაცემთა ბაზა შეიცავს მხოლოდ დაუმუშავებელ დეტალურ მონაცემებს.

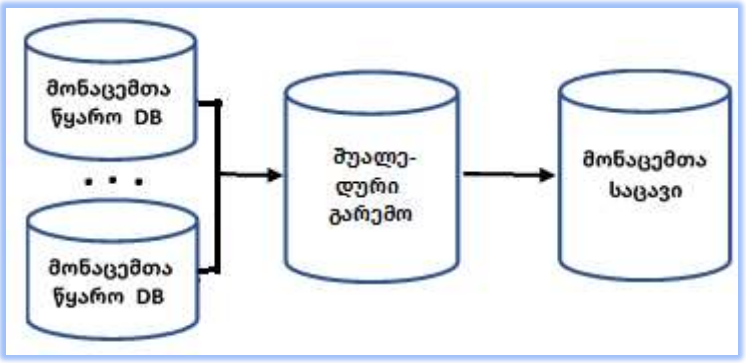
OLAP ტექნოლოგია მომხმარებელს აძლევს საშუალებას გადაწყვეტილება მიიღოს სამომავლო მოქმედებების შესახებ. მონაცემთა საცავებისგან განსხვავებით, OLAP ტექნოლოგია ემყარება რელაციურ ტექნოლოგიას, რაც აიოლებს ინფომაციის ანალიზს და არ ითხოვს დამატებით მეთოდების და ტექნოლოგიების გამოყენებას. იგი იყენებს ერთიან მთლიან მონაცემთა მრავალგანზომილებიან ხედვას, რათა უზრუნველყოს სტრატეგიული ინფორმაციის სწრაფი მიღება მათი შემდგომი ანალიზისთვის [20].

OLAP და მონაცემთა საცავი ერთიანი, მთლიანი ტექნოლოგიაა, სადაც მონაცემთა საცავი მართავს და ინახავს მონაცემებს, ხოლო OLAP კი გარდაქმნის მონაცემთა საცავის მონაცემებს „სტრატეგიულ ინფორმაციად“. მისი დიაპაზონი შედგება საბაზისო მონაცემთა ნავიგაციიდან მათ საბოლოო, რთულ ანალიზამდე.

1.4. მონაცემთა ნაკადი

მონაცემთა ნაკადი (Data flow) საცავში აღწერს იმ გარემოებას, თუ რომელი ობიექტებია საჭირო დაპროექტების (დიზაინის) დროს და რომელი – სისტემის ამუშავებისას, ეს საჭიროა მონაცემთა წყაროდან BI-ში გადასატანად და მონაცემების გაწმენდის, კონსოლიდაციისა და ინტეგრაციისათვის, რათა ისინი გამოყენებულ იქნას ანალიზის, რეპორტების და დაგეგმვისათვის.

მონაცემთა ნაკადი შედგება მონაცემთა ერთი ან რამდენიმე წყაროსგან, რომელიც ერთიანდება ერთ შუალედურ გარემოში, საიდანაც გადადის მონაცემთა საცავში (ან რეპოზიტორში) (ნახ.1.3).



ნახ.1.3

1.5. გადაწყვეტილების მიღება მონაცემთა საცავების გამოყენებით

გადაწყვეტილების მიღების მხარდამჭერი სისტემა (DSS - Decision Support System) არის ინსტრუმენტი, რომელიც გამოიყენება კომპლექსურ სისტემებში გადაწყვეტილების მიღების პროცესის გასაუმჯობესებლად და მხარდასაჭერად.

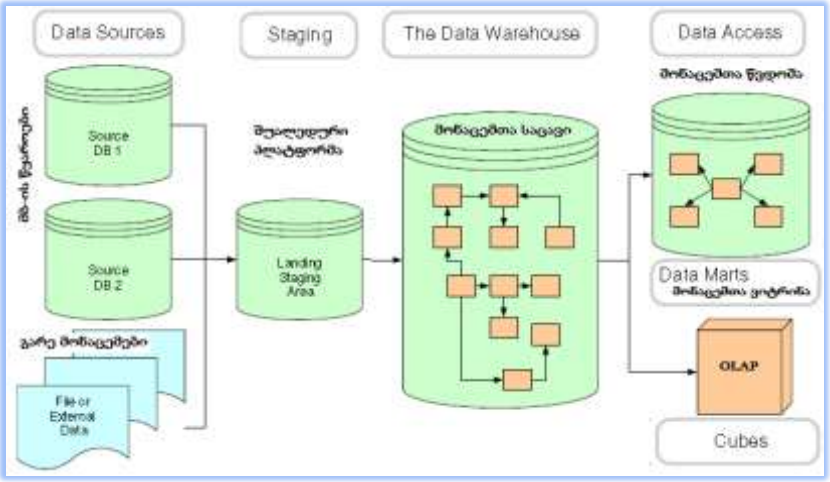
DSS შეიძლება იყოს სისტემაში, რომელიც პასუხობს მარტივ მოთხოვნებს და საშუალებას იძლევა მომხმარებელმა შემდგომი გადაწყვეტილებები თვითონ მიიღოს. სისტემა, იყენებს ხელოვნურ ინტელექტს და უზრუნველყოფს დეტალურ მონაცემთა მოპოვებასთან (Data mining) დაკავშირებულ ფართო სპექტრის საკითხებს.

DSS-ის გამოყენების ყველაზე მნიშვნელოვან სფეროებს შორისაა ის რთული სისტემები, რომლებიც უშუალოდ პასუხობს კითხვებს, რომლებიც უკავშირდება განსაკუთრებით მაღალი დონის რისკების შეფასებას და სცენარების მოდელირებას.

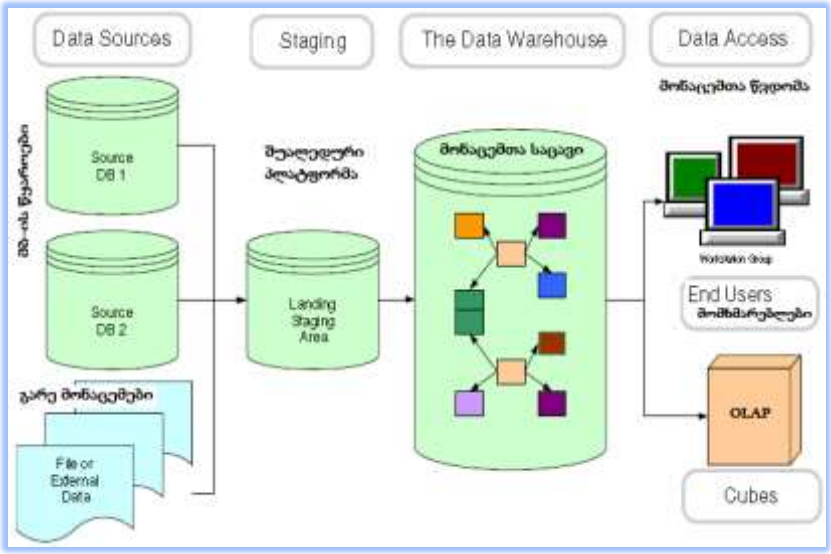
ბოლო ორი ათწლეულის განმავლობაში აქტიურად ხდება გადაწყვეტილების მხარდაჭერი სისტემის – მონაცემთა საცავების განვითარების და მისი გამოყენების თეორიულ-პრაქტიკული საკითხების დამუშავება [21-24].

მონაცემთა საცავის გარემო ექვემდებარება კონტროლს და შესაბამისად, უფრო საიმედოა გადაწყვეტილების მიღებისას, ვიდრე მანამდე არსებული მეთოდები.

მაგალითისათვის შეიძლება მოვიყვანოთ ამ მიმართულების პიონერების: ბ, ინმონის, რ. კიმბალის და მათი მიმდევრების ზოგიერთი მნიშვნელოვანი დეტალები, განსხვავებული მოსაზრებები მათ შორის (ნახ.1.4, 1.5) [22].



ნახ.1.4. ინმონის მონაცემთა საცავის მოდელი ეყრდნობოდა აღმავალ ტექნოლოგიას



ნახ.1.5. კიმბალის მონაცემთა სასიცოცხლო ციკლის მოდელი

როგორც აბრანსონმა აღნიშნა [22], ამ ორი ნახაზის შედარების საფუძველზე, ძირითადი განსხვავება მათ შეხედულებებში ის იყო, რომ კიმბალს მიაჩნდა მონაცემთა საცავი როგორც „მონაცემთა ვიტრინების“ (Data marts) გაერთიანება. იგი ეყრდნობოდა მონაცემთა საცავის აგების „აღმავალ“ (Botton-Up) მეთოდოლოგიას. ანუ ჯერ იქმნება ცალკეული „მონაცემთა ვიტრინები“ და შემდეგ ხდება მათი გაერთიანება დიდ მონაცემთა საცავში.

ინმონი იცავდა „დაღმავალი“ (Top-Down) მეთოდოლოგიის აზრს, ანუ მონაცემთა ვიტრინების შექმნა შესაძლებელია მხოლოდ მონაცემთა საცავის შექმნის შემდეგ.

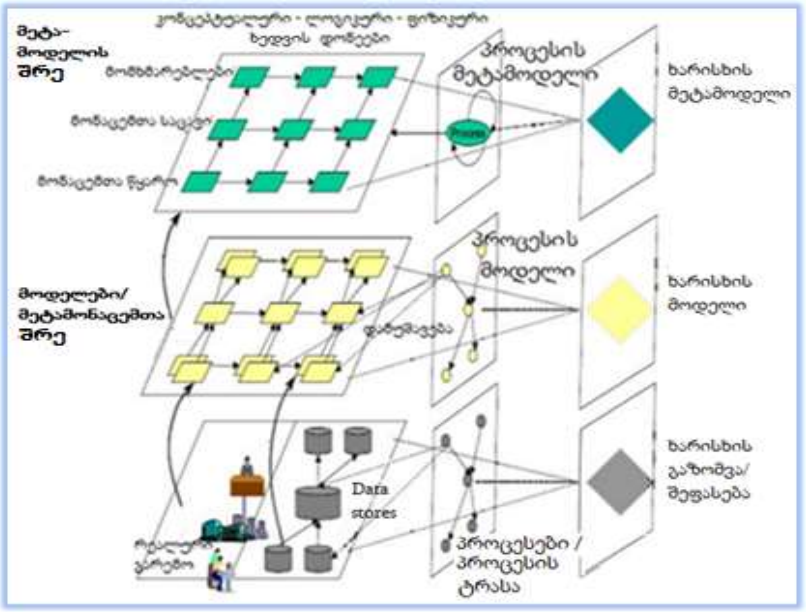
განსხვავება მონაცემთა საცავსა და მონაცემთა ვიტრინებს შორის მნიშვნელოვანია. კერძოდ, მონაცემთა საცავი ცენტრალიზებული, კორპორაციული აპლიკაციაა, ისტორიული მონაცემების ხანგრძლივი შენახვით, მოქნილი შესაძლებლობებით და ერთიანი რთული სტრუქტურით. მონაცემთა ვიტრინები კი - სპეციალური დეცენტრალიზებული დანართებია, რეზიუმეს ტიპის ისტორიული მონაცემების ხანმოკლე შენახვით, შეზღუდული შესაძლებლობებით და რამდენიმე მარტივი სტრუქტურით [22].

მონაცემთა საცავის გარემო მხარს უჭერს გადაწყვეტილების მიღების სრულ მოთხოვნებს მაღალი ხარისხის ინფორმაციის მიწოდებით, რომელიც ხელმისაწვდომია ზუსტი, ეფექტური და თანმიმდევრული მონაცემთა ტრანსფორმაციის წესების და მონაცემთა მნიშვნელობების დოკუმენტირებულ წარდგენით. იგი ეყრდნობა ზუსტ და სანდო ინფორმაციის წყაროს, რომელიც შეიძლება გამოყენებული იქნას მონაცემთა ანალიზისთვის. საცავი (DW) აერთიანებს მონაცემებს მრავალი ჰეტეროგენული ინფორმაციის წყაროდან და გარდაქმნის მათ მრავალგანზომილებიან გადაწყვეტილების მხარდამჭერ სისტემად [9].

მონაცემთა საცავს ასევე ახასიათებს რთული სასიცოცხლო ციკლი. მუდმივი განახლების და დიზაინის ტრანსფორმაციის ფაზაში. მონაცემთა საცავის დაპროექტებისას დიზაინერმა უნდა შექმნას კონცეპტუალური მოდელი და მოცულობითი ლოგიკური სქემა, რომელსაც თან ახლავს დეტალურად წარმოდგენილი მონაცემთა ფიზიკური სტრუქტურა. დიზაინერი ასევე უნდა უზრუნველყოფდეს მონაცემთა საცავის ადმინისტრაციულ პროცესებს, რომლებიც კომპლექსურია სტრუქტურით და შეიცავს დიდი რაოდენობით კოდირებულ ჩანაწერებს.

მონაცემთა საცავი მოიცავს „ევოლუციის“ ფაზას, სადაც კომბინირებულადაა წარმოდგენილი დიზაინის და ადმინისტრაციული ამოცანები. დროთა განმავლობაში იცვლება ორგანიზაციის ბიზნესწესები, მომხმარებლებს უჩნდებათ ახალი მოთხოვნები, ხელმისაწვდომი ხდება ინფორმაციის ახალი წყაროები, რის გათვალისწინებითაც უნდა განვითარდეს და მოთხოვნებზე ადაპტირდეს მონაცემთა საცავის არქიტექტურა. იგი ეფექტური უნდა იყოს გადაწყვეტილების მიღების პროცესის ორგანიზებისთვის.

მონაცემთა საცავის შემადგენელ ყველა კომპონენტზე, პროცესსა და მონაცემებზე უნდა ტარდებოდეს უწყვეტი მონიტორინგი. ადმინისტრირება უნდა მიმდინარეობდეს მეტამონაცემთა საცავის საშუალებით. ამ მხრივ საინტერესოა ბერძენი და გერმანელი მეცნიერების მოსაზრებები [23]. მათ განიხილეს მონაცემთა საცავის სტრუქტურა სამი პროექციით: *კონცეპტუალური, ლოგიკური და ფიზიკური* (ნახ.1.6).



ნახ.1.6. მონაცემთა საცავი სამი პროექციით [24]

თითოეული იყოფა სამ დონედ: მონაცემთა წყარო (data source), მონაცემთა სანახი (data view) და მომხმარებლის დონე (user level).

– *მეტამოდელის შრეზე* სქემატურად გამოსახულია მონაცემთა საცავის არქიტექტურა, მეტა–კლასების სპეციფიკაციების მითითებით, რაც უზრუნველყოფს მონაცემთა საცავისათვის ისეთი ობიექტების წვდომადობას, როგორიცაა: სანახი, ფუნქციონალური კავშირი, მონაცემთა წარმოდგენის წინა ხედი და ა.შ.;

– *მეტამონაცემთა შრეზე* მეტამოდელი არის მონაცემთა საცავის კონკრეტული მოდელი, მოცემული სქემა განსაზღვრავს ინდექსირებას, ცხრილების მოცულობას და ა.შ.;

– მონაცემთა საცავის, მისი სქემის განსაზღვრის, ინდექსების, ცხრილის მოცულობის და ა.შ. ყველაზე დაბალი შრე (ნახაზზე) წარმოადგენს აქტუალურ/რეალურ პროცესს და მონაცემებს.

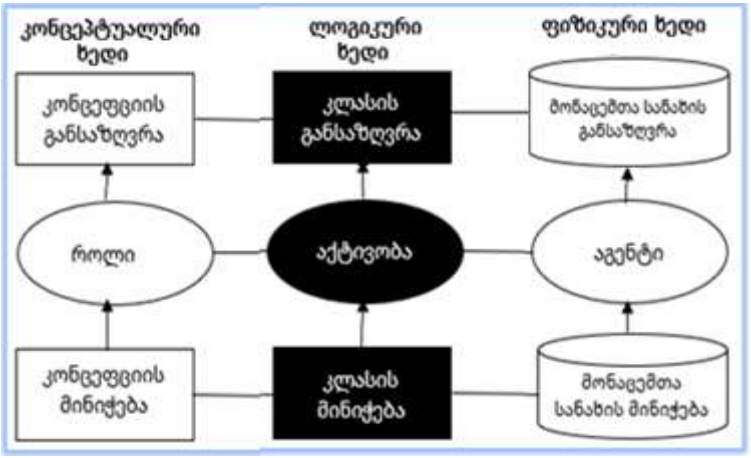
ასეთი სქემა შესაძლებელს ხდის სამივე პროექციაზე განხორციელდეს მონაცემთა ნაკადებზე დაკვირვება:

– *ლოგიკურ პროექციაზე* მოდელირდება ფუნქციონალური ქმედება, რომელიც აღწერს პრაქტიკულ საქმიანობას ინფორმაციის გამოყენების და დამუშავების თვალსაზრისით;

– *ფიზიკურ პროექციაზე* პროცესის შესრულებისთვის საჭირო დეტალები პირდაპირ კავშირშია მოდელირების ცენტრთან;

– *კონცეპტუალური პროექცია* მოიცავს პროცესის არსებობისათვის აუცილებელ მიზეზს, იგი მუშაობს პრინციპით and/or, რომელიც შეიძლება იყოს ინფორმაციის მიმღების დამოკიდებულება მომწოდებელზე ან ინფორმაციის ხანდაზმულობაზე.

1.7 ნახაზზე წარმოდგენილია მეტაპროცესის მოდელის 3 სახე: მონაცემთა საცავის დიზაინის ფორმირების ბიჯები [23].



ნახ.1.7. მეტაპროცესის მოდელის სამი ხედი

მონაცემთა საცავის დაპროექტება რთული პროცესია, რომელიც მოიცავს შემდეგ ქმედებებს:

- არქიტექტურის განსაზღვრა, სადაც წინასწარ დაგეგმილია ის მოცულობა, რომელშიც მოხდება სანახის სერვერების, მონაცემთა, OLAP სერვერების და ინსტრუმენტების განთავსება;
- სერვერზე სანახის და „client tools“ ინტეგრირებული წარმოდგენა;
- მონაცემთა საცავის და ხედების აწყობა;
- მონაცემთა საცავის ფიზიკური ორგანიზება, მონაცემთა განთავსება და განაწილება შესაბამისი მეთოდის გამოყენებით;
- მონაცემთა წყაროს და შემავალი კვანძების დაკავშირება, ODBC (Open Database Connectivity) დრაივერებთან და სხვა დამხმარე საშუალებებთან;
- მონაცემთა მიღების, მოწესრიგების, ტრანსფორმაციის, ჩატვირთვისა და განახლებისათვის სკრიპტების შემუშავება და გამოყენება;
- რეპოზიტორის სქემატური შევსება, სადაც განისაზღვრება სკრიპტების და სხვა მეტა მონაცემების დანიშნულება;
- საბოლოო მომხმარებლისთვის დიზაინის და დანერგვისთვის საჭირო აპლიკაციის განსაზღვრა;
- საცავის და მომხმარებლების დანიშნულების განსაზღვრა.

მონაცემთა საცავის მოდელები, არქიტექტურის თვალსაზრისით, წარმოდგენილია სამი სახის მოდელად:

➢ *ორგანიზაციული მონაცემთა საცავი*

- ორგანიზაციაში აერთიანებს ობიექტების შესახებ არსებულ მთელ ინფორმაციას;

- უზრუნველყოფს კორპორატიულ მონაცემთა ინტეგრაციას, ერთი ან რამდენიმე ოპერატიული სისტემიდან ან გარე საინფორმაციო პროვაიდერებიდან;

- შეიცავს დეტალურ და ჯამურ მონაცემებს, რომელთა ზომა შესაძლებელია იყოს რამდენიმე გიგაბაიტიდან ტერაბაიტამდე ან უფრო მეტი;

- შესაძლებელია დანერგვა მოხდეს ტრადიციულ „მეინფრეიმ-ზე“, როგორიცაა ე.წ. „სუპერ“ UNIX სერვერი ან პარალელურ არქიტექტურულ პლატფორმებზე.

➢ *მონაცემთა „მარკეტი“*

- შეიცავს კორპორატიულ მონაცემთა ქვეჯგუფს, რომლებიც განკუთვნილია მომხმარებელთა კონკრეტული ჯგუფებისთვის. ჯგუფის წაევრები განსაზღვრულია სპეციალური მახასიათებლების მიხედვით;

- ინერგება დაბალფასიან ორგანიზაციულ სერვერებზე, რომლებიც UNIX ან Windows / NT– ზეა დაფუძნებული;

- კატეგორიზებულია, როგორც ოპერატიულ მონაცემთა სისტემეხზე დამოკიდებული ან დამოუკიდებელი წყარო, ან გარე საინფორმაციო პროვაიდერებზე, ან ორგანიზაციაში ადგილობრივად არსებულ მონაცემების მიხედვით. თუმცა მონაცემთა „მარკეტი“ დამოკიდებულია უშუალოდ მონაცემთა საცავიდან მიღებულ მონაცემებზე;

- მონაცემთა „მარკეტში“ მონაცემთა წარმოდგენა ხდება შეჯამებული სახით

➢ *ვირტუალური მონაცემთა საცავი*

- წარმოადგენს სხვადასხვა ოპერატიულ მონაცემთა ბაზების ხედებს;

- შესაძლებელია მხოლოდ რამდენიმე ხედის გაერთიანება და მატერიალიზება შემდგომი დამუშავებისთვის;

- იოლია ფორმირება, მაგრამ ასეთი სახის მონაცემთა დამუშავებისთვის საჭიროა დიდი სიმძლავრის ოპერატიულ მონაცემთა სერვერი.

1.6. ინტეგრირებული საინფორმაციო მართვის სისტემების დაპროექტება

პროგრამული ინდუსტრიის არნახულად სწრაფი განვითარების ფონზე მონაცემთა საცავები განაწილებული მართვის საინფორმაციო სისტემების ერთ-ერთ მთავარ კომპონენტია. იგი მოიცავს საავტომატიზაციო მართვის ობიექტის სხვადასხვა ტიპის საინფორმაციო ბლოკებს: ტექსტები და სუპერტექსტები, ცხრილები და გრაფიკული დიაგრამები, აუდიო და ვიზუალური მასალები და სხვ. ამგვარად, მონაცემთა საცავი აღნიშნული ტიპების ფაილთა მართვის სისტემის რთული კონგლომერატია.

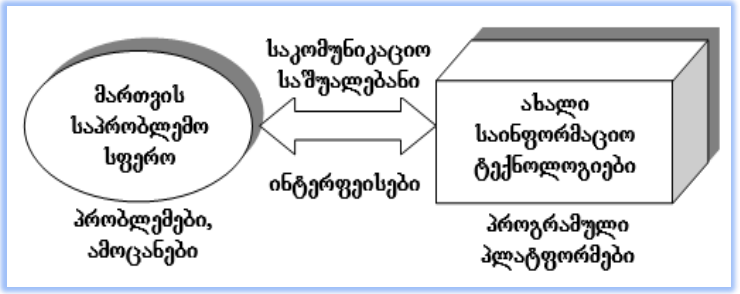
პირველადი წყაროებიდან მონაცემთა შეგროვება, მათი გაფილტვრა (მეთოდურად, სემანტიკურად და ტექნიკურად), ტრანსფორმაცია და კონვერტაცია (მონაცემთა წინასწარ განსაზღვრული სტრუქტურების მისაღებად), მეტაინფორმაციის იერარქიულად ორგანიზება (მონაცემთა კატალოგების და არქივების მართვის მიზნით), გამოყენებითი და სტანდარტული პროგრამული უზრუნველყოფის შექმნა (მონაცემთა პრაგმატული დამუშავებისთვის), ვებ-გვერდების უახლესი საინფორმაციო ტექნოლოგიებით დაპროექტება (ფართო მომხმარებელთა მარტივი ინტერფეისების ასაგებად) – ყველაფერი გათვალისწინებულია ინტეგრირებული ქსელური საინფორმაციო სისტემების დაპროექტების პროცესში.

განაწილებული მართვის ავტომატიზებული სისტემის კომპიუტერული რესურსების ეფექტური მართვის საკითხების კვლევა (ქსელის არხები, კლიენტ-სერვერული და სერვის-ორიენტირებული არქიტექტურები, ინფორმაციული ბაზები და ფაილები, მათი დაცვის, განახლების მოდულები და ა.შ.) – უმნიშვნელოვანეს ამოცანათა კლასს მიეკუთვნება თანამედროვე ინტეგრირებული სისტემების ასაგებად ციფრული ტექნოლოგიების ბაზაზე.

ჩვენი ნაშრომის მიზანი სწორედ ასეთი საკითხების კვლევა, ანალიზი და დამუშავებაა.

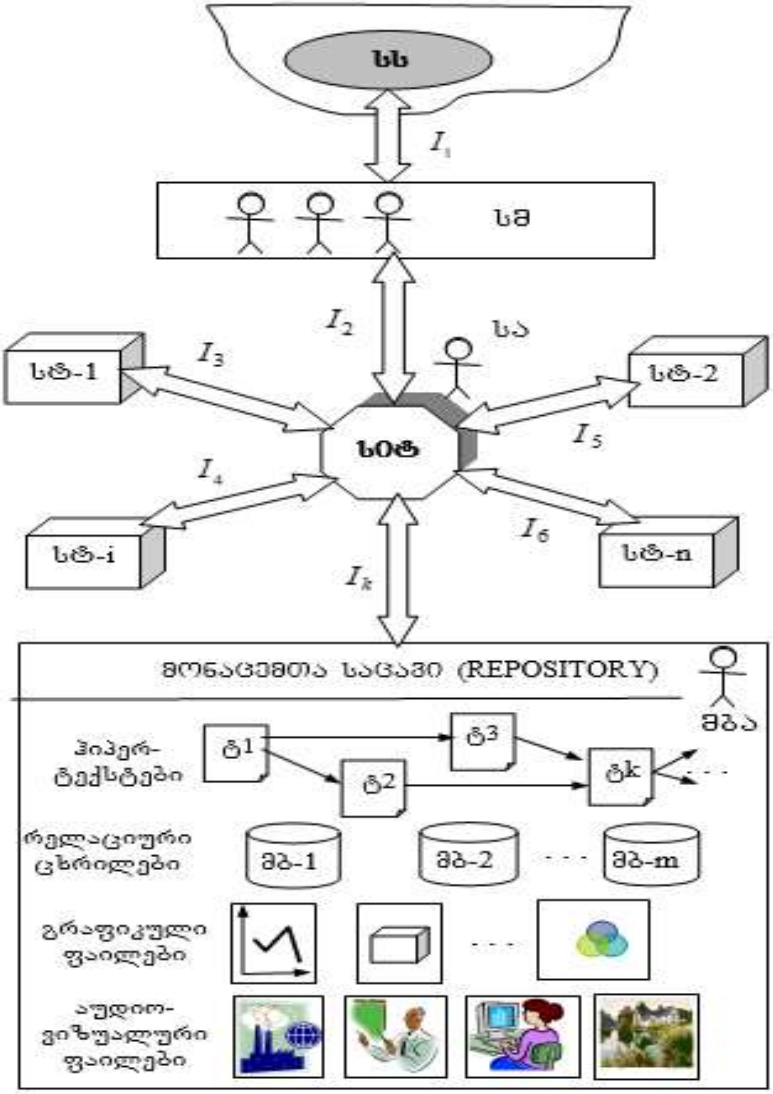
განაწილებული სისტემების პროექტირებისა და პროგრამული რეალიზაციისთვის ვიყენებთ სტრუქტურულ, ობიექტ-ორიენტირებულ და ვიზუალურ მეთოდებს, უნიფიცირებული მოდელირების ენის (UML) და მოქნილ (Agile) მეთოდოლოგიებს [25].

1.8. ნახაზზე მოცემულია „ადამიანი-მანქანა“ ტიპის ინტეგრირებული საინფორმაციო მართვის სისტემის (ისმს) ზოგადი სტრუქტურული სქემა.



ნახ.1.6. ინტეგრირებული საინფორმაციო მართვის სისტემის ზოგადი სქემა

კვლევის საგანი მოცემულ კონტექსტში ამ სისტემის კომპონენტთა (ინფორმაციული ბლოკები, პროგრამული პაკეტები, ინტერფეისები და სხვ.) სიმრავლე და მათი ურთიერთკავშირებია თანამედროვე საკომუნიკაციო და პროგრამული გარდამქმნელების ბაზაზე. 1.9 ნახაზზე მოცემულია ზემოგანხილული ინტეგრირებული საინფორმაციო მართვის სისტემის გაშლილი ზოგადი სქემა. საპრობლემო სფერო (სს) ჩვენ შემთხვევაში ელექტრონული ბიზნესის ან კომერციის კომპლექსური ობიექტია (მაგალითად, დიდი სავაჭრო ცენტრები – ფიზიკური ან ონლაინი) [9, 26].



ნახ.1.9. ინტეგრირებული საინფორმაციო მართვის სისტემის გაშლილი სქემა

სისტემის მომხმარებლები (სმ) კლასიფიცირდება მათი ფუნქციური დანიშნულების მიხედვით (მაგალითად, სისტემის ადმინისტრატორი, მონაცემთა საცავის ადმინისტრატორი, საბოლოო მომხმარებლები – სავაჭრო ცენტრის ხელმძღვანელები და ფუნქციურ ქვედანაყოფთა სპეციალისტები და ა.შ.). I1 – ინტერფეისია საპრობლემო სფეროს და სისტემის მომხმარებლებს შორის.

- ***სიტ*** (საბაზო ინფორმაციული ტექნოლოგია) – განაწილებული საინფორმაციო მართვის სისტემის ძირითადი პროგრამული პაკეტების კრებულია. ესაა ოპერაციული სისტემა (მაგალითად, Ms_Windows და .NET პლატფორმები) და მაღალი დონის პროგრამული ენები, რომელთაც აქვს ობიექტ-ორიენტირებული მეთოდები და ინსტრუმენტული საშუალებანი (მაგალითად, C#.NET, Visual Basic.NET, JavaScript.NET და სხვ.);
- ***სტi, i=1,n*** (საინფორმაციო ტექნოლოგიები) – შეიძლება განვიხილოთ ფართო სპექტრით. კონკრეტული ინტეგრირებული საინფორმაციო მართვის სისტემის დაპროექტებისას, ობიექტ-ორიენტირებული ანალიზის ეტაპზე დაზუსტდება აუცილებელი პროგრამული პაკეტებისა და მათ შორის საკომუნიკაციო ინტერფეისების შედგენილობა. მაგალითად, .NET საბაზო პლატფორმის შემთხვევაში ინფორმაციული ტექნოლოგიები შეიძლება იყოს C#.NET, Visual Basic.NET, MsOffice, ASP.NET (Web-აპლიკაციისათვის), ADO.NET, მონაცემთა ბაზების მართვის სისტემების სახით გამოიყენება SQL და NoSQL ტიპის ბაზები (SQL Server, Oracle, MySQL, MongoDB, Redis და ა.შ.).

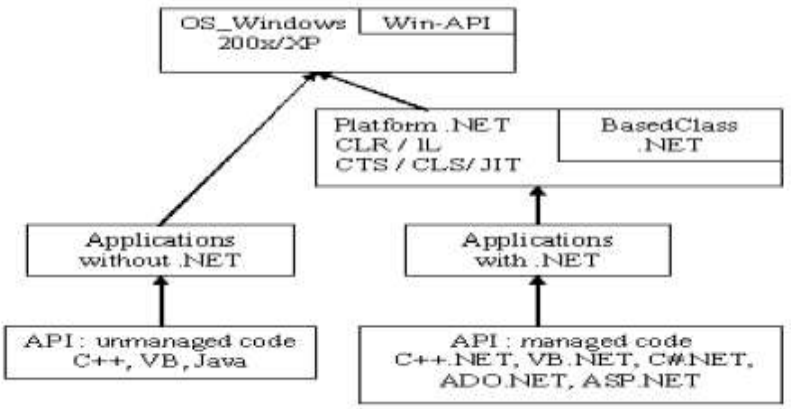
ყოველ ინფორმაციულ ტექნოლოგიას აქვს საბაზო ტექნოლოგიასთან კომუნიკაციის საინტერფეისო (Ij, j=4,J) ენა [26].

მონაცემთა საცავებისა და ბაზებიდან ინფორმაციის ამოსაღებად ფართოდაა მიღებული კლასიკური (სტანდარტული სტრუქტურული მოთხოვნების) SQL-ენის ან შედარებით ახალი, Json ფორმატის (key-value) ტიპის ინსტრუმენტების გამოყენება.

სხვადასხვა ინფორმაციული ტექნოლოგიების (მაგალითად, დაპროგრამების ენები, სხვადასხვა ბაზები) გამოყენებისას მონაცემების ან კოდების თავსებადობისათვის შეიძლება CORBA (Common Object Request Broker Architecture) ან DCOM (Distributed Component Object Model) ტექნოლოგიების გამოყენება [27].

მაიკროსოფთის კორპორაციამ .NET პლატფორმის ბაზაზე, ორიგინალურად მოახერხა და გაამარტივა ეს პროცესი. მან შექმნა შუალედური გარდაქმნის IL (Intermediate Language) ენა. პროგრამები, რომელთა საწყისი კოდები დაწერილია, მაგალითად, C++, C# ან .NET-ის IL-ის ენაზე, კომპილირდება მანქანურ კოდში, dll (dynamic link library) ფაილების სახით [28].

.NET-ის ყველა ენა ფლობს CTS (Common Type System) მონაცემთა შესათანხმებელ საერთო ტიპებს, რათა ენების სტანდარტიზაცია იქნეს მიღწეული. ამგვარად, ობიექტური dll-კოდები IL-ენის საშუალებით ისე მიიღება, რომ მათში არაა დაფიქსირებული, თუ რომელ ენაზეა დაწერილი საწყისი კოდი. .NET პლატფორმა მდებარეობს Windows ოპერაციულ სისტემასა და საავტომატიზაციო სისტემის აპლიკაციას შორის (იხ. ნახ.1.10).



ნახ.1.10. .NET-ის კონცეფცია

1.7. ელექტრონული ბიზნესი და ელექტრონული კომერცია

ელექტრონული ბიზნესი (electronic business, e-business, Online Business) მარკეტინგული სისტემაა ინტერნეტული სერვისებითა და ტექნოლოგიებით. ეფექტიანი საქმიანობისა და მეტი მოგების მიღების მიზნით, ფუნქციონირების პროცესში იგი იყენებს ინტერნეტ/ინტრანეტ საინფორმაციო ქსელებს კომპანიის შიგა და გარე კავშირებისათვის, ახორციელებს სამეურნეო, პარტნიორულ და საშუამავლო ურთიერთობებს და ა.შ.

ელექტრონული ბიზნესი კორპორაციებისა და ფირმებისათვის უზრუნველყოფს Web-საიტების შექმნას, ბიზნეს-პროცესების ინტეგრაციას, მათ კავშირს დამკვეთებთან, მიმწოდებლებთან და მომხმარებლებთან, ბაზრის მასშტაბების გაფართოებას, კომერციული ტრანზაქციების ავტომატიზაციას თანამედროვე ციფრული ტექნოლოგიებისა და ინფორმაციული სისტემების ბაზაზე.

ბიზნესის ავტომატიზაციის მიზნით გამოიყენება საინფორმაციო სისტემების რამდენიმე კლასი [9, 29]:

- ERP (Enterprise Resource Planning) – საწარმოს რესურსების მართვის სისტემა;
- CRM (Customer Relationship Management) – მომხმარებელთან ურთიერთობის მენეჯმენტი;
- BI (Business Intelligence) – ბიზნეს ინფორმაციის მოპოვება, ანალიზი და წარმოდგენა;
- ECM (Enterprise Content Management) – კორპორატიული კონტენტ (დოკუმენტების) მენეჯმენტი;
- HRM (Human Resource Management) – პერსონალის მენეჯმენტის ინფორმაციული სისტემა;
- SCM (Supply Chain Management) – მიწოდების ჯაჭვის მენეჯმენტი.

ელექტრონული კომერცია (electronic commerce, e-commerce) არის ელექტრონული ბიზნესის ერთ-ერთი მთავარი შემადგენელი ნაწილი. იგი ტექნიკური და ორგანიზაციული ფორმების ერთობლიობაა, რომლის საშუალებითაც შესაძლებელია მატერიალური და ფინანსური აქტივების გადაცემა ბიზნესის ერთი სუბიექტიდან მეორეზე. ბიზნეს-ურთიერთობები იყოფა ოთხ კატეგორიად:

- Business-to-Business (B2B) – ელექტრონული კომერციაა საწარმოებს შორის, რომელიც მოიცავს ყველა დონის ინფორმაციულ კავშირებს კომპანიებს შორის. ასეთი სისტემები ძირითადად გამოიყენება წარმოების მომარაგებასა და მზა პროდუქციის გასაღებაში;

- Business-to-Consumer (B2C) – ელექტრონული კომერციის ვარიანტია, სადაც მყიდველები წარმოადგენენ კერძო პირებს. მაგალითად, ინტერნეტ-მაღაზია; ფასიანი საინფორმაციო სამსახური და ა.შ.;

- Consumer-to-Consumer (C2C) – ელექტრონული კომერციაა სხვადასხვა კერძო პირებს შორის, მაგალითად ელექტრონული აუქციონი;

- Business-to-Government (B2G) – ელექტრონული კომერციაა საწარმოსა და სახელმწიფო ორგანოებს შორის.

ელექტრონული ბიზნესი და კომერცია, კომპანიის საბაზრო სტრატეგიის თვალსაზრისით, გვთავაზობს ისეთი ფორმების არსებობას ინტერნეტში, როგორიცაა ელექტრონული სავიზიტო ბარათი (e-business card), ელექტრონული კატალოგი (e-catalog ან online catalog), ელექტრონული ვაჭრობა (e-trading), ელექტრონული მაღაზია (e-shop), ელექტრონული ბუღალტერია (e-accounting), ელექტრონული ფული (e-cash) და ა.შ.

ჩვენი ცხოვრების დღევანდელი დინამკური რიტმი და ტემპი მრავალ საწარმოოს და, უპირველეს ყოვლისა, თვით ადამიანს

აიძულებს უფრო მოქნილი იყოს გარემოს ცვლილებათა მიმართ. ეკონომიკური საქმიანობის გლობალიზაციის და უნივერსალიზაციის პროცესი სულ უფრო და უფრო იწევა წინა პლანზე. საჭირო ხდება ენერგიის, დროის და, რა თქმა უნდა, ფინანსური რესურსების დაზოგვის ღონისძიებების გატარება, რისი უნიკალური საშუალებაცაა ელექტრონული ბიზნესისა და კომერციის გამოყენება [9,30,31].

ელექტრონული კომერცია (e-comerce), როგორც ცნობილია გამოიყენება ინტერნეტ-ტექნოლოგიის დახმარებით და მომხმარებელს მიეწოდება ინტერნეტ ქსელის საშუალებით.

„E“ – electronic (ლათ. სწრაფი);

„E“ – economical (ინგლ. ეკონომიური);

„E“ – extended business („ბიზნესი საზღვრების გარეშე˜).

ამ თვალასზრისით ელექტრონული კომერცია არის სწრაფი და ბიზნესის ეკონომიკური სახე, რომელიც არ ცნობს საზღვრებს. ინტერნეტის შესახებაც არსებობს სხვადასხვა მოსაზრებები [32].

თანამედროვე მსოფლიოში საკმაოდ პოპულარულია ელექტრონული კომერციის თემა – კომპიუტერის მონიტორიდან უშუალოდ მსოფლიო ბაზარზე ყიდვა-გაყიდვის წარმოება. მთელი რიგი კომპანიების საშუალებით წარმატებით იქნა ორგანიზებული თანამედროვე გასაღების ბაზარი (მაგალითად, Amazon). ასეთი პროცესების გაფართოებას და განვითარებას ძალზე შეუწყო ხელი „კოვიდ-19“ პანდემიამ, პროდუქტების შეკვეთისა და სახლებში მიტანის სისტემის შექმნამ.

კონკრეტულად, სავაჭრო ორგანიზაციების ინტერნეტულ ქსელში განთავსების მიზეზი რამდენიმეა: ინტერნეტ მაღაზია ფუნქციონირებს მთელი დღე და ღამე (24სთ). მისაწვდომია მსოფლიოს ყველა ქვეყნის მოსახლეობისთვის და უზრუნველყოფს მომხმარებლის ინდივიდუალურ მომსახურებასაც. კორპორაციები IBM, Oracle, Microsoft დიდი ხანია დაინტერესდნენ და

პერსპექტივით უყურებენ ამ მიმართულებას, აუმჯობესებენ თავიანთ პროდუქციას საწარმოებლად და ამ სფეროში დაკავებულ ბიზნესმენებს სთავაზობენ სხვადასხვა პროგრამულ უზრუნველყოფას (IBM NET, Commerce, Oracle ICS და სხვ.). ბოლო პერიოდში მნიშვნელოვნად განვითარდა ღრუბლოვანი ტექნოლოგიების გამოყენება ასეთი კორპორაციების მიერ [30, 33].

ელექტრონული კომერციის სისტემის პროგრამული რეალიზაციის მიზნით აქტიურად ვითარდება ვებ-ტექნოლოგიები და მათი შექმნის ინსტრუმენტული საშუალებები. შეიძლება აღინიშნოს რამდენიმე პროგრამული ენა და ტექნოლოგია, როგორებიცაა (HTML, Java, XML, JavaScript, Angular, ReactJS, TypeScipt და სხვ.), მონაცემთა ბაზების და Web-ინსტრუმენტები (ASP.NET MVC, NoSQL), სხვადასხვა ბრაუზერი (Browsers) – ვებ კვანძების დათვალიერების პროგრამები უზრუნველყოფს მომხმარებელს ინტერნეტში სამუშაოდ, გრაფიკული ბრაუზერები (Chrome, Firefox, Firefox, Edge, Internet Explorer, Opera, Safari) საშუალებას იძლევა მომხმარებელმა დაინახოს და „მოისმინოს“ ის ვებ-გვერდები, რომლებიც შეიცავს დინამიკურ ინფორმაციას, აუდიო და ვიდეო ფაილებს და ა.შ. [25, 34-36].

ვებ-სერვერების უმრავლესობას შეუძლია ერთდროულად დაამუშაოს რამდენიმე მოთხოვნა, ე.ი. ერთ ვებ-კვანძს შეუძლია ერთდროულად რამდენიმე მომხმარებელი მიიღოს.

ვებ-სერვერს და მომხმარებელს ესაჭიროება ინფორმაციის გაცვლის ერთიანი პროტოკოლი (Hyper-Text Transfer Protocol – http). გადასაცემი ინფორმაციის უსაფრთხოების (დაცვის) მიზნით იყენებენ ტექსტის შიფრაციას (https პროტოკოლს). კლიენტ-სერვერის ურთიერთობის დროს ვებ გვერდის მოთხოვნა იგზავნება ასეთ ფორმატში, ვებ სერვერი ინტერპრეტაციას უკეთებს ამ მოთხოვნას და უბრუნებს ბრაუზერს მონაცემებს.

ამ თვალსაზრისით განსაკუთრებით საყურადღებოა დღეს

ფართოდ გამოყენებული front-end და back-end ტექნოლოგიები. მათი დეველოპერები მუშაობენ ვებსაიტის სხვადასხვა მხარეს [35]:

- front-end ინტერფეისია მომხმარებლისათვის კლიენტის მხარეს. ბრაუზერი კითხულობს და შეუძლია ეკრანზე გამოტანა ან ამუშავება. ასეთი სისტემებია: HTML, CSS და JavaScript.

– HTML (Hyper Text Markup Language) ბრაუზერს აწვდის გვერდის შედგენილობას, მაგალითად, სათაური, პარაგრაფი, სია, ელემენტები და ა.შ.;

– CSS (Cascading Style Sheets) ბრაუზერს კარნახობს თუ როგორ ასახოს ელემენტები ეკრანზე. მაგალითად, „პირველი პარაგრაფის შემდეგ აბზაცი 25 პიკსელი“ ან „body ელემენტში მთლიანი ტექსტი იყოს ლურჯი და დაწერილი Sylfaen ფონტით“.

JavaScript აძლევს ბრაუზერს მითითებებს, თუ როგორ იმოქმედოს გარკვეულ მოვლენებზე. ამისათვის იგი იყენებს დაპროგრამების მსუბუქ ენას და ცვლის ეკრანზე გვერდის შიგთავსს.

- back-end გამოიყენება დეველოპერების მიერ სერვერის მხარეს. მისთვის გამოიყენება სერვერზე არსებული ყველა ინსტრუმენტი, რომელიც უზრუნველყოფს შეტყობინებების მომსახურებას. აქ გამოიყენება დაპროგრამების უნივერსალური ენები: Java, JavaScript/Node, PHP, Python და სხვ.

მონაცემთა ბაზების მართვის სისტემები (Database) გამოიყენება სერვერის მხარეს დაპროგრამების ენებთან და ტექნოლოგიებთან ერთად. აქ მოიაზრება რეალაციური და NoSQL მონაცემთა ბაზები: MsSQL_Server, PostgreSQL, Oracle, MySQL, MariaDB, MongoDB, Cassandra, Redis და სხვ. [36].

საჭიროდ მიგვაჩნია აქვე Node.js პროგრამული პლატფორმის ხსენება, რომელიც თავიდან შეიქმნა Google Chrom-ის JavaScript სამუშაო გარემოსთვის სწარაფი და მასშტაბირებადი ქსელური აპლიკაციების მარტივად ასაგებად [37]. მისი V8 შესრულების

მექანიზმი, რომელიც დაწერილია C++ ენაზე, უზრუნველყოფს JavaScript კოდის კომპილირებას მანქანურ კოდში.

Node.js გამოიყენება ძირითადად სერვერის მხარეს, როგორც back-end. შესაძლებელია მისი გამოყენება ასევე მომხმარებლის მხარესაც, როგორც front-end, მაგრამ თანამედროვე აქტუალური ენების (მაგალითად, Angular და სხვ.) პოპულარულ ინტერფეისებში ეს არაა რეკომენდებული.

Node.js ხელს უწყობს JavaScrip-ის შესაძლებლობების გაფართოებას C++ და სხვა ენების გარე ბიბლიოთეკების მისაერთებლად [37].

ფირმა Visa International და კომპანია Master Card International პროტოკოლში SET (Secure Electronic Transaction – დაცული ელექტრონული ტრანზაქცია) ხედავს ინტერნეტში ვაჭრობის ძლიერ ინსტრუმენტს, რომელიც აუმჯობესებს ქსელში ჩატარებული ოპერაციების უსაფრთხოებას და აზღვევს გარიგებაში მონაწილე ორივე მხარეს [38]. ინტერნეტში ელექტრონული ტრანზაქციების ჩატარება ხდება SSL-ის (Secure Socket Layer) გამოყენებით. იგი არის შიფრაციის პროტოკოლი და პირველ რიგში გამოიყენება ქსელში გადაცემული ინფორმაციის დასაცავად.

ელექტრონული კომერციის პრობლემები კი უფრო ფართო სპექტრს შეიცავს, ვიდრე შიფრაცია. აქ შეიძლება საქმე ეხებოდეს ისეთ პრობლემებს, როგორიცაა კლიენტის მიერ შეთანხმების პირობის ან მაღაზიის სინამდვილის შემოწმება. ამიტომ მსოფლიოს უდიდესი კომპანიები დიდ კაპიტალს აბანდებენ ამ მიმართულების გასავითარებლად და რადიკალურ ფორმებს მიმართავენ მსოფლიო ბაზარზე საკუთარი ადგილის დასამკვიდრებლად,

თუმცა მომავალი ციფრული ეკონომიკის ბირთვი (Digital economy - ციფრული ეკონომიკა) არის ეკონომიკური აქტივობების, კომერციული ტრანზაქციებისა და პროფესიული ურთიერთქმე-

დებების მსოფლიო ქსელი, რომელიც უზრუნველყოფილია საინფორმაციო და საკომუნიკაციო ტექნოლოგიებით (ICT) [39].

იგი ციფრულ ტექნოლოგიებზე დაფუძნებული ეკონომიკაა. არის არა მხოლოდ სავაჭრო, არამედ საქმიანი ტრანზაქციების ჩატარების საშუალება, რაც გულისხმობს ინდივიდუალურ და კორპორაციულ კონტრაქტებს, ინვესტიციების და კაპიტალის გადაადგილებას. ციფრული ეკონომიკა და ინტერნეტული კომერცია, როგორც გლობალური მოვლენები, განსაკუთრებულ გავლენას ახდენს თავის შედეგებით სოციალურ გარემოზე. ინდივიდუალური სფერო ინტერნეტ-შეკვეთებით, დოკუმენტბრუნვის პროცესების ავტომატიზაცია ორგანიზაციებში და ელექტრონული მთავრობა ამის კარგი მაგალითებია.

1.8. ელექტრონული კომერციის ინფრასტრუქტურული სისტემა

კომპიუტერული ქსელის ინფრასტრუქტურა მნიშვნელოვნად განსაზღვრავს ელ-კომერციის ავტომატიზებული სისტემის ფუნქციონირების ხარისხს. იგი მოიცავს ინფორმაციის გადაცემის მატარებლებს. ამიტომ მასში შედის ინტერნეტი, საკაბელო ტელევიზია, ტელეკომუნიკაციური და კერძო კორპორაციული ქსელები და ა.შ.

კორპორაციული (საწარმოო) ინფრასტრუქტურა ორიენტირებულია პროდუქციაზე და იმ რესურსებზე, რაც საჭიროა მის შესაქმნელად. გაყიდვის ინფრასტრუქტურის მიზანია შეათანხმოს მომხმარებელთან საქონელი და მომსახურება. მომსახურების ინფრასტრუქტურა კი მოიცავს გაყიდვის, მისი შემდგომი მხარდაჭერის და უსაფრთხოების პროცესებს [5].

ქსელის მუშაობა მთლიანად დამოკიდებულია პროტოკოლებზე – ქსელის მუშაობის წესებზე. პროტოკოლები განსაზღვრავს. თუ როგორ უკავშირდება ელ-კომერციის გამოყენებითი

ნაწილი ქსელს, თუ როგორ იყოფა მონაცემები გამოყენებით პაკეტებად საკაბელო გადაცემებისთვის. ამასთანვე როგორი ელექტრონული სიგნალები წარმოადგენს მონაცემებს ქსელურ კაბელში. მონაცემები დაჯგუფებულია ციფრული ქსელებისათვის გადასაცემად. პაკეტები ამ მონაცემების გარდა შეიცავს მათზე საკონტროლო ინფორმაციასაც. პროტოკოლის შემუშავებისას განისაზღვრება ის, თუ როგორ გაცვლის იგი მონაცემებს მეზობელ დონეებს შორის. მეცნიერულად დასაბუთებული და ეფექტურად ფუნქციონირებადი ელ-კომერციული სისტემა გვთავაზობს ძირითადი ელემენტების ჯგუფს, რომელიც უზრუნველყოფს კონკრეტული პირობების შესრულებას.

ელექრტონული კომერციის ძირითადი ელემენტები იყოფა ორ ჯგუფად: პირველში განთავსებული ელემენტები არის ელექრტონული კომერციის ინფრასტრუქტურული სისტემის ყველა კომპონენტი, ხოლო მეორე ჯგუფისა კი – მათი რეალიზაციის ორგანიზაციული ფორმის მრავალსახეობა. ეს ძირითადი ელემენტები მნიშვნელოვნად განსხვავდება ტრადიციული მაღაზიების შემთხვევისაგან. ელ-კომერციის სისტემა იყენებს აბსოლიტურად განსხვავებულ ბიზნეს-საშუალებებს [9]. ელექრტონული კომერციის ინფრასტრუქტურული სისტემის ძირითადი ელემენტებია:

- სპეციალური პროგრამული უზრუნველყოფა;
- მონაცემთა ბაზების მართვის სისტემა;
- სატელეკომუნიკაციო კავშირები;
- უსაფრთხოების სისტემა საქონლის ყიდვა-გაყიდვის და მომსახურების სფეროში;
- იურიდიული სამართლიანობის უზრუნველყოფა;
- ვირტუალური საბაზო სისტემა;
- სპეციალური საგადამხდელო სისტემა;
- საწყობის ავტომატური მომსახურება;
- საქონლის მიწოდებისა და მომსახურეობის სისტემა;

- საფინანსო ინსტიტუტები (საბროკერო და სხვ.);
- საგადასახადო სისტემა და საბაჟო ტარიფები;
- მარკეტინგული მომსახურეობა, რომელიც მოიცავს: სარეკლამო თუ ფასების რეგულირების და დიზაინის განყოფილებებს: (ვებ-გვერდები, ვებ-სერვერი).

დავახასიათოთ ზოგიერთი კომპონენტი მათი არსის გასაგებად:

- *ელექტრონული მაღაზია* – განთავსებულია ინტერნეტ-ქსელში, ვებ-სერვერის სახით. ასეთი სახის მაღაზიის შექმნის მთავარი მიზანია უზრუნველყოს, დროის მინიმალურ შუალედში საქონლის გასაღება და დახმარება გაუწიოს ინტერნეტის სხვა მომხმარებლებს, რათა იაფად და სწრაფად შეარჩიონ მათთვის სასურველი პროდუქცია, ასევე ყველა მომხმარებელს საშუალება აქვს აწარმოოს შეკვეთები;
- *სპეციალური პროგრამული უზრუნველყოფა* – მისთვის გამოიყენება შემდეგი ენები: Java, HTML, XML, Javascript და ა.შ. მონაცემთა შეტანა-გამოტანის შაბლონები, დიზაინი და ვებ-გვერდების მომზადების საშუალება, სპეციალური პროგრამული უზრუნველყოფა და ა.შ.;
- *იურიდიული უზრუნველყოფა* – ელექტრონული კომერციის ორგანიზაცია, პირველ რიგში ბაზირებული უნდა იყოს ტრადიციული იურიდიული ნორმებით და წესებით, მეორე რიგში კი უნდა მოხდეს ახალი სპეციალიზირებული სამართლის ინსტიტუტების მიერ მიღებული ცვლილებების და პროცედურების გათვალისწინება. ამას გარდა აქტუალურია კანონმდებლობის უნიფიკაცია, აგრეთვე პროცედურების და წესების გამარტივება, რომლებიც გამოიყენება სხვადასხვა ქვეყნებში. იგი ხელს უწყობს თანამშრომლობას ბიზნესის წარმოებასა და შესაბამის სახელმწიფო მართვის სტრუქტურებს შორის;
- *სპეციალური საგადასახადო სისტემა* – ინტერნეტის საშუალებით გადახდის წარმოება ხორციელდება სხვადასხვა ბარათების

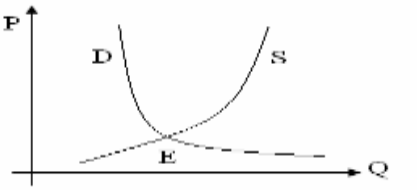
დახმარებით. ინტერნეტ ქსელის შესაძლებლობების გამოყენება მომხმარებლის უშუალოდ აქტიური ჩარევით. ეს შესაძლოა იყოს, როგორც მინიმუმი ვებ-გვერდი ან საკუთარი ვებ-სერვერი, ხოლო როგორც მაქსიმუმი – უსაზღვრო შესაძლებლობები, რაც გამოიხატება ინტერაქტიული მაღაზიების არსებობაში და მსხვილი ფირმების ხელმძღვანელებთან უშუალო კავშირებში.

ეკონომიკაზე საუბრისას მასში მიმდინარე პროცესების შესახებ, აუცილებელია გავითვალისწინოთ რამდენიმე ძირითადი პრინციპი:

ეკონომიკურ მოვლენას განსაზღვრავს მრავალი ცვალებადი ფაქტორი, ამიტომ ამ მოვლენის შესწავლის გასაადვილებლად აფიქსირებენ (უცვლელად თვლიან) ყველა ფაქტორს გარდა ერთისა და სწავლობენ თუ როგორ რეაგირებს მოვლენა ამ ერთი ფაქტორის ცვლილებაზე სხვა ფაქტორების უცვლელობის პირობებში.

მაგალითად, ყველასათვის ცნობილია, რომ მიწოდება, მოთხოვნა და ფასი ურთიერთდაკავშირებულია. იზრდება ფასი კლებულობს მოთხოვნა და პირიქით კლებულობს ფასი მოთხოვნა იზრდება, ეს ურთიერთკავშირი ეკონომიკაში აქსიომად ითვლება (არის გამონაკლისი შემთხვევები, როდესაც ეს პრინციპი ირღვევა).

მოთხოვნის და მიწოდების განმსაზღვრელი ცენტრალური ფაქტორი ფასია. გამყიდველებისთვის უკეთესია მღალი ფასი, ხოლო მყიდველისათვის – დაბალი, თუ ყველა სხვა პირობა ერთნაირია. თუ დავადებთ ერთმანეთს გრაფიკებს, მოთხოვნის და მიწოდების მრუდები გადაიკვეთება (ნახ.1.11).



ნახ.1.11. მოთხოვნა-მიწოდების მრუდები

D – მოთხოვნის მრუდი; S – მიწოდების მრუდი; P- ფასების ღერძი; Q - მიწოდებული საქონლის რაოდენობის ღერძი; E- წონასწორობის წერტილი

მოცემული ნახაზიდან ჩანს, რომ არსებობს ერთადერთი ფასი, რომლის დროსაც მომხმარებელის და მიმწოდებელის ინტერესები ერთმანეთს ემთხვევა. ამ ფასს წონასწორულ საბაზრო ფასს (Market Equilibrium Price) უწოდებენ. იგი ზუსტად იმ წერტილშია, სადაც გადაიკვეთება მოთხოვნისა და მიწოდების მრუდები. წონასწორობის წერტილი – ის ადგილია, რომელშიც მოთხოვნისა და მიწოდების სიდიდეები ტოლია, ხოლო საბაზრო ფასი - ეს ისეთი ფასია, რომელიც განისაზღვრება მოთხოვნისა და მიწოდების საშუალებით.

1.9. მონაცემთა საცავის ანალიზის OLAP-ინსტრუმენტი

მონაცემთა მრავალგანზომილებიან კომპლექსურ ანალიზს უწოდებენ OLAP (Online Analytical Processing) ტექნოლოგიას, რომელიც განიმარტება როგორც „მონაცემთა ოპერატიული ანალიზი“. მონაცემთა საცავში იგი მნიშვნელოვანი საკვანძო კომპონენტია [9,40].

OLAP ინსტრუმენტი პირველად მონაცემებს წარმოადგენს ინფორმაციის სახით, რომლითაც შესაძლებელია წარმოების მოცულობის შესახებ მივიღოთ რეალური სურათი. იგი არის უნიკალური, რადგან იძლევა საშუალებას:

1) ჩატარდეს ინფორმაციის მრავალგანზომილებიანი ანალიზი სხვადასხვა ჭრილში;

2) სწრაფად განხორციელდეს ბიზნესის ძირითადი მეტრიკების გამოთვლა, დაგეგმვის და პროგნოზირების ფუნქციები, მონაცემთა დიდი მოცულობის „თუ . . . , მაშინ . . ." ანალიზი.

რატომაა საჭირო OLAP ტექნოლოგია მონაცემთა საცავში ?

ბიზნეს მომხმარებლებისთვის იგი იძლევა სწრაფ და ინტუიციურ წვდომას ცენტრალიზებულ მონაცემებზე და დაკავშირებულ გამოთვლებზე ანალიზისა და რეპორტების მიზნებისათვის. OLAP გადაწყვეტა IT-ისათვის აძლიერებს მონაცემთა საცავს ან სხვა რელაციურ მონაცემთა ბაზას მთლიანი მონაცემებით და ბიზნეს-გამოთვლებით.

1.9.1. OLAP-კუბი – მონაცემთა მრავალგანზომილებიანი წარმოდგენა

ინფორმაციის ონლაინ ანალიზის OLAP ინსტრუმენტს აქვს საკუთარი კონცეფცია, პრინციპები და მოთხოვნები, რომლებიც კარგადაა ადაპტირებული შესაბამის პროგრამულ პროდუქტში და ანალიტიკოსებს საგრძნობლად უადვილებს მონაცემებთან წვდომას და ურთიერთობას.

ბიზნეს-ანალიტიკოსი პროგრამული სისტემის განსაკუთრებული მომხმარებელია კორპორაციული ინფორმაციით (ცოდნით). მისი ამოცანაა „წესრიგის დაამყარება“ დიდ მონაცემთა საცავში. მისთვის მნიშვნელოვანი არაა ცალკეული ფაქტები. მონაცემთა საცავში არსებული ფაქტებით შეიძლება დაინტერესდეს, მაგალითად, კომპანიის ბუღალტერი ან რომელიმე განყოფილების ხელმძღვანელი, რომელთა კომპეტენციაშიც შედის საქმიანი შეთანხმებები. ანალიტიკოსისათვის კი ერთი რომელიმე სახის მონაცემი თითქმის არაფერს წარმოადგენს, მისი ინტერესის სფერო

შეიძლება გახდეს დროის რაღაც შუალედში (მაგალითად, ერთი თვის, წლის) რომელიმე მსხვილი ობიექტის, ყველა სახის საქმიანი შეთანხმება. ამავდროულად, მას შეუძლია არ გაამახვილოს ყურადღება ისეთ ინფორმაციაზე, როგორიცაა მომხმარებლის მისამართი, მობილურის ნომერი, კონტრაქტის ინდექსი და ა.შ.

მისთვის საჭირო ინფორმაცია წარმოდგენილი უნდა იყოს რიცხვითი მნიშვნელობებით, ეს განაპირობებს მისი მოქმედების ძირითად არსს. ეს უკანასკნელი მიუთითებს იმაზე, რომ ანალიტიკოსი მუშაობს 1.1 სახის ცხრილთან.

ქვეყანა და საქონელი - გაყიდვების მოცულობით ცხრ.1.1

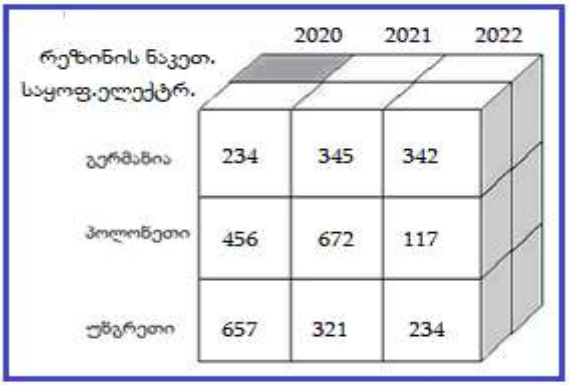
ქვეყანა	საქონელი	წელი	ყიდვა-გაყიდვის მოცულობა
გერმანია	საყოფაცხ_ელექტრონიკა	2020	234
პოლონეთი	საყოფაცხ_ელექტრონიკა	2020	456
უნგრეთი	საყოფაცხ_ელექტრონიკა	2021	321
ერმანია	საყოფაცხ_ელექტრონიკა	2022	342
პოლონეთი	რეზინის ნაკეთობა	2022	123
უნგრეთი	საყოფაცხ_ელექტრონიკა	2020	657
პოლონეთი	რეზინის ნაკეთობა	2021	143
გერმანია	რეზინის ნაკეთობა	2021	345
პოლონეთი	რეზინის ნაკეთობა	2020	56
უნგრეთი	საყოფაცხ_ელექტრონიკა	2022	234
გერმანია	რეზინის ნაკეთობა	2022	309
უნგრეთი	რეზინის ნაკეთობა	2021	345
პოლონეთი	საყოფაცხ_ელექტრონიკა	2022	117
უნგრეთი	რეზინის ნაკეთობა	2020	57
გერმანია	რეზინის ნაკეთობა	2021	152
პოლონეთი	საყოფაცხ_ელექტრონიკა	2022	672
უნგრეთი	რეზინის ნაკეთობა	2022	212

ქვეყანა, საქონელი და *წელი* - ატრიბუტებია, ხოლო ყიდვა-გაყიდვის მოცულობა გამოსახულია რიცხვითი მნიშვნელობით.

ანალიტიკოსის ამოცანაა განსაზღვროს დამოკიდებულება ატრიბუტებსა და რიცხვით პარამეტრებს შორის.

თუ ცხრილს დავაკვირდებით, შევამჩნევთ, რომ მისი წარმოდგენა შესაძლებელია სამ განზომილებაში (ნახ.1.12):

- ერთი ღერძის გასწვრივ დალაგდება *ქვეყნები*,
- მეორეზე *საქონელი*, ხოლო
- მესამეზე - *წელი*.



ნახ.1.12. OLAP კუბი

ნახაზზე განსხვავებული ფერი გამოხატავს სეგმენტს სადაც იმ წელს არ არსებობს მონაცემი. მაგალითად, გერმანიას 2020 წელს რეზინის ნაკეთობებზე არა აქვს ინფომაცია. სწორედ ასეთი სახის სამ განზომილებიან მასივს უწოდებენ კუბს.

აუცილებელი არაა კუბის ყველა ელემენტის შევსება. თუ არ იქნება ინფორმაცია, როგორც აღვნიშნეთ გერმანიაში 2020 წელს რეზინის ნაკეთობებზე, მაშინ შესაბამისი უჯრედი უბრალოდ არ იქნება განსაზღვრული OLAP დანართში და ამავე დროს არც არის

საჭიროება დანართში მოხვდეს მრავალგანზომილებიანი სტრუქტურის ყველა ელემენტი. მთავარია მომხმარებელმა მიიღოს მხოლოდ საჭირო მონაცემები. კუბში მრავალგანზომილებიან მონაცემთა კომპაქტური შენახვა განაპირობებს, რომ არ მოხდეს მეხსიერების უსარგებლო დაკარგვა არასასურველ ინფორმაციაზე.

თუ ნამდვილ კუბს და OLAP-კუბს შევადარებთ, მაშინ მათ შორის შევნიშნავთ განსხვავებას – ნამდვილი კუბის ელემენტების რაოდენობა ყველა განზომილებაში ერთნაირია, ხოლო OLAP-კუბი არ არის ზუსტად განსაზღვრული. იგი შეიძლება იყოს ორ-, სამ-, ოთხ- და ა.შ. n-განზომილებიანი, ეს დამოკიდებულია იმაზე, თუ როგორი სახის ამოცანასთან გვაქვს საქმე.

OLAP კუბის თითოეული განზომილება შედგება ე.წ. შრეების ან ცალკეული ნაწილებისგან. მაგალითად, განზომილება „ქვეყანა~ შედგება შრეებისგან: გერმანია, პოლონეთი, უნგრეთი და ა.შ.

თვითონ მრავალგანზომილებიანი კუბის გამოყენება ანალიზისათვის შეუძლებელია. მაგალითად, თუ მოცემული გვაქს ექვს- ან ცხრამეტგანზომილებიანი კუბი, მათი ადეკვატური გარდაქმნა სირთულეებთანაა დაკავშირებული. ამიტომ გამოყენების წინ მრავალგანზომილებიანი კუბიდან გამოყოფენ, ორგანზომილებიან ცხრილს ამ ოპერაციას უწოდებენ კუბის „გაკვეთას“.

ანალიტიკოსები მრავალგანზომილებიან კუბიდან ,ამოკვეთენ~ მათთვის საინტერესო შრეებს. შესაბამისად „გაუკვეთავი“ რჩება ორი განზომილება. როდესაც კუბი შეიცავს რამდენიმე სახის რიცხვით მნიშვნელობას, მაშინ ერთი სახის მნიშვნელობები განთავსდება ცხრილის ერთ განზომილებაში.

1.1 ცხრილში მოცემული მონაცემები პირველადი არაა. ისინი დაჯგუფებულია და წარმოდგენილია სრულყოფილი სახით. მაგალითად, წელიწადი შეიძლება დავყოთ კვარტალებად, კვარტალები – თვეებად, თვეები – კვირეებად, კვირები – დღეებად და ა.შ.,

ასევე, ქვეყანა შედგება რეგიონებისგან, რეგიონები – დასახლებული პუნქტებისგან და ა.შ.

OLAP-ში ასეთი სახის მრავალდონიან გაერთიანებას უწოდებენ იერარქიას. ცალკეული ელემენტი შეიცავს რამდენიმე სახის იერარქიას, მაგალითად, დღე-კვირა-თვე ან დღე-დეკადა-კვარტალი. მონაცემთა დაჯგუფება იწყება ყველაზე დაბალი დონიდან და ბოლოს ჯგუფდება მაღალი დონის მონაცემამდე.

იმისათვის, რომ დაჩქარდეს ერთი მდგომარეობიდან მეორეში გადასვლის პროცესი და მონაცემთა სხვადასხვა დონეზე დაჯგუფება, ყველა მონაცემი უნდა ინახებოდეს კუბში. ერთი შეხედვით მომხმარებელი ხედავს მხოლოდ ერთ კუბს, მაგრამ შეგვიძლია ვთქვათ, რომ ეს კუბი შეიცავს პრიმიტიული კუბების მთელ სიმრავლეს.

1.9.2. იერარქიული გაერთიანებები OLAP-კუბში

OLAP ტექნოლოგიის ძალზე მნიშვნელოვანი, ეფექტური და მაღალმწარმოებლური შესაძლებლობაა იერარქიული გაერთიანებების არსებობა.

ბიზნეს-ანალიტიკოსი, დამოუკიდებლად ან პროგრამისტის დახმარებით დასვავს შესაბამის SQL-მოთხოვნას და შედეგად ღებულობს მონაცემებს ელექტრონული ცხრილის სახით. ამ დროს წარმოიქმნება დაბრკოლებათა მთელი სიმრავლე, საიდანაც შეიძლება გამოვყოთ რამდენიმე მათგანი:

1) ანალიტიკოსი გარკვეული დროის განმავლობაში ელოდება პროგრამისტისგან ინფორმაციის მიღებას;

2) ინფორმაციის მიწოდება ხდება მხოლოდ ერთი ცხრილის საშუალებით, რის გამოც შეუძლებელია ჩატარდეს სრულყოფილი კვლევა. ამიტომ ერთსა და იმავე პროდუქტზე უნდა ჩატარდეს რამდენიმე გამოკვლევა;

3) ბიზნეს-ანალიტიკოსს სჭირდება მხოლოდ მნიშვნელოვანი ინფორმაცია გარკვეულ საკითხზე (უმნიშვნელო მონაცემები არ ჭირდება). ეს იმას ნიშნავს, რომ კორპორაციული რელაციური მონაცემთა ბაზების მართვის სისტემის სერვერმა, რომელსაც ანალიტიკოსი მიმართავს, უნდა მიაწოდოს მას ყველაზე აქტუალური ინფორმაცია და, ამავე დროს, დაბლოკოს სხვა დანარჩენი ტრანზაქციები.

არსებული პრობლემის გადასაჭრელად OLAP-ტექნოლოგია არის ერთ-ერთი საშუალება. OLAP კუბი მეტა-ანგარიშთა ნაკრებია. გაკვეთილი მეტა-ანგარიში (ანუ კუბი) ანალიტიკოსს საშუალებას აძლევს სხვადასხვა განზომილებაში ასახოს მისთვის საინტერესო ორგანიზაციის სტრუქტურული მონაცემები ან ანგარიში.

OLAP ინსტრუმენტის გამოყენებისას სისტემის სრულყოფილი ანალიზის მიზნით მნიშვნელოვანია 5 ძირითად პრინციპი. მას უწოდებენ FASMI ტესტს. აქ კონკრეტულად განსაზღვრულია OLAP პროდუქტის მოთხოვნები.

FASMI აბრევიატურა განისაზღვრება ტესტის შემადგენელი თითოეული პუნქტისაგან:

- Fast (სწრაფი) – OLAP დანართმა უნდა უზრუნველყოს ანალიზური მონაცემების მიღება მინიმალურ დროში. ანალიზი უნდა ჩატარდეს რაც შეიძლება სწრაფად და უნდა მოხდეს ყველა ინფორმაციული ასპექტის გათვალისწინება. ამ პერიოდის ხანგრძლივობა საშუალოდ უნდა იყოს ხუთი წამი ან ურფო ნაკლები;
- Analysis (ანალიზი) – OLAP დანართმა მომხმარებელს უნდა მისცეს საშუალება განახორციელოს რიცხვითი და სტატისტიკური ანალიზი;
- Shared (ერთდროულად გამოყენება) – OLAP დანართმა უნდა უზრუნველყოს ერთსა და იმავე დროს რამდენიმე მომხმარებლის ერთდროული მუშაობა. ამავე დროს უნდა შემოწმდეს, რამდენად დაცულია კონფიდენციალური ინფორმაცია;

- Multidimensional (მრავალგანზომილება) – ეს არის OLAP –ის დამახასიათებელი არსებითი თვისება;
- Information (ინფორმაცია) – OLAP დანართმა მომხმარებელს უნდა მიაწოდოს მისთვის საინტერესო ყველა არსებული და წარმოებული მონაცემი, მიუხედავად იმისა, საჭიროა თუ არა ეს ინფორმაცია პროგრამული დამუშავებისთვის.

1.9.3. აგრეგატულ მონაცემთა შენახვის ასპექტები

მონაცემთა საცავში თავმოყრილი *აგრეგატული მონაცემები* მრავალგანზომილებიანი შენახვის იერარქიის სხვადასხვა დონეზე ერთმანეთისგან განსხვავდება. მაგალითად, ყიდვა-გაყიდვის მოცულობა ერთ დღეში, თვეში, წელიწადში, საქონლის კატეგორია და ა.შ.

აგრეგატულ მონცემთა შენახვის მიზანია მომხმარებელთა მოთხოვნების დაკმაყოფილებაზე დახარჯული დროის შემცირება, რამდენადაც უმეტეს შემთხვევაში ანალიზის და პროგნოზის სფეროს წარმოადგენს არა დეტალური ინფორმაცია, არამედ გაერთიანებული, საბოლო სახემდე მიყვანილი დაჯგუფებული მონაცემები.

ამიტომ, მრავალგანზომილებიანი მონაცემთა ბაზის შექმნის დროს, ყოველთვის გასათვალისწინებელია, რომ შეტანილ იქნეს რამდენიმე აგრაგატული მონაცემი. ასევე ის ფაქტიც, რომ ყველა მონაცემის აგრაგატული სახით შენახვა გაუმართლებელია, რადგან მონაცემთა მოცულობაში ახალი განზომილების დამატების შემთხვევეაში კუბი, სადაც ამ მონაცემის ჩამატება მოხდება, დაიწყებს ზრდას ექსპონენციალურად.

ამ პროცესს უწოდებენ მონაცემთა მოცულობის „ფეთქებად ზომას“, ამიტომ აგრეგატულ მონაცემთა მოცულობის ზომის ხარისხი დამოკიდებულია კუბების რაოდენობაზე და იერარქიის სხვადასხვა დონეზე მიმდინარე გაზომვებზე. „ფეთქებად ზომის“ პრობლემის გადასაჭრელად გამოიყენება სხვადასხვა სახის

მონაცემთა შენახვის სქემები: MOLAP, ROLAP, JOLAP და HOLAP [9].

- MOLAP – (მრავალგანზომილებიანი OLAP) პროდუქტს საფუძვლად უდევს მონაცემთა არარელაციური სტრუქტურა, რომელიც უზრუნველყოფს მონაცემთა მრავალგანზომილებიან შენახვას.
- ROLAP – (რელაციური OLAP) ასეთ ინსტრუმენტში მრავალგანზომილებიანი სტრუქტურა რეალიზებულია რელაციური ცხრილებით, ხოლო მონაცემები ანალიზის პროცესში შესაბამისად შეირჩევა მონაცემთა რელაციური ბაზებიდან ანალიზური ინსტრუმენტით.
- JOLAP – (Java Specification Requet) ინსტრუმენტის გამოყენებისას თითოეული მოთხოვნის მოდელირება შესაძლებელია, როგორც ერთიან ობიექტთა ჯგუფი. მის განხორციელებაში ეხმარება გრაფიკულ ინტერფეისში ოპერაციათა თანამიმდევრული სტრუქტურა და მოთხოვნათა მოდიფიკაციის განსაზღვრა. JOLAP გმოიყენება სხვადასხვა სპეციფიკაციაში, მაგალითად, როგორიცაა: ჯგუფი OMG მეტამოდელისთვის Common Warehouse Metamodel (CWM), ობიექტური მოდელი Meta Object Facility (MOF), XML Metadata Interchange (XMI) და აგრეთვე Java Metadata Interface (JMI, JSR-40) [41].

MOLAP უზრუნველყოფს მაღალმწარმოებლურობას, მაგრამ სტრუქტურები არ შეიძლება გამოვიყენოთ დიდი მოცულობის მონაცემთა დამუშავებისთვის, რადგანაც დიდი გაბარიტები დიდ აპარატულ რესურსებს მოითხოვს. ამასთან ერთად ჰიპერკუბის დატვირთვა შეიძლება იყოს ძალიან მაღალი, რაც შემდგომში თავს იჩენს ინფორმაციის მიწოდების სისწრაფეზე.

შეიძლება ითქვას პირიქითაც, რომ ROLAP უზრუნველყოფს შენახული მონაცემების დიდი მასივების დამუშავებას, ასე, რომ გარანტირებულია უფრო ეკონომიური შენახვა, მაგრამ ამასთან ერთად, მნიშვნელოვან როლს ითამაშებს მრავალგანზომილებიან სამუშაოს შესრულების სიჩქარეზე.

მსგავს მსჯელობას მივყავართ ახალი კლასის გამოყოფაზე, რომელიც არის HOLAP (Hybrid Online Analytical Processing) ანალიზური ინსტრუმენტი. ასეთი ჰიბრიდული ტიპის მონაცემთა ოპერატიულ-ანალიზური დამუშავების ინსტრუმენტი საშუალებას იძლევა შეთანხმდეს ორივე მიდგომა – რელაციურიც და მრავალგანზომილებიანიც. მონაცემთა ბაზის ერთი ნაწილი შენახულ იქნება რელაციურში, ხოლო მეორე – მრავალგანზომილებიანში.

ასეთ შემთხვევაში ხელმისაწვდომი გახდება როგორც MOLAP (მრავალგანზომოლებიანი), ასევე ROLAP (რელაციური) მონაცემების ერთდროული გამოყენება, იმისადა მიხედვით თუ მონაცემთა როგორი ტიპის დამუშავება იქნება უპირატესად გამოსაყენებელი.

არსებობს HOLAP-ის განსხვავებული ტიპები [42]:

- Web-Enabled OLAP (WOLAP) სერვერი – ვრცელდება OLAP აპლიკაციაზე, რომელიც ხელმისაწვდომია ვებ ბრაუზერის საშუალებით. ტრადიციული კლიენტის/სერვერის OLAP აპლიკაციებისგან განსხვავებით, WOLAP განიხილება სამსაფეხურიანი არქიტექტურით, რომლის კომპონენტებია: კლიენტი, შუალედური პროგრამა და მონაცემთა ბაზის სერვერი;
- Desktop OLAP (DOLAP) სერვერი – მომხმარებელს საშუალებას აძლევს ჩამოტვირთოს მონაცემთა არეალი მონაცემთა ბაზიდან ან წყაროდან და იმუშაოს ამ მონაცემთა ბაზასთან ლოკალურად (თავის სამუშაო მაგიდაზე);
- Mobile OLAP (MOLAP) სერვერი – მომხმარებლებს აძლევს წვდომის საშუალებას OLAP მონაცემებთან და პროგრამულ უზრუნველყოფასთან დისტანციურად სამუშაოდ, საკუთარი მობილური მოწყობილობების გამოყენებით .
- Spatial OLAP (SOLAP) სერვერი – მოიცავს გეოგრაფიული საინფორმაციო სისტემების (GIS) და OLAP-ის შესაძლებლობებს ერთ მომხმარებლის ინტერფეისში. იგი მხარს უჭერს როგორც სივრცითი, ასევე არასივრცითი მონაცემების ადმინისტრირებას.

1.9.4. მონაცემთა საცავის შექმნის და მოდელირების ინსტრუმენტი

მონაცემთა საცავების შექმნის და მოდელირების ინსტრუმენტები გამოიყენება ინფორმაციის სხვადასხვა წყაროებიდან მიღებული მონაცემთა შესაგროვებლად, მათი გარდაქნის და შემდეგ საცავში ასატვირთად (გლობალურ ან ლოკალურ დონეზე).

მონაცემთა საცავის სამრეწველო რეჟიმში მართვისათვის გამოიყენება, მაგალითად, IBM DB2 Warehouse Manager ინსტრუმენტი, რომელიც არის საცავის ძირითადი მექანიზმი [43].

დაპროექტების ეტაპზე Data Warehouse Center – სათადარიგო ინსტრუმენტული საშუალებაა მონაცემთა საცავის შექმნისათვის. მის შემადგენლობაში შედის Warehouse Schema Modeler და Process Modeler. ისინი გამოიყენება დაპროექტებისთვის, სხვადასხვა სქემათა გენერაციისა და მონაცემთა ატვირთვისათვის. აგრეთვე შესაძლებელია ჩატარებულ მოქმედებათა გრაფიკული აღწერა, რომელიც სრულდება საცავის აგებისას. Process Modeler ახორციელებს მონაცემთა უპირობო დამუშავებას, არსებულ მოვლენებზე შეტყობინებას და მართვის კალენდარულ დაგეგმვას.

IBM DB2 UDB Data Warehouse Editions – ახალი, მრავალფუნქციური პაკეტია მონაცემთა საცავების ორგანიზებისათვის, რომლის ბაზა არის IBM DB2 Universal Database Version 8.1 Enterprise Server Edition (ESE). ამავე დროს იგი დამატებითი საშუალებაა საცავის ადმინისტრირების, მონაცემთა ინტეგრაციისა, მართვისა და ანალიზისათვის.

DB2 Data Warehouse Enterprise Edition – პაკეტი ორიენტირებულია ნებისმიერი ზომის მონაცემთა საცავის შექმნაზე ნებისმიერ დარგში, რაც საშუალებას გვაძლევს ავაგოთ მძლავრი საცავი ფართო ანალიზური დანართებისთვის. პაკეტის მოცულობა შეიძლება იყოს 10 ტერაბაიტზე მეტი.

DB2 Data Warehouse Enterprise Edition – შეიძლება გამოყენებული იქნას შემდეგ პლატფორმებზე: IBM AIX, Microsoft Windows 2000/2003, Linux Sun Solaris.

IBM DB2 Cube Views – გადაწყვეტილების მიღების თანამედროვე საშუალებაა, რომელიც OLAP-ოპერაციებს ახორციელებს მონაცებთა ბაზების მართვის IBM DB2 სისტემებში. მისი დახმარებით მომხმარებელს შეუძლია შექმნას მონაცემთაA სწრაფად გარდამქმნელი და იოლად მართვადი OLAP-გადაწყვეტილება, რომელიც ხელს უწყობს ამაღლდეს ანალიზური დანართების მთელი სპექტრის წარმადობა.

IBM DB2 Cube Views – გამჭვირვალედ არის მრავალგანზომილებიანი მონაცემთა ბაზების სტრუქტურას შემდეგი ფაქტორების დახმარებით:

- ქმნის მეტამონაცემებს სხვადასხვა განზომილების, იერარქიის, ატრიბუტების და ანალიზური ფუნქციისათვის;
- ხელს უწყობს და ამაღლებს OLAP – ინსტრუმენტის ეფექტუროობას;
- DB2-კატალოგში შეტანილი OLAP მეტამონაცემები ამაღლებს ამ ინსტრუმენტების მაღალეფექტურობას;
- DB2-მაღალეფექტური ტექნოლოგიის გამოყენება იმდენად მარტივია, რომ იგი შეიძლება შევადაროთ ჯამურ ცხრილებს და ანალიზურ ფუნქციებს.

IBM DB2 Query Patroller – არის მოთხოვნათა მართვის მაღალმწარმოებლური სისტემა, რომელიც ახორციელებს დინამიკურ კონტროლს მოთხოვნათა ნაკადზე და DB2–მონაცემთა ბაზაზე შემდეგი საშუალებებით:

- მოთხოვნათა შორის რაციონალურად უნდა განაწილდეს სისტემის რესურსები, მიუხედავად იმისა თუ როგორი სირთულისაა ეს მოთხოვნები;

- მოთხოვნათა ავტომატური გადაყვანა ლოდინის რეჟიმში;

- კონტრილიდან გამოსულ მოთხოვნათა სისტემიდან გამოყვანა.

Query Patroller უზრუნველყოფს მონაცემთა საცავზე დატვირთვის რეგულირებას, მცირე და დიდ მოთხოვნათა სწრაფად დასაკმაყოფილებლად სისტემის რესურსების ეფექტურ გამოყენებას. აგრეთვე იგი ხელს უწყობს მონაცემთა მოგროვებას და ინფორმაციის ანალიზს მოთხოვნაზე პასუხის გასაცემად.

IBM DB2 Warehouse Manager – უზრუნველყოფს მონაცემთა საცავების ავტომატიზებულ ორგანიზებას [43]. მისი მიზანია ბიზნესთან დაკავშირებული ინფორმაციის ღრმად ჩაწვდომა და მართვა, რათა სწორად მოახდინოს ბიზნესის წარმოება. ამ ინფრასტრუქტურაში შეიძლება გაერთიანდეს სხვა ინსტრუმენტები და Business Intelligence (ინტელექტუალური ბიზნესი) დანართები, რომლებიც გვაძლევს იმის გარანტიას, რომ მივიღებთ იმ დროისთვის არსებულ ყველაზე აქტუალურ ინფორმაციას, რაც დაგვეხმარება ბიზნესის სხვადასხვა სფეროში გააზრებული გადაწყვეტილების მიღებაში. IBM DB2 – არის მონაცემთა ბაზების მართვის საუკეთესო სისტემა, რომელიც ხელს უწყობს საინფორმაციო საცავის შექმნას.

DB2 – საცავის მენეჯერი გვთავაზობს გაფართოებულ ფუნქციონალურ გაერთიანებას, სადაც საბაზისო DB2-ს შესაძლებლობას ემატება (ETL – Extraction, Transformation and Loading) მონაცემთა გარდაქმნის და ატვირთვის მექანიზმები;

- მეტამონაცემთა და საინფორმაციო კატალოგის მართვა. საინფორმაციო კატალოგი იძლევა მეტამონაცემთა დასმული მოთხოვნის გაერთიანების საშუალებას;

- განაწილებულ სისტემებში მონაცემთა ჩატვირთვას და გარდაქმნას - ETL;

მასში გაერთიანებულია QMF for Windows – დანართი, რომელიც ხელს უწყობს DB2 პაკეტისთვის, Windows ან Web ინტერფეისის დახმარებით მოთხოვნების დაყენებას.

DB2 Warehouse Manager V8-ში აღსანიშნავია აგრეთვე ის, რომ:

- გამოყენებისას იგი საკმაოდ მარტივია, გამოირჩევა მაღალი მწარმოებლურობით;

- ერთდროულად ახორციელებს რამდენიმე ოპერაციას;

- ახდენს საწყის მონაცემთა თავმოყრას და არის გაერთიანებული საცავის ტიპი;

- ახედნს მონაცემთა სწრაფად შევსებას და საინფორმაციო კატალოგისთვის ახალი ინტერფეისის სრულყოფას;

- უზრუნველყოფს IBM AIX და Linux – ოპერაციული სისტემის-თვის მონაცემთა საცავის სერვერის დაკავშირებას.

II თავი
მონაცემთა საცავის დაპროექტება

2.1. მონაცემთა საცავის დაპროექტების ორგანიზაციულ-მეთოდური საფუძვლები

მონაცემთა მეცნიერების (Data science) სფეროში საცავის ცნება აქტუალური გახდა დაახლოებით 2000 წლიდან და იგი დღემდე ითვლება პერსპექტიულ მიმართულებად მართვის საინფორმაციო სიტემების შესაქმნელად. განსაკუთრებით გამოიკვეთა მისი მნიშვნელობა დიდ მონაცემთა (Big data) სისტემების მიმართულების სწრაფი განვითარების ფონზე.

საწყის ეტაპზე მართვის საინფორმაციო სისტემების (Management information systems – MIS) დაპროექტება ვითარდებოდა „ინფოლოგიკური“ გზით [51]. ახალი მიმართულების განვითარების ტენდენციამ ფართო გამოვლინება ჰპოვა „მონაცემთა საცავის“ სახით [9,21]. ინფორმატიკაში, როგორც მრავალი ტერმინი, ასევე ეს ცნებაც მოკლებულია ზუსტ განმარტებას, რაც პრაქტიკაში წარმოქმნის მცირე შეუთავსებლობას [20].

არსებობს განმარტების მრავალგვარი ახსნა რომელთაგან ერთ-ერთი ასეთია: „მონაცემთა საცავი არის ბიზნეს-მონაცემების დიდი კოლექცია, რომელიც გამოიყენება ორგანიზაციის დასახმარებლად გადაწყვეტილების მიღების დროს“ [52]. ტექნიკურ დონეზე, მონაცემთა საცავი სისტემატურად (პერიოდულად) აგროვებს ახალ. მიმდინარე ინფორმაციას. შემდეგ, ეს მონაცემები გადის ფორმატირებისა და იმპორტის პროცესებს, რათა იყოს თავსებადი საცავში უკვე არსებული მონაცემების. ამგვარად, იგი ინახავს დამუშავებულ მონაცემებს, რათა გადაწყვეტილების მიმღებთათვის წვდომა იყოს მზად. რამდენად ხშირად ხდება მონაცემთა ამოღება, ან როგორ ხდება მათი ფორმატირება და ა.შ., განსხვავდება ორგანიზაციის საჭიროებიდან გამომდინარე.

ასეთი სახის მონაცემთა საცავის გამოყენება იმ შემთხვევაშია შესაძლებელი, თუ უზრუნველყოფილ იქნება შემდეგ ფუნქციათა შესრულება:

2.1.1. F1( ) – მონაცემთა შერჩევის ფუნქცია

საწყის ეტაპზე ერთ-ერთი მნიშვნელოვანი ფუნქციაა „მონაცემთა მოხვედრა საცავში~. იგი მთელი რიგი პროცესების ერთობლიობაა, რომელიც ემყარება გარკვეულ კანონზომიერებებს.

პირველ რიგში, მთავარი პირობაა, რომ მონაცემები საცავში მოხვდეს საჭირო სახით, ხოლო მეორე რიგში უნდა ხდებოდეს ამ მონაცემთა რეგულირება საჭიროებისამებრ. „საჭირო სახეში~ იგულისხმება, მინიმუმ, მონაცემთა ფორმატირება. ერთი და იმავე სახის ინფორმაცია (მონაცემებით) შეიძლება მივიღოთ სხვადასხვა წყაროებიდან განსხვავებული ფორმატით, რაც წარმოქმნის შეუთავსებლობას, ამიტომ უნდა მოხდეს დასახელებათა და ტიპების უნიფიცირება.

ინფორმაციის პირველადი დამუშავებისთვის მნიშვნელოვანია მისი კონტექსტური აანალიზი, რათა საცავში წარმოდგენილ ერთიდაიმავე მონაცემებ არ ჰქონდეს სხვადასხვა მნიშვნელობები (სინტაქსურ-სემანტიკური პრობლემა). ამან შეიძლება მიგვიყანოს არასასურველ შედეგებამდე (მონაცემთა არა-თავსებადობა).

მონაცეთა პირველადი დამუშავებისას ხდება შეცდომების გამოაშკარავება სტატისტიკური დამუშავების სხვადასხვა საშუალებებით, შესაძლებელია უკვე გამოყენებულ მონაცემთა ჩამოცილება და აგრეთვე გამოტოვებულ მონაცემთა აღდგენა. ოპერატიული დამუშავების სისტემიდან წამოსული მონაცემები არ არის ერთჯერადი გამოყენების, ამიტომ მისი ნორმალური ფუნქციონირებისათვის უნდა მოქმედებდეს საცავში მონაცემთა გადაცემის პროცედურის შემსრულებელი პროგრამები და მათი პირველადი დამუშავება კავშირში უნდა იყოს გარე მოვლენებთან.

2.1.2. F2( ) – მონაცემთა ურთიერთშეთანხმების ფუნქცია

როგორც ყველა მონაცემთა ბაზა, ასევე მონაცემთა საცავიც შეიძლება იყოს განაწილებული კომპიუტერული ქსელის კვანძებში, სადაც ინფორმაცია მიიღება სხვადასხვა წყაროდან.

იმისათვის, რომ ურთიერთშეთანხმებულად მოქმედებდეს ინფორმაციის მომწოდებელი სხვადასხვა წყარო, მონაცემთა საცავის მართვის სისტემა აუცილებელია სარგებლობდეს მონაცემთა სტრუქტურის ერთიანი (საერთო) აღწერით. ამისათვის იყენებენ ლექსიკონ-ცნობარს (თეზაურუსი), სადაც თავმოყრილია ცნობები მონაცემთა ფორმატზე, სტრუქტურაზე, ინფორმაციის მიმღებ არხებსა და წყაროებზე და ა.შ.

2.1.3. F3( ) – მონაცემთა მოპოვების ფუნქცია

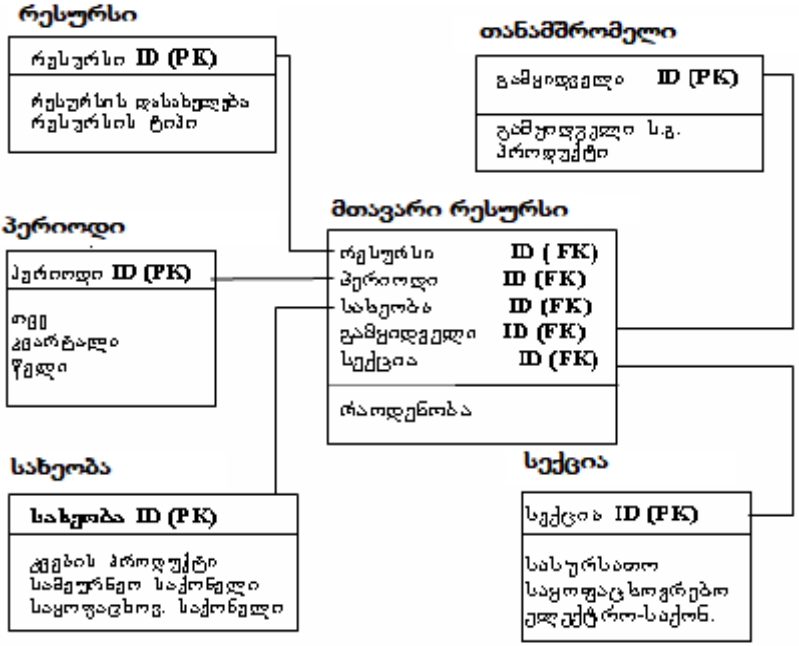
მონაცემთა საცავსა და ოპერატიულ მონაცემთა ბაზას შორის არსებული განსხვავების შესაბამისად, გამოიყენება მათთან მიდგომის განსხვავებული საშუალებები.

მონაცემთა საცავთან დაკავშირება ხდება არარეგლამენტირებულ მოთხოვნათა სისტემით (ad hoc query). იგი, მოთხოვნების ფორმულირებისას, გამოირჩევა მოქნილობით, საშუალებას იძლევა მონაცემებზე ანალიზის ჩატარებისას, ვისარგებლოთ როგორც ზოგადი, ასევე მათი დეტალური ანალიზით.

ანალიზის მეორე სახე, რომელიც გამოიყენება ასეთ სისტემებში, გაითვლის შესაძლო შემთხვევებს (what-if-analysis), როგორიცაა, მაგალითად, რეალურ მონაცემებში რამდენიმე მაჩვენებლის შეცვლა და მათი ანალიზი. შედეგად იცვლება საერთო მდგომარეობა. ზოგადად, არარეგლამენტირებულ მოთხოვნათა სისტემის მომსახურებისთვის შეიძლება გამოყენებულ იქნას OLAP [17, 40]. მონაცემთა საცავი ეყრდნობა რელაციურ და მრავალგანზომილებიან ბაზებს. პირველადი სახის მონაცემები ინახება

რელაციურ მონაცემთა ბაზაში და მოთავსებულია „რადიალურ“სქემაში (star schema).

ვ. ინმონი მონაცემთა საცავის კონკრეტულ სტრუქტურას თითქმის არ განიხილავს, ხოლო რელაციური ბაზების ფუძემდებელმა, ე. კოდმა წარმოადგინა ოპტიმიზირებული მონაცემთა ბაზების სქემა [15,17,51]. მას აქვს ერთი მთავარი ცხრილი, რომელთანაც დაკავშირებულია რამდენიმე დამოკიდებული ცხრილი, მთავარი ცხრილის საშუალებით ხდება დამოკიდებულ ცხრილთა მონაცემებით მომარაგება. მაგალითად, კომერციული ობიექტის სტრუქტურის რადიალური სქემა (ნახ.2.1).



ნახ.2.1. რადიალირი სქემა

მთავარი ცხრილი ანუ ფაქტების ცხრილი ასახავს კვლევის ობიექტს, იგი შესაძლოა შედგებოდეს მილიონობით სტრიქონისგან.

მონაცემთა ორგანიზების განსხვავებული მიდგომაა მონაცემთა საცავში, კერძოდ, მრავალგანზომილებიანი ანალიზის გამოყენება [52,53]. აქ მონაცემები ინახება არა ცხრილში არამედ მრავალგანზომილებიან კუბში. ჰიპერკუბის მოდელი მომხმარებლისთვის არის თვალსაჩინო და OLAP ანალიზისთვის ხელსაყრელი.

მონაცემთა მრავალგანზომილებიან ასახვას ახლავს გარკვეული სირთულეებიც, ინფორმაციის დამუშავების მეთოდოლოგიის შერჩევის თვალსაზრისით.

2.1.4. F4( ) – კლასიფიკაციის ფუნქცია

ეს არის მონაცემთა კვლევის ერთ-ერთი ცნობილი მეთოდი, სადაც განიხილება მონაცემთა ტიპები (კლასები). კლასი არის ერთი ტიპის ობიექტების სიმრავლე, რომელთაც აქვს მსგავსი სტრუქტურა და ქცევა [54]. მათი საშუალებით შეგვიძლია განვსაზღვროთ მოვლენები, სიტუაციები და პროცესები.

ობიექტები (ობიექტი განიხილება, როგორც გარკვეული არსი რომელიც ხასიათდება მდგომარეობით და ქცევით) უნდა აღიწეროს შემდეგი სიდიდეებით, როგორიცაა: სიმპტომები, მაჩვენებლები, პარამეტრები. ინფორმაცია ყოველი კლასის შესახებ მოცემულია ობიექტების დახმარებით (დაკვირვება, პრეცენდენტი) აქედან გამომდინარე შეგვიძლია განვსაზღვროთ, ობიექტთა გარკვეული ჯგუფი თუ რომელ კლასს მიეკუთვნება.

უნდა განისაზღვროს კრიტერიუმები, რათა დადგინდეს თუ რომელ კლასიფიკაციის კატეგორიას მიეკუთვნება ობიექტი. კრიტერიუმის შერჩევა ეყრდნობა უკვე კლასიფიცირებული ობიექტის ხასიათის შესწავლას. მაგალითად, ამოცნობის ალგორითმი გამოიყენება მედიცინაშიც, როდესაც გარკვეული სიმპტომებით და ამბულატორიული გამოკვლევის მონაცემებით ხდება დაავადების დიაგნოსტიკა, აგრეთვე ტექნიკაში, როდესაც მაკონტროლებელი მოწყობილობის მაჩვენებლებით და ექსპერტული მონაცემების

შეჯერებით ხდება ოპტიმალური კრიტერიუმების დადგენა.

ბიზნესის ბევრ სფეროში შეინიშნება მუდმივი დამკვეთის დაკარგვის პრობლემა. კლასიფიკაციის ინსტრუმენტის გამოყენებით შესაძლოა გამოვყოთ დამკვეთის მოთხოვნათა ხასიათის შესაბამისი მონაცემთა ჯგუფი. სავარაუდოა, რომ წინასწარ შექმნილ მოდელს გამოუჩნდება სასურველი დამკვეთი, ასეთი სახის მოდელი შესაძლოა იყოს დამკვეთზე ზემოქმედების ისეთი საშუალება, როგორიცაა რეკლამა, იგი საშუალებას იძლევა სხვადასხვა კატეგორიის დამკვეთის მოსაზიდად.

2.1.5. F5( ) –კლასტერიზაციის ფუნქცია

კლასტერიზაცია მოგვაგონებს კლასიფიკაციის ფორმას, იმ განსხვავებით, რომ კლასიფიკაციის კრიტერიუმები არაა მოცემული. კლასტერიზაცია მონაცემთა კვლევისას საშუალებას გვაძლევს აღმოვაჩინოთ მონაცემები, რომლებიც დაჯგუფებულია რაიმე ნიშანთვისებების მიხედვით ისე, რომ ერთ ჯგუფში გაერთიანებული მონაცემები „მსგავსია“, ხოლო სხვადასხვა ჯგუფში „არაა მსგავსი“. კლასტერიზაციის ალგორითმი, როგორც პირველადი ანალიზის ინსტრუმენტი, შეუცვლელია მრავალგანზომილებიან მონაცემთა ჯგუფის დამუშავებისას, ამასთან იგი ძალიან ეკონომიურია [55, 56].

2.1.6. F6() – ოპერატიული ანალიზის ფუნქცია OLAP ინსტრუმენტით

მონაცემთა საცავების დაპროექტებისა და მისი ფუნქციონირებისათვის, მეთოდური თვალსაზრისით, გამოიყენება ედგარ კოდის მიერ ჩამოყალიბებული პრინციპები [20,57]. ესაა ის 12 წესი, რომელსაც უნდა აკმაყოფილებდეს ნებისმიერი განაწილებული სისტემა მონაცემთა საცავით, რათა ჩატარდეს საინფორმაციო ბლოკების სრულფასოვანი ოპერატიული ანალიზური სამუშაოები.

1) **მონაცემთა მრავალგანზომილებიანი კონცეპტუალური ხედვა.** მომხმარებელ-ანალიტიკოსი საპრობლემო სფეროს, თავისი ბუნების მიხედვით, ხედავს როგორც მრავალგანზომილებიანს. შესაბამისად OLAP- მოდელიც უნდა იყოს მრავალგანზომილებიანი. ასეთი ტიპის კონცეპტუალური სქემა (მომხმარებელთა წარმოდგენები) აიოლებს მოდელირებას, ანალიზს და გამოთვლებს;

2) **გამჭვირვალეობა.** OLAP წარმოდგენილი უნდა იყოს ღია არქიტექტურის კონტექსტში, სადაც მომხმარებელს საშუალება ექნება ნებისმიერ დროს ანალიზური ინსტრუმენტის საშუალებით დაუკავშირდეს სერვერს საჭირო ინფორმაციის მისაღებად;

3) **წვდომადობა.** OLAP – ის მომხმარებელ ანალიზატორს უნდა ჰქონდეს ანალიზის ჩატარების საშუალება, რომელიც ემყარება საერთო კონცეპტუალურ სქემას. აქ განთავსებულია რელაციური მონაცემთა ბაზა საწარმოთა შესახებ არსებული ყველა ახალი და ძველი მონაცემებით. ეს ნიშნავს, რომ OLAP-მა უნდა წარმოადგინოს თვისი საკუთარი ლოგიკური სქემა, რათა შეასრულოს შესაბამისი გარდაქმნა და მომხმარებელს წარუდგინოს მონაცემები. გარდა ამისა აუცილებელია წინასწარ იმაზე ზრუნვა, თუ სად, როგორ და როგორი სახის ფიზიკური ორგანიზაციის მონაცემები იქნას გამოყენებული. OLAP სისტემამ უნდა შეასრულოს ისეთი მონაცემების დამუშავება, რომელთა მოთხოვნაც რეალურად არსებობს;

4) **სტაბილური ანგარიშგების წარმადობა.** თუ ანალიტიკოსის მიერ ჩატარებული გაზომვათა რაოდენობა ან მონაცემთა ბაზების რიცხვი მნიშვნელოვნად იზრდება, მომხმარებელ ეს პროცესი უნდა დარჩეს შეუმჩნეველი და არ უნდა აისახებოდეს საწარმოო პროცესების წარმადობის შემცირებაზე;

5) **კლიენტ-სერვერული არქიტექტურა.** OLAP ინსტრუმენტების სერვერის კომპონენტი უნდა იყოს საკმარისად ინტელექტუალური, რომ კლიენტებმა შეძლონ მიერთება მინიმალური რესურსებით.

სერვერს უნდა შეეძლოს განსხვავებული ტიპების მონაცემთა ასახვა და კონსოლიდაცია სხვადასხვა მონაცემთა ბაზებს შორის;

6) **ზოგადი განზომილება.** მონაცემთა ყველა განზომილება უნდა იყოს ეკვივალენტური თავისი სტრუქტურით და ოპერატიული შესაძლებლობებით;

7) **დინამიკური მართვა** გამონთავისუფლებული რეჟიმით. OLAP– ინსტრუმენტის ფიზიკური სქემა უნდა ადაპტირდებოდეს სპეციფიკურ ანალიტიკურ მოდელთან, რათა ოპტიმალურად მართოს გამონთავისუფლებული მატრიცა. ერთი ცარიელი მატრიცისთვის არსებობს ერთადერთი ოპტიმალური ფიზიკური სქემა. OLAP – ინსტრუმენტის ბაზური ფიზიკური მონაცემები პრაქტიკული ოპერაციებისთვის, რომელთაც აქვს დიდი ანალიზური მოდელი, უნდა კონფიგურირდებოდეს ნებისმიერ ქვესიმრავლესთან. თუ OLAP–ინსტრუმენტს არ შეუძლია გააკონტროლოს და დაარეგულიროს გასაანალიზებელი მონაცემების მნიშვნელობები, ის ჩაითვლება უსარგებლოდ და არასაიმედოდ;

8) **მრავალი მომხმარებლის მხარდაჭერა.** ხშირ შემთხვევაში მომხმარებელ-ანალიტიკოსი დასმულ მოთხოვნებს აყენებს ერთ ანალიზურ მოდელთან ან ქმნის განსხვავებულ მოდელს ერთი სახის მონაცემებიდან. OLAP ინსტრუმენტი უნდა უზრუნველყოფდეს ერთდროულად მონაცემთა მოძიებასა და განახლებას, წვდომას, მთლიანობას და უსაფრთხოებას;

9) **შეუზღუდავი გადამკვეთი ოპერაციები.** მონაცემთა შემოწმების სხვადასხვა დონე და გაერთიანების გზა, მათი იერარქიული ბუნების გათვალისწინებით მჭიდრო კავშირშია OLAP - მოდელთან ან დანართთან. თვითონ ინსტრუმენტი უნდა მოიაზრებოდეს შესაბამის გამოთვლებთან და არ უნდა მოსთხოვოს მომხმარებელს თავიდან განსაზღვროს გამოთვლები და ოპერაციები. გამოთვლები მოითხოვს რომელიმე გამოყენებულ ენაში განსხვავებული ფორმულების განსაზღვრას. ასეთი ენა შეიძლება გამოვიყენოთ ნებისმიერი სიდიდის მონაცემთა მანიპულირებისთვის და არ

შეზღუდოს მონაცემები არსებული კუბის უჯრედებს შორის და კონკრეტული უჯრედების საერთო ატრიბუტებზე;

10) **მონაცემთა ინტუიციური მანიპულაცია**. მონაცემთა დეტალიზაციის, გაერთიანების და სხვა მანიპულაციები უნდა იყენებდეს ცალკეულ უჯრედებზე ანალიზური მოდელის შედეგებს და არ უნდა იყენებდეს მომხმარებლის ინტერფეისებს. მომხმარებელ ანალიტიკოსს უნდა ჰქონდეს ყველა აუცილებელი პირობა იმისა, რომ მიიღოს სრულყოფილი ინფორმაცია;

11) **ანგარიშების მიღების მოქნილი საშუალება**. შეტყობინებათა დამუშავება და პასუხის გაცემა უნდა იყოს მოქნილი და ელასტიური. მომხმარებელს უნდა შეეძლოს მონაცემთა კომბინირება და გაანალიზება. მოქნილობა მნიშვნელოვანია, რათა ყურადღება გამახვილდეს მონაცემთა განმასხვავებელ ნიშნებზე. ანუ სისტემის ანგარიშგების ინსტრუმენტულმა საშუალებებმა უნდა წარმოადგინოს ინფორმაცია ისე, როგორც მომხმარებელს სურს მისი ნახვა;

12) **შეუზღუდავი ზომები და აგრეგაციის დონეები**. მონაცემთა მხარდაჭერილ განზომილებათა რაოდენობა ყველა მიზნისთვის უნდა იყოს შეუზღუდავი. გამოკვლევებმა აჩვენეს, რომ აუცილებელი გაზომვა ერთდროულად შეიძლება ჩატარდეს 19-ჯერ. აქედან გამომდინარე შეიძლება ვთქვათ, რომ ანალიტიკური ინსტრუმენტი საშუალებას გვაძლევს ერთდროულად ვაწარმოოთ 15-დან 20-მდე გაზომვა, ამასთან თითოეული გაზომვის მცდელობა არ არის შემოსაზღვრული დადგენილი რიცხვით.

ეს პირობები შეიძლება ჩავთვალოთ OLAP-ის (ოპერატიული ანალიზური დამუშავების) თეორიულ ბაზისად. როგორც უკვე აღვნიშნეთ, OLAP-ში ჩადებულია მონაცემთა დამუშავების მრავალგანზომილებიანი სტრუქტურის იდეა. როდესაც ვსაუბრობთ მასზე, უნდა ვიგულისხმოთ, რომ ეს არის მონაცემთა ლოგიკური სტრუქტურის მრავალგანზომილებიანი ანალიზური ინსტრუმენტი.

2.2. მონაცემთა ტიპიზაცია, კლასიფიკაცია და კატალოგიზაცია

დასაპროექტებელი ობიექტის განაწილებული სისტემის მონაცემთა საცავში განსათავსებელი ინფორმაციული ბლოკები მათი წინასწარი დამუშავების საფუძველზე შეივსება ოპერატიული მონაცემთა ბაზებიდან (მეთოდური საფუძვლები წინა პარაგრაფში იყო განხილული). როგორც ვიცით, ოპერატიულ ბაზაში საწყისი მონაცემები თავს იყრის გარე ორგანიზაციებიდან ინტერნეტის საშუალებით, ან სხვა სახის კომუნიკაციებიდან.

საინფორმაციო ბლოკები, ან უფრო ზოგადად, მონაცემთა მანქანური ფაილები განსხვავებული ტიპებისაა: ტექსტები და სუპერტექსტები (H), ცხრილები (T), გრაფიკები (G), აუდიო (A) და ვიდეო (V) მასალა და ა.შ.

მათემატიკური მოდელი შეიძლება ჩავწეროთ ოთხეულით:

$$B = < R, M, S, F >, \qquad (2.1)$$

სადაც

R – საცავის კლასიფიცირებული ობიექტებია.

$R = \{R_i\}$, აქ i კლასიფიკაციის ტიპია და იგი შეესაბამება ზემოაღნიშნულ ცვლადებს: {H, T, G, A, V};

M – მონაცემთა ბლოკების მეტაინფორმაციაა, განთავსებული სამდონიან იერარქიულ კატალოგსა (C_R, C_A, C_V) და ინდექსურ ცხრილებში (T_I);

$$M = \{C_R, C_A, C_V, T_i\}, \qquad (2.2)$$

სადაც

- C_R - საინფორმაციო ბლოკების აღწერის რელაციათა სიმრავლეა: $C_R=\{C_{Rj}\}$, სადაც j=1,m რელაციათა ინდექსებია (რელაციათა სიმრავლის სიმძლავრე);

- C_A - საინფორმაციო ბლოკების აღწერის ატრიბუტების სიმრავლეა: $C_A=\{C_{Ak}\}$, სადაც k=1,n ატრიბუტების ინდექსებია (ატრიბუტთა სიმრავლის სიმძლავრე);

- C_V - ატრიბუტების მნიშვნელობათა სიმრავლეა:

$C_V=\{C_{Vz}^{Ak}\}$, სადაც k=A_k - ატრიბუტის ინდექსები, ხოლო z ამ ატრიბუტის შესაბამის მნიშვნელობათა ინდექსებია (ატრიბუტის მნიშვნელობათა ქვესიმრავლის სიმძლავრე).

T_I – ინდექსების ცხრილის კორტეჟების (ან სტრიქონების, ჩანაწერების) სიმრავლეა: $T_I=\{T_{AkVz}^{Rj}\}$, სადაც j=1,m რელაციათა ინდექსები, k=1,n – Ak-ატრიბუტის ინდექსები, ხოლო V_z ამ ატრიბუტის შესაბამისი მნიშვნელობათა ინდექსებია. ინდექსების ცხრილის სვეტების რაოდენობაა n, ხოლო სტრიქონების რაოდენობა - m.

S – საცავის მეხსიერების ცენტრალური ცხრილია, რომელთა კორტეჟებშიც მოთავსებულია საინფორმაციო ბლოკების ფიზიკური მისამართები და ინდექსური ცხრილების ელემენტთა მნიშვნელობები. ამგვარად, მყარდება ლოგიკური კავშირი მოთხოვნაში დასახელებულ j-ურ რელაციის k-ურ ატრიბუტის z-ურ მნიშვნელობასა და მეხსიერების ფიზიკურ a–ურ მისამართს შორის;

F – მონაცემთა საცავის ელემენტების (R), ანუ კლასიფიცირებული ობიექტების დამუშავების ოპერაციებია (კლასთა მეთოდები). ჩვენ წინა პარაგრაფში განვიხილეთ ზოგიერთი ძირითადი ოპერაცია, მაგალითად:

F1() – მონაცემთა შერჩევა;

F2() –მონაცემთა ურთიერთშეთანხმება;

F3() – მონაცემთა მოპოვება;

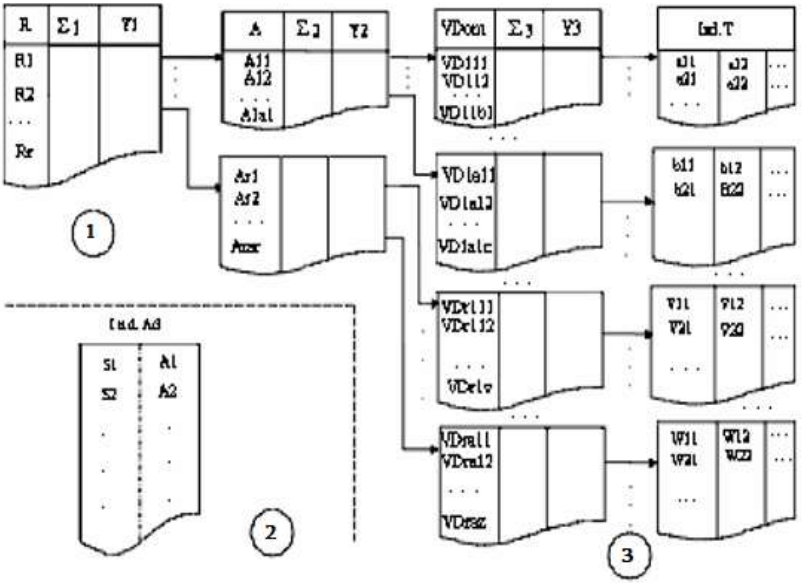
F4() – კლასიფიკაცია;

F5() –კლასტერიზაცია;

F6() – ოპერატიული analizi OLAP ინსტრუმენტით და სხვ. აქ ემატება მონაცემთა ტრანსფორმაციის, კონვერტირების და რესტუქტურიზაციის ფუნქციებიც.

მომხმარებელთა მოთხოვნების დამუშავების ამოცანა განიხილება 2.4 პარაგრაფში.

2.3 ნახაზზე მოცემულია მონაცემთა საცავის მეტაინფორმაციის ორგანიზებისა და მართვის სამდონიანი კატალოგის და ინდექსური ცხრილების გრაფიკული წარმოდგენა,

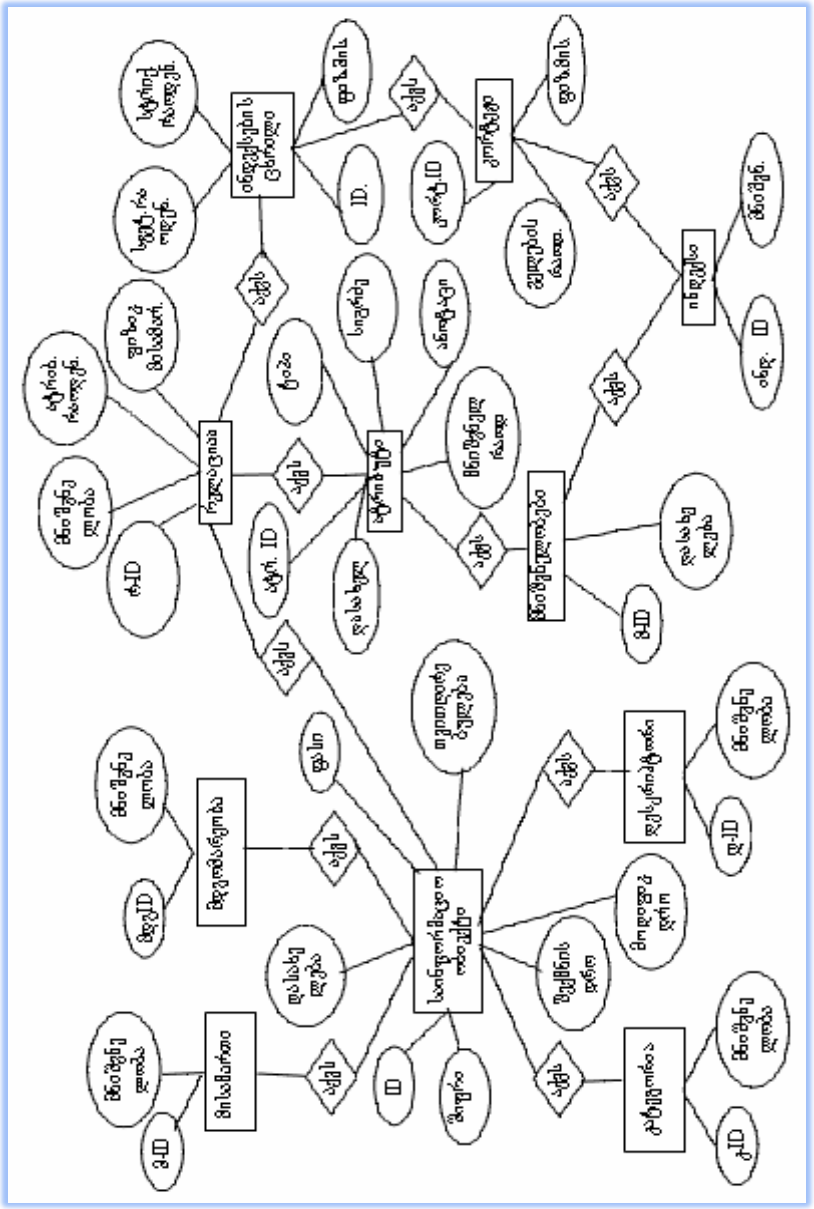


ნახ. 2.3. მეტაინფორმაცია - მართვის სამდონიანი კატალოგი

2.4-ზე კი ილუსტრირებულია საინფორმაციო ობიექტების დეტალური აღწერის კონცეპტუალური მოდელი არსთა-დამოკიდებულების (ER) დიაგრამის საშუალებით.

პროგრამული რეალიზაციის საკითხი დაკავშირებულია მონაცემთა განაწილებული ბაზების მართვის სისტემის, მაგალითად, Ms SQL Server-ის გამოყენებასთან.

პირველ ეტაპზე საჭიროა დაპროექტდეს მონაცემთა საცავის საინფორმაციო ობიექტების მეტაინფორმაციის სტრუქტურა, რომელიც საერთო იქნება ყველა ტიპის (H, T, G, A, V) კლასისათვის. ქვემოთ მოცემულია ამ სტრუქტურის ცხრილი შესაბამისი ატრიბუტებითა და განმარტებებით:



ნახ. 2.4

ინფორმაციულ-ობიექტთა მეტაინფორმაციის სტრუქტურა. ცხრ.2.1

№	ატრიბუტი	დასახელება	ტიპი	შენიშვნა
1.	ObjID	ობიექტის იდენტიფიკატორი	int	უნიკალური
2.	Shifri	ობიექტის შიფრი	varchar	
3.	Name	ობიექტის დასახელება	varchar	
4.	Category	ობიექტის კატეგორია	smallint	მეორადი გასაღები
5.	StateCode	მდგომარეობის კოდი	smallint	მეორადი გასაღები
6.	Summary	ობიექტის ანოტაცია	varchar	MEMO-ველი
7.	DescrID	ძირითად დესკრიპტორთა იდ.	smallint	მეორადი გასაღები
8.	FileSize	ფაილის ზომა ბაიტებში	int	
9.	CreatDate	ობიექტის შექმნის თარიღი	Date	
10.	ModifDate	ბოლო მოდიფიკაციის თარიღი	Date	
11.	PriceA	თვითღირებულება	Float	შენახვის ხარჯი
12.	PriceB	ფასი	Float	გასაყიდი ფასი
13.	PhizAdrID	ფიზიკური მისამართის იდენტ.	char	მეორადი გასაღები

2.3. მოთხოვნების წინმსწრები ანალიზის და ტრანზაქციების სინქრონიზაციის სერიალიზაციის ამოცანის გადჭრა

შეტყობინებათა წინმსწრები ანალიზით შესაძლებელია დადგინდეს თუ რომელი საინფორმაციო ობიექტი აინტერესებს ამა თუ იმ მომხმარებელს (ან ჯგუფს) [59].

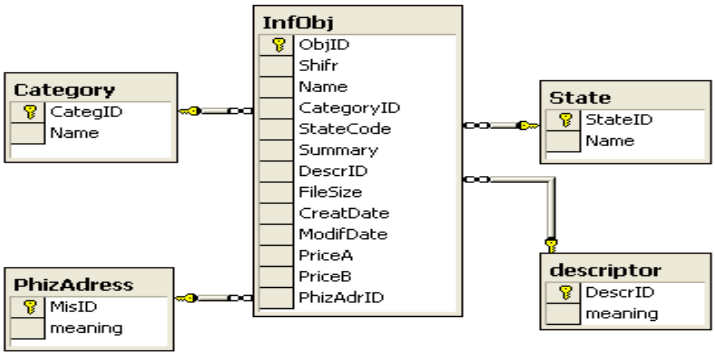
წინმსწრები ანალიზის ალგორითმის მიზანია დროის ერთეულში შემოსული მოთხოვნების პაკეტის (რამდენიმე შეტყობინება) დესკრიპტორული ანალიზის ჩატარება და მათი დაკმაყოფილების მიზნით აუცილებელი და საკმარისი ინფორმაციული რესურსების მოძიების და პასუხების მომზადების მიმდევრობით-პარალელური ოპერაციების დაგეგმვის განხორციელება.

„პაკეტის“ წინმსწრები განხილვის აზრი მდგომარეობს იმაში, რომ გამოვლინდეს სხვადასხვა შეტყობინებაში ერთიდაიმავე საინფორმაციო ობიექტებზე მოთხოვნა, შემდგომ ეტაპზე საცავში განმეორებითი ძებნის ოპერაციების გამოსარიცხად. ეს ამაღლებს სისტემის მწარმოებლურობას და ამცირებს მოთხოვნების დაკმაყოფილების საერთო დროს.

წინა პარაგრაფიდან (2.1) მოდელში შევიტანოთ მომხმარებლის მოთხოვნის, ანუ შეტყობინების ელემენტი Q (Query). მოდელი მიიღებს შემდეგ სახეს:

$$B = < R, M, S, F, Q > \qquad (2.3)$$

ჩვენთვის აუცილებელია ლოგიკური მექანიზმის შემუშავება, რომელიც ყოველ კონკრეტულ შეტყობინებას ცალსახად განუსაზღვრავს მისთვის საჭირო კლასიფიცირებულ ობიექტთა სიმრავლეს მონაცემთა საცავიდან. აღნიშნული ამოცანის გადასაწყვეტად წინასწარ საჭიროა დამუშავდეს მონაცემთა საცავის საინფორმაციო ობიექტებისა და მათი მეტაინფორმაციის (კატალოგების) სისტემის კლასთაშორისი - კავშირების დიაგრამა. მისი აგება შესაძლებელია წინა პარაგრაფში დაპროექტებული კონცეპტუალური მოდელის საფუძველზე. 2.5 ნახაზზე ნაჩვენებია ობიექტ-ორიენტირებული დიაგრამის კომპიუტერული რეალიზაციის ფრაგმენტი.



ნახ. 2.5

ამოცანის გადაწყვეტის რეალიზაციის მიზნით ჩვენ ვიყენებთ მონაცემთა რელაციური ბაზების მართვის სისტემის MsSQL Server პროგრამულ პაკეტს. ინფორმაციულ მონაცემთა და მათი მეტა-ინფორმაციის ძირითადი ცხრილები აგებულია რელაციური პრინციპით. მონაცემთა ძირითად დესკრიპტორთა სიმრავლი-სათვის შექმნილია ინდექსური ფაილები (სწრაფი მოძებნის ცხრილები). მომხმარებელთა შეტყობინებების ფორმირება ხორცი-ელდება სტანდარტული SQL-ენის საფუძველზე.

მოთხოვნების დამუშავების პროცედურები (მაგალითად, სელექცია) ბაზებში ძირითადად შეიცავს ისეთ კრიტერიუმებს, რომელთა შესრულება ხორციელდება რელაციური ალგებრის ოპერაციებით ცხრილების კორტეჟებსა და ატრიბუტებზე.

(2.3) მოდელში Q შეტყობინებათა წინმსწრები ანალიზის ჩასატარებლად ვიყენებთ ე. კოდის რელაციური მოდელებისა და ალგებრის კონცეფციას:

$$Ra = < R, f > \qquad (2.4)$$

სადაც R – რელციური დამოკიდებულებებია მონაცემთა საცავის საინფორმაციო ობიექტების ცხრილების ატრიბუტთა სიმრავლეზე განსაზღვრული, ხოლო f – სიმრავლეთა თეორიისა და კოდის ალგებრის ოპერაციები [51, 58].

დასმული ამოცანის კვლევის საგნად ჩვენ გამოვიყენებთ პეტრის ქსელის ინსტრუმენტს, იზომორფული სისტემის თვალ-საზრისით [60].

პეტრის ქსელების გრაფულ წარმოდგენას აქვს ასეთი სახე:

$$P = < S, T, I, O > \qquad (2.5)$$

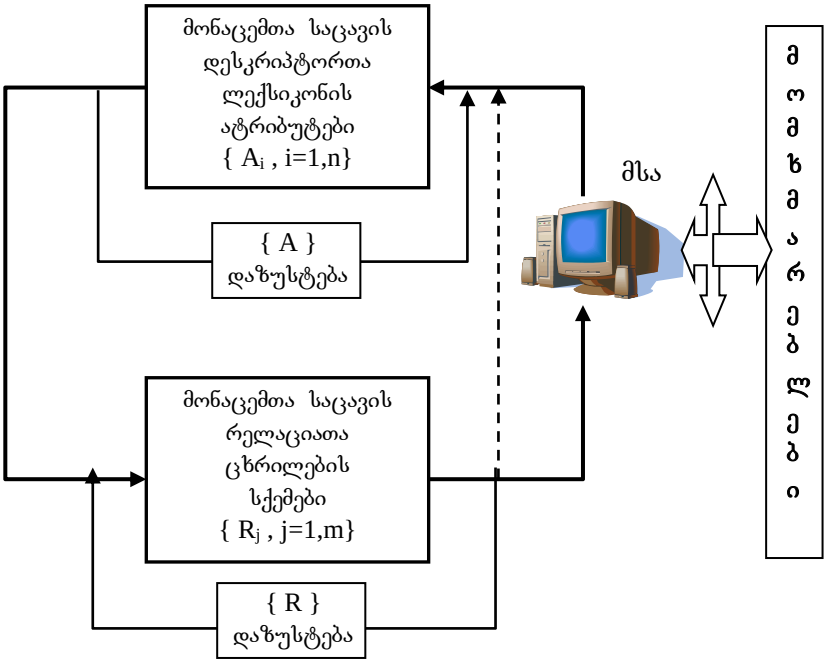
სადაც S – ქსელის პოზიციებია (ობიექტის ან შესაბამისი ცხრილის მდგომარეობები), T – ქსელის გადასასვლელები (რელაციური ოპე-რაციები გარკვეული დაყოვნებით), I და O შესაბამისად არის შემავალი და გამომავალი ფუნქციები. ქსელის პოზიციაში შეიძლება იყოს მარკერი, რომელიც ამოდელირებს მაგალითად, ქსელში

მოთხოვნის შემოსვლას, არსებობას ან შედეგის მიღებას.

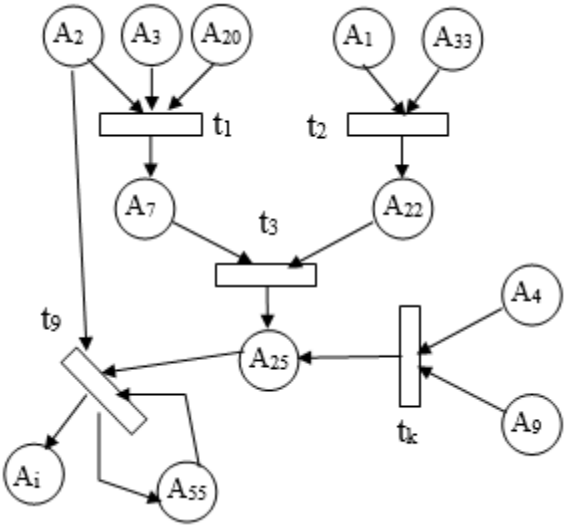
ვინაიდან მომხმარებლის მოთხოვნა პრედიკატულ ფორმაში ყალიბდება (SQL სტანდარტი), მასში ძირითადად ლოგიკური და რელაციური პროცედურები სჭარბობენ.

მოთხოვნაში შეიძლება იყოს მათემატიკური, სტატისტიკური, აგრეგაციის, დაჯგუფებისა და სხვა სახის ფუნქციები, რომლებიც პროგრამულად მეთოდების სახითაა რეალიზებული და შენახული პროგრამულ ბიბლიოთეკაში.

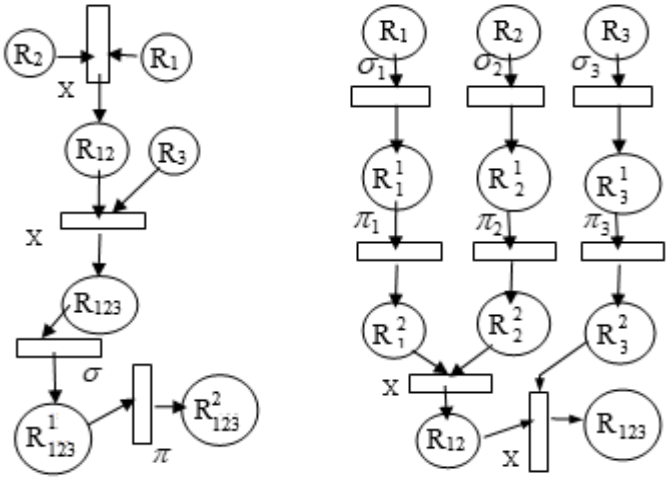
2.6 ნახაზზე ნაჩვენებია მომხმარებელთა მოთხოვნების წინმსწრები ანალიზის ორეტაპიანი მართვის პროცესის სქემა და მისი შესაბამისი პეტრის ქსელის გრაფის ფრაგმენტები ატრიბუტებისა (ნახ.2.7) და რელაციათა სქემებისათვის (ნახ.2.8).



ნახ. 2.6



ნახ. 2.7. ატრიბუტების სქემის პეტრის ქსელის გრაფი



ნახ. 2.8. რელაციათა სქემის პეტრის ქსელი გრაფი

აღნიშნული საკითხის გადასაწყვეტად მულტიმომხმარებლურ რეჟიმისათვის ჩვენ შემოთავაზებული გვაქვს ტრანზაქციათა სინქრონიზაციის სერიალიზაციის ალგორითმი, რომელიც რეპოზიტორ-მენეჯერ პროგრამის კლასის ერთ-ერთი მეთოდია.

მომხმარებელთა ტრანზაქციები (საინფორმაციო ობიექტების წაკითხვა/ჩაწერის პროცედურები) იყენებს ქსელის საერთო გამოყენების რესურსებს, ამიტომაც საჭიროა მათი ბლოკირება-დებლოკირების კონფლიქტური სიტუაციების მართვა, რათა თავიდან იქნას აცილებული „ჩიხური“, უსასრულო მოლოდინის მდგომარეობები.

ტრანზაქციათა სერიალიზაცია კი უზრუნველყოფს მათ პარალელურ შესრულებას. რომელიმე საინფორმაციო ობიექტში ცვლილებების შეტანისას ერთი ტრანზაქციით იგი ბლოკირებულია სხვა ტრანზაქციებისათვის ამ პროცესის დამთავრებამდე.

ამასთანავე, მონაცემთა საცავის მთლიანობის მოთხოვნის დაცვის მიზნით რეპოზიტორ-მენეჯერი მთლიან საცავში (კატალოგებსა და განაწილებულ ბაზებში) განახორციელებს შესაბამის ცვლილებებს.

2.4. მონაცემთა საცავის პროგრამული კომპონენტები

მონაცემთა საცავი მოიცავს ტრანსფორმაციისა და კონვერტაციის პროგრამებს, საბაზო მეტამონაცემთა სისტემას, არქივირებული შენახვის სისტემას, ინტეგრირებულ მონაცემთა ბაზებს და საცავის ინტელექტუალური ანალიზის ტექნოლოგიას [61,62].

- *ტრანსფორმაციის პროგრამა:* ახორციელებს ინტერფეისის ფუნქციას მონაცემთა საცავსა და მონაცემთა წყაროებს შორის. მონაცემები (ინფორმაცია) განსხვავებულ მონაცემთა ბაზებიდან (იერარქიულ, რელაციურ ან ობიექტ-ორიენტირებულ) ან თანამიმდევრულ განსხვავებული ფორმატის ფაილებიდან (ASCII, ANSI, EBCDIC და ა.შ.) ექსტრადირდება. ტრანსფორმაციის წესის

თანახმად ისინი ერთიანდება, როგორც ინტეგრირებული, სუბიექტორიენტირებული, მუდმივი და დროში ცვალებადი სტრუქტურები. ტრანსფორმაციის პროგრამამ უნდა უზრუნველყოს ტრანსპორტირებისათვის ფუნქციათა წარმოდგენა და ასევე მონაცემთა მომზადება საცავში გადასაგზავნად.

საცავში შიგა მონაცემების (ინფორმაციის) უდიდესი ნაწილი შესაძლოა მიღებულ იქნას განაწილებული ოპერატიული სისტემიდან. მონაცემთა საცავში ისტორიული და მიმდინარე ინფორმაციის შევსება ხდება ბაზებიდან, სადაც პერიოდულად მიმდინარეობს მონაცემთა აქტუალიზაცია. თუ ოპერატიულ მონაცემთა ჩანაწერების რეგისტრაციაში შეტანის თარიღი უფრო ახალია, ვიდრე ბოლო ტრანსფორმაციის შეტანის დრო, მაშინ მოხდება ამ უკანასკნელის ლიკვიდაცია. ე.ი. აუცილებლად გასათვალისწინებელია ცალკეული ტრანსფორმაციის პროცესებს შორის ვადები.

მონაცემთა გარე მიმწოდებლებია შეტყობინებათა სამსახურები, ბირჟები, პოლიტიკური საინფორმაციო ბლოკები, სამეცნიერო კვლევითი ინსტიტუტები და ბაზარი. ინფორმაციის მოწოდებისათვის გამოიყენება ისეთი საშუალებები, როგორიცაა მაგალითად: Internet, CD-ROM, Flash-Memory, FD და ა. შ. ბევრი ამ მონაცემთაგან ტრანსფორმაციის პროგრამის საშუალებით, სანამ გადავა მონაცემთა საცავში უნდა წარმოდგეს გარკვეული Internet – სტანდარტული ფორმატის სახით.

- *მონაცემთა ბაზების სიტემა:* მონაცემთა საცავს ემსახურება ისე, როგორც სხვა მონაცემთა ბაზების სისტემები, რომელთა საშუალებითაც ხდება, მონაცემთა დამოუკიდებელ პროგრამათა ინტეგრაცია, შენახვა და მართვა. მონაცემთა ბაზის მართვის სისტემა უზრუნველყოფს ადამიან-კომპიუტერს შორის ინფორმაციის გაცვლას, შეიცავს მონაცემთა დიდ რაოდენობას კანონზომიერი კავშირებით. მონაცემთა ბაზების მართვის სისტემა შეადგენს მონაცემთა საცავის ცენტრალურ ნაწილს.

➢ *არქივირებული შენახვის სისტემა:* უზრუნველყოფს მონაცემთა დაცვას და მათ არქივირებულ შენახვას მონაცემთა საცავში. მონაცემთა არქივირებული შენახვა, როგორც ცალკე სისტემა, მონაცემთა საცავში ამცირებს მეხსიერების უჯრედებს და ზრდის მუშაობის ეფექტურობის ხარისხს. არქივირების ეფექტური სისტემა მნიშვნელოვანია, რადგან მოკლე ვადაში შესაძლებელია მონაცემთა გადმოტვირთვა მომხმარებელთა მოთხოვნების შესაბამისად.

ხშირად სისტემაში თავს იყრის უსარგებლო ინფორმაციის ნაკადი, იკავებს დიდ ადგილს და აფერხებს სისტემის მუშაობის ეფექტურობას, არქივირებული სისტემის დახმარებით ხდება ასეთი ინფორმაციის განადგურება. შესაძლოა ასევე დეფექტური ტრანზაქციის შედეგად მოხდეს მონაცემთა „დაზიანება“. ამ შემთხვევაში ამოქმედდება მონაცემთა დაცვის სისტემა, რაც უზრუნველყოფს დეფექტების აღმოფხვრას და არასასურველი ინფორმაციის წაშლას.

არქივირებული შენახვისას ყურადღება უნდა გავამახვილოთ ინფორმაციის შენახვის კანონებზე, რომელიც ითვალისწინებს არქივში ინფორმაციის შენახვის ვადებს, რადგან შენახული ინფორმაცია გარკვეული პერიოდის შემდეგ კარგავს აქტუალობას.

➢ *მონაცემთა საცავის ანალიზური დამუშავება:* OLAP ტექნოლოგიის გამოყენებით. მონაცემთა მრავალგანზომილებიან კომპლექსური ანალიზის ტექნოლოგიას უწოდებენ OLAP-ს, რომელიც ნიშნავს „მონაცემთა ოპერატიულ ანალიზს“. მონაცემთა საცავში იგი წარმოადგენს მნიშვნელოვან ძირითად კომპონენტს.

OLAP ინსტრუმენტი პირველად მონაცემებს ასახავს ინფორმაციის სახით, რომლის დახმარებითაც შესაძლებელი ხდება საწარმოო მოცულობის შესახებ ვიქონიოთ რეალური წარმოდგენა. ამავე დროს იგი უნიკალური ინსტრუმენტია, რომელიც საშუალებას გვაძლევს სხვადასხვა ანალიზური ჭრილით ჩავატაროთ ინფორმაციის მრავალგანზომილებიანი ანალიზი.

2.5. ოპერატიულ მონაცემთა ბაზის ინფორმაციის კონვერტირების ინსტრუმენტი

მონაცემთა საცავებში ინფორმაციული ბლოკები შეივსება ოპერატიულ მონაცემთა ბაზებიდან. გარე ორგანიზაციების საწყისი მონაცემები საცავში თავს იყრის ინტერნეტის ან სხვა სახის კომუნიკაციებიდან მოსული ინფორმაციით.

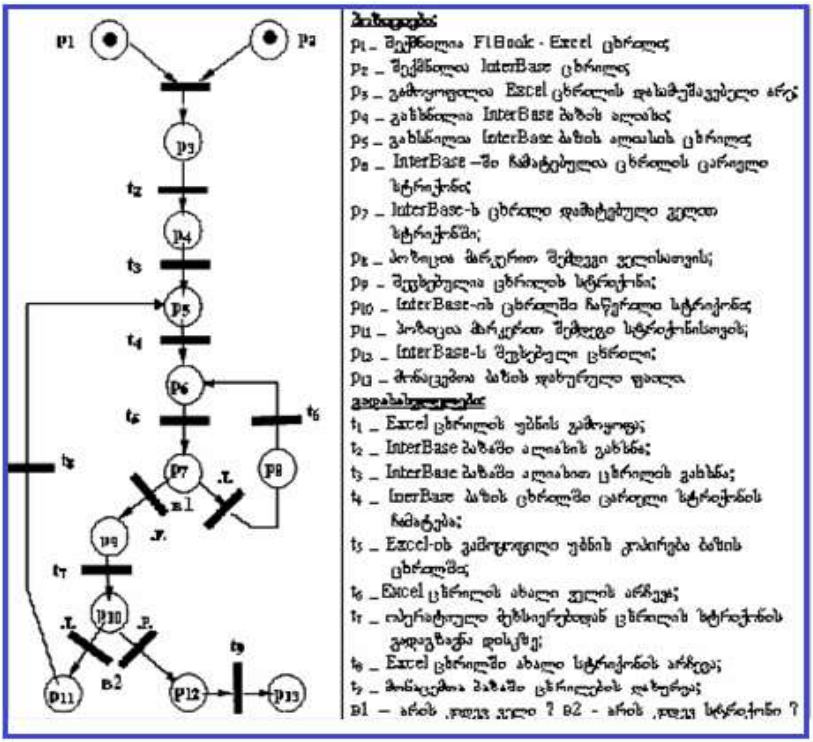
როგორც ანალიზმა გვიჩვენა, ელექტრონული ბიზნესისა და კომერციის ობიექტებზე (მაგალითად, დიდი სავაჭრო ცენტრები) თითქმის 60-70% გამოიყენება MsExcel პაკეტზე აგებული ცხრილები (შეკვეთები, ფაქტურები და სხვა ანგარიშები). მონაცემთა საცავი რელაციური ტიპის ბაზებთან სამუშაოდაა ორიენტირებული, როგორებიცაა, მაგალითად, MsAccess, SQL Server, Oracle, MySQL, PostgreSQL და ა.შ. [63,64].

ამ მონაცემთა ბაზებს აქვს როგორც ერთმანეთთან, ასევე Ms Excel-თან მონაცემთა ცხრილების გაცვლის შესაძლებლობა (Import/Export, MsQuery ფუნქციების ან ინსტრუმენტების სახით და ODBC დრაივერით).

წინამდებარე პარაგრაფში ჩვენ მიერ შემუშავებული და წარმოდგენილია ექსცელის ცხრილების კონვერტაციის ფუნქციის საილუსტრაციო ალგორითმული სქემა მონაცემთა განაწილებული ბაზების მართვის სისტემისთვის. წლების განმავლობაში იგი აქტიურად გამოიყენებოდა Borland C++ Builder და Borland Delphi საბაზო ტექნოლოგიების პროგრამული აპლიკაციებით [27].

ამოცანა მდგომარეობს მონაცემთა საცავის განაწილებული რელაციური ბაზის ავტომატიზებულად შესავსებად მონაცემთა პირველადი წყაროების ელექტრონული ცხრილებიდან.

ნახაზზე ნაჩვენებია აღნიშნული ამოცანის გადაწყვეტის ალგორითმის შესაბამისი პეტრის ქსელის გრაფი, რომელშიც ასახულია შესასრულებელ პროცედურათა მიმდევრობა.

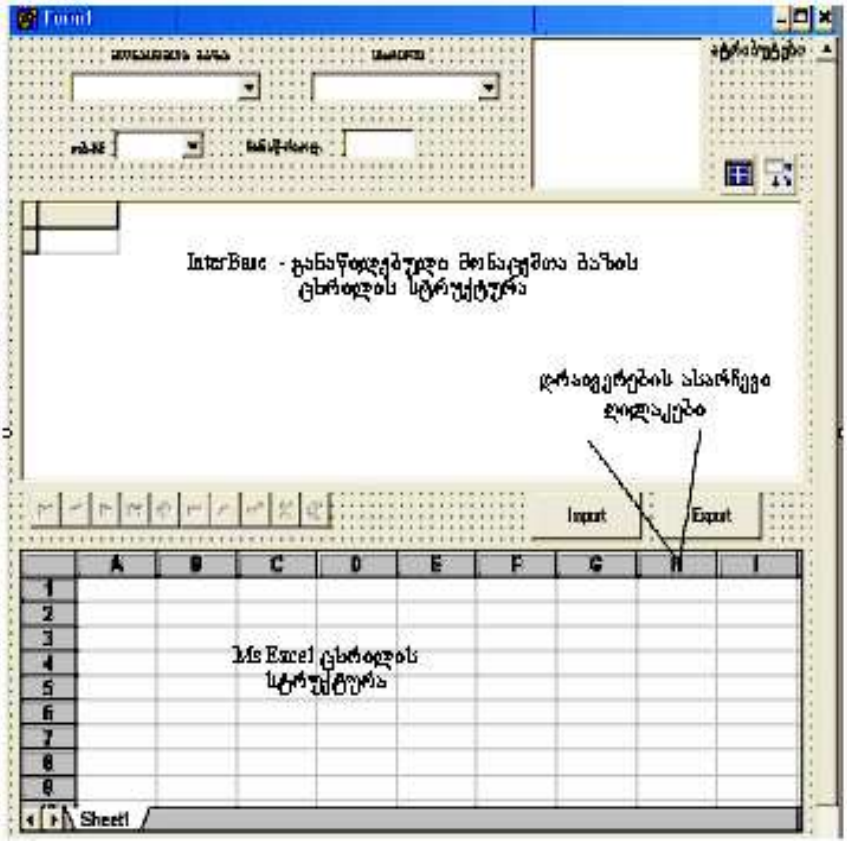


ნახ.2.9. პეტრის ქსელის გრაფი ღილაკისთვის ButtonClick - Import

ქვემოთ, 2.1 ლისტინგში მოცემულია ამ ღილაკის შესაბამისი C++ პროგრამის ტექსტი ხოლო 2.10 ნახაზზე მონაცემთა ბაზის ადმინისტრატორის ინტერფეისის ფანჯარა.

```
//— Unit1.cpp —programis teqsis fragmenti –
int i,j, Ob_Num, Dan_Num, Rec_Num;
void  fastcall TForm1::FormCreate(TObject *Sender)
  {
    Session->GetAliasNames(cbAlias->Items);
   }
void  fastcall TForm1::cbAliasChange(TObject *Sender)
```

```
  { Session->GetTableNames(cbAlias->Text,"",true,false,cbTable->Items);
cbTable->ItemIndex = 0;
if (cbTable->Text == "")
  {
    ShowMessage("Table not selected"); return; } Table1->Active = false;
    Table1->DatabaseName = cbAlias->Text; Table1->TableName = cbTable-
>Text;
    Table1->Active = true;
    if(Table1->Active)  Table1->GetFieldNames(cbField->Items);
  }
//————————— Button of Import ———————————— void      fastcall
TForm1::Button1Click(TObject *Sender)
 {
   AnsiString Info; if(Table1->Active)
     {
       int t=Table1->FieldCount;
       for(i=3; i<Rec_Num+3; i++)          // str_num
          {  Table1->Append();
             for(j = 1; j < t; j++)         // col_num
                {
                  Label3->Caption=t;
                  Table1->FieldValues["OBJNUM"] = Ob_Num; Info = Table1->
                           Fields->Fields[j]->FieldName; Label1->Caption=j;
                  Excel->SetSelection(i,j,i,j);    // choice of Range if(Excel->Text != 0)
                    {
                      Label2->Caption=Excel->Text;
                      Table1->FieldValues[Info] = Excel->Text;
                     }
                } // for j Table1->Post();
          }  // for i
     } // for if( )
 }
```



ნახ.2.10. მბ ადმინისტრატორის ინტერფეისი
მონაცემთა კონვერტაციის ამოცანისათვის

III თავი

Data Mining - მონაცემთა ინტელექტუალური ანალიზი

3.1. მონაცემთა მოპოვება (Data Mining)

Data Mining – მონაცემთა მოპოვება ან *მონაცემთა ინტელექტუალური ანალიზი* არის დიდ მონაცემთა ერთობლიობის დახარისხების პროცესი კანონზომიერებებისა და ურთიერთკავშირების დასადგენად, რის ბაზაზეც შესაძლებელია ბიზნეს-პრობლემების გადაჭრა მონაცემთა დახმარებით.

მონაცემთა მაინინგი აერთიანებს ხელოვნურ ინტელექტს, სტატისტიკურ ანალიზს და მონაცემთა ბაზის მართვის სისტემებს, რათა მონაცემთა საცავიდან მოხდეს საჭირო *ცოდნის მოპოვება.*

იგი გამოიყენება გარე ინტერფეისის მართვისთვის. მისი ცხრილური ნაწილი განკუთვნილია ონლაინ ანალიზის პროცესისთვის (OLAP). გამოწვევები, რომელიც უკავშირდება OLAP სისტემის მხარდამჭერ გარემოს, არის ცხრილური ნაწილი, მრავალ გიგაბაიტიან მონაცემთა ბაზით.

მონაცემთა მაინინგის პროცესში, ხდება ინტელექტუალური ანალიზის შედეგად ინფორმაციის მიღება, რაც მნიშვნელოვნად განსხვავდება ტრადიციული მონაცემთა ბაზის მოთხოვნებით მიღებული ინფორმაციისგან.

მონაცემთა ბაზის მოთხოვნა დამოკიდებულია იმ დონეებზე თუ რომელ დონეზე გვინდა ოპერატიული მონაცემების ქვეჯგუფის გამოვლენა და გამომავალი ინფორმაციის მიღება.

მონაცემთა მაინინგში არაა სტანდარტულად განსაზღვრული მოთხოვნის ენა და, შესაბამისად, მოთხოვნებიც არ არის მკაფიოდ წარმოდგენილი. ამრიგად მიღებული შედეგობრივი ინფორმაცია არის ბუნდოვანი, არაზუსტი და არ წარმოადგენს მონაცემთა ბაზის ქვეჯგუფს, ისევე როგორც გამოყენებული საბაზისო ოპერატიული მონაცემები.

მონაცემთა საცავის ფორმირების პროცესში ოპერატიული მონაცემები დაჯგუფებულია განსხვავებული ნიშანთვისებების მიხედვით.

მაგალითად, საბანკო სისტემაში პირობითად რომ გავაერთიანოთ სამი თვის პერიოდის განმავლობაში არსებული საბანკო ვალდებულებები. მოთხოვნებს ექნება ასეთი სახე:

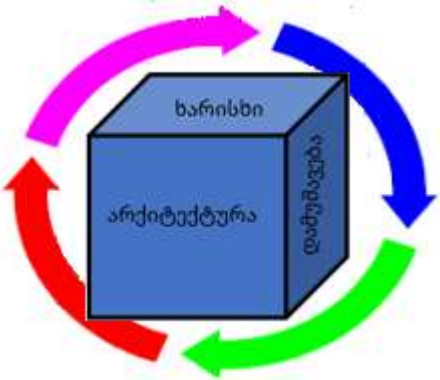
„ყველა მსესხებლის იდენტიფიცირება, რომელთაც აქვთ მსგავსი ინტერესები“ ან „წევრები, რომელთაც ხშირად წარმოექმნებათ საბანკო ვალდებულება“.

მიუხედავად შინარსობრივი მსგავსებისა და განსხვავებული სინტაქსისა, რომელმაც შესაძლოა ერთიდაიმავე შედეგზე გაგვიყანოს, მონაცემთა ტრანზაქციისას ჩაიწერება განსხვავებული მოთხოვნების სახით.

სუპერმარკეტებში მოთხოვნათა იდენტიფიკაცია დაკავშირებულია მონაცემთა მაინინგთან. არსებობს სტანდარტულად განსაზღვრული რამდენიმე ნივთი/პროდუქტი, რომლებიც ლოგიკურად დაკავშირებულია ერთმანეთთან. მაგალითად: „ლუდი და ჩიპსი“, „პური და რძე“, „ყველი და კარაქი“ ეს ნიშნავს იმას, რომ მომხმარებელთა უმეტესობა აღნიშნულ პროდუქციიდან თუ ყიდულობს ერთ-ერთ რომელიმეს, ყიდულობს მეორესაც.

მობილური კომპანიებიც ტარიფებთან და სპეციალურ პაკეტებთან დაკავშირებით, რომელთაც მომხმარებელს სთავაზობს, გადაწყვეტილებას იღებს ბაზრის კვლევის საფუძველზე, პიკურ სიტუაციებში თუ როგორი გამოყენება აქვს მათ პროდუქტს.

მომხმარებლებს შეუძლიათ გამოიყენონ მონაცემთა მაინინგის ტექნოლოგია, მონაცემთა საცავიდან განსხვავებული სახის ინფორმაციის მიღებისთვის, რაც დაეხმარებათ გადაწყვეტილების მიღების პროცესში. როგორც 3.1 ნახზიდან ჩანს, შესაძლებელია ხარისხის, არქიტექტურის და მონაცემთა დამუშავების მეთოდების მიხედვით ინფორმაციის მიღება.



ნახ.3.1. მეტამონაცემთა რეპოზიტორის სხვადასხვა ხედებით წარმოდგენა

მაგალითად, თუ გარკვეული წიგნების ჩამონათვალს მკითხველები იყენებენ იშვიათად ერთ-ერთ ბიბლიოთეკაში, ხოლო იგივე წიგნებზე მეტი მოთხოვნა დაფიქსირდება სხვა ბიბლიოთეკაში, აღნიშნული წიგნების გადატანა მოხდება მეორე ბიბლიოთეკაში, რათა ეფექტურად მოხდეს მათი გამოყენება.

ასეთი სახის ცოდნის აღმოჩენა შესაძლებელია მხოლოდ ბიბლიოთეკების მონაცემთა ბაზების ინტეგრირებით, საიდანაც სპეციალური ალგორითმების საშუალებით, რომელსაც იყენებს მონაცემთა მაინინგი, მიიღება ცოდნა, რომელიც განხილული მაგალითის საფუძველზე ხელს უწყობს ბიბლიოთეკების ეფექტურ მუშაობას [65].

ტრადიციული მათემატიკური სტატისტიკა დიდი ხნის განმავლობაში ინარჩუნებდა ერთ-ერთ მნიშვნელოვან ადგილს, როგორც მონაცემთა ანალიზის ინსტრუმენტი, მაგრამ თანამედროვე გამოწვევების წინაშე აღნიშნული მიდგომა არაეფექტური აღმოჩნდა, რადგან მათემატიკური ანალიზის საფუძველზე შეუძლებელი იყო ჰიპოთეზათა წინასწარი ფორმულირება, მონაცემთა

ოპერატიული ანალიზი, მრავალგანზომილებიანი ანალიზი და ა.შ.

მეოცე საუკუნის 90-იანი წლებიდან აქტიურად ინერგება და გამოიყენება მონაცემთა ანალიზის OLAP და Data Mining ტექნოლოგიები.

მონაცემთა მაინინგის პროცესი შედგება რამდენიმე ეტაპისგან, სადაც გათვალისწინებულია მონაცემთა შედარების, ტიპიზაციის, კლასიფიკაციის, გაერთიანების, აბსტრაგირების და განმეორების ელემენტები. მონაცემთა მოპოვების პროცესი პირდაპირ კავშირშია გადაწყვეტილების მიღების პროცესთან.

მონაცემთა მაინინგის პროცესის დროს გათვალისწინებული უნდა იყოს შემდეგი ეტაპები:

- საგნობრივი სფეროს ანალიზი;
- ამოცანის დასმა;
- მონაცემების მომზადება;
- მოდელების აგება;
- მოდელების შეფასება და შემოწმება;
- მოდელის შერჩევა;
- მოდელების კორექცია და განახლება.

➢ *ეტაპი 1. საგნობროვი სფეროს ანალიზი.*

აღნიშნულ ეტაპზე უნდა განისაზღვროს საგნობრივი სფეროს მახასიათებელი ნიშან-თვისებები და მათ შორის ურთიერთდამოკიდებულებები.

საგნობრივი სფერო შედგება სხვადასხვა ნიშან-თვისებების მქონე ობიექტებისგან, რომლთა შორის არის გარკვეული ლოგიკური კავშირები და შეიცავს ინფორმაციას. საგნობრივი სფეროს აღწერისთვის გამოიყენება სხავადსხვა მეთოდები, მაგალითად, როგორიცაა: სტრუქტურული ანაიზის მეთოდი, მონაცემთა ნაკადების დიაგრამის აგების ჰეინ-სარსონის მეთოდი, ობიექტორიენტირებული ანალიზის UML მეთოდი და სხვ. [66,67].

- *ეტაპი 2. ამოცანის დასმა*

ამ ეტაპზე უნდა მოხდეს ამოცანის ფორმულირება და ფორმალიზაცია ასევე გათვალისწინებული უნდა იყოს ობიექტის სტატიკური და დინამიკური მდგომარეობის აღწერა. სტატიკურ აღწერაში იგულისხმება ობიექტების და მათი ნიშან-თვისებების ასახვა, ხოლო დინამიკურ აღწერაში კი - ობიექტების ქცევის და ამ ქცევის გამომწვევის მიზეზების ასახვა.

ზოგიერთ შემთხვევაში საგნობრივი სფეროს აღწერა და ამოცანის დასმა ერთიანედება, როგორც ერთი ეტაპი.

- *ეტაპი 3. მონაცემთა მომზადება*

ეტაპის მიზანია ბაზის ფორმირება მონაცემთა მაინინგისათვის. აქ ხდება მონაცემთა მოდელირება ანუ იმ მონაცემთა ანალიზი და განსაზღვრა, რომელიც საჭიროა მონაცემთა მოპოვებისათვის. მესამე ეტაპზე ხდება გეოგრფიულად, ორგანიზაციულად და ფუნქციონალურად განაწილებულ მონცემთა შესწავლა, ასევე შიგა და გარე მონაცემთა წყაროების ანალიზი. მოცემულ ეტაპზე ხდება მონაცემთა კოდირებაც.

- *ეტაპი 4. მოდელების აგება*

მონაცემთა გარკვეული კანონზომიერების და თვისებების გამოკვლევისათვის ხდება სხვადასხვა მოდელების აგება. ერთერთია პროგნოზირების მოდელი, რომელიც ისტორიულინფორმაციულ ფაქტებს ეფუძნება. ასეთი ტიპის მოდელებში ხდება უცნობი მნიშვნელობის მინიჭება და შემდეგ პროგნოზირება იმისა, თუ რა გავლენას მოახდენს უცნობი მნიშვნელობა მიმდინარე პროცესზე. როგორც დასაწყისში აღვნიშნეთ, მონაცემთა მაინინგი პირდაპირ კავშირშია მონაცემთა ანალიზთან, ამდენად პროგნოზირების მოდელი, საკმაოდ მნიშვნელოვანია მოდელირების პროცესში.

- *ეტაპი 5. მოდელების შეფასება და შემოწმება*

მოწმდება ყველა მოდელის კორექტულობა და ეფექტურობა. არსებული სიტუაციიდან გამომდინარე ხდება მოდელის ქცევის შემოწმება და ეფექტურობის შეფასება, რაც ძალიან მნიშვნელოვანია მაინინგის პროცესისთვის.

- *ეტაპი 6. მოდელის შერჩევა*

მოდელის შერჩევის დროს გამოიყენება ალგორითმები, რომლებიც ორიენტირებულია საგნობრივ სფეროზე. ალგორითმში აუცილებელია განისაზღვროს იმ შესაძლო ცვლადების რაოდენობა, რომელიც უშუალო კავშირშია საგნობრივი სფეროს შემადგენელი ობიექტების მახასიათებლებთან.

- ეტაპი 7. მოდელების კორექცია და განახლება

ახალი ინფორმაციის მიღებისას შესაძლებელია არსებული მოდელის კორექცია და განახლება ობიექტის კლასიფიკაციის შესაბამისად და იმ ანომალიების გათვალისწინებით რომელთა წარმოქმნის ალბათობაც არსებობს.

როგორც ზემოთ აღვნიშნეთ Data Mining არის ე.წ. ინტელექტუალური მოდელის ტიპი, რომელიც ერთის მხრივ, განისაზღვრება როგორც გენერირებულ ჰიპოთეზათა სახეობა, ხოლო მეორეს მხრივ, მნიშვნელოვანი გადაწყვეტილების მიღება ანალიზის პროცესისას. ხშირ შემთხვევაში მონაცემთა ანალიზის მარტივი მოდელი წარმოდგენილია ფორმით, რომელიც პასუხობს „თუ – მაშინ“ ლოგიკას. მონაცემთა მაინინგში გამოიყენება ე.წ. „გადაწყვეტილების ხე“, სადაც არის ხშირად გამოყენებადი სპეციალური წესების ერთობლიობათა ფორმა. კომპლექსური პროცესი დამყარებულია პრედიკტების ლოგიკაზე ან გამოიყენება სპეციალური ნეირონული ქსელები როგორიცაა 1980 წელს ფინელი პროფესორის ტ. კოჰონენის მიერ შემოთავაზებული კოჰონენის ქსელი [68].

მონაცემთა სტრუქტურული ასახვა მრავალგანზომილებიან ნეირონულ ქსელებში, მათი ფარული შრეების სტრუქტურიდან გამომდინარე, დაკავშირებულია საკმაოდ რთულ პროცესებთან. კოჰონენის ქსელი კი უზრუნველყოფს მონაცემთა დამუშავების პროცესში ქსელს შესასვლელზე მათ თანმიმდევრულად მიწოდებას, რის შემდეგაც ქსელი იწყებს გადაწყობას შემავალი მონაცემების კანონზომიერების მიხედვით და არა გამომავალი მონაცემების სტანდარტული შაბლონების შესაბამისად.

მონაცემთა მიწოდების დროს, ქსელის შესასვლელზე თანმიმდევრული გადაცემისას განისაზღვრება ე.წ. „დომინანტი ნეირონი“, რომელიც შემდეგში გამოიყენება მეზობელ ნეირონებში „წონების“ განაწილების ცენტრად.

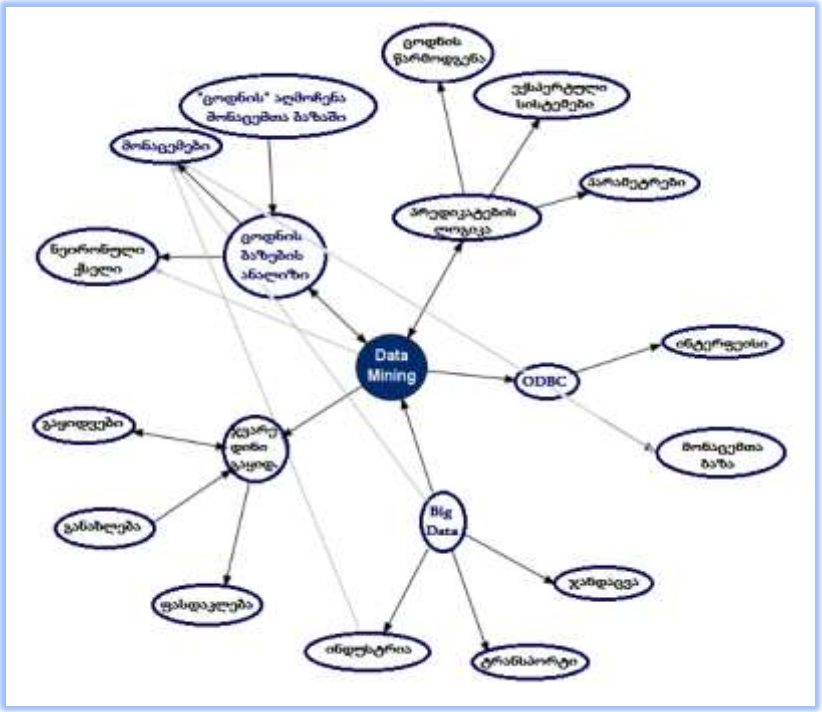
აღნიშნული მიდგომის გამოყენება, ერთის მხრივ, შესაძლებელს ხდის საწყისი და საბოლოო მონაცემთა კანონზომიერებების ძებნას და ანალიზს, და ამავდროულად, კოჰონენის მეთოდი უზრუნველყოფს მრავალგანზომილებიანი შემავალი სივრცის გამოსახვას ორგანზომილებიან გამავალ სივრცედ.

ხდება მონაცემთა შეკუმშვა, რაც საბოლოოდ აჩქარებს ალგორითმის შესრულებას. ნეირონული ქსელის გამოყენების პროცესში აუცილებელ პირობაა ე.წ. „დომინანტი ნეირონის“ განსაზღვრა, რომლის მიხედვითაც ხდება ნეირონული ქსელის კოეფიციენტების კორექტირება.

კოჰონენის ქსელი, მრავალშრიანი ნეირონული ქსელისგან განსხვავებით, შეიცავს მხოლოდ ორ შრეს: შემავალს და გამომავალს, რაც ამარტივებს და ეფექტურს ხდის აღნიშნული ქსელის გამოყენებას. ქსელში არსებული მონაცემების ძიების პროცესში არსებული კომპლექსური მოდელებიდან უნდა შეირჩეს ისეთი მოდელი, რომელიც შეძლებს ევრისტიკული მეთოდების გამოყენებას, რადგან იგი აჩქარებს მისაღები შედეგის დადგომის პროცესს.

მონაცემთა მაინინგი უზრუნველყოფს მონაცემთა საცავთან დაკავშირებას და კონკრეტულ კომპანიის მონაცემებთან წვდომას. საჭირო მონაცემები მომხმარებელს მიეწოდება ცხრილების სახით.

ტექნოლოგია Data Mining (discovery-driven data mining) ეფუძნება შაბლონების (templates) კონცეფციას, რომელიც აისახება მონაცემთა მრავალასპექტიან ფრაგმენტებთან ურთიერთკავშირში. ეს შაბლონები წარმოდგენილია კანონზომიერად, მონაცემთა ქვეჯგუფებად და მომხმარებელს კომპაქტურად მიეწოდება გასაგები ფორმით. შაბლონთა ძიება მიმდინარეობს მეთოდების გამოყენებით, რომელიც აპრიორულად არაა შეზღუდული. მონაცემთა მაინინგის სტრუქტურა მოცემულია 3.2 ნახაზზე [24]:



ნახ.3.2. Data Mining-ის ზოგადი სტრუქტურა

Data Mining -ის კონცეფცია მოიაზრება, როგორც მეთოდების და ალგორითმების გამოყენების შესაძლებლობა, როდესაც მონაცემთა ბაზასა და სამიზნე ობიექტს შორის ავტომატურად ხდება ემპირიულ მონაცემთა ექსტრაქცია.

მას ასევე შესაძლებელია ვუწოდოთ ინტელექტუალური ანალიზის ინსტრუმენტი, რომელიც მნიშვნელოვან მონაცემებს აგენერირებს, როგორც „ცოდნა“-ს. მონაცემთა მაინინგის მიზანია მომხმარებელს დაეხმაროს გადაწყვეტილების მიღებაში საინტერესოსა და საჭიროს შორის.

აქ გამოიყენება ხელოვნური ინტელექტის და სტატისტიკის ინტეგრირებული მთოდები. იგი არის კომპლექსური პროცესი, როდესაც კლასიფიცირდება ცოდნის ბაზები. მონაცემთა მოპოვების ძირითადი ამოცანაა განსაზღვროს მონაცემთა:

- კლასიფიკაცია;
- სეგმენტაცია;
- პროგნოზირება;
- დამოკიდებულებათა ანალიზი;
- გადახრათა ანალიზი.

მონაცემთა *კლასიფიკაციისას*, ყალიბდება კლასები ერთგვარ მონაცემთა ობიექტების სახით;

სეგმენტაციისას, მონაცემთა ობიექტები ლაგდება ჯგუფებად ერთი მახასიათებელი ნიშანთვისების შესაბამისად. ამავდროულად ერთ ჯგუფში შემავალი ყველა ობიექტი უნდა იყოს ჰომოგენური;

პროგნოზირებისას წინასწარ ხდება უცნობი ნიშანთვისების განსაზღვრა და ყალიბდება როგორც „ცოდნა“;

დამოკიდებულებათა ანალიზი აღწერს ურთიერთდამოკიდებულებას ერთი ნიშანთვისების მქონე ობიექტებს ან განსხვავებულ ობიექტებს შორის;

გადახრათა ანალიზისას სრულად ხდება ისეთი ობიექტების იდენტიფიკაცია, რომლებიც არ შეესაბამება დამოკიდებულ

ობიექტებს. ამის გათვალისწინებით დგინდება ობიექტებს შორის ახალი დამოკიდებულება.

თანამედროვე მოთხოვნათა სპეციფიკაცია უნდა პასუხობდეს შემდეგ გამოწვევებს:

- მონაცემებს აქვს უსაზღვრო მოცულობა;
- მონაცემები არის სხვადასხვაგვარი (რაოდენობრივად, ხარისხობრივად, სინტაქსურად);
- შედეგი უნდა იყოს კონკრეტული და გასაგები;
- პირველადი მონაცემების დამუშავების ინსტრუმენტი უნდა იყოს მარტივი.

3.2. Data Mining და Big Data

ხშირ შემთხვევაში ტერმინს Big Data და Data Mining ერთ კონტექსტში გამოიყენებენ. აქ მნიშვნელოვანია ის გარემოება, რომ ორივე თვალსაზრისი ერთმანეთისგან განუყოფელია. Big Data მოიცავს განსაკუთრებით დიდი მოცულობის მონაცემებს, სადაც ტრადიციული მეთოდები და ინსტრუმენტები არაეფექტურია, მოცემულ დროის ერთეულში მათ დასამუშავებლად. Data Mining ხშირად გამოიყენება დიდი მოცულობის მონაცემთა ბაზებში, ისევე როგორც Big Data -ში.

Data Mining ანალიზის დროს აღწერს შემავალ საწყის მონაცემებს მათი რელევანტური დამომოკიდებულების შესაბამისად, ამავე დროს შესაძლებელია გამოყენებულ იქნას პატარა მონაცემთა ბაზებიც. Big Data შემთხვევაში მიეწოდება დიდი მოცულობის მონაცემები და გარდაიქმნება ტექნიკურ პლატფორმად, რომელიც არის ეფექტური საშუალება მონაცემთა ამოცნობისა და ახალი ჯგუფის გამოვლენისათვის, სადაც ხდება ჰიპოთეზათა გენერირება, რაც ვერიფიკაციის შედეგად იხვეწება და ზუსტდება [24].

მონაცემთა მაინინგი იყენებს სტატისტიკურ და ხელოვნური

ინტელექტის ალგორითმებს, რაც აძლევს იმის საშუალებას, რომ Big-Data ტექნოლოგიის სფეროში მიღწეულ იქნას მნიშვნელოვანი წარმატება. შესაძლებელი ხდება როგორც სტრუქტურირებული, ისე არასტრუქტურირებული მონაცემების დამუშავება, როდესაც Big-Data და Data Mining გამოიყენება ერთად და მოიპოვება ცოდნა იმის შესახებ თუ რომელი მონაცემებია სტატისტიკური თვალსაზრისით დაჯგუფებადი, ან რომელი კატეგორიის ჯგუფია ჯერ კიდევ ამოუცნობი და აღუწერელი. ამავე დროს განისაზღვრება ის წესები და ურთიერთდამოკიდებულებები, რომლებიც საანალიზო ელემენტების სივრცეშია მოცემული.

3.3. Data Mining -ის გამოყენების სფეროები

მონაცემთა მაინინგს მომოვალში აქვს ძალიან დიდი გამოყენების პოტენციალი, თუმცა დღესაც გამოიყენება ბევრი მიმართულებით მაგ. საბანკო სექტორში, სადაზღვევო სისტემებსი, ტელეკომუნიკაციაში მედიცინაში და ა.შ.

3.3.1. კომერციული საბანკო საქმიანობა

მონაცემთა მოპოვების მოწინავე ტექნოლოგიები ფართოდ გამოიყენება საბანკო სექტორში [85,86]. იგი ხელს უწყობს შემდეგი ამოცანების შესრულებას:

- *საკრედიტო ბარათების თაღლითური მოხმარების გამოვლენა.* წინა შესრულებული ტრანზაქციების ანალიზის შედეგად, რაც შემდგომში აღმოჩნდა თაღლითური, ბანკს აძლევს შესაძლებლობას გამოავლინოს მსგავსი თაღლითობის გარკვეული სტერეოტიპი;
- *მომხმარებელთა (კლიენტთა) სეგმენტაცია.* მომხმარებლის სხვადასხვა კატეგორიებად დაყოფა, ხელს უწყობს ბანკების მარკეტინგული პოლიტიკის მიზანმიმართულ და ეფექტურ

განვითარებას, რაც გულისხმობს განსხვავებული სახის სხვადასხვა მომხმარებლთა ჯგუფებისთვის შესაბამისი სერვისების შეთავაზებას;

- მომხმარებლის *ცვალებადობის პროგნოზირება.* Data Mining ხელს უწყობს ბანკებს საკუთარი მომხმარებლის ფასეულობათა განსაზღვრაში და შესაბამისი კატეგორიის მოდელის პროგნოზირების ჩამოყალიბებაში. ამის საფუძველზე დგინდება კლიენტზე ორიენტირებული განსაზღვრული წესების ჩამონათვალი თითოეული კატეგორიის მომსახურებაში.

3.3.2. ტელეკომუნიკაციების სფერო

ტელეკომუნიკაციების ინდუსტრიაში Data Mining ხელს უწყობს კომპანიებს მარკეტინგის და ფასების საკუთარი პროგრამების აქტიურად წარდგენაში, რათა შეინარჩუნოს არსებული მომხმარებელი და მოიზიდოს ახალი [87,88].

ტიპურ ღონისძიებათა შორის, აღსანიშნავია შემდეგი:

- გამოწვევების დეტალური მახასიათებლების შესახებ ჩანაწერების ანალიზი, რაც ემსახურება მსგავსი სტერეოტიპების მქონე მომხმარებელთა კატეგორიების გამოვლენას, მათი მომსახურებისას სასურველი ფასების და მომსახურების პაკეტის შემუშავებას;

- მომხმარებელთა ლოიალობის გამოვლენა: შესაძლებელია Data Mining-ის გამოყენება იმ მომხმარებლის დახასიათების ამოცნობის მიზნით, რომელმაც ერთხელ მაინც ისარგებლა მოცემული კომპანიის მომსახურებით და დიდი ალბათობით შეინარჩუნებს ერთგულებას მის მიმართ. შესაბამისად, მარკეტინგისთვის გამოყოფილი თანხების გამოყენება მიზანშეწონილია გარანტირებული დაბრუნების სფეროში.

3.3.3. სადაზღვევო საქმიანობა

წლების განმავლობაში სადაზღვევო კომპანიები ახორციელებს დიდი მოცულობის მონაცემთა ბაზის შევსებას. ამ სფეროში

Data Mining-ს აქვს მოქმედებისა და მეთოდების გამოყენებისათვის დიდი შესაძლებლობები [89,90].

- თაღლითობის გამოვლენა: სადაზღვევო კომპანიებს შესწევს უნარი შეამციროს თაღლითობის ზღვარი, გარკვეული სტერეოტიპების შესაბამისი გადახდილი სადაზღვევო ანაზღაურების განცხადებებში მოძიების მეთოდით, რაც ექიმებს, იურისტებსა და განმცხადებლებს შორის ურთიერთობის დამახასიათებელ ფაქტორზე იქნება დაფუძნებული;

- რისკის შეფასების ანალიზი. გადახდილ განაცხადებთან დაკავშირებული რიგი ფაქტორების გამოვლენის შედეგად, დამზღვეველს შეუძლია შეამციროს საკუთარი ზარალი ვალდებულებებზე. ცნობილია შემთხვევა, როდესაც აშშ-ში მსხვილმა სადაზღვევო კომპანიამ გამოავლინა, რომ ქორწინებაში მყოფ დაავადებულ პირებზე გადახდილი თანხები ორმაგად აღემატებოდა თანხებს, რაც გადახდილი იყო მარტოხელა პირებზე. კომაპნიამ მოახდინა სწრაფი რეაგირება აღნიშნულ ფაქტზე, შეცვალა საკუთარი პოლიტიკა დაოჯახებულ მომხმარებლეზე შეღავათიანი სესხების გაცემის წესებში და დაზოგა მილიონები.

3.3.4. სატრანსპორტო წარმოების და გადაზიდვების სფერო

საავტომობილო ინდუსტრიისა და სატრანსპორტო გადაზიდვების სფეროებში მნიშვნელოვანი ადგილი უკავია ინტელექტუალური ანალიზისა და პროგნოზირების ამოცანების გადაწყვეტას. მონაცემთა მოპოვების, მანქანური დასწავლის და ხელოვნური ინტელექტის მეთოდების საფუძველზე [91-93].

- საავტომობილო მრეწველობის განვითარება: ავტომობილების აწყობისას აუცილებელია თითოეული მომხმარებლის მოთხოვნის გათვალისწინება. აქედან გამომდინარე საჭიროა გარკვეული, მოთხოვნადი მახასიათებლების პროგნოზირება და ცოდნა იმისა თუ რაზეა მომხმარებელი ორიენტირებული [94];

- გარანტიების პოლიტიკა: მწარმოებელმა უნდა ივარაუდოს მომხმარებელთა ის რაოდენობა, რომელმაც შესაძლოა წარადგინოს საგარანტიო მოთხოვნები და მოთხოვნების საშუალო ღირებულება;
- ავიახაზებით მგზავრი-მომხმარებლების წახალისება: შესაძლებელია, რომ ავიაკომპანიებმა გამოავლინოს მომხმარებელთა გარკვეული ჯგუფი, რომელიც ხშირად სარგებლობს აღნიშნული ავიაკომპანიის მომსახურებით, გარკვეული წახალისების შედეგად.

3.3.5. სამედიცინო სფერო

სამედიცინო მონაცემთა მოპოვება არის მონაცემთა მეცნიერების მეთოდებისა და ინსტრუმენტების ერთობლიობა, რომელიც გამოიყენება მტკიცებულებებზე დაფუძნებული სამედიცინო ინფორმაციის გენერირებისთვის. ჯანდაცვის მონაცემთა მოპოვების ტექნოლოგიები გამოიყენება მრავალ სფეროში, როგორიცაა მაგალითად, ბიოტექნოლოგია, ფარმაცევტული კვლევა და სამედიცინო მეცნიერება [95,96].

არსებობს სამედიცინო დიაგნოსტიკის რამდენიმე საექსპერტო სისტემა. შემუშავებულია გარკვეული წესები (ცოდნა), რომლებიც აღწერს სხვადასხვა დაავადების და სხვადასხვა სიმპტომის ჯგუფებს. ასეთი წესების დახმარებით, შესაძლებელია დადგინდეს არა მარტო პაციენტის დაავადება, არამედ მისი მკურნალობის მეთოდიკა. აღნიშნული წესების მიხედვით, ასევე შესაძლებელი ხდება მედიკამენტოზული საშუალებების შერჩევა, მათი ჩვენების განსაზღვრა, უკუჩვენების დადგენა და სამკურნალო პროცედურებზე ორიენტირება, გაცილებით ეფექტური მკურნალობის პირობების შემუშავება.

Data Mining-ის ტექნოლოგიის გამოყენება უზრუნველყოფს სამედიცინო მონაცემებში შაბლონების შექმნას, რაც წარმოადგენს აღნიშნული წესების საფუძველს და შესაძლებელს ხდის ონლაინ რეჟიმში პაციენტის დიაგნოსტირებას და დანიშნულების მიცემას.

3.4. მონაცემთა მოპოვების სპეციალური ალგორითმები და პროგრამები

Data Mining ანალიზის პროცესში იყენებს ალგორითმებს, განსხვავებული ტიპის, კატეგორიისa და სტრუქტურის მონაცემებისათვის.

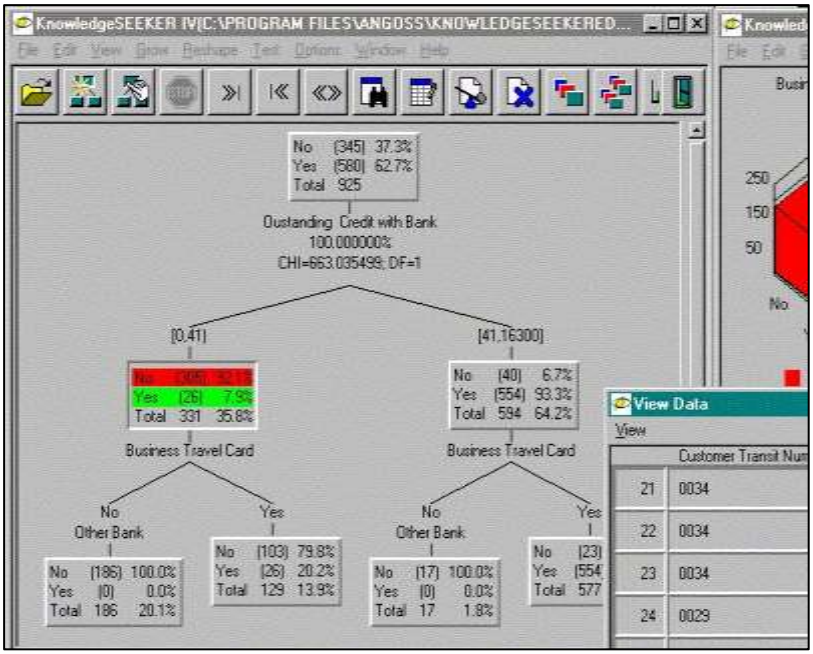
ალგორითმი შეიცავს კლასიფიქტორებს, რომელთაც აქვს განსხვავებული სტრუქტურა. კლასიფიკატორი ალგორითმში წინასწარ საზღვრავს მონაცემებს კლასიფიკაციის მიხედვით თუ სად განთავსდეს და რომელ კლასს ან კატეგორიას მიეკუთვნოს ახალი მონაცემი. არსებობს რამდენიმე ცნობილი ალგორითმი, რომელთა გამოყენებით შესაძლებელი ხდება მონაცემთა ვიზუალიზაცია, დამუშავება და პრიორიტეტების განსაზღვრა. განვიხილოთ რამდენიმე მაგალითი.

3.4.1. გადაწყვეტილების მიღების ალგორითმი C4.5

მონაცემთა მოპოვების ამოცანებში გამოყენებული ალგორითმებიდან ერთ-ერთი ყველაზე პოპულარულია C4.5 (ეს ალგორითმი ხშირად გამოიყენება მანქანური დასწავლში), როგორც გადაწყვეტილების ხის კლასიფიკატორი, რომელიც შეიძლება გამოყენებულ იქნას გადაწყვეტილების გენერირებისთვის, მონაცემთა გარკვეულ ნიმუშზე დაფუძნებული (ერთვარიანტული ან მრავალვარიანტული პროგნოზები) [69-70].

გადაწყვეტილებათა მიღების ხე **(decision trees)** მონაცემთა მაინინგში არის ერთ-ერთი ყველაზე განსაკუთრებული მიდგომა, სადაც ხის იერარქიულ სტრუქტურა კლასიფიცირებულია „თუ-მაშინ“ (if-then) ლოგიკით. გადაწყვეტილების მიღებისას ხდება განსაზღვრა, თუ რომელ კლასს მიეკუთვნის რომელიმე ობიექტი ან არსებული სიტუაცია, ამავე დროს ხდება პასუხის გაეცემა კითხვებზე, რომელიც ხის კვანძებზეა განთავსებული და ღებულობს დადებით ან უარყოფით პასუხს.

საკითხი დგას შემდეგნაირად, მაგალითად. „A პარამეტრის მნიშვნელობა მეტია x -ზე“, თუ პასუხი დადებითია, მაშინ გადასასვლელი იხსნება ხის მარჯვენა კვანძის მომდევნო დონეზე, ხოლო უარყოფითი პასუხის შემთხვევაში მარცხენა მხარეს, შემდეგ ისევ დგება მომდევნო კვანძის შესაბამისი კითხვა და ა.შ. ხის ხტრუქტურა საკმაოდ მარტივი და თვალსაჩინოა 3.3 ნახაზზე მოცემულია გადაწყვეტილებათა მიღების ხის სტრუქტურის ფრაგმენტი.



ნახ.3.3 გადაწყვეტილებათა მიღების ხის სტრუქტურა

გადაწყვეტილებათა ხე შესაძლებელია არ იყოს ერთადერთი საუკეთესო გამოსავალი მონაცემთა ასახვისას. აქ თანმიმდევრულად ხდება იმ ძირითადი მახასიათებლების წარმოდგენა და შეფასება, რომლებიც უშუალოდ განსაზღვრავს მონაცემის შესაბამისობას ამა თუ იმ კლასის მახასიათებელ ძირითად პარამეტრებთან.

დღეს ეს მეთოდი საკმაოდ გავრცელებულია და იყენებენ ისეთ დიდი სისტემებში როგორიცაა: See5/C5.0 (RuleQuest, ავსტრალია), Clementine (Integral Solutions, დიდი ბრიტანეთი), SIPINA (University of Lyon, საფრანგეთი), IDIS (Information Discovery, ამერიკა), KnowledgeSeeker (ANGOSS, კანადა) და ა.შ. [71-73].

როგორც დასაწყისში აღვნიშნეთ, **C4.5** ალგორითმი მონაცემთა წარმოდგენისას იყენებს გადაწყვეტილებათა მიღების ხეს, სადაც მონაცემები წინასწარ უნდა იყოს კლასიფიცირებული. კლასიფიკატორი არის ე.წ. ინსტრუმენტი, რომელიც გამოიყენება მონაცემთა მაინინგში, შედგება კლასიფიციცირებული მონაცემებისგან, რომლის საფუძველზეც ხდება წინასწარ განსაზღვრა, თუ რომელ კლასს უნდა მიეკუთვნოს ახალი მონაცემი.

მარტივი მაგალითის საფუძველზე ეს პროცესი შეიძლება ასე ავხსნათ: მაგალითად, სატრანსპორტო გადაზიდვებში განსხვავებული ტვირთების შესახებ გვაქვს მონაცემთა ჯგუფები, რომელიც მოიცავს რამდენიმე ძირითად ატრიბუტს [93]. მათი საშუალებით ხდება ტვირთის აღწერა. მაგალითად, *ლოკაციის ადგილი, ტრანსპორტირების საბოლოო ადილი, ტვირთის სახეობა, ღირებულება, მოცულობა, წონა, ზომა, კონტეინერთან თავსებადობა* და ა.შ.

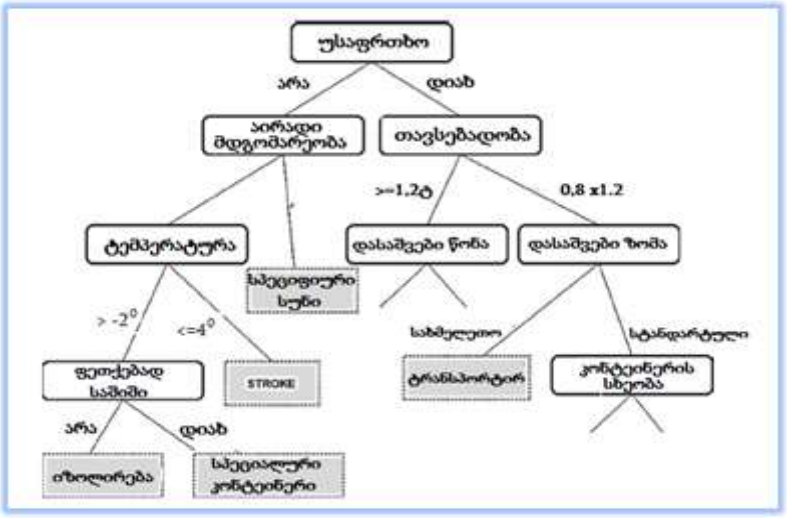
ჩვენ წინასწარ ვიცით განსხვავებული ტვირთების შესახებ ძირითადი მახასიათებელი პარამეტრები, რომელთაც ატრიბუტების სახით ვაერთიანებთ კლასებად, აღნიშნული ატრიბუტების საფუძველზე წინასწარ გვინდა განვსაზღვროთ ტვირთი საშიშია თუ არა. ტვირთი შესაძლებელია მოხვდეს ორი კლასიდან ერთში, „საშიში“ ან „უსაფრთხო“. ალგორითმი შეტყონინებას უგზავნის თითოეული ტვირთის შესაბამის კლასს.

ტვირთის განმსაზღვრელი ატრიბუტების და შესაბამისი კლასების საფუძველზე ალგორითმი C4.5 აგებს გადაწყვეტილებათა ხეს, რომლის დახმარებით წინასწარ ატრიბუტების საფუძველზე მოხდება ტვირთის შესაბამისი კლასისი განსაზღვრა.

ტვირთის კლასიფიკაციის განსაზღვრა გადაწყვეტილებათა ხის მეთოდის გამოყენებით ხდება სპეციალური ბლოკ-სქემით. მაგალითად:

- თავსებადობა;
- მდგომარეობა;
- იზოლირება;
- სპეციფიური სუნი;
- განსაზღვრულ ტემპერატურაზე ტრანსპორტირება;
- სპეციალური კონტეინერი;
- დაზიანების საფრთხე;
- აალებადი;
- მაკოროზირებელი;
- და ა. შ.

ბლოკ-სქემა ყოველი წერტილიდან აგზავნის შეკითხვას ამა თუ იმ ატრიბუტის შესახებ, მოცემული ატრიბუტების გათვალისწინებით განისაზღვრება თუ რომელ კლასს ეკუთვნის მოცემული პროდუქტი. 3.4. ნახაზზე მოცემულია გადაწყვეტილებათა მიღების შესაბამისი ხე.



ნახ.3.4 გადაწყვეტილებათა მიღების ხე

- ალგორიათმი C4.5 იყენებს შემავალ ინფორმაციულ ნაკადს, რის საფუძველზევ იგება გადაწყვეტილებათა მიღების ხე;
- C4.5 ალგორითმს აქვს ერთარხიანი დატოტვილი გადაწყვეტილებათა მიღების ხის ფორმა, სადაც დანაწევრებული ტოტები აუმჯობესებს მოდელის მუშაობას;
- C4.5 ალგორითმს აქვს შესაძლებლობა იმუშაოს როგორც დისკრეტულ, ასევე უწყვეტ მნიშვნელობებთან. ალგორითმი ქმნის შეზღუდვის დიაპაზონს, სადაც ადგენს მონაცემთა საზღვრებს და უწყვეტი მონაცემები გადაყავს დისკრეტულში;
- გამოტოვებული მონაცემები, რომელიც არ კლასიფიცირდება, გამოიმუშავებს თავის საკუთარ შესაძლებლობებს, რათა დეტალიზების საშუალებით დაუახლოვდნენ რომელიმე კლასტერს.

3.4.2. კლასტერული ანალიზის მეთოდი k-საშუალო

მეთოდი k-საშუალო ქმნის ობიექტთა ჯგუფს, ისე რომ ჯგუფის მახასიათებელი წევრები არის ერთგვაროვანი. ეს არის კლასტერული ანალიზის ცნობილი მეთოდი, რომელიც გამოიყენება მონაცემთა ჯგუფების კვლევისას.

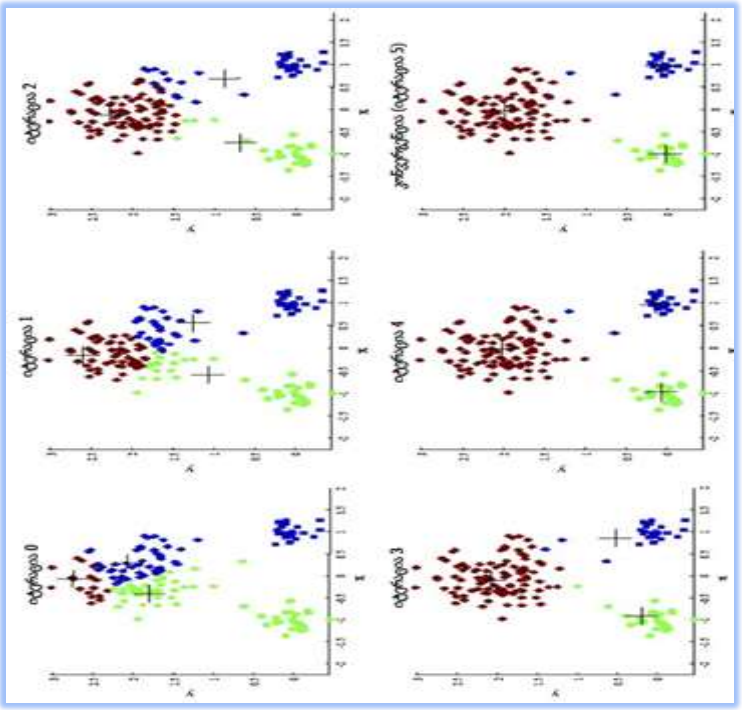
კლასტერული ანალიზი – ალგორითმების ერთობლიობაა, რომელიც ახდენს ჯგუფების ფორმირებას, ისე რომ ჯგუფის წევრები მეტნაკლებად ერთმანეთის მსგავსია და მკვეთრად განსხვავდება ჯგუფის არაწევრი ელემენტებისგან. კლასტერი და ჯგუფი კლასტერულ ანალიზში არის სინონიმები [56].

კლასტერული ანალიზის ჩატარებისას მაგალითად, როდესაც განიხილება ტვირთის შესაბამისი მახასიათებელი ნიშან-თვისებები, კლასტერულ ანალიზში უწოდებენ დაკვირვებას. წინასწარ ყოველთვის ცნობილია გარკვეული სახის ინფორმაცია ტვირთის შესახებ, მაგალითად, ტიპი, სახეობა, მოცულობა, ზომა, წონა და ა.შ. მას პირობითად უწოდებენ ვექტორს, რომელიც აღწერს და წარმოადგენს ტვირთს. ვექტორი შესაძლებელია წარმოვიდგინოთ

როგორც ჩამონათვალში მოცემული რიცხვები, რომლებიც ინტერპრეტირდება კოორდინატებად მრავალგანზომილებიან სივრცეში.

მაგალითად, ტიპი ერთ განზომილებაში, მოცულობა მეორეში და ა.შ., მაგრამ იმისთვის რომ ერთმანეთთან დავაჯგუფოთ კონკრეტული ტვირთის შესაბამისი ტიპი, ზომა და წონა მოცემული ვექტორის დახმარებით, გამოიყენება მეთოდი k - საშუალო.

მეთოდი k-საშუალოს გამოყენებით, განსხვავებული ტიპის მონაცემების იტერაციის შედეგად მიიღება განსხვავებულ ვარიანტთა სიმრავლე, რაც შესაძლებელია წარმოდგენილი იქნას შემდეგ პუნქტებად, რომელთა ვიზუალიზაცია მოცემულია 3.5 ნახაზზე.



ნახ.3.5. ახალი კლასტერის ჩამოყალიბების პროცესი

მეთოდი k-საშუალო, მრავალგანზომილებიან სივრციდან ირჩევს წერტილებს, რომელიც შემდგომში იტერაციის შედეგად წარმოდგენილი იქნება, როგორც ახალი k-კლასტერი. ამ წერტილებს ეწოდებათ დატვირთვის ცენტრები.

1) ყოველი ტვირთის შესაბამისი ატრიბუტები მიახლოებული უნდა იყოს მოცემული სივრცის დატვირთვის წერტილებთან, რის შედეგადაც მათ გარშემო ჩამოყალიბდება ახალი კლასტერი;

2) მიღებულ k-კლასტერში ყოველთვის აღმოჩნდება ახალი ტვირთის სახეობა, დამახასიათებელი ატრიბუტების შესაბამისად;

3) მეთოდი k-საშუალოს გათვალისწინებით კლასტერების წევრების ცენტრი მდებარეობს k-კლასტერში;

4) მიღებული ცენტრი არის კლასტერის დატვირთვის ცენტრი;

5) შესაძლებელია დატვირთვის ცენტრის შეცვლა, ტვირთი შესაბამისი მახასიათებლების გათვალისწინებით ანალიზის პროცესში შესაძლებელია აღმოჩნდეს დატვირთვის სხვა ცენტრში, ანუ შეუძლია შეცვალოს ადგილი და გადაინაცვლოს სხვა კლასტერში;

6) ბიჯი 2-6 მეორდება მანამ, სანამ დატვირთვის ცენტრი არ შეწყვეტს ცვლილებას და მიიღებს სტაბილურ სახეს, მას უწოდებენ კონვერგენციას (დაახლოებას).

სქემატურად დატვირთვის ცენტრის განსაზღვრა და ახალი k-კლასტერის ჩამოყალიბების პროცესი მოცემულია 3.5 ნახაზზე. აღნიშნული ალგორითმის გამოყენების უპირატესობა მის სწრაფქმედებაში, ეფექტურობასა და სიმარტივეშია. მეთოდი k-საშუალო გამოიყენება დიდ მონაცემთა ჯგუფის დანაწევრებისთვისაც, რის შემდეგაც ტარდება შედარებით დიდი მოცულობის კლასტერების ანალიზი და ქვეკლასტერებად წარმოდგენა. მეთოდი k-საშუალო ასევე გამოიყენება კლასტერების რაოდენობის შესამცირებლად.

ასეთი ალგორითმის სუსტი მხარეა ის, რომ იგი არ მოიხმარება დისკრეტულ მონაცემებში. გამოიყენების სფეროებია: Apache Mahout; Julia; R; SciPy; Weka; MATLAB; SAS და სხვ, [74-78].

3.4.3. აპრიორების ალგორითმი

აპრიორების ალგორითმი ეძებს ასოციაციურ წესებს და კანონზომიერებებს მონაცემთა ბაზაში, სადაც არის დიდი რაოდენობის ტრანზაქციები. ასოციაციურ წესებში მოიაზრება გარკვეული სახის ტექნიკური საშუალება, რომელიც გამოიყენება მონაცემთა მაინინგში, რათა მონაცემთა ბაზაში ცვლადებს შორის განისაზღვროს დამოკიდებულება [79].

აპრიორულ ალგორითმზე განვიხილოთ მარტივი მაგალითი, რომელიც შეეხება სუპერმარკეტის მონაცემთა ბაზის ტრანზაქციებს. მონაცემთა ბაზა შესაძლებელია განვიხილოთ, როგორც დიდი მოცულობის ცხრილი, სადაც ყოველი სტრიქონი არის ტრანზაქციების ნომერი, ხოლო სვეტები შეიცავს ცალკეული გაყიდვების შესახებ ინფორმაციას (ნახ.3.6).

Transaction ID	არაჟანი	რძე	კარაქი	ყველი	ვაშლი
1	X	X	X		
2	X	X			X
3	X		X	X	

ნახ.3.6. ცხრილი

აპრიორების ალგორითმის გამოყენებით შესაძლებელია განისაზღვროს ის პროდუქტები, რომელსაც მომხმარებელი ყიდულობს ერთად, ასე, რომ განისაზღვრება მათ შორის ასოციაციური დამოკიდებულება, რაც მოგვცემს საშუალებას განვსაზღვროთ ის პროდუქტები, რომელიც ხშირ შემთხვევაში იყიდება ერთად.

ძირითადი მარკეტინგული ამოცანაა – რაც შეიძლება მეტი პროდუქტი შეიძინოს მომხმარებელმა, აღნიშნული მეთოდის გამოყენება მარკეტინგის მენეჯერს უადვილებს მომხმარებლისთვის შეარჩიოს ის პროდუქტი ან პროდუქტები რაზეც იოლად მიიღებს შეძენის გადაწყვეტილებას. ერთმანეთზე დამოკიდებულ პროდუქტებს ეწოდებათ ჯგუფი, ერთი ჯგუფის პროდუქტები,

მაგალითად. არაჟანი, რძე კარაქი, ხშირად შეგვინიშნავს, რომ მარკეტში თაროებზე მოთავსებულია ერთმანეთის გვერდიგვერდ. ორ რომელიმე სახის პროდუქტს, რომელზეც მომხმარებელი უმეტეს შემთხვევაში აკეთებს არჩევანს, უწოდებენ ორელემენტიან ჯგუფს. როდესაც მონაცემთა ბაზა საკმაოდ დიდია, უფრო რთულია პროდუქტებს შორის დამოკიდებულების დანახვა, განსაკუთრებით მაშინ, როდესაც ხდება სამ-, ოთხ- ან უფრო მეტ ელემენტიანი რთული ჯგუფის განსაზღვრა, სწორედ ასეთ შემთხვევებში გამოიყენება აპრიორების ალგორითმი.

სანამ უშუალოდ ალგორითმს განვსაზღვრავთ, უნდა ჩამოვაყალიბოთ სამი ძირითადი პარამეტრი:

1. პირველ ეტაპზე უნდა განისაზღვროს ჯგუფის *ზომა*, ანუ, რამდენელემენტიანი ჯგუფი გვჭირდება;

2. მეორე ეტაპზე განისაზღვრება *მხარდაჭერა* – ტრანზაქციების საერთო რაოდენობიდან იმ ტრანზაქციების გამოყოფა და რაოდენობის განსაზღვრა, რომელიც შედის ჯგუფში. ჯგუფი, რომელიც უზრუნველყოფილია მხარდაჭერით, ის ითვლება ყველაზე ხშირად განმეორებად ჯგუფად;

3. მესამე ეტაპზე განისაზღვრება *საიმედოობა,* ალბათობა იმისა, რომ კონკრეტული პროდუქტი მოხვდება კალათაში სხვა პროდუქტთან ერთად. მაგალითად, არაჟანს აქვს 65% ალბათობა იმისა, რომ მოხვდება რძესთან ერთად კალათაში და ა.შ.

აპრიორების ალგორითმი შედგება სამი ძირითადი ბიჯისგან:

1. *გაერთიანება.* მონაცემთა ბაზაში უნდა განისაზღვროს ცალკეული პროდუქტის განმეორების სიხშირე;

2. *განცალკევება.* ის ჯგუფები, რომელთაც აქვს მხარდაჭერა და საიმედოობის საკმარისი მაჩვენებელი, გადადის მომდევნო იტერაციაზე ორ-კომპონენტიან ჯგუფში;

3. *განმეორება.* პირველი ორი ბიჯი მეორდება ჯგუფში შემავალი ყოველი სიდიდისთვის მანამდე, სანამ არ მიიღება ადრე განსაზღვრული ზომა.

3.4.4. მოლოდინის მაქსიმიზაციის კლასტერული ალგორითმი (EM)

Data Mining-ში გამოიყენება *მოლოდინის მაქსიმიზაციის კლასტერული* ალგორითმი (expectation-maximization – EM), რომლის დანიშნულებაცაა „ცოდნის

მათემატიკურ სტატისტიკაში EM-ამოპოვება“ [80,81]. ცოდნის მოპოვება (knowledge mining) – ესაა ინტელექტუალური ანალიზი – ახალი დისციპლინა ხელოვნურ ინტელექტში, რომელიც იყენებს ინტელექტუალური სერვისების კომბინაციას, რათა სწრაფად შეისწავლოს დიდი მოცულობის ინფორმაცია. ლგორითმი ითვლება იტერაციულად და გამოიყენება სტატისტიკური მოდელის მაქსიმალურად მსგავსი პარამეტრების გამოთვლისას, როდესაც ცვლადები დაფარულია. სტატისტიკური ისეთი მოდელებია, როდესაც აღწერილია უკვე ცნობილი, დადასტურებული მონაცემები.

მაგალითად, გამოცდაზე მიღებული შეფასება შესაძლებელია წარმოვადგინოთ როგორც ნორმალური განაწილება, ამიტომ მოსალოდნელია, რომ შეფასებები დაგენერირდეს შესაბამისად ნორმალურ განაწილების მოდელში.

განაწილება არის ყველა ის ალბათური შედეგი, რომელიც შეიძლება მიღებული იქნას გამოცდის შედეგად. მაგალითად, გამოცდაზე მიღებული შეფასება შეესაბამება ნორმალურ განაწილებას, სადაც არის ალბათობა იმისა, რომ წარმოდგენილი იქნება გამოცდაზე მიღებული ყველა სავარაუდო შედეგი. ე.ი. განაწილება გვეხმარება, რომ გავიგოთ გამოცდაზე გასულმა რამდენმა ადამიანმა მიიღო ესა თუ ის შეფასება.

მოდელის მნიშვნელოვანი მახასიათებელია პარამეტრი, რომელიც აღწერს მის ძირითად ნაწილს, განაწილებას. მაგალითად,

საშუალო განაწილება აღიწერება საშუალო არითმეტიკულით და დისპერსიით. ზოგადად, ნორმალური განაწილებისთვის აუცილებელია ორი პარამეტრის განსაზღვრა: საშუალო არითმეტიკულის და დისპერსიის.

იმ შემთხვევაში, როდესაც არ ვიცით ყველა შეფასების საშუალო არითმეტიკული ან დისპერია, შესაძლებელია შეფასდეს მხოლოდ ერთი მაგალითის მონაცემები.

EM-ალგორითმის ერთ-ერთი მნიშვნელოვანი მახასიათებელია, ხდომილება possibility – შესაძლებლობა, event – მოვლენა). იგი არის ალბათობა იმისა თუ რამდენად მოხდა გადახრა ნორმალური განაწილებიდან. ანუ ალბათობა იმისა, ნორმალური განაწილების მრუდი რამდენად სწორად აღწერს გამოცდაზე მიღებული შედეგების საშუალო არითმეტიკულს და დისპერსიას.

ალბათობას (probability) და ხდომილებას (possibility) თითქმის მსგავსი მნიშვნელობა აქვს, მაგრამ მათ შორის მცირე განსხვავებაცაა. ალბათობა ნიშნავს, რომ რაღაც შეიძლება მოხდეს, მაგრამ უფრო სავარაუდოა, რომ იგი მოხდეს. *ხდომილება* (possibility) კი ნიშნავს, რომ რაღაც შეიძლება მოხდეს, მაგრამ ჩვენ არ ვიცით რამდენად ეს სავარაუდოა (რამდენად ალბათურია). იგი აღწერს გაურკვევლობას (uncertainty), შეიძლება მოხდეს თუ არა მოვლენა (event).

მონაცემთა მაინინგში და კლასტერიზაციაში მნიშვნელოვანია შეფასდეს კლასები მასში გაერთიანებული მონაცემების შესაბამისად, როგორც გამოტოვებული მონაცემები. ვინაიდან თავიდან უცნობია კლასის სახეობა, ამიტომ გამოტოვებული მონაცემების იტერაცია საკმაოდ მნიშვნელოვანი პროცესია კლასტერიზაციის ამოცანაში EM-ალგორითმის გამოყენებისას.

EM (Expectation-Maximization) იტერაციული ალგორითმია, რომელიც გამოიყენება მაქსიმალურად თანხვედრი პარამეტრების მქონე სავარაუდო მოდელის შესაფასებლად. როდესაც მოდელში

არის რამდენიმე არაცნობადი პარამეტრი, ფასდება მათი მაქსიმალური თანხვედრა სხვა მოდელთან, რის შედეგადაც EM – ალგორითმი ქმნის ახალ მოდელს. ახალი მონაცემთა ერთობლიობა აისახება როგორც კლასი. კლასტერიზაციის პროცესში კი სრულდება სამ ბიჯიანი იტერაციული პროცესი:

1) **E** – ბიჯი: აღნიშნულ ბიჯზე მოდელის ძირითადი პარამეტრების საფუძველზე გამოითვლება ალბათობა იმისა, ეკუთვნის თუ არა ეს მონაცემები მოცემულ კლასტერს;

2) **M** – ბიჯი: ხდება მოდელის პარამეტრების განახლება შესაბამის კლასტერულ განაწილებაში, რომელიც ჩატარდა E ბიჯზე;

3) პირველი ორი ბიჯი მეორდება მანამ, სანამ მოდელის პარამეტრები და კლასტერული განაწილება არ გათანაბრდება.

EM-ალგორითმის მთავარი ფასეულობაა გამოყენების სიმარტივე, ასევე მას შეუძლია მოახდინოს არა მარტო მოდელის პარამეტრების ოპტიმიზაცია, არამედ განსაზღვროს რამდენად ღირებულია გამოტოვებული მონაცემები მოდელისათვის. EM – შესაძლებელია ჩავთვალოთ როგორც საუკეთესო მეთოდი კლასტერიზაციისა და პარამეტრებზე დამოკიდებული მოდელის შექმნისათვის.

3.4.5. რანჟირებული ბმის ალგორითმი (PageRank)

PageRank – რანჟირებული ბმის ალგორითმი განსაზღვრავს ობიექტის მნიშვნელობას და კავშირს ქსელში არსებულ სხვა ობიექტებთან (მაგალითად, ვებ-საიტის გვერდების მნიშვნელობა – რეიტინგი). აღნიშნულ ალგორითმს ეფექტურად იყენებს საძიებო სისტემები: Google, Wikipeida და ა.შ. [82,83].

რანჟირებული ბმა – ქსელური ანალიზის ტიპია, რომელიც ადგენს ობიექტებს შორის (წაიკითხე, დააკავშირე) ასოციაციურ კავშირს. ინტერნეტის თითქმის ყველა ვებ-გვერდი კავშირშია

ერთმანეთთან. მაგალითად, თუ გავააქტიურებთ ვებ-გვერდს gtu.ge მივიღებთ ბმას https://eqe.ge/ -ზე. აქ განთავსებულია „განათლების ხარისხის განვითარების ეროვნული ცენტრი“, ეს ნიშნავს რომ გვერდი gtu.ge , რელევანტურად თვლის გვერდს eqe.ge -ს, ამავე დროს აღსანისნავია ის გარემოება, რომ ნებისმიერი ვებ გვერდი სადაც ხდება მიმართვა gtu.ge-დან, ამაღლებს gtu.ge-ს რელევან-ტურობას. მეთოდი PageRank ვებ. გვერდებს ანიჭებს 0-დან 10-მდე პრიორიტეტს. 3.1 ცხრილში მოცემულია კომპანია Google- ს მიერ გამოქვეყნებული რეიტინგი.

ინფორმაციული ცხრილი ცხრ.3.1

Website	PageRank
twitter.com	10
facebook.com	9
reddit.com	8
stackoverflow.com	7
tumblr.com	6
crucial.com	5
programmingzen.com	4
dearblogger.org	3

ცხრილში მაქსიმალური მნიშვნელობა (10 ქულა) ენიჭება ვებ. გვერდს რომელიც ყველაზე პოპულარული და რელევანტურია. ალგორითმი PageRank სპეციალურად შექმნილია გლობალური ქსელისთვის, რომელსაც ადამიანების დამოკიდებულება გადაყავს ციფრებში.

PageRank ალგორითმი არის სუპერეფექტური საშუალება, რომელიც ახდენს ბმების რანჟირებას. გასათვალისწინებელია ის

გარემოება, რომ დასაკავშირებელი ობიექტები არ არის აუცილებელი იყოს ვებ. გვერდები [84]. PageRank ალგორითმის ინოვაციური გამოყენების სფეროებია:

1) ეკოლოგია, სადაც აღნიშნული ალგორითმის გამოყენებით განისაზღვრება ეკოსისტემების სასიცოცხლო ციკილი;

2) ტვიტერმა PageRank ალგორითმის გამოყენებით შეიმუშავა WTF (Who-to-Follow) – პერსონალიზირებული სარეკომენდაციო ვარიანტები, სადაც ჩამონათვალში წარმოდგენილია იმ ადამიანთა სია, რომელთაც სჭირდებათ სხვადასხვა სახის შეთავაზებები და რეკომენდაციები.

3) PageRank ალგორითმი აქტიურად გამოყენება ჰონკონგის პოლიტექნიკური უნივერსიტეტის პროფესორის ბინ ჟენის (Bin Jiang) მიერ ტოპოლოგიურ ჩანაწერებში, სადაც წინასწარ ხდება ადამიანების აქტიური მოძრაობის განსაზღვრა.

PageRank ალგორითმის მთავარი ღირებულება საიმედოობაა, მიუხედავად იმ სირთულისა, რომელიც უკავშირდება რელევანტური ბმის პროცესს.

გრაფიკული ან სქემური მონაცემების შესაბამისი პარამეტრების, პრიორიტეტების, რელევანტურობის, განსაზღვრისათვის ასევე ყველაზე ეფექტური საშუალებაა PageRank-ის გამოყენება.

საფირმო ნიშანი PageRank ეკუთვნის კომპანია Google-ს. ალგორითმი PageRank შეიქმნა და დაპატენტდა სტენფორდის უნივერსიტეტში.

PageRank ალგორითმი ასევე რეალიზებულია შემდეგ პროგრამულ პაკეტებში:

1. C++ OpenSource PageRank;
2. Python PageRank;
3. ქსელური ანალიზის პაკეტი - igraph.

ზოგადად, შესაძლებელია ითქვას, რომ მონაცემთა მაინინგი არის *მულტიდისციპლინური დარგი*, სადაც მონაცემთა ბაზების

შეფასება ხდება გამოყენებითი სტატისტიკის მეშვეობით, ხოლო ამოცნობის თვალსაზრისით გამოიყენება ხელოვნური იტელექტის მეთოდები, მონაცემთა ბაზების თეორია და ა.შ.

3.7 ნახაზზე მოცემულია ის სისტემები და ტექნოლოგიები, რომლებიც უშუალო კავშირშია მონაცემთა მაინინგთან.

ნახ. 3.7. Data Mining - მულტიდისციპლინური დარგი

წარმოდგენილი სისტემებიდან ბევრი ინტეგრირდება სხვა მეთოდებთან და ტექნოლოგიებთან, მაგრამ თითოეული მათგანი შეიცავს ე.წ. საკვანძო კომპონენტს, რაც მნიშვნელოვანს და ეფექტურს ხდის მათში მონაცემთა მაინინგის გამოყენებას.

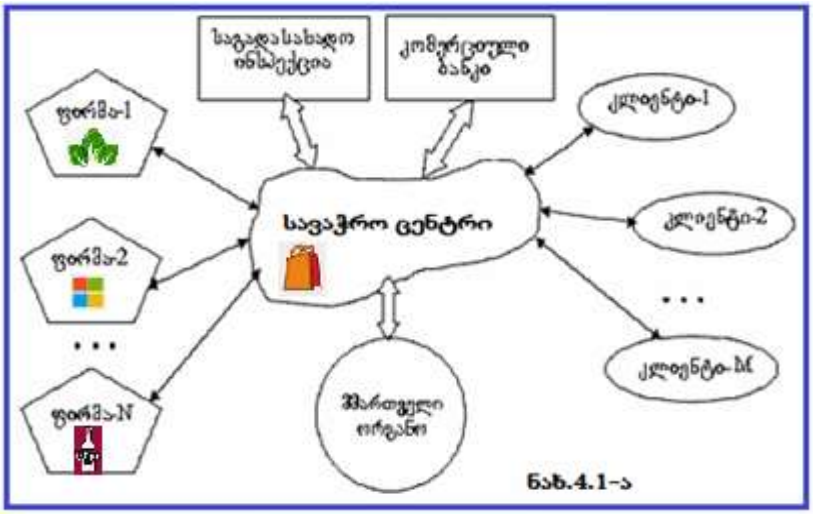
IV თავი

განაწილებული ბიზნეს სისტემების მონაცემთა საცავის პროგრამული უზრუნველყოფა

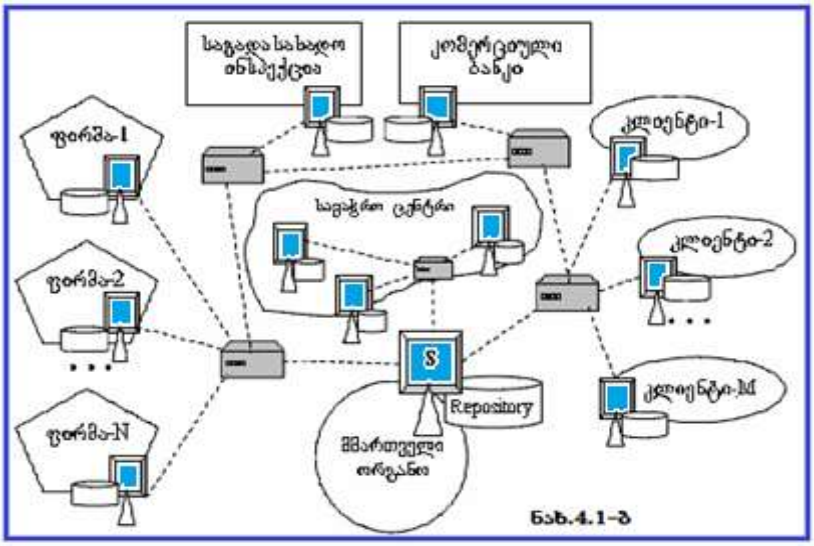
1.1. სისტემის არქიტექტურა და ინფორმაციული ნაკადები

ელექტრონული ბიზნესისა და კომერციის განაწილებული სისტემების დაპროექტების ტექნიკური რეალიზაციის მხარე მოითხოვს მისი ცალკეული კვანძების ფუნქციური ანალიზის საფუძველზე აპარატურულ-პროგრამულ უზრუნველყოფათა ამოცანების გადაწყვეტას. აპარატურულში იგულისხმება კომპიუტერული და ქსელური ტექნიკა, რომლის საფუძველზედაც უნდა მოხდეს სისტემის გლობალურ/ლოკალურ ქსელში ფიზიკურად ჩართვის ორგანიზება. პროგრამული კი – ქსელში ჩართული ოპერაციული სისტემის, პლატფორმის, საერთო-სერვისული გარემოს და კერძო-ფუნქციური პაკეტების ერთობლიობაა.

4.1-ა,ბ ნახაზებზე მოცემულია ბიზნესის *ტრადიციული* და *ელექტრონული კომერციის* ზოგადი სტრუქტურული სქემები.



ნახ.4.1-ა

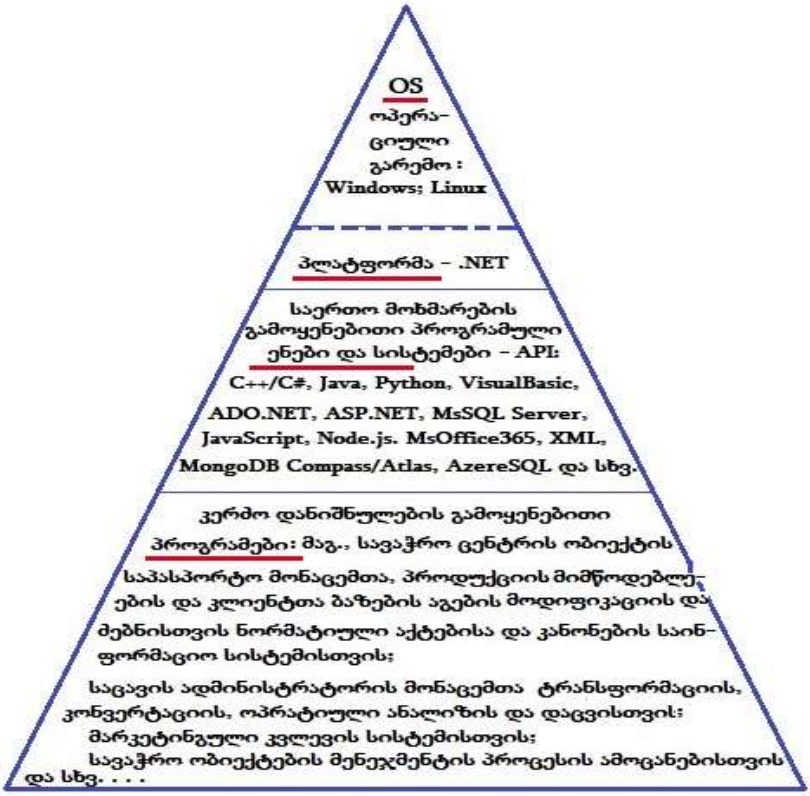


ნახ.4.1–ბ

კომპიუტერული ტექნიკისა და საინფორმაციო ტექნოლოგიების საფუძველზე რეალიზებულია თვისობრივად ახალი ურთიერთობები სავაჭრო ცენტრებს (მაღაზიათა ქსელი), პროდუქციის მიმწოდებელ ფირმებს, მაკონტროლებელ ორგანოებს, კომერციულ ბანკებსა და კლიენტებს შორის.

განაწილებული კომერციული სისტემის მონაცემთა საცავის არსებობა ხელს უწყობს აღნიშნულ ობიექტებს შორის ინფორმაციის დროულად გაცვლისა და გამჭვირვალე პროცესების მართვის უზრუნველყოფას. ახალი სისტემის პირობებში საჭირო ხდება საცავების შესაბამისი აპარატურული და პროგრამული დაცვის მექანიზმების გამოყენება.

4.2 ნახაზზე მოცემულია (ჩვენი კვლევის სფეროს შესაბამისი) სისტემის ფუნქციონირებისათვის აუცილებელი პროგრამული პლატფორმისა და პაკეტების მრავალდონიანი არქიტექტურა. მისი შედგენილობა დროის ფუქნქციაა და განისაზღვრება აპარატულ-პროგრამული სისტემების განვითარების მდგომარეობით.



ნახ. 4.2.

ბიზნეს-სისტემის მუშაობის პირობები უნდა ითვალისწინებდეს შემდეგ ძირითად მოთხოვნებს:

- სისტემა ფუნქციონირებს პერსონალური კომპიუტერების ლოკალურ ქსელში ინტერნეტ-ინტრანეტის გამოყენებით;

- სისტემამ უნდა უზრუნველყოს ინფორმაციის უსაფრთხოება და დაცვა;

- სისტემას უნდა ჰქონდეს მომხმარებლებთან ურთიერთობის მეგობრული ინტერფეისი;

- სისტემის მომსახურება უნდა იყოს მოქნილი და მარტივი.

დიდ სავაჭრო ცენტრებში (მაღაზიათა ქსელი) ელექტრონული კომერციის განსახორციელებლად საჭიროა შიგა და გარე ინფორმაციული ნაკადების ანალიზის ჩატარება. არსებობს შემდეგი სახის ინფორმაციული ნაკადები:

- კორესპოდენცია და წერილები;
- ნორმატიული აქტები და კანონები;
- კონტრაქტები (ხანგრძლივი) და შეკვეთები (ერთჯერადი);
- ფაქტურები;
- სავაჭრო ობიექტების საქონელბრუნვის გეგმები, ფაქტობ- რივი შესრულებები, ანალიზის მასალები;
- ინტერნეტიდან მიღებული ინფორმაცია (ფურცლები);
- აუდიო და ვიდეო ინფორმაცია (ელექტრონული გამოფე- ნები, სალონები, პროდუქციის კატალოგები);
- საბანკო ანგარიშები, ბუღალტრული აღრიცხვა;
- კადრების აღრიცხვისა და შრომითი დასაქმების დოკუმენტაცია;
- ინფორმაცია პარტნიორებისა და კონკურენტების შესახებ;
- ინფორმაცია პროდუქციის ადგილობრივი და საერთაშო- რისო ბაზრების შესახებ (კონიუნქტურა, ფასები);
- სტატისტიკური ანალიზის მასალები და სხვ.

ინფორმაციული ნაკადების მოცულობათა საანგარიშოდ ინფორმაციის ერთეულად მივიღოთ:

I_T : ერთი ნაბეჭდი A4 ფორმატის გვერდის ტექსტური ინფორმაციის სიდიდე; I_A : ერთი აუდიო ინფორმაციის სიდიდე; I_V : ერთი ვიდეო ინფორმაციის სიდიდე.

პირობითად მივიღოთ, რომ I_T = 4 Კბ, I_A = 20 Კბ, I_V = 30 Კბ. ინფორმაციული ნაკადების ზომები დამოკიდებული იქნება კომერციული ობიექტების მასშტაბებზე (ზომები, კონიუნქტურა, წლიური ფონდბრუნვა და საქონელბრუნვა, ფილიალების რაოდენობა, სეზონი, რეგიონი და ა.შ.). ინფორმაციული ნაკადების მოცულო-

ბების საანგარიშოდ შეიძლება ჩავატაროთ მიახლოებითი, გასაშუალებული გათვლები (თვის, კვარტლის, წლის და ხანგრძლივი პერიოდისთვის), რომელთა საფუძველზე შესაძლებელი იქნება საერთო ინფორმაციული ფონდის მოცულობის შეფასება და მონაცემთა განაწილებული საცავის ფიზიკური მოწყობილობების საჭირო მეხსიერების დადგენა.

4.1 ცხრილში განიხილება ერთი პირობითი (ვირტუალური) კომერციული ობიექტის ინფორმაციული ნაკადების მოცულობები (საშუალოდ). სავაჭრო ცენტრებისათვის ის გამრავლდება, შესაბამისად ფუნქციური ფილიალების რაოდენობაზე და გამოაკლდება საერთო გამოყენების ინფორმაციის რაოდენობა.

ინფორმაციული ნაკადები **ცხრ.4.1**

№	ინფორმაციული ნაკადი	სახე	მოცულობა (საშუალოდ)			
			თვეში Mb	კვარტ. Mb	წელში Mb	ხანგრძლივი პერიოდი, წლებით
1.	კორესპონდენცია და წერილები	T	8	24	288	5
2.	ნორმატიული აქტები და კანონები	T			200	3
3.	კონტრაქტები და შეკვეთები	T	16	48	576	10
4.	ფაქტურები	T	20	60	720	12
5.	საკონსულტაციო გეგმები, ფაქტობრივი შესრულებები, ანალიზის მასალები	T			400	6
6.	ინტერნეტიდან მიღებული ინფორმაცია (ფურცლები)	T	20	60	720	12
7.	აუდიო და ვიდეო ინფორმაცია (ელექტრონული გამოფენები, სალონები, პროდუქციის კატალოგები)	A, V	50	150	1800	30
8.	საბანკო ანგარიშები, ბუღალტრული აღრიცხვა	T	20	60	720	12
9.	გაფრების აღრიცხვისა და შრომითი დასაშვების დოკუმენტაცია	T			10	2
10	ინფორმაცია პარტნიორებისა და კონკურენტების შესახებ	T	5	15	180	3
11.	ინფორმაცია პროდუქციის ალვილობრივი და საერთაშორისო ბაზრების შესახებ (კონიუნქტურა, ფასები)	T	15	45	540	10

<u>შენიშვნა:</u> ცხრილში ინფორმაცია აღებულია ექსპერტული შეფასებების საფუძველზე.

საანგარიშო ფორმულებად ვიყენებთ შემდეგ გამოსახულებებს:

$$V_{jT} = k_j * \sum_{k=1}^{n} R_i * I_T, \text{ სადაც}$$

V_{jT} არის ტექსტური სახის ინფორმაციის თვიური, კვარტალური და წლიური დოკუმენტების ჯამური მოცულობა მეგაბაიტებში;

k_j – თვიური, კვარტალური და წლიური კოეფიციენტი (1, 3, 12 - თვე);

R_i – ტექსტ-დოკუმენტის A4-გვერდების რაოდენობა;

აუდიო ინფორმაციული ნაკადებისათვის შესაბამისად გვექნება:

$$V_{jA} = k_j * \sum_{k=1}^{n} A_i, \text{ სადაც}$$

A_i – აუდიო ინფორმაციის ფაილის ზომაა;

საჭიროა გავითვალისწინოთ ხმის გადაცემის მახასიათებელი, რომელიც საშუალოდ წარმოშობს 64 Kbit/sec წარმადობის ინფორმაციულ ნაკადებს.

ვიზუალურისათვის შესაბამისად გვექნება:

$$V_{jV} = k_j * \sum_{k=1}^{n} V_i, \text{ სადაც}$$

V_i – ვიდეო ინფორმაციის ფაილის ზომაა;

საჭიროა გავითვალისწინოთ, რომ ვიდეო გამოსახულების გადაცემა არქივირების გარეშე წარმოშობს 9.216 Mbit/sec, ხოლო არქივირებით 1.5 Mbit/sec წარმადობის ინფორმაციულ ნაკადებს.

მთლიანად ინფორმაციული ნაკადების ჯამური მოცულობა იქნება:

$$S = T_i * K_i * \sum_{j=1}^{n} V_j^i, \text{ სადაც}$$

T_i – i-ური კომერციული ობიექტის არსებობის მთლიანი პერიოდია (წლები);

K_i – i-ურ კომერციულ ობიექტზე ფილიალების რაოდენობაა;

V_j^i – i-ური კომერციული ობიექტის j-ური სახის ინოფორმაციული ნაკადის მოცულობა.

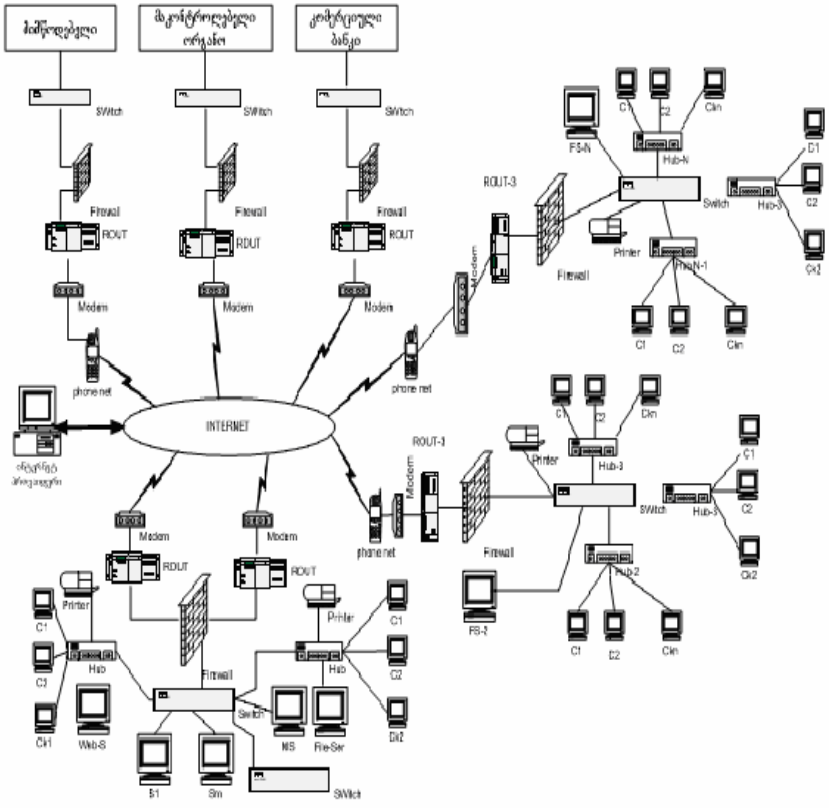
ექსპერტული ინფორმაციის საფუძველზე, როგორც ჩვენი პირობითი გათვლებიდან გამომდინარეობს, ერთ (მცირე ბიზნესის) კომერციულ ობიექტზე დაახლოებით 10 წლიანი არსებობის პერიოდში მონაცემთა საცავისათვის საშუალოდ დაგვჭირდება 200GB–ანი მეხსიერება (გაითვალისწინება ძირითადი სტატისტიკური და ისტორიული ფაილების შენახვაც).

დიდი სავაჭრო ცენტრისათვის (მაღაზიების ქსელი) ეს რიცხვი უნდა გამრავლდეს ობიექტების რაოდენობაზე და დაემატოს მათი მენეჯმენტისათვის საჭირო ინფორმაციული ნაკადი. ამგვარად, მონაცემთა საცავისათვის საჭირო ფიზიკური მეხსიერება მიაღწევს რამდენიმე ტერაბაიტ-მოცულობას (!).

4.2. ქსელური არქიტექტურა: მონაცემთა საცავის დაცვისა და უსაფრთხო გამოყენების მექანიზმები

განვიხილოთ დიდი კომერციული ობიექტის (მაღაზიათა ქსელი) ზოგადი ინტერნეტ/ინტრანეტ ქსელური სქემა (ნახ.4.3). კონკრეტულად, ჩვენ აქ შევეხებით სისტემასთან უსაფრთხო მუშაობის და ინფორმაციის დაცვის ორგანიზაციულ და ტექნიკურ საკითხებს.

კომერციული ობიექტების ქსელური კომპიუტერული სისტემა ორიენტირებულია მრავალ-მომხმარებლურ რეჟიმში სამუშაოდ, რაც აუცილებლად მოითხოვს სისტემის ფუნქციონირების უსაფრთხოების გარანტიებს Ms Windows, SQL Server, AzureSQL, MongoDB Atlas და სხვა ქსელურ სისტემებს. მათ აქვს სპეციალური პაროლური სისტემები მომხმარებლების დასარეგისტრირებლად (Login ID).



ნახ.4.3

ასევე მნიშვნელოვანია სისტემის მომხმარებელთა ლოგიკური როლების განაწილების საკითხი. აქ წყდება მომხმარებელთა მიკუთვნება რომელიმე წინასწარ განსაზღვრულ როლზე, ხოლო როლს გააჩნია მონაცემთა ბაზაში ინფორმაციის წვდომის შეზღუდვები. მაგალითად, მომხმარებელი შეიძლება იყოს სისტემური ადმინისტრატორი, მონაცემთა ბაზის ადმინისტრატორი ან სიტემის საბოლოო მომხმარებელი (ფუნქციური თანამშრომელი ან კლიენტი).

ასეთი კატეგორიის მომხმარებლები სხვადასხვა პრიორიტეტებით ისარგებლებენ. კერძოდ, მათი როლების შესაბამისად, განესაზღვრებათ ბაზიდან მონაცემთა მხოლოდ წაკითხვის და/ან წაკითხვა-ჩაწერის უფლებებიც.

შეიძლება ითქვას, რომ მონაცემთა დაცვის მექანიზმები კარგადაა დამუშავებული განაწილებული რელაციური მონაცემთა ბაზების მართვის თანამედროვე სისტემებში. იმისდა მიხედვით, თუ რომელი პროგრამული პაკეტი იქნება არჩეული (Ms SQL Server, MySQL, MongoDB ან სხვ.) სისტემის სარეალიზაციოდ, მოხდება შესაბამისი უსაფრთხოების სისტემის გამოყენება [63,64].

კომერციული ობიექტის კომპიუტერული სისტემა ქსელური მოხმარების სისტემაა, ამიტომაც მრავალმომხმარებლური რეჟიმიდან გამომდინარე, საჭიროა სისტემის ექსპლუატაციის დროს გარკვეული რეგლამენტის შემუშავება მონაცემთა დაცვის თვალსაზრისით. აქ პირველ რიგში იგულისხმება სისტემის საიმედოობის გაზრდა (როგორც პროგრამული პაკეტების, ასევე მონაცემთა ბაზების ფაილებისათვის). ერთის მხრივ, უსაფრთხო მუშაობის პრინციპი (პაროლური სისტემა, როლები და ა.შ.) გარკვეულად ამცირებს არაავტორიზებულ მიმართვებს მონაცემთა ბაზებთან, მაგრამ მეორეს მხრივ, აუცილებელია არსებობდეს არაკორექტული მონაცემების დამახსოვრების თავიდან აცილების შესაძლებლობაც.

ინფორმაციის არაკორექტულობა შეიძლება გამოწვეული იყოს შემთხვევით, მონაცემების ან პროგრამის შეცდომით ჩაწერისას ბაზაში ან წინასწარი გაწზრახვით. ამგვარად, ინფორმაციის დაცვა ორი ამოცანის გადაწყვეტას ითხოვს: მონაცემთა მთლიანობის უზრუნველყოფას და საიდუმლოების გარანტიას (მონაცემთა მისაღებად შეზღუდვების დაყენებას).

მონაცემთა ბაზის მთლიანობის უზრუნველსაყოფად გამოიყენება სტრუქტურული შეზღუდვები ან უშუალოდ მონაცემთა მნიშვნელობების შეზღუდვები.

შეზღუდვები სტრუქტურულ დონეზე ეფუძნება მონაცემთა ბაზებში ფუნქციონალური დამოკიდებულებების (რელაციების, ატრიბუტების და ა.შ.) აღწერას. შემოიტანება სპეციალური გასაღებური ატრიბუტების, ინდექსების ცნებები (მარტივი ან შედგენილი). მათი საშუალებით ხორციელდება რელაციურ ფაილებში ინფორმაციის მოწესრიგება, ძებნა და ამორჩევა.

მთლიანობის შეზღუდვები მონაცემთა მნიშვნელობების ცვლილებებზე ხორციელდება სპეციალური მათემატიკური დამოკიდებულებებით. თუ მონაცემთა მნიშვნელობა გამოვა განსაზღვრული დიაპაზონიდან, ან არ შეესაბამება არსებულ მათემატიკურ დამოკიდებულებას, მაშინ ხდება სპეციალური დამცველი ფინქციების ამუშავება, რათა არ დაირღვეს ბაზის მთლიანობა. ასეთი ფუნქციის როლს თანამედროვე მონაცემთა ბაზების მართვის სისტემებში ტრიგერები (Triggers) ასრულებს.

ესაა ჩართვა-გამორთვის ფუნქციები, რომლებიც უზრუნველყოფს მონაცემთა მთლიანობას ლოგიკურად დაკავშირებულ ცხრილებში (რელაციებში). სისტემაში სტანდარტული ან კერძო ფუნქციის ამუშავება განისაზღვრება მომხმარებლის მიერ, ტრიგერები კი არაა დამოკიდებული პროგრამაზე. იგი ჩაირთვება ყოველთვის, როდესაც ადგილი აქვს მონაცემთა ბაზაში ინფორმაციის განახლებას: ახლის ჩამატებას, ძველის წაშლას ან შეცვლას. ამგვარად, ტრიგერების ერთ-ერთი მთავარი ფუნქციაა სისტემაში მონაცემთა ცვლილების სტატისტიკის წარმოება.

tempdb.mdf და tempdblog.ldf – ბაზაში ინახება დროებითი ცხრილები. იგი SQL Server-ის გლობალური რესურსია. მომხმარებლის მიერთებისას SQL Server-თან ყოველთვის იხსნება ეს ბაზა, მუშაობის დამთავრებისას კი იგი ავტომატურად წაიშლება.

მონაცემთა საიმედოობის უზრუნველსაყოფად კოლექტიური მოხმარების კომპიუტერულ სისტემებში, სადაც

მაღალია მონაცემთა დაკარგვის ან დამახინჯების ალბათობა (რისკი), გამოიყენებენ, როგორც ზემოთ აღვნიშნეთ, მონაცემთა ბაზების პერიოდულ დუბლირებას.

მაგალითად, რეგლამენტით დადგინდება, რომ ყოველი დღის ბოლოს (ან კვირის ბოლოს, ეს განისაზღვრება ორგანიზაციის ხელმძღვანელობის მიერ) მოხდეს არსებული მონაცემთა ბაზების არქივირება და შენახვა. თუ მომდევნო დღეს (კვირას) მოხდა ინფორმაციის დაკარგვა ან დაზიანება, მაშინ არსებული არქივირებული ფაილიდან მოხდება წინა დღის (კვირის) ინფორმაციის აღდგენა. ეს კი ნიშნავს, რომ დაიკარგება მხოლოდ ბოლო დღის (კვირის) მონაცემები, რაც უკეთესია, ვიდრე საერთოდ დაკარგვა.

არქივირების პროგრამები, როგორიცაა მაგალითად, Winzip, Winrar და სხვა პაკეტები, ასრულებს უნივერსალურ ფუნქციას. ასევე შეიძლება მონაცემთა ბაზების სისტემებსაც ჰქონდეს სპეციალური არქივატორები (Backup-შეკუმშვა, Restore-გახსნა) და ა.შ.

4.3. ანტივირუსული აპარატულ-პროგრამული საშუალებანი

ინტერნეტში საიმედო და უსაფრთხო მუშაობისათვის ვირუსების წინააღმდეგ აუცილებელია სპეციალური აპარატულ/პროგრამული მექანიზმების გამოყენება. ლოკალურ საოფისე ქსელსა და ვებ-ბრაუზერს შორის საურველია მოხდეს „გადატიხვრა", ისე რომ ლოკალურ ქსელში ვერ შეძლონ „დივერსანტებმა" არავტორიზებული შეღწევა.

იმისათვის, რომ საიმედოდ გადავტიხროთ ჩვენი საოფისე ქსელი და დავიცვათ გარე „მავნე" ქსელისგან გამოიყენება პროგრამულ–აპარატული კომპლექსი, რომე-

ლიც ემსახურება დამცავი ეკრანის შექმნას შიგა ლოკალურ საოფისე ქსელსა და ინტერნეტის საშიშ ქსელს შორის. აუცილებელია აგრეთვე პროგრამული ანტივირუსების გამოყენებაც და მათი ხშირად განახლება.

ბრანდმაუერები (Firewalls – ქსელთაშორისი ეკრანი) ქმნის „ცეცხლოვან კედელს“ ვირუსებისთვის და ისინი ვერ აღწევს მომხმარებელთა კომპიუტერებამდე (97).

4.3.1. Firewall – ქსელური ეკრანი

ქსელური ეკრანის მუშაობის ყველაზე მარტივი სქემა გამოიყურება შემდეგნაირად: ყველა ქსელი იყოფა „გარე“ (რომლისგანაც ვიცავთ თავს) და „შიგა“ (რომელსაც ვიცავთ) ქსელისგან. თუ გარე ქსელისგან შემოვა მოთხოვნა შიგა კომპიუტერულ ქსელში ჩასართავად, უმეტეს შემთხვევაში firewall (ქსელის დამცავი მოწყობილობა) ამ მოთხოვნას უარყოფს. მხოლოდ იმ შემთხვევაში, თუ შიგა ქსელიდან იქნება მოთხვნა გარე ქსელთან დასაკავშირებლად (მომხმარებელი ცდილობს ბრაუზერში რომელიმე საიტზე გვერდის გახსნას), მაშინ firewall ამოწმებს, მომხმარებელს აქვს თუ არა უფლება დაუკავშირდეს ამ კონკრეტულ საიტს.

ზოგადად Firewall-ს აქვს შემდეგი ფუნქციები:

1) ახდენს ინტერნეტის ანალიზს;

2) უკრძალავს გარედან შემოსულ მოთხოვნებს (თანხმობის გარეშე) შიგა ქსელთან დაკავშირებას;

3) არეგულირებს ურთიერთობას შიგა ლოკალურ საოფისე ქსელსა და გარე ქსელს შორის;

4) ამოწმებს ვირუსზე ისეთ მომსახურებას, როგორიცაა საფოსტო ფუნქციები;

5) წარმოადგენს სტატისტიკას, რომელიც ასახავს გარე ქსელიდან უარყოფილ და მიღებულ მოთხოვნებს;

6) ასახავს შიგა მომხმარებელთა სტატისტიკას (ვინ, სად, როდის და რამდენჯერ შევიდა ინტერნეტში);

7) ახორციელებს სისტემურ ადმინისტრატორთან ავტომატურ შეტყობინებას, რომელიც ასახავს მნიშვნელოვან მოვლენებს;

8) ახორციელებს ავტომატურ დაცვა-დამისამართების ბლოკირებას.

9) უზრუნველყოფს საკუთარი პროგრამული პაკეტის მთლიანობის დაცვას;

10) გამოავლენს უჩვეულო და საშიშ პროცესებს შიგა მოვლენების ავტომატური ანალიზის საფუძველზე;

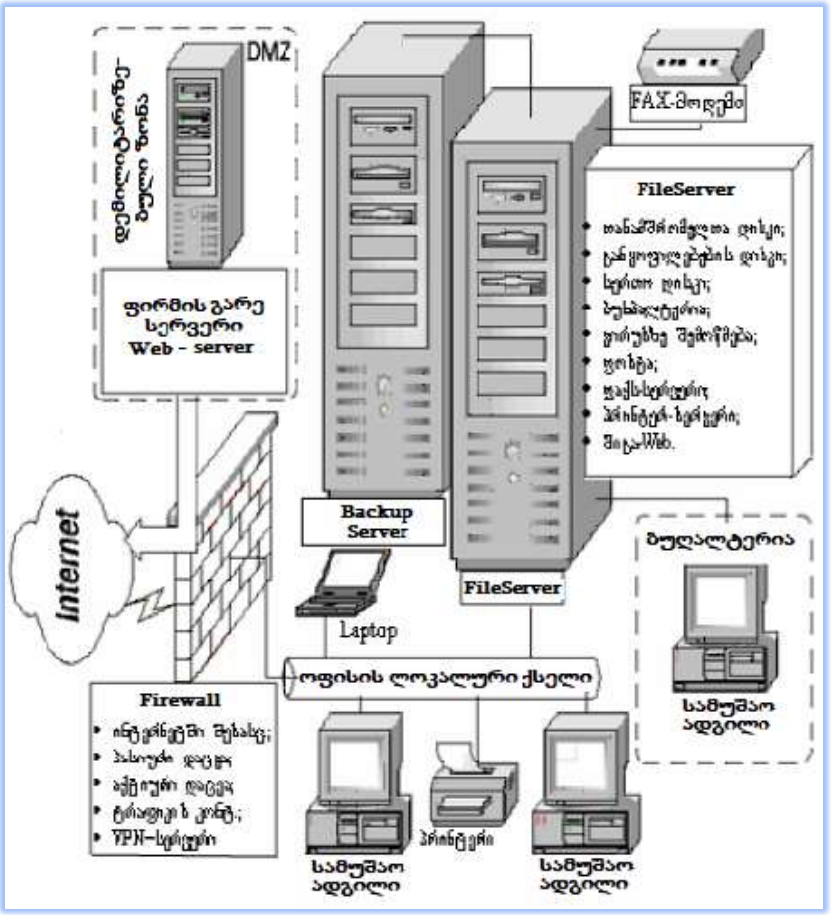
11) უზრუნველყოფს ინტერნეტ ქსელთან დაშიფრულ დაკავშირებას.

4.3.2. DMZ – დემილიტარიზებული ზონა

შიგა და გარე ქსელის გარდა არსებოს ქსელი რომელსაც DMZ დემილიტარიზებული ზონა ეწოდება [98,112].

DMZ-ს ვიყენებთ იმ შემთხვევაში, როდესაც სავაჭრო ცენტრს ინტერნეტში აქვს თავის გარე ქსელი (ვებ-სერვერი) და იგი საჭიროებს დაცვას. 4.4 ნახაზზე ნაჩვენებია სავაჭრო ცენტრის ქსელის ზოგადი არქიტექტურა DMZ ზონით.

ამ შემთხვევაში ეს სერვერი თავსდება DMZ-დემილიტარიზებულ ზონაში, სადაც იგი გაცილებით დაცულია გარე „შემოტევებისგან“.



ნახ.4.4.

4.3.3. ოპტიმიზაციის სისტემა და ინტერნეტ-ტრაფიკზე კონტროლი (Proxy-სერვერი)

ინტერნეტში მუშაობისას ხშირად რამდენიმე თანამშრომელი მუშაობს ერთსა და იმავე დოკუმენტთან ან შედიან ერთსა და იმავე საიტზე. ასეთი დროს იყენებენ ოპტიმიზაციის ინტერნეტ-ტრაფიკ

სისტემას. იგი ყველა ტრაფიკს უშვებს თავის-თავის გავლით და ინახავს მათ ასლებს დისკზე. თუ მომხმარებელს ექნება სურვილი განმეორებით გამოიძახოს გვერდი, მას აღარ დასჭირდება ინტერნეტის გამოძახება, იგი პირდაპირ უკავშირდება ოპტიმიზაციის ინტერნეტ-ტრაფიკ სისტემურ დისკს. პროცესი გაცილებით სწრაფაა და იაფი. სისტემას უწოდებენ Proxy-სერვერს [99].

ამგვარად, პროქსი სერვერი შუამავალი სერვერია კლიენტსა და ინტერნეტს შორის. მისი ძირითადი ფუნქციებია:

- ბრანდმაუერის (Firewall) და ქსელის მონაცემთა გაფილტვრა;
- ქსელის კავშირის გაზიარება;
- მონაცემთა ქეშირება.

Proxy-სერვერი არეგულირებს კომპანიის თანამშრომლებს შორის ინტერნეტ-რესურსების გადანაწილებას. მას აქვს მოქნილი მოწყობილობა, რომელიც განსაზღვრავს მომხმარებელთა ურთიერთობას კონკრეტულ საიტთან, აკონტროლებს ინტერნეტიდან გადმოტვირთულ ინფორმაციის მოცულობას და უზრუნველყოფს ამა თუ იმ ინტერნეტ მომხმარებლის მუშაობას დროის ნებისმიერ მომენტში.

Proxy-სერვერი მომხმარებელს საშუალებას აძლევს არ გამოვიდეს თავის ინტერნეტ მისამართიდან და მოთხოვნის წინ შეცვალოს მისამართი. მოთხოვნა დააყენოს არა როგორც ქსელის რეალური მომხმარებლის სახელით, არამედ თავის პირადი მისამართით. ეს ეხმარება არა მხოლოდ შიგა ქსელის სტრუქტურის დამალვას უსაფრთხოების დაცვის მიზნით, არამედ თავის ინტერნეტ-პროვაიდერს არ მოუწევს თითოეულ თანამშრომლის მისამართზე დამატებითი იჯარის გაცემა, რადგან ფიქსირებული იქნება Proxy-სერვერისთვის ერთადერთი მისამართი.

Proxy-სერვერი ახორციელებს ინტერნეტიდან მონაცემთა

კეშირებას (მონაცემთა შენახვა განმეორებითი მოთხოვნისთვის), და განმეორებითი მოთხოვნის დროს იგი პირდაპირ თავის დისკიდან მიაწვდის მომხმარებელს ინფორმაციას. ეს მნიშვნელოვნად აჩქარებს მრავალი გვერდის ჩატვირთვას და რაც მთვარია პროვაიდერს უმცირდება თანხა ინტერნეტიდან ჩატვირთულ გვერდების მოცულობაზე.

4.3.4. ვირტუალური კერძო ქსელი -VPN

ხშირად ფირმას აქვს განთავსებული ობიექტები (მაგალითად, მაღაზიათა ქსელი) სხვადასხვა ტერიტორიაზე. ასეთი ფირმების არსებობის შემთხვევაში დგება საკითხი თუ როგორ უნდა დაუკავშირდეს ფილიალები ერთმანეთს ერთ სერთო საინფორმაციო ქსელში. ამასთან ერთად ეს კავშირი უნდა იყოს მაქსიმალურად მოხერხებული, უსაფრთხო და იაფი.

იმ მიზნით, რომ შევქმნათ ამ პირობების გათვალისწინებით ლოკალური ქსელი ინტერნეტ ქსელის ინფრასტრუქტურის გამოყენებით, უნდა სრულდებოდეს შემდეგი პირობები:

1) გადაწყვეტილება უნდა იყოს უნივერსალური და შეესაბამისობაში მოდიოდეს დანართებთან. ე.ი ის არ უნდა იყოს დამოკიდებული მხოლოდ ერთ კონკრეტულ პროგრამაზე, არამედ ყველა ფილიათან იყოს თავსებადი და ემსაუხრებოდეს ერთ მიზანს;

2) მომხმარებელს არ უნდა შეხვდეს რაიმე სირთულე ამ ტექნოლოგიის გამოყენებისას;

3) დიდად არ განსხვავდებოდეს არსებული ქსელისგან;

4) ფილიალებს შორის ურთიერთკავშირისთვის უნდა გამოიყენებოდეს უკვე არსებული ინტერნეტ ქსელი ან საკუთარი კავშირის ხაზები;

5) გადაწყვეტილების მიღება არ უნდა უკავშირდებოდეს

კონკრეტულ მწარმოებელს, იგი ორიენტირებული უნდა იყოს ღია პროტოკოლზე და სპეციფიკაციაზე;

6) ნებისმიერ შემთხვევაში ძირითადი პროტოკოლი უნდა უკავშირდებოდეს არსებულ პროტოკოლს, რომელიც ფართოდ გამოიყენება ინტერნეტში – TCP/IP;

7) ქსელებს შორის ინფორმაციული კავშირი უნდა იყოს საიმედოდ დაშიფრული;

8) დაშიფვრის ალგორითმი უნდა იყოს საიმედო, მრავალმხრივად შემოწმებული. შესაძლებელი უნდა იყოს მომხმარებლის მიერ, თავისი შეხედულებისამებრ, დაშიფვრის ალგორითმის და მისი პარამეტრების შერჩევა.

ამ ტექნოლოგიის გამოყენება საკმაოდ იაფია, რადგან მისი ძირითადი პრინციპია კავშირებისთვის გამოიყენოს ლოკალური განაწილებული ქსელი ინტერნეტთან ერთად. VPN – ტექნოლოგიის გამოყენების ძირითადი არსია ყველა არხის შიფრირება, რომელიც ქსელში ინფორმაციის გადაცემის დროს გამოიყენება [100]. ორივე ფილიალში დგას VPN–თან მიერთებული ბრანდმაუერი, რომელიც უკავშირდება ერთმანეთს ინტერნეტით. როდესაც რომელიმე მომხმარებელი ქსელიდან LAN-1 უკავშირდება მეორე LAN-2 ქსელს, მისი მოთხოვნა გაივლის VPN+Firewall-1 ბარიერს.

ეს ნიშნავს, რომ LAN-1 დან ბარიერის გავლით გასული მოთხოვნის მონაცემები და მისამართი გაივლის შიფრაციას და ინტერნეტის საშუალებით გადაეცემა ბრანდმაუერს, რომელსაც თავის VPN+Firewall-2 დამცველი ბარიერი აქვს იმავე შემოწმების შემდეგ მიაღწევს LAN-2-თან. ანალოგიური პროცესი ხორციელდება უკუკავშირის დროსაც.

მოცემული ტექნოლოგიის დანერგვის შემდეგ უსაფრთხო ხდება ლოკალური ქსელის დახმარებით ფილიალებს შორის ურთიერთკავშირი და საკმაოდ ამაღლებს კომპანიის მუშაობის წარმადობას.

4.3.5. მობილური კლიენტი

დისტანციაზე (მაგალითად, მივლინებაში) მყოფ თანამშრომელს ჭირდება თავისი კომპანიის შიგა ქსელში შესვლა, რომ დაათვალიეროს ფირმის განახლებული, „პრაიზლისტი“, შიგა ბრძანებები, ანდაც მან უშუალოდ შეძლოს თანამშრომელთა შემოწმება და ბრძანებების გაცემა.

ასეთი პროცესების რეალიზაციის საფუძველია პროტოკოლები: ინტერნეტ- პროტოკოლი IPSec და Ms PPTP [101, 102].

IPsec პროტოკოლების ჯგუფია, რომლებიც გამოიყენება მოწყობილობებს შორის დაშიფრული კავშირების დასაყენებლად. ის ეხმარება დაცული იყოს საჯარო ქსელებით გაგზავნილი მონაცემები. IPsec ხშირად გამოიყენება VPN-ების დასაყენებლად და ის მუშაობს IP პაკეტების დაშიფვრით, წყაროს ავთენტიფიკაციით, საიდანაც მოდის პაკეტები.

ქსელის პროტოკოლი (PPTP – Point to Point Tunneling Protocol) გამოიყენება VPN გვირაბების შესაქმნელად საჯარო ქსელებს შორის. PPTP ცნობილია ასევე, როგორც ვირტუალური პირადი სატელეფონო ქსელის (VPDN) სერვერები. PPTP უპირატესობას ანიჭებს სხვა VPN პროტოკოლებს, რადგან ის უფრო სწრაფია და აქვს მობილურ მოწყობილობებზე მუშაობის შესაძლებლობა.

4.3.6. განაწილებილი ოფისი

გეოგრაფიულად განაწილებული კორპორაციებისა და ფილიალების ერთ ინფორმაციულ ქსელში ინტეგრაციიის მიზნით Internet Engineering Task Force (IETF) ფირმამ შეიმუშავა ინტერნეტ ქსელის ყველა სტანდარტი და პროტიკოლი [103]. ასეთი გახლდათ, ზემოთ ნახსენები IPSec პროტოკოლი [101].

IPSec ყველაზე ეფექტურია განაწილებულ ოფისებში. იგი ითვლება როგორც ყველაზე საიმედო და მოხერხე-ბული უნივერსალური საშუალება, რომელიც ფილიალებს აკავშირებს ორგანიზაციასთან..

საშუალოდ, ყოველი ფილიალის ლოკალური ქსელის კონფიგურაცია აწყობილია ბაზაზე, სადაც მუშაობს 10-150 თანამშრომელი. ყოველ ფილიალში ფირმები აყენებს ქსელურ HUB-მოწყობილობას, რომელიც ასრულებს შემდეგ ფუნქციებს: უზრუნველყოფს ფილიალების დაკავშირებას ინტერნეტთან; ასრულებს ქსელური ეკრანის (Firewall) ფუნქციას და მოიცავს VPN ტექნოლოგიას. VPN-ჰაბი შედგება რამდენიმე ვირტუალური არხისგან და უზრუნველყოფს ყველა ფილიალის გამჭვირვალე კავშირს სისტემის ლოკალურ ქსელთან (მხოლოდ თანამშრო-მლებისთვის).

ყველა ფილიალში ინახება ქსელის პირვანდელი სტრუქ-ტურა, რადგან ყოველი თანამშრომლისთვის ლოკალური ქსელი ფართოვდება და მოიცავს ფირმის ყველა ფილიალს. ამიტომ ყველა თანამშრომელს შეუძლია იოლად გამოიყენოს ნებისმიერი დოკუმენტი, რომელიც ფირმის ნებისმიერ კომპიუტერშია მოთავსებული. ამასთან არცერთ თანამშრომელს არ შეუძლია წაშალოს კომპანიის რაიმე დოკუმენტი.

ყველა მონაცემი გადაიცემა შიფრირებულად და მისი გაშიფრვა შეუძლებელია სპეციალური შიფრატორის გარეშე, რომელზეც მხოლოდ ფირმის ხელმძღვანელს მიუწვდება ხელი.

4.4. Ms SQL Server -ის არქიტექტურა

განვიხილთ მოკლედ Ms SQL Server სისტემის არქიტექტურა 2019 ვერსიის მაგალითზე [104, 105].

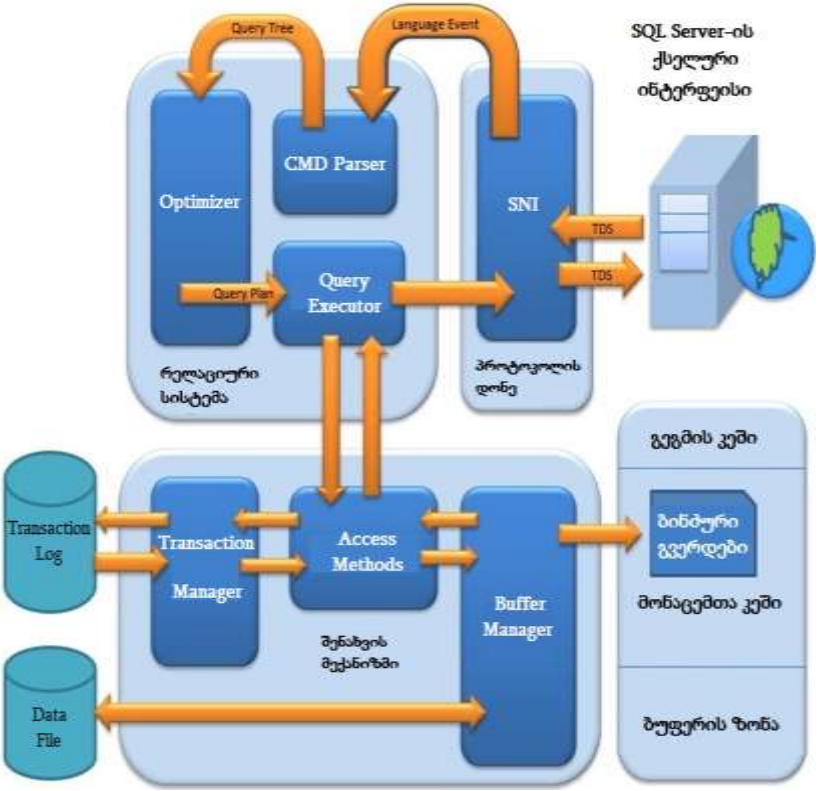
SQL Server 2019/22 ამ დროისათვის ბოლო ვერსიაა. იგი ბიზნეს აპლიკაციების წარმატებული მხარდამჭერი სისტემაა. ამასთანავე, მნიშვნელოვნად გაფართოვდა მისი ფუნქციები, განსაკუთრებით სხვადასხვა სახის სერვისებით [106]. კერძოდ:

- SQL Server Relational Engine – მონაცემთა შენახვა, მოთხოვნების მართვა და რეალურ დროში მათი დამუშავება;
- SQL Server Agent – სისტემა დაგეგმვისა და შეტყობინებებისთვის, რომლებიც მიეწოდება რელაციური მექანიზმით;
- SQL Server Integration Services (SSIS) – მონაცემთა ექსპორტის, იმპორტის, ტრანსფორმაციის და ჩატვირთვის სერვისები;
- SQL Server Reporting Services (SSRS) – ანგარიშების (რეპორტების) შექმნის, მართვის და მიწოდების სერვისები;
- SQL Server Analysis Services (SSAS) – ბიზნეს ანალიზისთვის მონაცემთა შექმნის, მართვის, ანალიზის, აგრეგირების და შეტყობინებათა მიწოდების სერვისები;
- Power BI – ანგარიშების შექმნის ინსტრუმენტი დესკტოპ- და ღრუბელზე ბაზირებული.

SQL Server-ის ოფისში ამუშავების შემდეგ გვაქვს შემდეგი ღრუბლოვანი პარამეტრები:

- Azure – Microsoft-ის საერთო ღრუბლოვანი შეთავაზება;
- Azure SQL Database – Microsoft-ის მონაცემთა ბაზის სერვისი;
- AWS – Amazon-ის საერთო ღრუბლოვანი შეთავაზება;
- GCP – Google-ს ღრუბლოვანი პლატფორმა SQL სერვერისთვის.

Ms SQL Server რელაციური ბაზის არქიტექტურის საკითხები კარგადაა დეტალურად წარმოდგენული რიჩარდ პეტერსონის ელ-სტატიაში [107]. განვიხილოთ ზოგიერთი ძირითადი ცნება და პროცესი ამ ნაშრომის მიხედვით 4.5 ნახაზის საფუძველზე.



ნახ.4.5. SQL Server-ის არქიტექტურა [107]

Ms SQL Server არის კლიენტ-სერვერული არქიტექტურა. პროცესი იწყება კლიენტის აპლიკაცია აგზავნის მოთხოვნას. SQL სერვერი იღებს, ამუშავებს და პასუხობს ამ მოთხოვნას დამუშავებული მონაცემებით. დეტალურად განვიხილოთ ქვემოთ ნაჩვენები მთლიანი არქიტექტურა:

SQL სერვერის არქიტექტურაში სამი ძირითადი კომპონენტია:

- პროტოკოლის დონე (Protocol Layer);
- რელაციური სისტემა (Relational Engine);
- შენახვის მექანიზმი (Storage Engine).

სერვერის ქსელური ინტერფეისი (Server Nnetwork Interface – SNI) არის ცენტრალიზებული კოდი, გაზიარებული SQL Server და SQL Server კლიენტის პროვაიდერებს შორის, როგორც Windows, ასევე Linux, რომელსაც შეუძლია გაგზავნოს, გაანალიზება და იმოქმედოს TDS კომუნიკაციებზე. ღია მონაცემთა სერვისები (ODS) იყო ცალკე, საჯარო API ნაკრები, რომელიც საშუალებას გაძლევთ შექმნათ TDS კარიბჭე.

ცხრილური მონაცემთა ნაკადი (Tabular Data Stream – TDS) არის აპლიკაციის დონის პროტოკოლი, რომელიც გამოიყენება კლიენტებსა და მონაცემთა ბაზის სერვერის სისტემებს შორის მოთხოვნებისა და პასუხების გადასაცემად. ასეთ სისტემებში კლიენტი, როგორც წესი, დაამყარებს ხანგრძლივ კავშირს სერვერთან. TDS პროტოკოლი და Dataverse-ს ბოლო წერტილია. ეს ბოლოო წერტილი საშუალებას გვაძლევს დავუკავშირდეთ Dataverse-ს, რათა მივიღოთ მონაცემები Dynamics 365 CE ან Power Platform გარემოში. ეს წერტილი იძლევა სისტემაში მხოლოდ წაკითხვის საშუალებას ანგარიშგების და ანალიზის მიზნით. Dataverse ახორციელებს მონაცემთა უსაფრთხოდ შენახვას და მართვას, რასაც იყენებს ბიზნეს აპლიკაციები. მონაცემები Dataverse-ში ინახება ცხრილების ერთობლიობაში. ცხრილი არის სტრიქონების (ჩანაწერების) და სვეტების (ატრიბუტების) ერთობლიობა

სახელმინიჭებული არხი (Named pipes) – არის დასახელების მქონე კომუნიკაციის ცალმხრივი ან ორმხრივი არხი სერვერსა და ერთ ან მეტ კლიენტს შორის. სახელიანი არხის ყველა ეგზემპლარს აქვს ერთიდაიმავე არხის სახელი, მაგრამ თითოეულ ეგზემპლარს აქვს საკუთარი ბუფერები და დესკრიპტორები და უზრუნველყოფს ცალკე არხს კლიენტის/სერვერის კომუნიკაციისთვის.

ენობრივი მოვლენა (Language Event) – ტექსტური ტიპის მოთხოვნა მიღებულია SQL Server-ზე. სხვა ვარიანტია RPC (Remote

Procedure Call) Event, რაც პროცედურის დისტანციურ გამოძახებას შეესაბამება.

ბრძანებათა ანალიზატორი (CMD-Parser): პროტოკოლის შრიდან მონაცემები გადაეცემა რელაციურ სისტემას. CMD ანალიზატორი იღებს მოთხოვნის მონაცემებს და ახდენს მისი სინტაქსური და სემანტიკური შეცდომების შემოწმებას. საბოლოოდ, იგი ქმნის მოთხოვნათა ხეს (Query Tree).

მოთხოვნების ხე (Query Tree) – მონაცემთა სტრუქტურაა, რომელიც შეესაბამება RA გამოხატულებას. ფოთლის კვანძები: შეყვანის ურთიერთობები, შიდა კვანძები: RA ოპერაციები. • შეკითხვის ხის შესრულება შედგება შიდა კვანძის შესრულებისგან. ოპერაცია, როდესაც მისი ოპერანდები ხელმისაწვდომია და შემდეგ ჩანაცვლება. შიდა კვანძი მიღებული ურთიერთობით.

ოპტიმიზატორის (Optimizer) საქმეა მომხმარებლის მოთხოვნის შესრულების გეგმის აგება. ყველა მოთხოვნა არ არის ოპტიმიზირებული. ოპტიმიზაცია სრულდება DML (მონაცემთა მანიპულირების ენის) ბრძანებებისათვის, როგორიცაა SELECT, INSERT, DELETE და UPDATE. ასეთი მოთხოვნები ჯერ მოინიშნება, შემდეგ იგზავნება ოპტიმიზატორში. DDL (მონაცემთა აღწერის ენის) ბრძანებები, როგორიცაა CREATE და ALTER, არაა ოპტიმიზირებადი, მაგრამ ისინი კომპილირდება შიგა (მანქანურ) ფორმაში. მოთხოვნის ღირებულება გამოითვლება ისეთი ფაქტორების საფუძველზე, როგორიცაა პროცესორის (CPU) და მეხსიერების (Memory) გამოყენება და შეტანა/გამოტანის (I/O) მოთხოვნილებები. ოპტიმიზატორის როლია იპოვოს ყველაზე იაფი და რენტაბელური შესრულების გეგმა.

Query executer (მოთხოვნების შემსრულებელი) იძახებს მონაცემებთან წვდომის მეთოდს. იგი უზრუნველყოფს შესრულების გეგმას საჭირო მონაცემების მოპოვების ლოგიკის შესასრულებლად. როგორც კი მონაცემები მიიღება Storage_Engine-ს

Access_Metod-დან, შედეგი გამოქვეყნდება პროტოკოლის დონეზე. საბოლოოდ, მონაცემები ეგზავნება საბოლოო მომხმარებელს.

მონაცემთა შენახვის მექანიზმის (Storage Engine) - დანიშნულებაა მონაცემების შენახვა, მაგალითად, დისკზე (Disk) ან მონაცემთა შენახვის ქსელში (SAN - Storage Area Network), აგრეთვე მონაცემთა ამოღება მეხსიერებიდან, საჭიროების შემთხვევაში. მონაცემთა ფაილი ფიზიკურად ინახავს მონაცემებს გვერდების სახით, გვერდი 8-კბაიტია, რაც ქმნის SQL Server-ში ყველაზე პატარა შენახვის ერთეულს. 8 გვერდი ქმნის ერთ ექსტენტს (64-კბაიტი). ობიექტის მომსახურება ხდება ექსტენტების მიხედვით. გვერდს აქვს 96 ბაიტიანი „გვერდის სათაური", რომელიც შეიცავს მეტა-მონაცემების ინფორმაციას, როგორიცაა გვერდის ტიპი, ნომერი, გამოყენებული და თავისუფალი სივრცის ზომები, მისამართები წინა და შემდეგ გვერდებზე.

ბუფერების მენეჯერი (Buffer Manager) მართავს ქვემოთ მოცემულ მოდულების ძირითად ფუნქციებს:

- გეგმის ქეში Plan Cache);
- მონაცემთა ანალიზი: ბუფერული ქეში და მონაცემთა საცავი (Data Parsing: Buffer cache & Data storage);
- ბინძური გვერდი (Dirty Page).

თუ მონაცემები არაა ბუფერების მენეჯერში, საჭირო იქნება მათი ძებნა მონაცემთა საცავში. თუ სისტემა ინახავს მათ მონაცემთა ქეშში მომავალი გამოყენებისთვის.

ბინძური გვერდი (Dirty Page) ინახება როგორც ტრანზაქციის მენეჯერის დამუშავების ლოგიკა (იხ. ქვემოთ, ტრანზაქციების მენეჯერის აღწერისას).

ტრანზაქციის მენეჯერი გამოიძახება მაშინ, როდესაც წვდომის მეთოდი ადგენს, რომ მოთხოვნა არის *ოპერატორი არჩევის გარეშე.*

ჟურნალის მენეჯერი (Log Manager) ინახავს სისტემაში განხორციელებულ ყველა განახლებას ტრანზაქციების ჟურნალში. ჟურნალებს აქვს მიმდევრობითი ნომერი ტრანზაქციის ID-ით და მონაცემთა მოდიფიკაციის ჩანაწერით. იგი გამოიყენება დასრულებული ტრანზაქციის ფიქსირებისათვის და, საჭიროების შემთხვევაში, ტრანზაქციის წინა მდგომარეობის აღსადგენად.

ბლოკირების მენეჯერი (Lock Manager). ტრანზაქციის დროს მონაცემთა შენახვისას დაკავშირებული მონაცემები ბლოკირებულ მდგომარეობაშია. ამ პროცესს მართავს Lock Manager. იგი უზრუნველყოფს მონაცემთა მთლიანობას და იზოლაციას (ეს ცნობილია, როგორც ACID თვისებები [36,51].

- Atomicity (ატომარობა) – ტრანზაქციის პროცესი ან მთლიანად სრულდება (უწყვეტად) ან საერთოდ არ სრულდება. თუ ტრანზაქციის პროცესი ნაწილობრივ შესრულდა, მაშინ ატომარობა დარღვეულია;
- Consistency (მთლიანობა) – ტრანზაქციის პროცესი სრულდება მაშინ, როცა ბაზა მთლიანია. ტრანზაქციის პროცესის დასრულებისას ბაზა ისევ მთლიანობის მდგომარეობაში რჩება (მთლიანობა ნიშნავს, რომ თითოეული სტრიქონი და მნიშვნელობა უნდა შეესაბამებოდეს რეალურ სიტუაციას და უნდა სრულდებოდეს ყველა შეზღუდვა. მაგალითად, თუ წერია შეკვეთები და არ წერია საქონელი, მთლიანობის პრინციპი დარღვეულია);
- Isolation (იზოლირება) – ტრანზაქციის თითოეული პროცესი უნდა იყოს იზოლირებული, რაც ნიშნავს, რომ ყველა ტრანზაქციის პროცესი უნდა ხორციელდებოდეს ერთმანეთისაგან დამოუკიდებლად. ახალმა პროცესმა არ უნდა შეუშალოს ხელი უკვე დაწყებული ტრანზაქციის პროცესის დასრულებას. ეს პრინციპი განსაკუთრებით მნიშვნელოვანია, როცა ბაზასთან მუშაობს რამდენიმე მომხმარებელი;

- Durability (მდგრადობა) – გულისხმობს ტრანზაქციის პროცესის შესრულებას სისტემური შეფერხებებისაგან დამოუკიდებლად. მონაცემთა ბაზების მართვის სისტემა ისე უნდა იყოს დაპროექტებული, რომ ახალი ტრანზაქციის შესრულებისას შეფერხებების არსებობის შემთხვევაში, შესაძლებელი გახდეს ბოლოს, სრულად შესრულებული ტრანზაქციის პროცესის შემდეგ არსებული მდგომარეობის აღდგენა.

რელაციურ ბაზაში ნებისმიერი ოპერაცია ხორციელდება პრინციპით: *ყველაფერი ან არაფერი*.

შესრულების პროცესი (Execution Process). ჟურნალის (Log) მენეჯერი იწყებს რეგისტრაციას, ხოლო ბლოკირების მენეჯერი - კი ბლოკავს დაკავშირებულ მონაცემებს. მონაცემთა ასლი ინახება ბუფერულ ქეშში. მონაცემების ასლი, რომელიც უნდა განახლდეს, ინახება ჟურნალის ბუფერში და ყველა მოვლენა განაახლებს მონაცემებს მონაცემთა ბუფერში.

გვერდები (pages), რომლებიც ინახავს მონაცემებს, „ბინძური" გვერდები ეწოდება. საკონტროლო წერტილი (Checkpoint) და ჟურნალში წინასწარ ჩაწერა (Write-Ahead Logging): პროცესი იწყება და მოინიშნება ყველა „ბინძური გვერდი". გვერდი რჩება ქეშში. იგი ჯერ გადადის ბუფერული ჟურნალიდან ჟურნალის ფაილის მონაცემთა გვერდზე (ეს ცნობილია როგორც წინასწარ ჩაწერა).

ზარმაცი მწერალი (Lazy Writer): ბინძური გვერდი შეიძლება დარჩეს მეხსიერებაში. როდესაც SQL სერვერი აკვირდება დიდ დატვირთვას და ახალი ტრანზაქციისთვის საჭიროა ბუფერული მეხსიერება, იგი ათავისუფლებს ბინძურ გვერდებს ქეშიდან. ის მუშაობს LRU ალგორითმით (Least recently used – ბოლო დროს გამოყენებული), დისკზე, ბუფერული ზონიდან გვერდის გასუფთავების მიზნით [107].

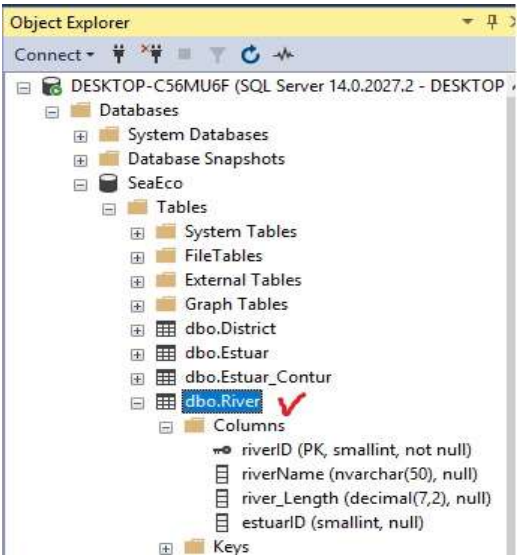
4.5. მომხმარებლის ინტერფეისის აპლიკაციის აგება C#, SQL Server და ADO.NET დრაივერით

განვიხილოთ კონკრეტული ამოცანა მომხმარებლის ინტერფეისის დასაპროგრამებლად C# ენისა და SQL Server მონაცემთა ბაზის გამოყენებით.

ამოცანა_1: მოცემულია SQL Server მონაცემთა ბაზა (მაგალითად, შავი ზღვის ეკოლოგიური მონიტორინგის სისტემა [28]), რომლის ერთ-ერთი ცხრილი (Table) არის River.dbo (შავი ზღვის მდინარეები საქართველოს აკვატორიაში). საჭიროა ავაგოთ Windows Forms აპლიკაცია (მომხმარებლის ინტერფეისი), რომელიც ბაზიდან ამოიღებს მდინარეების მონაცემებს DataGridView ცხრილში, შეძლებს Insert, Update და Delete ოპერაციების განხორციელებას. გამოყენებულ იქნება ADO.NET დრაივერის საშუალებები.

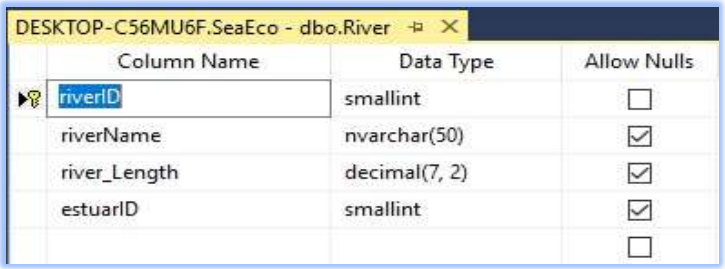
➢ **ექსპერიმენტული SQL Server მონაცემთა ბაზის მომზადება**

4.6 ნახაზზე ნაჩვენებია საწყისი მონაცემთა ბაზა, რომელიც Ms SQL Server პაკეტითაა რეალიზებული. იგი შეესაბამება ბაზის ცხრილების იერარქიას.

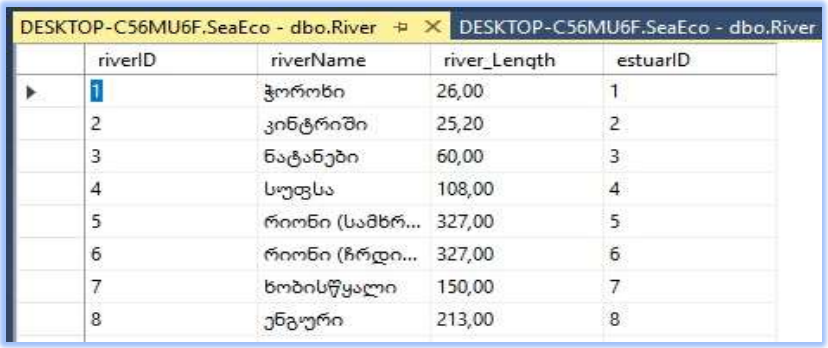


ნახ.4.6. SeaEco მონაცემთა ბაზა 4 ცხრილით

4.7-ა ნახაზზე ჩანს River ცხრილის სტრუქტურა, მონაცემთა შესაბამისი ტიპებით, ხოლო 4.7-ბ ზე კი – მოცემულია ჩანაწერები, რომელთა შეტანა მოხდა წინასწარ (შემდგომში იგი უნდა შეივსოს ინტერფეისიდან – პროგრამულად).



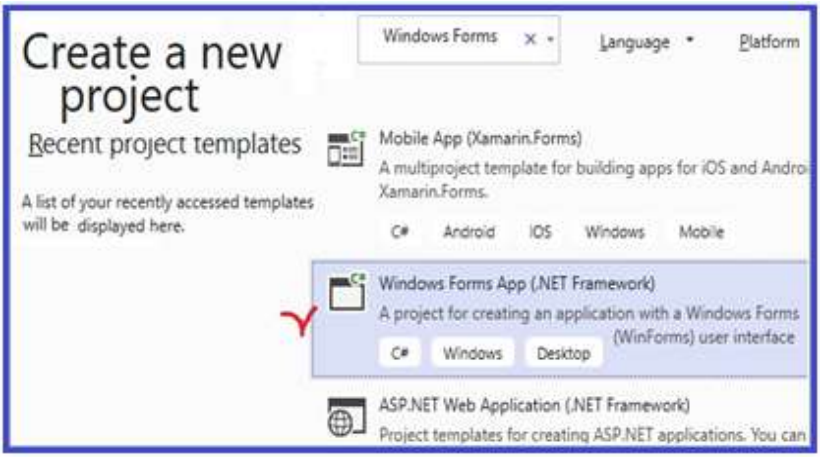
ნახ.4.7-ა. River (მდინარეების) ცხრილის სტრუქტურა



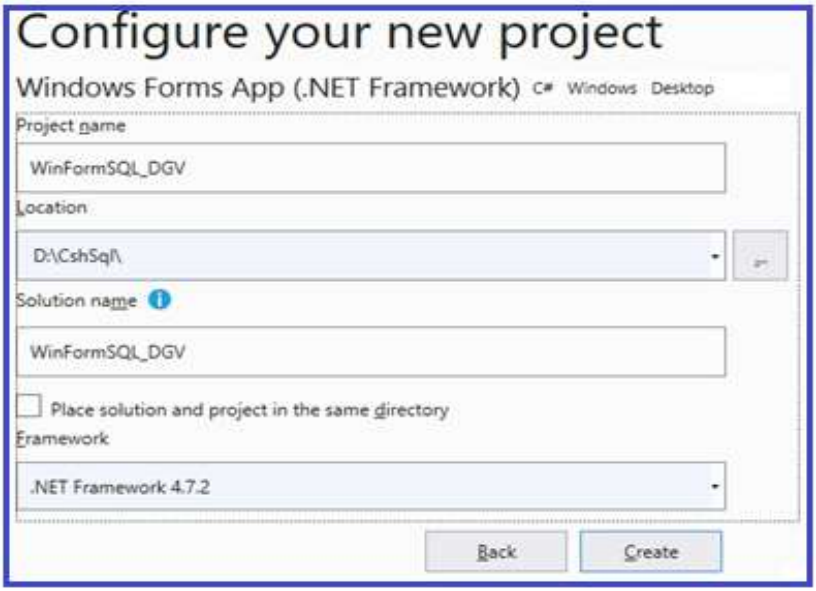
ნახ.7-ბ. River (მდინარეების) საწყისი ცხრილი
Ms SQL Server-ში

- **ახალი პროექტის შექმნა .NET პლატფორმაზე C# ენის გამოყენებით**

მომხმარებლის ინტერფეისის აპლიკაციის პროექტის აგება Ms Visual Studio.NET სამუშაო გარემოში რამდენიმე ეტაპს მოიცავს. განვიხილოთ ეს პროცესი დეტალურად. 4.8-ა,ბ ნახაზზე ნაჩვენებია პროექტის შექმნის პროცედურა.



ნახ.4.8-ა. WinFormSQL_DGV პროექტის შექმნა

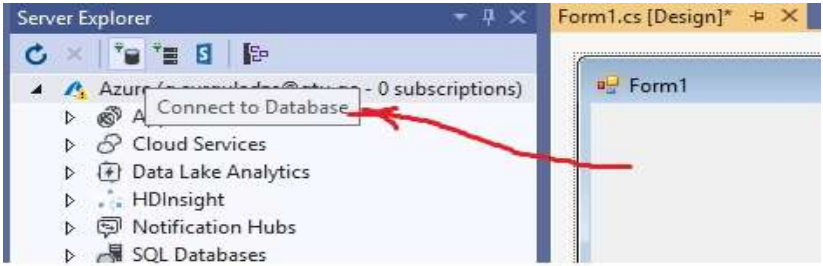


ნახ.4.8-ბ. WinFormSQL_DGV პროექტის შექმნა და განთავსება სისტემის კატალოგში

ვირჩევთ პროექტის სახელს (Name), მისი შენახვის ადგილს (Browse-ს დახმარებით) და Solution name-ს. შემდეგ OK.

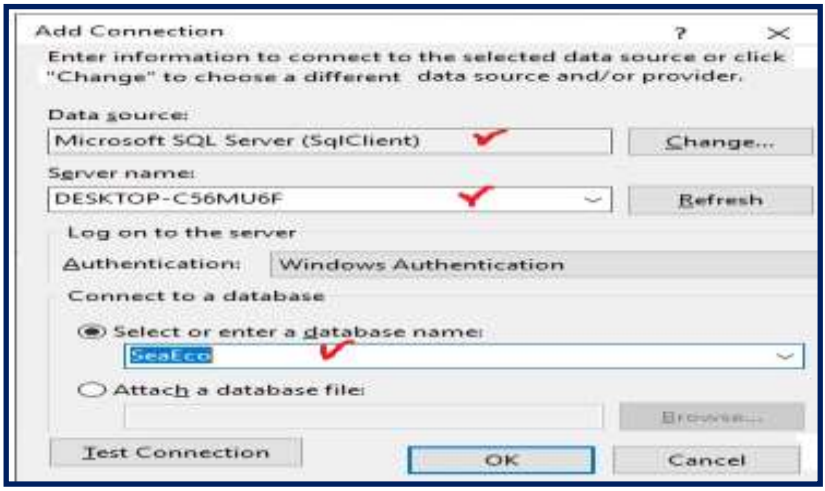
➢ **პროექტთან მონაცემთა ბაზის მიერთება**

საჭიროა პროექტისთვის განვახორციელოთ მონაცემა ბაზის მიერთება (Connect to Database), რაც 4.9 ნახაზზეა ნაჩვენები.



ნახ.4.9. Connect Database ამოქმედება

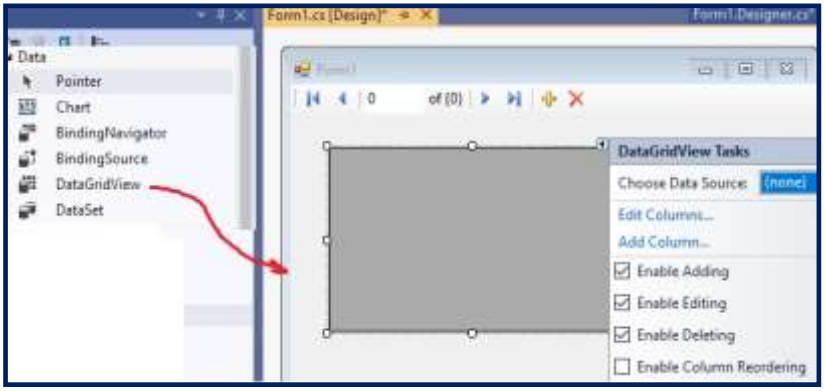
გამოიტანება ახალი ფანჯარა (ნახ.4.10), რომელშიც უნდა შეირჩეს შესაბამისი წყარო (Data Source), სერვერი (Server name) და მონაცემთა ბაზა (Database name).



ნახ. 4.10. მბ-ის წყაროს არჩევა

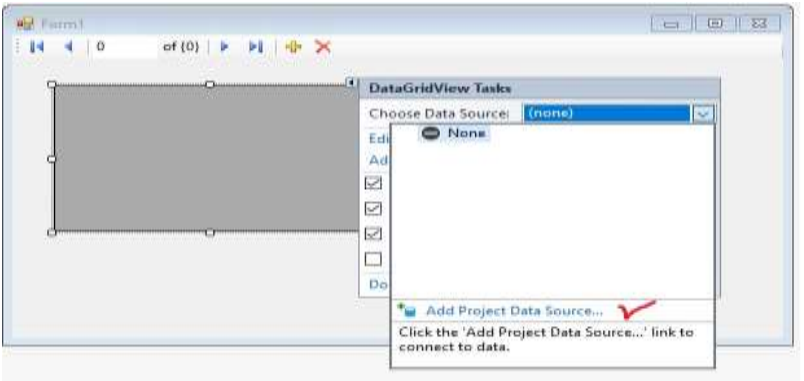
- **DataGridView ელემენტის გააქტიურება და ცხრილის პარამეტრების განსაზღვრა**

ინსტრუმენტების პანელიდან ფორმაზე გადავიტანოთ DataGridView ელემენტი და ზედა მარჯვენა კუთხე პატარა ისრით ავამოქმედოთ. მივიღებთ 4.11 ნახაზს.



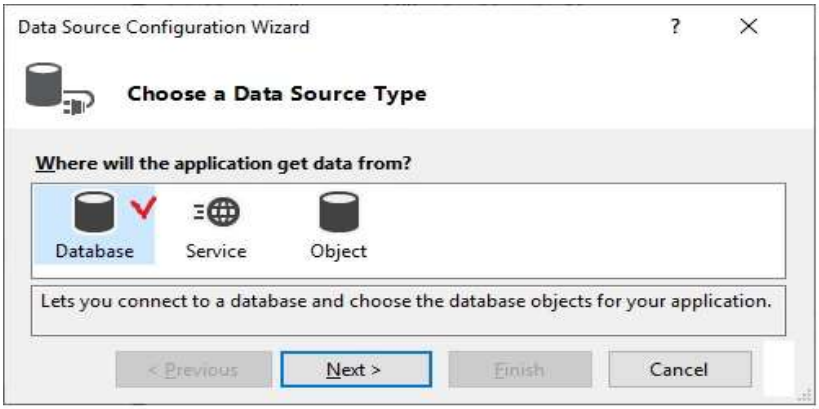
ნახ.4.11. პარამეტრების განსაზღვრა

ნახაზზე ჩანს, რომ ჩამატების, რედაქტირებისა და წაშლის ოპერაციები ნებადართულია (ჩეკბოქსები მონიშნულია). ავირჩიოთ Choose Data Source კომბობოქსის ღილაკი, მივიღებთ 4.12 ნახაზს.



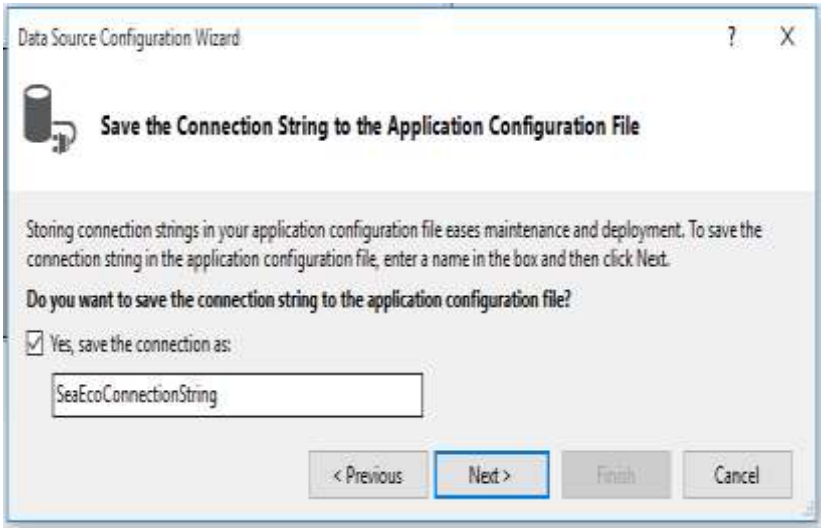
ნახ.4.12. მონაცემთა წყაროს პროექტის დამატება

ავამოქმედოთ Add Project Data Source და გადავიდეთ 4.13 ნახაზზე.



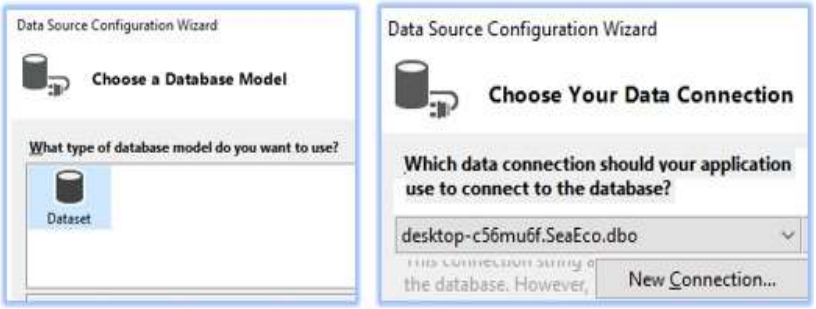
ნახ.4.13. მონაცემთა წყაროს ტიპის არჩევა

ვირჩევთ Database-ს და Next. მივიღებთ 4.14 ნახაზზე მოცემულ ფანჯარას.



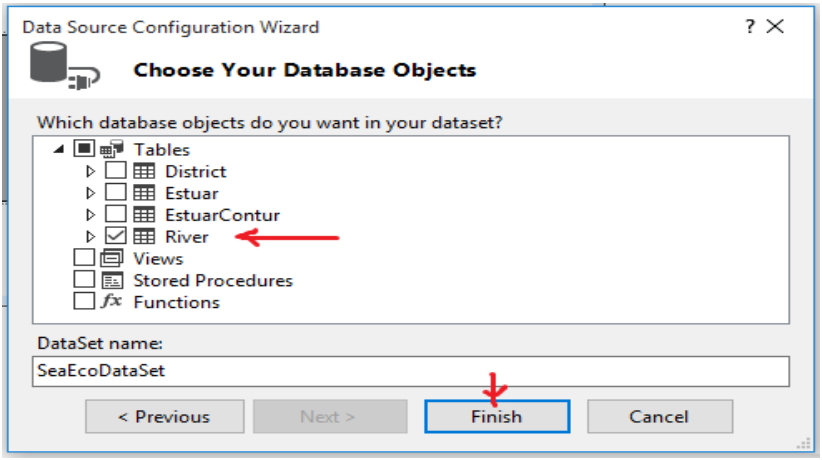
ნახ.4.14. SeaEco Connection (მიერთების) შერჩევა

Connection პარამეტრი ყველა კომპიუტერს ექნება თავისი. მისი განსაზღვრა შესაძლებელია Server Explorer-იდან (ჩვენ შემთხვევაში იგი არის: DESKTOP-C56MU6F.SeaEco.dbo). ბოლოს Next და გადავალთ 4.15 ნახაზზე.



ნახ.4.15. Connection String-ის შენახვა აპლიკაციის კონფიგურაციის ფაილში

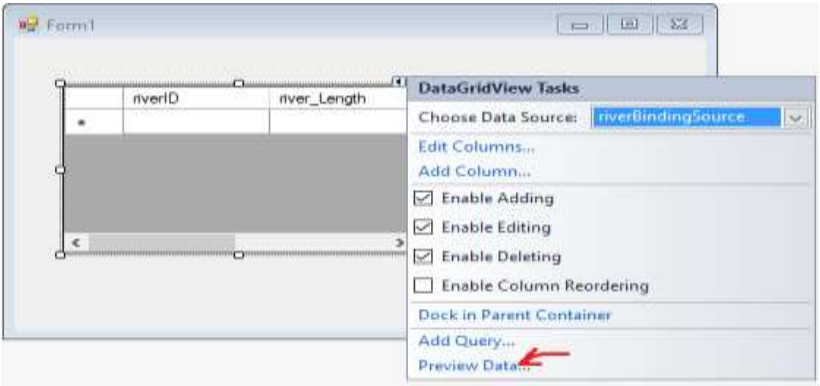
შემდეგ გამოიძახება მონაცემთა ბაზის ობიექტების არჩევის ფანჯარა (ნახ.4.16).



ნახ.4.16. ობიექტების მონიშვნა (მაგალითად, River)

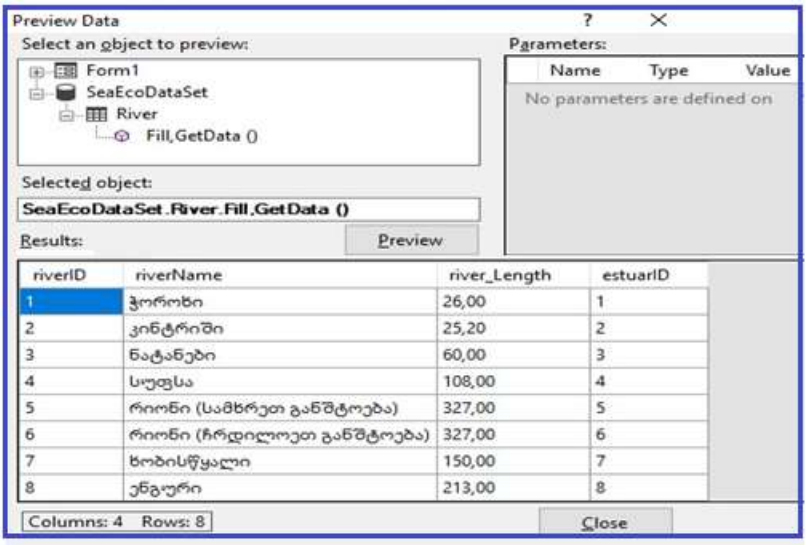
4.17 ნახაზზე ნაჩვენებია მონაცემთა წყაროს (Data Source)

განსაზღვრის შედეგის მნიშვნელობა, ჩვენ შემთხვევაში იგი არის: **riverBindingSource.**



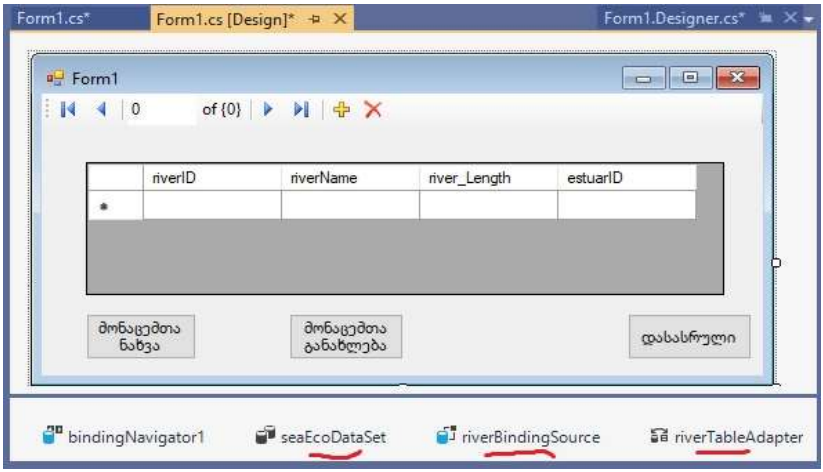
ნახ.4.17. Data Source შედეგი: "riverBindingSource"

აქვე შეიძლება გამოვიყენოთ Preview Data ღილაკი და ვნახოთ წინასწარ მიერთებული ბაზის ცხრილის ჩანაწერები (ნახ.4.18).



ნახ.4.18. Preview Data ცხრილი

ბოლოს, Form1-ს Properties-ში შევუცვალოთ სახელი „შავი ზღვის მდინარეები (საქართველოს საზღვრებში)“, ინსტრუმენტების პანელიდან გადმოვიტანოთ სამი ღილაკი (Button1,2.3), დავარქვათ სახელები. მიიღება 4.19 ნახაზი.



ნახ.4.19. პროექტის ძირითადი ინტერფეისი

- **პროგრამული რეალიზაციის საკითხები**

ახლა გადავიდეთ სისტემის ინტერფეისის ღილაკების ფუნქციების დაპროგრამებაზე, ანუ უნდა განხორციელდეს SQL Server-ის შესაბამისი ბაზის ცხრილის ჩანაწერების შექმნა, წაკითხვა, შეცვლა და წაშლა (CRUD ოპერაციები).

– თავდაპირველად ჩავამატოთ სახელსივრცის სტრიქონი:

```
using System.Data.SqlClient;
```

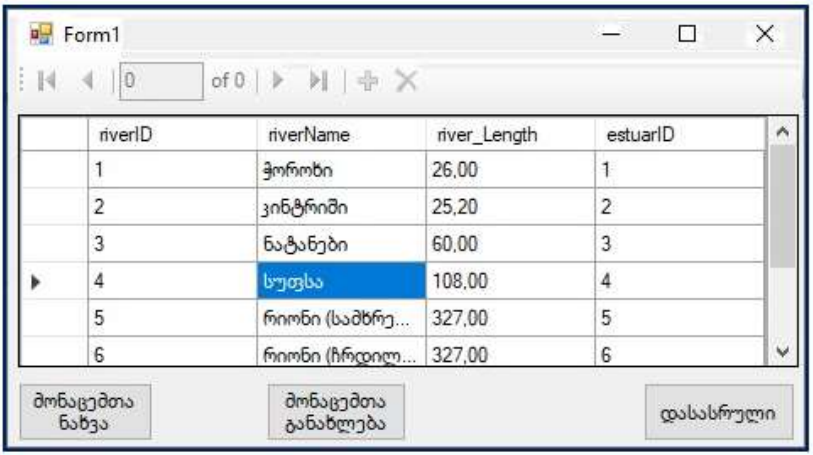
– შემდეგ გამოვაცხადოთ გლობალური ცვლადები:

```
SqlDataAdapter sda;
SqlCommandBuilder scb;
DataTable dt;
```

„მონაცემთა ნახვის“ ღილაკისათვის (button1) გვექნება შემდეგი კოდი:

```
private void Button1_Click(object sender, EventArgs e)
 {
   SqlConnection con = new SqlConnection("Data Source=DESKTOP-
       C56MU6F; Initial Catalog = SeaEco; Integrated Security = True");
   sda = new SqlDataAdapter(@"SELECT riverID,
                       river_Length, riverName,
                       estuarID from River", con);
   dt = new DataTable();
   sda.Fill(dt);
   dataGridView1.DataSource = dt;
 }
```

პროგრამის ამუშავებით, თუ მასში არაა შეცდომები, მიიღება 4.20 ნახაზი.



ნახ.4.20. „მონაცემთა ნახვის“ (DataShow) ღილაკის ამოქმედებით მიღებული შედეგი

„მონაცემთა განახლების“ ღილაკის კოდი (Insert, Update, Delete) ნაჩვენებია ქვემოთ, Button2-ის ლისტინგში:

```
// --- ლისტინგი --- Insert, Update, Delete for DataGridView_SQL----
```

```
private void Button2_Click(object sender, EventArgs e)
 {
   scb = new SqlCommandBuilder(sda);
   sda.Update(dt);
 }
```

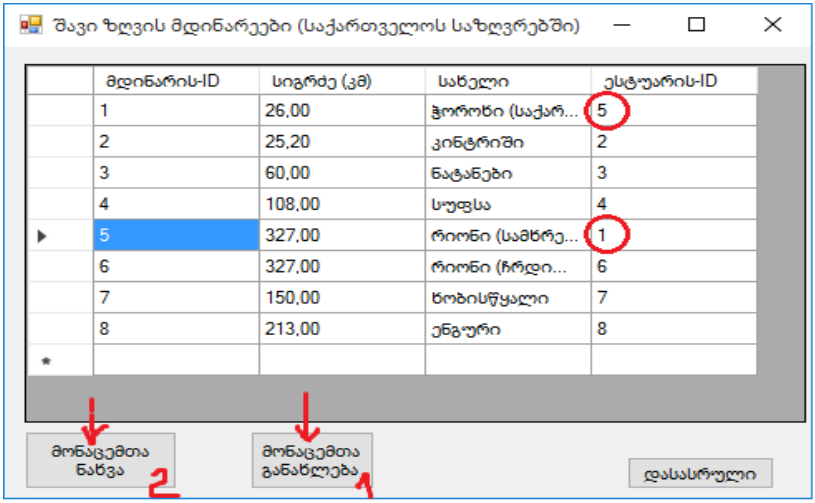
მთლიანი პროგრამის კოდი მოცემულია ლისტინგში.

// --- ლისტინგი 4.1 -- Insert, Update, Delete for DataGridView_SQL--

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient; // !!!
namespace WinFormSQL_DGV
{
  public partial class Form1 : Form
  {
    SqlDataAdapter sda;
    SqlCommandBuilder scb;
    DataTable dt;
    public Form1() { InitializeComponent(); }

  private void Form1_Load(object sender, EventArgs e)
  {
   this.riverTableAdapter.Fill(this.seaEcoDataSet.River);
  }
 // მონაცემთა ნახვა ---------
 private void Button1_Click(object sender, EventArgs e)
  {
   SqlConnection con = new SqlConnection("Data Source =
       DESKTOP-C56MU6F; Initial Catalog = SeaEco;
       Integrated Security = True");
  sda = new SqlDataAdapter(@"SELECT riverID, river_Length,
            riverName, estuarID from River", con);
```

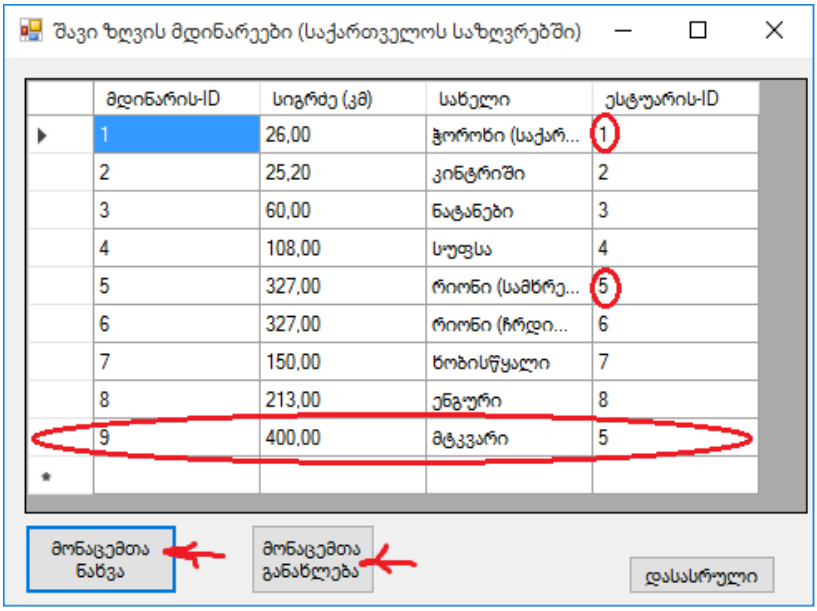
```
  dt = new DataTable();
  sda.Fill(dt);
  dataGridView1.DataSource = dt;
 }
// განახლება --------
private void Button2_Click(object sender, EventArgs e)
 {
   scb = new SqlCommandBuilder(sda);
   sda.Update(dt);
 }

private void Button3_Click(object sender, EventArgs e)
  { Close(); }
 }
}
```

4.21 ნახაზზე მოცემულია არსებულ ცხრილში ორი მნიშვნელობის (ესტუარის ID) შეცვლა (ახალი რიცხვები ნაჩვენებია წრეში), შემდეგ 1-ელი და მე-2 ღილაკების ამოქმედებით ბაზაში ჩაიწერება ეს შეცვლილი მნიშვნელობები (შეიძლება დავხუროთ პროგრამა და თავიდან ავამუშავოთ).



ნახ.4.21. Update ცვლილების განხორციელება

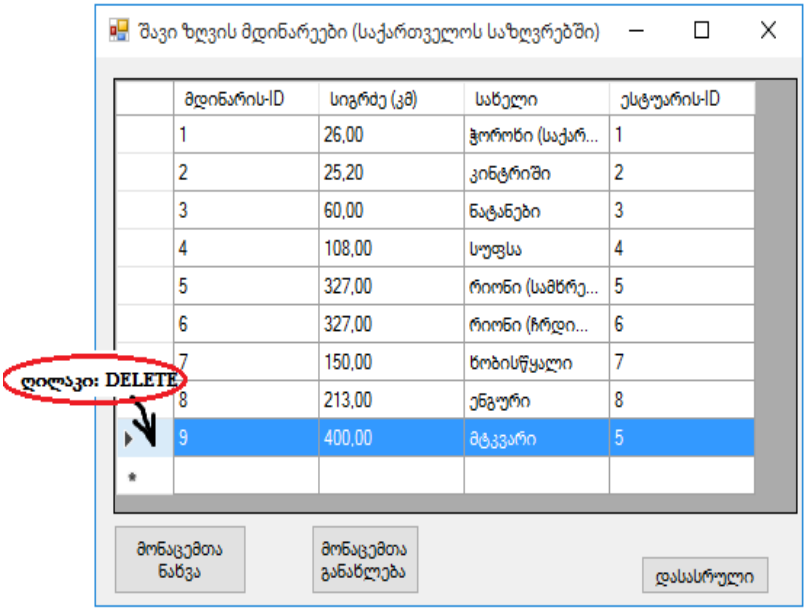
4.22 ნახაზზე ნაჩვენებია ახალი სტრქონის ჩამატების პროცედურა (Insert).



ნახ.4.22. Insert ოპერაციის შედეგი

ბოლოს ნაჩვენებია სტრიქონის წაშლის (Delete) ოპერაცია. იგი ხორციელდება მაგალითად, „**მდინარის-ID**“ შესაბამისი სტრიქონის მონიშვნით და შემდეგ კომპიუტერის კლავიატურის Delete-ღილაკის ამოქმედებით (ნახ.4.23).

ამით დავასრულეთ ვიზუალური C# ელემენტების საფუძველზე მონაცემთა ბაზასთან დაკავშირების რეალიზაციის ამოცანის განხილვა, რომლის შედეგადაც აგებულ იქნა შავი ზღვის ეკომონიტორინგის ინფორმაციული სისტემის ფრაგმენტი.



ნახ.4.23. Delete ოპერაციის განხორციელება

4.6. მომხმარებელთა Web-ინტერფეისების რეალიზაცია ASP.NET- პაკეტით, უსაფრთხოება და სერვისები

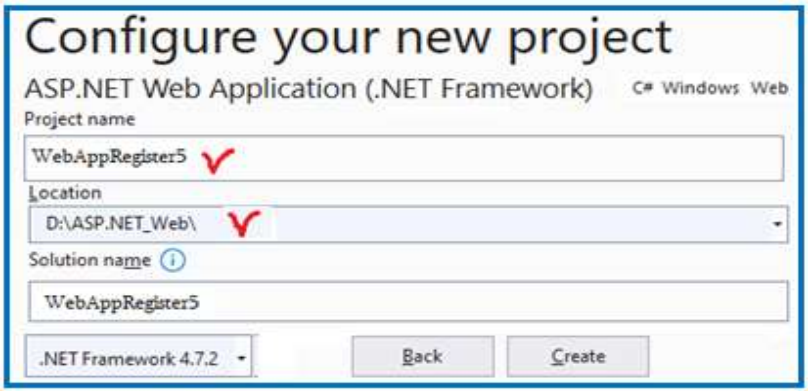
4.6.1. ინტერაქტიული Web-გვერდის აგება ASP.NET-ით

ამჯერად განვიხილოთ ამოცანა მომხმარებლის ინტერაქტიული ინტერფეისის დასაპროგრამებლად Web-აპლიკაციის სახით.

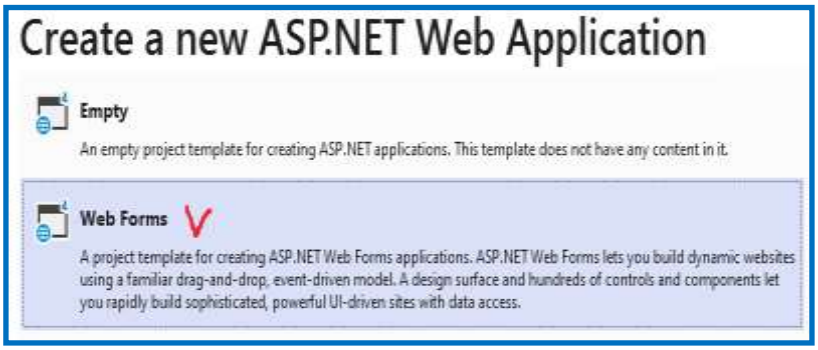
ამოცანა_2: Visual Studio .NET პლატფორმაზე ASP.NET-ის გამოყენებით შევქმნათ Web-გვერდი, რომელზეც მომხმარებელი შეიტანს საკუთარ მონაცემებს და გადააგზავნის სერვერზე [63,108].

– შევქმნათ ახალი ASP.NET აპლიკაცია პროექტის სახელით WebAppRegister5 (ნახ.4.24, 4.25-ა,ბ);

ნახ.4.24. ახალი ASP.NET პროექტის შექმნა .NET Framework პლატფორმაზე

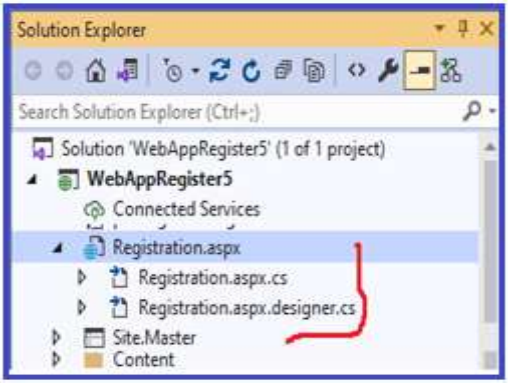


ნახ.4.25-ა



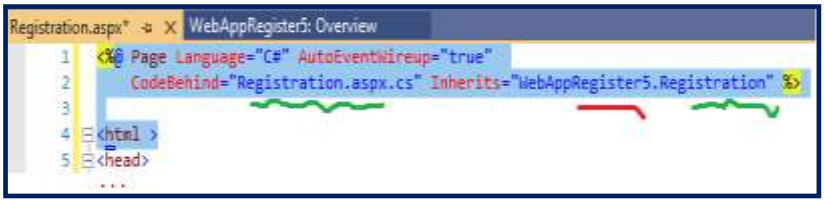
ნახ.4.25-ბ

– Solution Explorer-ში Default.aspx შევცვალოთ Registration-ით (ნახ.4.26).



ნახ.4.26

– Registration.aspx ფაილის 1-ელ სტრიქონს ექნება 4.27 ნახაზზე ნაჩვენები სახე, სადაც ასახულია პროექტის და .aspx და .aspx.cs პროგრამების სახელები;



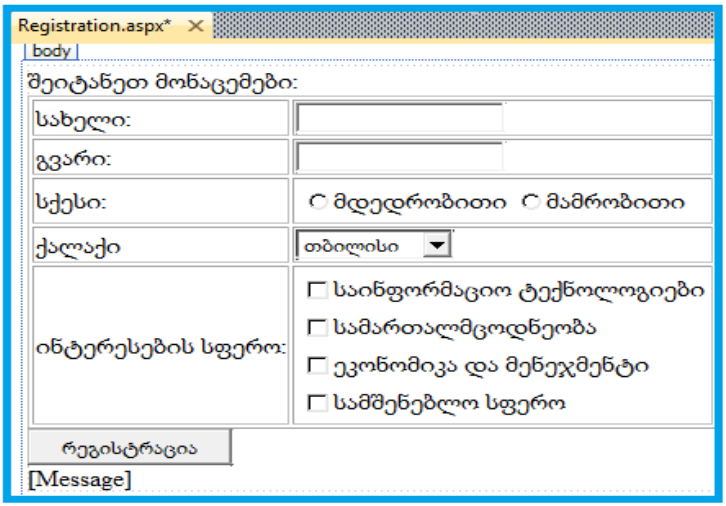
ნახ.4.27

– Registration.aspx.cs ფაილის საწყისი ტექსტის ლისტინგი ასე გამოიყურება;

```
// ----- ლისტინგი_4.2  Registration.aspx.cs.------
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
```

```
namespace WebAppRegister5
{
   public partial class Registration : Page
   {
      protected void Page_Load(object sender, EventArgs e) {  }
   }
}
```

– Web-გვერდის სარეგისტრაციო ფორმის მაკეტი ნაჩვენებია 4.28 ნახაზზე.



ნახ.4.28. სერვისის ინტერფეისი

ფორმაზე მოთავსებულია სერვერული მართვის ელემენტები form, asp:TextBox, asp:DropDownList, asp:CheckBoxList, asp:Button, asp:Label და ა.შ., რომლებიც ასახულია Registration.aspx ფაილის 4.3 ლისტინგში. გავხსნათ Registration.aspx და შევიტანოთ შემდეგი კოდი (ან გამოვიყენოთ წიგნიდან მისი შესაბამისი პროტოტიპი):

```
<!— ლისტინგი_4.3 ———— –>
    <%@ Page Language="C#" AutoEventWireup="true"
      CodeBehind="Registration.aspx.cs"Inherits="WebAppRegister8
```

```
      .Registrati on" %>
<html >
  <head>
   <title>რეგისტრაციის ფორმა</title>
    <style type="text/css">
     .auto-style1 {
        width: 253px;     }
   </style>
  </head>
<body>
  <form method="post" runat="server" id="registration">
  შეიტანეთ მონაცემები:
  <table border="1">
   <tr>
    <td>სახელი:</td>
    <td class="auto-style1">
     <asp:TextBox id="FirstName" runat="server"></asp:TextBox></td>
   </tr>
   <tr>
    <td>გვარი:</td>
    <td class="auto-style1">
     <asp:TextBox id="LastName" runat="server"></asp:TextBox></td>
   </tr>
   <tr>
    <td>სქესი:</td>
    <td class="auto-style1">
      <asp:RadioButtonList id="Sex"
                    runat="server" RepeatDirection ="Horizontal">
       <asp:ListItem Value="მდედრობითი"></asp:ListItem>
       <asp:ListItem Value="მამრობითი"></asp:ListItem>
      </asp:RadioButtonList></td>
   </tr>
   <tr>
    <td>ქალაქი</td>
    <td class="auto-style1">
```

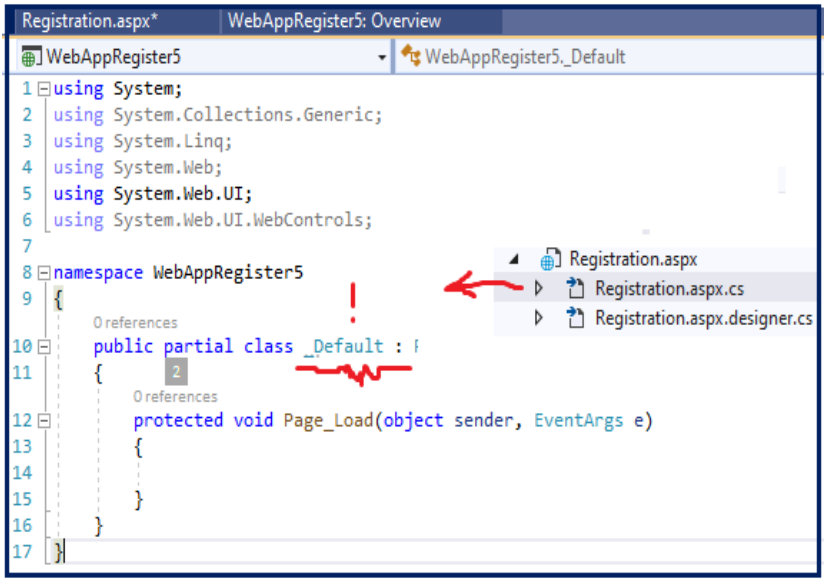
```
        <asp:DropDownList id="City" runat="server">
          <asp:ListItem Value="თბილისი"></asp:ListItem>
          <asp:ListItem Value="ქუთაისი"></asp:ListItem>
          <asp:ListItem Value="რუსთავი"></asp:ListItem>
          <asp:ListItem Value="გორი"></asp:ListItem>
          <asp:ListItem Value="ბათუმი"></asp:ListItem>
          <asp:ListItem Value="თელავი"></asp:ListItem>
        </asp:DropDownList>
      </td>
     </tr>
     <tr>
      <td>ინტერესების სფერო:</td>
      <td class="auto-style1">
       <asp:CheckBoxList id="Interests" runat="server">
         <asp:ListItem Value="საინფორმაციო ტექნოლოგიები">
         </asp:ListItem>
        <asp:ListItem Value="სამართალმცოდნეობა"></asp:ListItem>
        <asp:ListItem Value="ეკონომიკა და მენეჯმენტი">
        </asp:ListItem>
        <asp:ListItem Value="სამშენებლო სფერო"></asp:ListItem>
       </asp:CheckBoxList></td>

     </tr>
     </table>
        <asp:Button id="Register" runat="server" Text="რეგისტრაცია"
              OnClick="Register _Click">
        </asp:Button>
      </br>
      <asp:Label id="Message" runat="server"></asp:Label>
    </form>
   </body>
  </html>
```

– Web-გვერდზე „რეგისტრაცია“ Button-ის დაკლიკვით უნდა გადავიდეთ C# კოდში... მაგრამ ეს ვერ მოხერხდება, რადგანაც .aspx

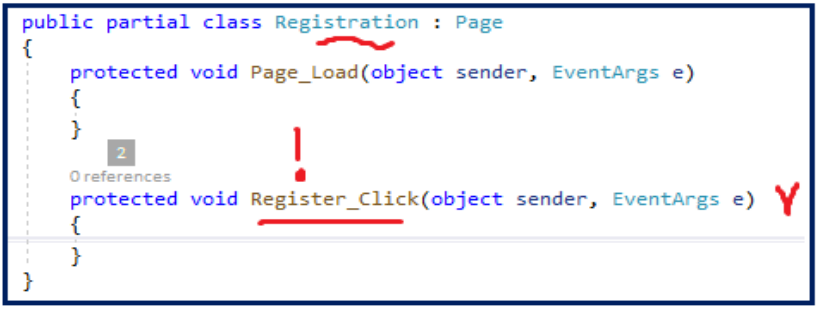
და .aspx.cs ფაილებს შორის კავშირი არაა გასწორებული...

კერძოდ Registration.aspx.cs გადასვლით (ნახ.4.29-ა,ბ) ჩანს, რომ მე-10 სტრიქონში დარჩენილია საწყისი კლასის სახელი:

_Default : Page.



ნახ.4.29-ა. _Default -ის შეცვლა Registration-ით



ნახ.4.29-ბ. შეცვლის შედეგი

Web-გვერდის ჩატვირთვის და მონაცემების შევსების შემდეგ „რეგისტრაცია“ Button-ის დაჭერისას გამოიძახება OnClick მოვლენაზე მიბმული მეთოდი Register_Click. ის აღიწერება C# კოდში, რომლის 4.4 ლისტინგი მოცემულია ქვემოთ.

გავხსნათ Registration.aspx.cs ფაილი და შევიტანოთ შემდეგი კოდი:

```
// ── ლისტინგი_4.4 ──────────
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
  namespace WebAppRegister5
  {
     public partial class Registration : Page
      {
       protected void Page_Load(object sender, EventArgs e)
        {
        }
       protected void Register_Click(object sender, EventArgs e)
       {
        System.Text.StringBuilder sb = new System.Text .StringBuilder();
             sb.Append("თქვენი გადაცემული მონაცემები:<br>");
             sb.AppendFormat("სახელი: {0}<br>", FirstName.Text);
             sb.AppendFormat("გვარი: {0}<br>", LastName.Text);
             sb.AppendFormat("სქესი: {0}<br>", Sex.SelectedValue);
             sb.AppendFormat("ქალაქი: {0}<br>", City.SelectedValue);
```

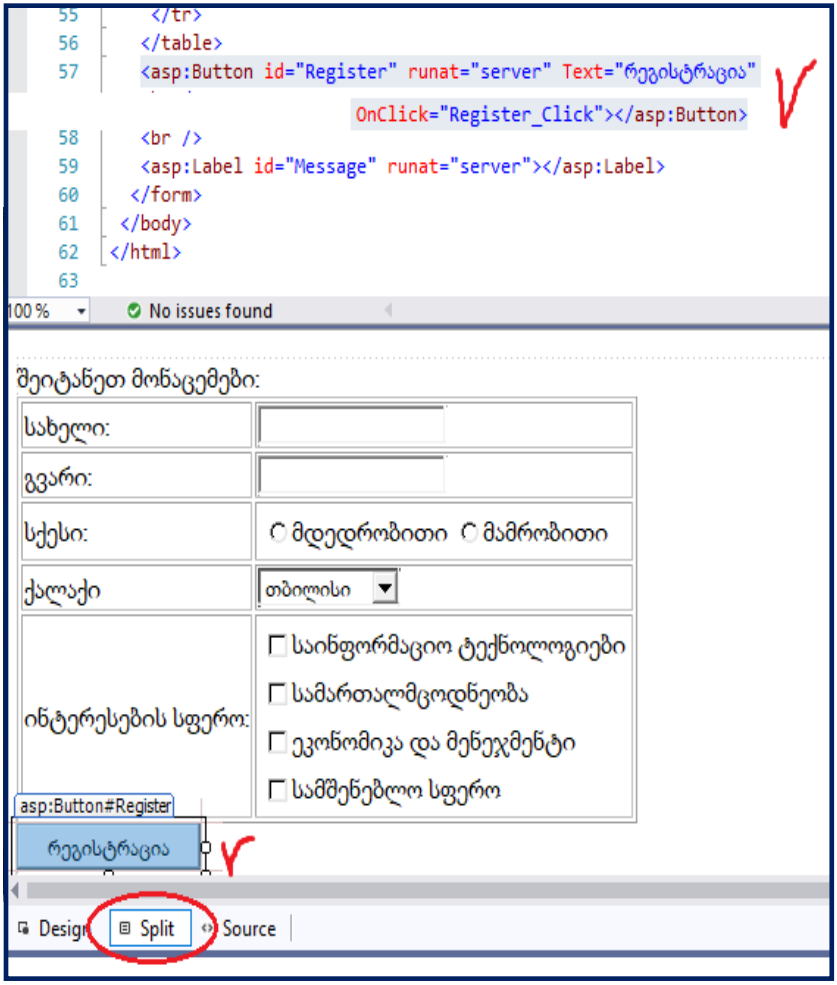
```
            sb.Append("ინტერესები: ");
        foreach (ListItem item in Interests.Items)
            {
               if (item.Selected)
                  sb.AppendFormat("{0}, ", item.Value);
            }
            sb.Append("<br>გმადლობთ რეგისტრაციისთვის");
            Message.Text = sb.ToString();
      }
    }
  }
```

ამის შემდეგ Registration.aspx დიზაინის „რეგისტრაცია“ ღილაკი ამუშავდება. Split რეჟიმში ჩანს ფორმის დიზაინიც და შესაბამისი .aspx პროგრამის ტექსტის ფრაგმენტიც (ნახ.4.30). აქ გამუქებულია <asp:Button id …>.

– „რეგისტრაცია“ ღილაკის ამოქმედებით სისტემა გადაგვიყვანს C# -ის კოდში, Registracion.aspx.cs-ის „Register_Click“ - მოვლენაზე;

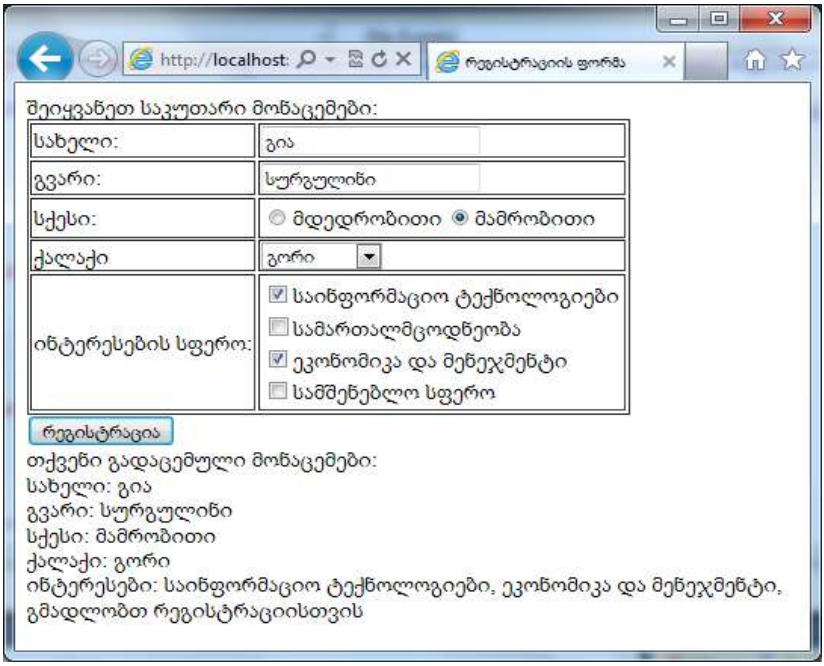
– ავამუშავოთ პროგრამა შესრულებაზე;

– Web-გვერდი, მომხმარებლის მიერ შეტანილი მონაცემებით, „რეგისტრაცია“ ღილაკზე დაკლიკვის შემდეგ, შეასრულებს C# კოდის Register_Click -ში ჩაწერილ ოპერაციებს.



ნახ.4.30

სინტაქსური და სისტემური შეცდომების არარსებობის შემთხვევაში შედეგებს ექნება 4.31 ნახაზზე მოცემული სახე.



ნახ.4.4.31. სწორი შედეგით დასრულება

4.6.2. Web-ის უსაფრთხოება ASP.NET-ში. სერვერის, კლიენტების, ფორმების და როლების აუთენტიფიკაცია

ვებ-აპლიკაციებში ხშირად საჭიროა მომხმარებელთა იდენტიფიკაცია და მათთვის სხვადასხვა რესურსებზე წვდომის უფლების განსაზღვრა. ASP.NET-ში არსებობს *აუთენტიფიკაციის* (ან ავტორიზაციის) რამდენიმე მეთოდი: Windows, Forms, Passport. ეს მოთოდები განისაზღვრება კონფიგურაციის ფაილში authentication ტეგის mode ატრიბუტის საშუალებით [108, 109].

```
<configuration>
        <system.web>
                <authentication />
```

```
        </system.web>
</configuration>
```

მეთოდები:

```
<authentication mode="Windows" />
<authentication mode="Forms" />
<authentication mode="Passport" />
<authentication mode="None" />
```

Web–აპლიკაციაზე მომხმარებლის წვდომის უფლებას განსაზღვრავს authentication ელემენტი, ხოლო აპლიკაციის გარკვეულ ნაწილებზე განისაზღვრება authorization ელემეტით: deny და allow ქვეელემენტების საშუალებით:

***** – *განსაზღვრავს ყველა მომხმარებელს,*

? – *ანონიმურ მომხმარებლებს.*

მაგალითად, ანონიმურ მომხმარებელთათვის საიტზე წვდომის უფლების გასათიშად ვებ-საიტის კონფიგურაციის ფაილში ჩაიწერება:

```
<configuration>
  <system.web>
  <authorization>
     <deny users="?" />
  </authorization>
 </system.web>
</configuration>
```

შესაძებელია ცალკეულ მომხმარებელზე და მომხმარებელთა ჯგუფებზე წვდომის უფლების მინიჭება. შემდეგი ჩანაწერი ნიშნავს, რომ წვდომის უფლება აქვთ someone და Admins ჯგუფში შემავალ მომხმარებლებს:

```
<authorization>
   <allow users="someone" />
   <allow roles="Admins" />
```

```
   <deny users="*" />
</authorization>
```

შესაძლებელია აგრეთვე ცალკეულ ვებ-გვერდებზე წვდომის უფლებების დაყენებაც:

```
<location path="webpage.aspx">
 <authorization>
   <allow roles="managers" />
   <deny users="?" />
 </authorization>
</location>
```

წინა ფრაგმენტი გვიჩვენებს, რომ webpage.aspx წვდომის უფლება აქვთ მხოლოდ managers ჯგუფის მომხმარებლებს.

Forms მეთოდით მომხმარებელთა ავტორიზაციის დროს საჭიროა მომხმარებლის სარეგისტრაციო გვერდის მითითება:

```
<configuration>
  <system.web>
  <authentication mode="Forms">
   <forms name=".ASPXCOOKIE" loginUrl="login.aspx" protection="All"
                  timeout="30" path="/"></forms>
  </authentication>
 </system.web>
</configuration>
```

ვებ საიტზე მომხმარებლების ავტორიზაციის მაგალითი:

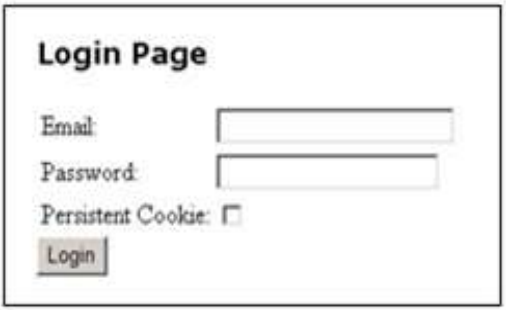
```
// - - - - - ლისტინგი 4.5 - - - Web.config  ფაილი - - - - -
<configuration>
  <system.web>
     <authentication mode="Forms">
      <forms name=".ASPXUSERDEMO" loginUrl="login.aspx"
protection="All" timeout="60" />
     </authentication>
```

```
    <authorization>
      <deny users="?" />
    </authorization>
   <globalization requestEncoding="UTF-8" responseEncoding="UTF-8"
/>
  </system.web>
</configuration>
      // ფაილი Default.aspx:
<%@ Import Namespace="System.Web.Security " %>
<html>
 <script language="C#" runat="server">
  void Page_Load(Object Src, EventArgs E )
   { Welcome.Text = "Hello, " + User.Identity.Name;
   }
  void Signout_Click(Object sender, EventArgs E)
   { FormsAuthentication.SignOut();
     Response.Redirect("login.aspx");
   }
 </script>
 <body>
  <h3><font face="Verdana">Using Cookie Authentication</font></h3>
  <form runat="server" ID="Form1">
   <h3><asp:label id="Welcome" runat="server" /></h3>
   <asp:button text="Signout" OnClick="Signout_Click" runat="server"
ID="Button1" NAME="Button1" />
  </form>
 </body>
</html>

// ფაილი Login.aspx:
<%@ Import Namespace="System.Web.Security " %>
<html>
 <script language="C#" runat="server">
```

```
    void Login_Click(Object sender, EventArgs E) {
//authenticate user: this samples accepts only one user with a
//name of someone@www.contoso.com and a password of 'password'
if((UserEmail.Value=="someone")&&(UserPass.Value=="password"))
 {
   FormsAuthentication.RedirectFromLoginPage(UserEmail.Value,
                    PersistCookie.Checked);
 }
else
 { Msg.Text = "Invalid Credentials: Please try again"; }
 }
 </script>
 <body>
   <form runat="server" ID="Form1">
  <h3><font face="Verdana">Login Page</font></h3>
  <table>
   <tr>
    <td>Email:</td>
    <td><input id="UserEmail" type="text" runat="server"
         NAME="UserEmail" /></td>
    <td
    </td>
   </tr>
   <tr>
    <td>Password:</td>
    <td><input id="UserPass" type="password" runat="server"
         NAME="UserPass" /></td>
    <td>
    </td>
   </tr>
   <tr>
    <td>Persistent Cookie:</td>
    <td><ASP:CheckBox id="PersistCookie" runat="server" />
    </td>
```

```
    <td></td>
   </tr>
  </table>
  <asp:button text="Login" OnClick="Login_Click"
        runat="server" ID="Button1" NAME="Button1" />
  <asp:Label id="Msg" ForeColor="red" Font-Name="Verdana"
        Font-Size="10" runat="server" />
 </form>
</body>
</html>
```

შედეგი ნაჩვენებია 4.32 ნახაზზე

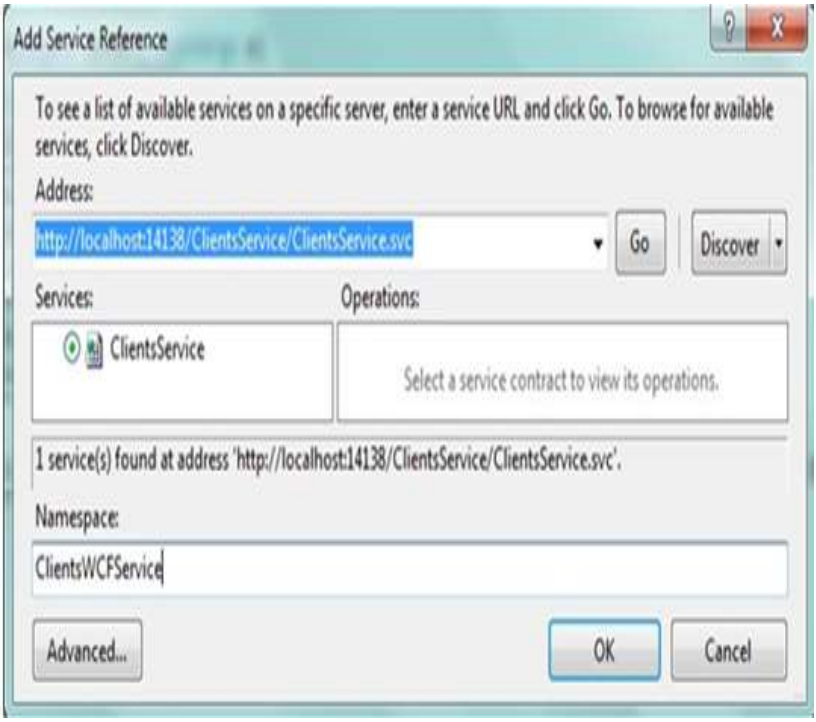


ნახ.4.32

4.6.3. Web-სისტემის მომხმარებელთა ინტერფეისების აგება (სერვისების რეალიზაცია)

ვებ ინტერეფისის ფორმა შექმნილია ASP.NET აპლიკაციაში. იგი შედგება ვებ-გვერდებისა და კოდის ფაილებისაგან. ეს აპლიკაცია მონაცემების ბაზაში წვდომისათვის გამოიყენებს ვებ-სერვისებს.

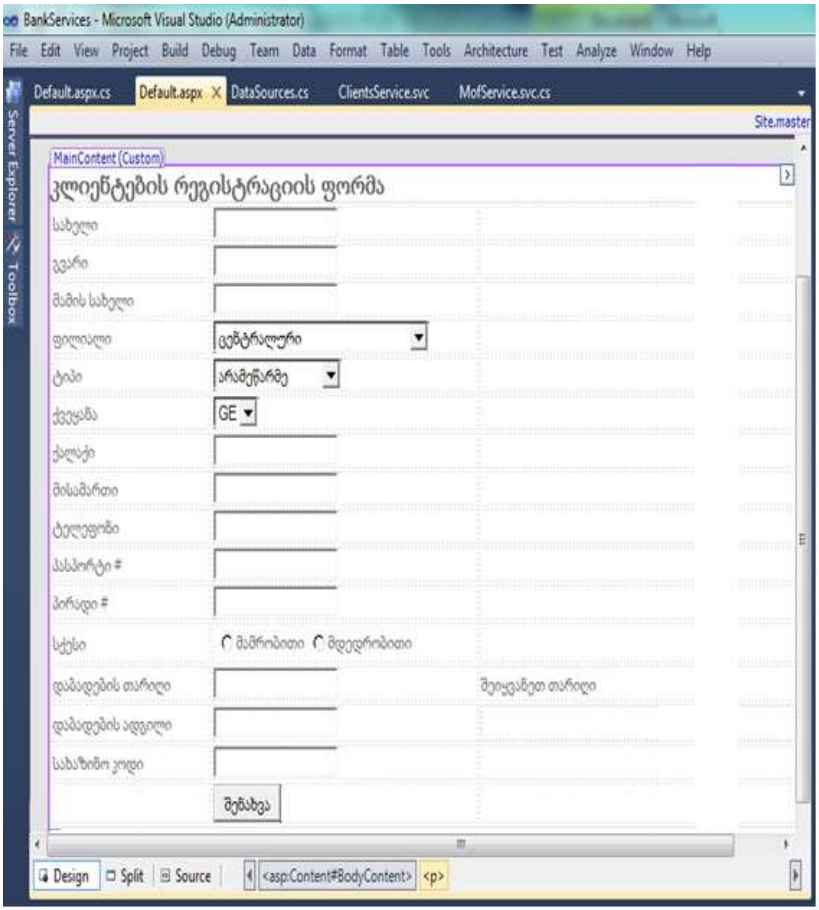
ვებ სერვისები დამატებულია Add Service Reference დიალოგური ფანჯრის საშალებით. ეს ფანჯარა მოცემულია 4.33 ნახაზზე.



ნახ.4.33. Add Service Reference დიალოგური ფანჯარა

ამ ფანჯარაში გაიწერება ვებ-სერვისი URL. იქმნება სპეციალური სახელების სივრცე (Namespace) და ხდება ვებ-სერვისთან დასაკავშრებელი კლასების გენერაცია.

ახალი კლიენტების რეგისტრაციის ფორმა, რომელიც გამოიყენება მათი რეგისტრაციისათვის სისტემაში, მონაცემების ბაზაში შესანახად, მოცემულია 4.34 ნახაზზე.



ნახ.4.34. მომხმარებლის ინტერფერისი ASP.NET გარემოში

Web-გვერდი შედგება HTML და კოდის ფაილებისგან. კოდში გამოყენებულია C# ენა. მომხმარებლის მონაცემების შეტანის შემდეგ „შენახვა“ ღილაკზე დაჭერით მოხდება მონაცემების გადაგზავნა სერვერზე და AddClient_Click პროცედურის გამოძახება. ქვემოთ, მოცემულია ღილაკის Click მეთოდის ლისტინგი 4.6.

// - - - - - ლისტინგი 4.6 - - - -

```
protected void AddClient_Click(object sender, EventArgs e)
 {
   ClientsWCFService.ClientsServiceClient service = new
                                ClientsWCFService.ClientsServiceClient();
   string firstName = txtFirstName.Text;
   string lastName = txtLastName.Text;
   string fatherName = txtFatherName.Text;
   int brachID = Convert.ToInt32(ddlBranch.SelectedValue);
   byte type = Convert.ToByte(ddlType.SelectedValue);
   string country = ddlCountry.SelectedValue;
   string city = txtCity.Text;
   string address = txtAddress.Text;
   string phone = txtPhone.Text;
   string passport = txtPassport.Text;
   string personalID = txtPersonalID.Text;
   bool gender =  rdlGender.SelectedValue=="0"?false:true;
   DateTime birtDate = DateTime.Parse(txtBirtDate.Text);
   string birtPlace = txtBirtPlace.Text;
   string taxCode = txtTaxCode.Text;
   try
   {
     int clientNo = service.OpenClient(firstName, lastName,
           fatherName, brachID, type, gender, passport, personalID,
           country, city, address, phone, birtPlace, birtDate, taxCode);
    if (clientNo != -1)
      Response.Redirect(String.Format("~/Result.aspx?clientId={0}",
                        clientNo));
   }
   catch (Exception ex)
    {
       throw ex;
     }
   }
```

ამ პროცედურაში გამოიძახება ვებ-სერვისი ClientsWCFService და მისი მეთოდი OpenClient. ამ მეთოდს გადაეცემა ახალი მომხმარებლის მონაცემები. ვებსერვისი კი უზრუნველყოფს ამ მონაცემების ასახვას ბაზაში.

4.7. ღრუბლოვანი მონაცემთა ბაზები და საცავები

ღრუბლოვანი მონაცემთა საცავი (Cloud data warehouse - CDWH) არის მონაცემთა (ბაზების) ერთობლიობა, რომელიც ინახება მართვადი სერვისის სახით საერთო გამოყენების ღრუბელში და ოპტიმიზებულია მასშტაბირებადი ბიზნესანალიზისათვის (BI) [44].

ბოლო წლებში განსაკუთრებით მოიმატა კორპორაციებში ინფორმაციული მოცულობის დინამიკურმა ზრდამ, რამაც ბევრი მათგანი დააყენა პრობლემის წინაშე – ვეღარ წყვეტდნენ ფიზიკური რესურსების (მონაცემთა ცენტრების) ექსტენსიონალურ გაფართოებას (საკმაოდ დიდი ფინანსური ხარჯებით).

დროული გადაწყვეტა იყო ღრუბლოვანი მონაცემთა საცავები (არქიტექტურები), რამაც ხელი შეუწყო, მეტწილად, მცირე და საშუალო ბიზნესის ობიექტებს შეზღუდული დანახარჯებით გაეგრძელებინათ დიდი მონაცემების შენახვა და ანალიზი. შესაძლებელი გახდა ინფორმაციის განსხვავებული წყაროებიდან, როგორიცაა ERP, CRM, IoT და სხვ. მონაცემების უწყვეტი მიღება.

მონაცემთა ღრუბლოვანი საცავების ფუნქციონირება (ინფორმაციის შეტანა/გამოტანა და დამუშავება) ხორციელდება მრავალი სერვერის პარალელური მუშაობის რეჟიმში, მათი დატვირთვების გადანაწილებით და მართვით.

მონაცემთა სვეტოვანი სანახები (Columnar data stores) ყველაზე მოქნილი და ეკონომიურია ანალიტიკისთვის. ასეთი მონაცემთა ბაზები ინახავს და ამუშავებს მონაცემებს სვეტების მიხედვით, ნაცვლად სტრიქონებისა. იგი ახორციელებს მთლიანი მოთხოვნების მკვეთრად უფრო სწრაფად მუშაობს და ამიტომაც ხშირად გამოიყენება ანგარიშგებისთვის.

ამგვარად, ღრუბლოვანი არქიტექტურა აერთიანებს სამ ძირითად ელემენტს: *მონაცემთა შენახვის სიმძლავრეს, დიდ მონაცემთა პლატფორმის მოქნილობას და ღრუბლის ელასტიურობას.* ეს საკითხები და რომელი ვარიანტი იქნას არჩეული ჩვენი კონკრეტული დასაპროექტებელი სისტემისთვის, დეტალურადაა გადმოცემული აღნიშნულ ნაშრომში [44].

აქ ერთ-ერთი მნიშვნელოვანი საკითხია მონაცემთა ღრუბლოვანი საცავების ავტომატიზაცია (Cloud Data Warehouse Automation). მონაცემთა ინტეგრაციის ზოგიერთ თანამედროვე პლატფორმაში ავტომატიზიებულია მონაცემთა საცავის მთელი სასიცოცხლო ციკლი, რათა დაჩქარდეს ანალიტიკის პროცესისთვის მზა მონაცემების წვდომადობა. მოდელზე ორიენტირებული მიდგომა ეხმარება მონაცემთა ბაზების და საცავების ინჟინრებს შექმნან, განათავსონ და მართონ ღრუბლოვანი მონაცემთა საცავები.

მომდევნო პარაგრაფებში დეტალურად განვიხილავთ SQL და NoSQL ტიპის ბაზების გამოყენების საკითხებს ღრუბლოვან საცავებში.

4.7.1. Microsoft Azure SQL

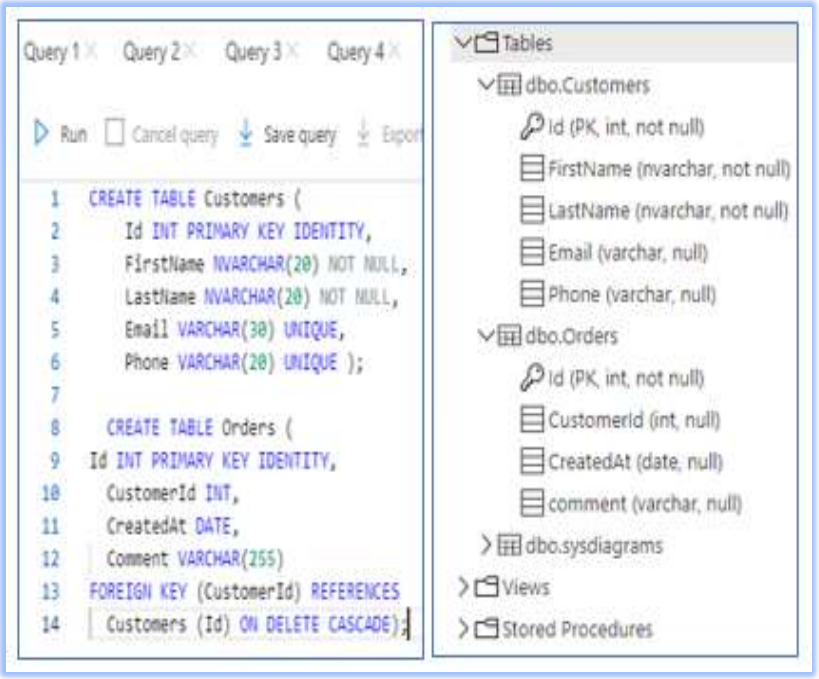
Ms Azure SQL მართული ღრუბლოვანი მონაცემთა ბაზაა. იგი მუშაობს ღრუბლოვან გამოთვლით პლატფორმაზე და მასზე წვდომა უზრუნველყოფილია როგორც სერვისი. მართული მონაცემთა ბაზის სერვისები ზრუნავს მონაცემთა ბაზის მასშტაბურობაზე, სარეზერვო და მაღალ წვდომადობაზე.

პროგრამული აპლიკაცია თუ front_end-ია, Azure SQL არის back_end. იგი გამოიყენება OLTP-ში (ტრანზაქციების ონლაინ-დამუშავება) დროის რეალურ რეჟიმისთვის. Azure SQL *მონაცემთა საცავი* კი ოპტიმიზებულია მონაცემთა ანალიზის ამოცანების შესასრულებლად, დიდ მონაცემთა დასამუშავებლად.

Azure პორტალზე ავირჩიოთ ბრძანება SQL Database →Create და შევავსოთ დიალოგური ფანჯარა (ნახ..4.35) [45.46].

ნახ.4.35. მონაცემთა ბაზის შექმნა Azure SQL-ში

მაგალითისთვის, შევქმნათ ცხრილები Customers და Orders. რომელთა ილუსტრაავია 4.36 ნახაზზეა მოცემული.



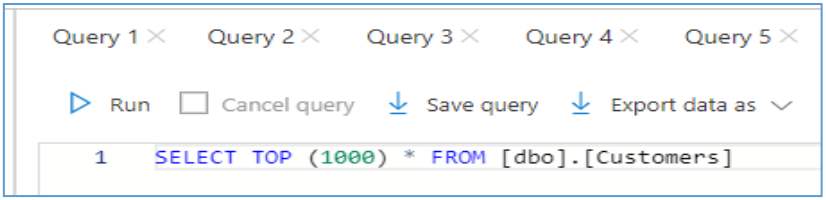
ნახ.4.36. ცხრილების დაპროექტება Azure SQL-ში

ბაზის Customers ცხრილი შევავსოთ ჩანაწერებით:

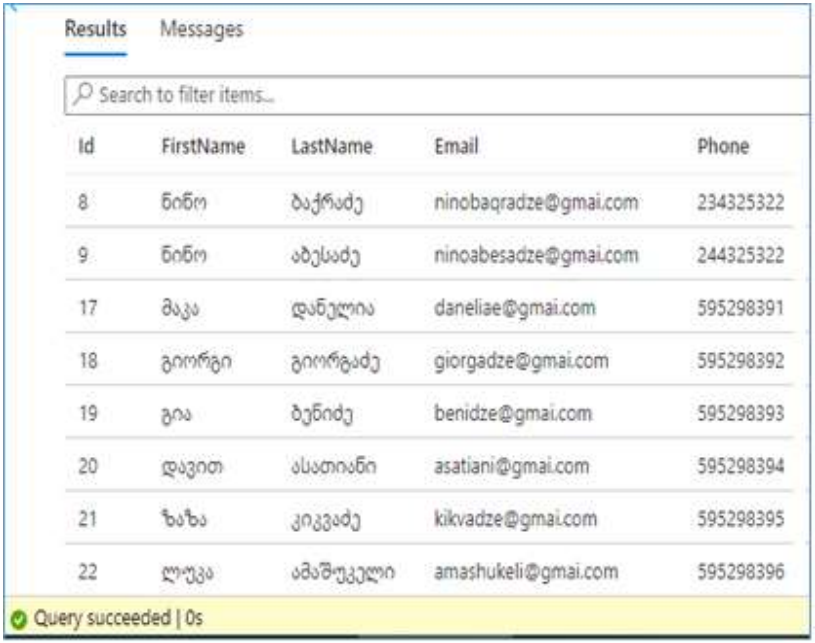
```
insert into [dbo].[Customers] values (N'მაკა', N'დანელია', 'daneliae@g
                                    mail.com', 595298391)
insert into [dbo].[Customers] values (N'გიორგი', N'გიორგაძე',
                                    'giorgadze@gmail.com', 595298392)
insert into [dbo].[Customers] values (N'გია', N'ბენიძე',
                                    'benidze@gmail.com', 595298393)
insert into [dbo].[Customers] values (N'დავით', N'ასათიანი',
                                    'asatiani@gmail.com', 595298394)
```

```
insert into [dbo].[Customers] values (N'ზაზა', N'კიკვაძე',
                                      'kikvadze@gmail.com', 595298395)
insert into [dbo].[Customers] values (N'ლუკა', N'ამაშუკელი',
                                      'amashukeli@gmail.com', 595298396)
```

შედეგები ნაჩვენებია 4.37-ა,ბ ნახაზებზე.



ნახ.4.37-ა



ნახ.4.37-ბ

Office 365-ის ერთ-ერთი აპლიკაციაა *Power Automate.* იგი ღრუბლოვანი სერვისია, რომლის საშუალებით მომხმარებელს შეუძლია დამოუკიდებლად შექმნას სამუშაო პროცესები, განახორციელოს განმეორებადი, ხელით შესასრულებელი სამუშაოების ავტომატიზაცია. ეს კი ამარტივებს ბიზნეს პროცესებს და მათ ეფექტურად მართვას [47].

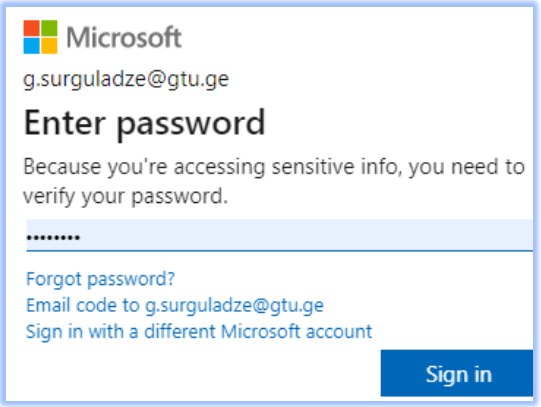
Power Automate დაფუძნებულია ტრიგერებზე (triggers) და მოქმედებებზე (actions). ტრიგერის საშუალებით იწყება ნაკადი ანუ ვირჩევთ მოვლენას, რომლის მიხედვითაც იგეგმება მოქმედება, ანუ ის, რაც ხდება ნაკადის გააქტიურების შემდეგ. ეს შეიძლება იყოს დავალების შექმნა ან ერთი ან მეტი მოქმედების შესრულება.

არსებობს *ხუთი ტიპის ნაკადის (flow)* შექმნის საშუალება:

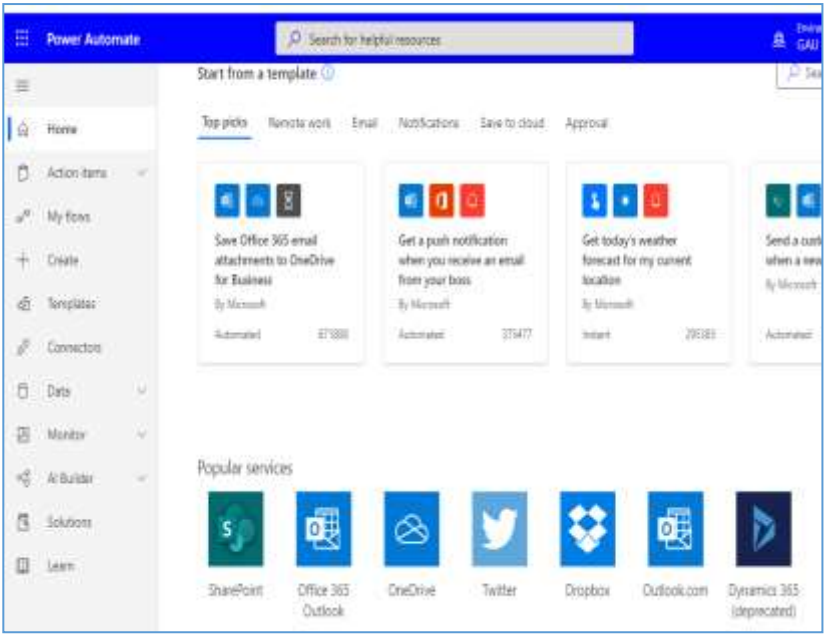
- *ავტომატიზებული* – მოვლენის შედეგად გამოწვეული ნაკადი, მაგალითად, გაიგზავნოს მეილი, თუ SharePoint სიის ელემენტი შეიცვალა;
- *მყისიერი* – მომხმარებლებს საშუალებას აძლევს ხელით იმოქმედონ მობილურიდან ან აპლიკაციიდან ღილაკის დაჭერით. მაგალითისთვის, მარტივად გაგზავნონ შეხსენების მეილი ორგანიზაციის წევრებთან შეხვედრის დაწყებამდე;
- *დაგეგმილი* – იწყებს მუშაობას გარკვეულ დროს;
- *ბიზნეს პროცესების ნაკადები,* რომელიც ემყარება განსაზღვრულ მოქმედებათა ერთობლიობას;
- *UI (user interface) ნაკადები,* რომლებიც გამოიყენება Windows და Web პროგრამებში განმეორებადი ამოცანების ავტომატიზაციისათვის.

➢ **ბიჯი 1:** *Power Automate გააქტიურება*

რომელიმე ინტერნეტ-ბრაუზერში მივცეთ: **portal.office.com** (ნახ.4.38), შევიტანოთ მომხმარებლის სახელი და პაროლი. გააქტიურდება Office 365 მთავარი გვერდი, ავირჩიოთ **Power Automate** (ნახ.4.39).



ნახ.4.38

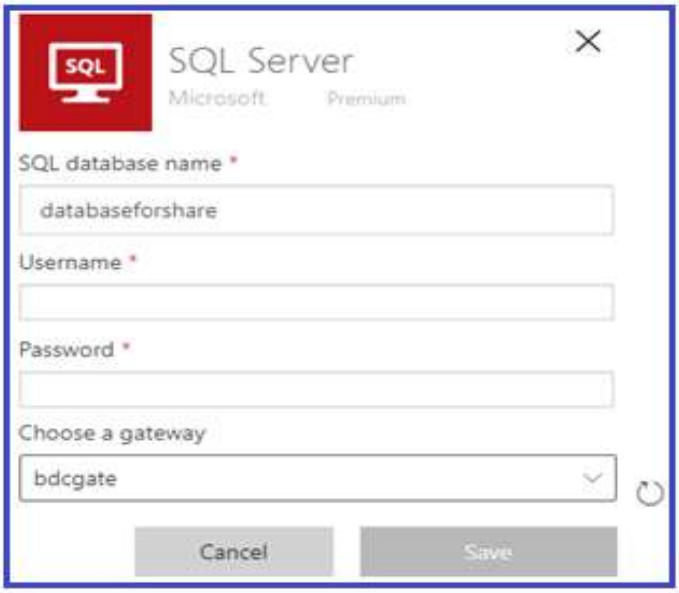


ნახ.4.39

➢ **ბიჯი 2:** *ეკრანზე გამოჩნდება Power Automate-ის შაბლონები. ძებნის ველში ვუთითებთ Sharepoint. მიიღება* ფანჯარა (ნახ.4.40).

ნახ.4.40

შევავსოთ 4.41 ნახაზზე ნაჩვენები ფანჯრის საჭირო ველები,



ნახ.4.41

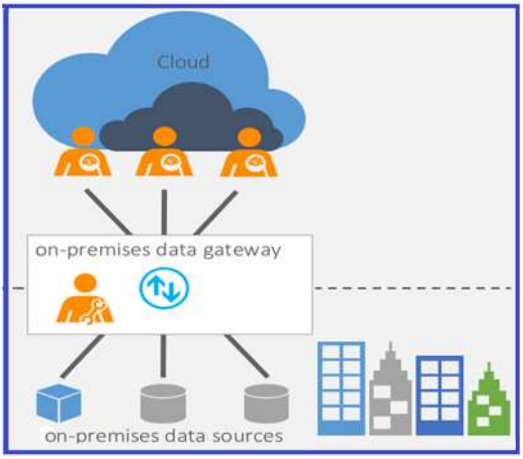
მაიკროსოფთის საიტიდან ჩამოვტვირთოთ და დავაინსტა-ლიროთ Gateway.

on-premises data gateway მოქმედებს როგორც ხიდი, რათა უზრუნველყოს მონაცემთა სწრაფი და უსაფრთხო გადაცემა შიგა მონაცემებს შორის (მონაცემები, რომლებიც არ არის ღრუბელში) და Microsoft-ის სხადასხვა ღრუბლოვან სერვისს შორის.

ეს ღრუბლოვანი სერვისები მოიცავს (ნახ.4.42):

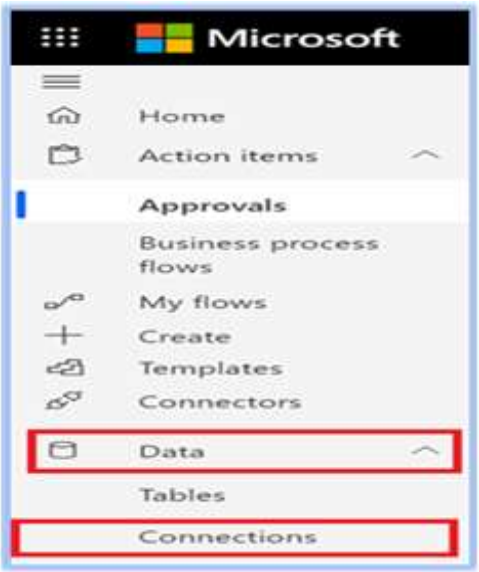
- Power BI, Power Apps;
- Power Automate;
- Azure Analysis Services და
- Azure Logic Apps.

გამოვიყენოთ Power Automate სერვისი სამუშაო პროცესების ავტომატიზაციის მიზნით. კერძოდ, მოვახდინოთ Sharepoint-ის სიაში შეტანილი მონაცემთა სინქრონულად გადატანა Azure SQL-ში.



ნახ.4.42

დავამატოთ ახალი კავშირი (ნახ.4.43).



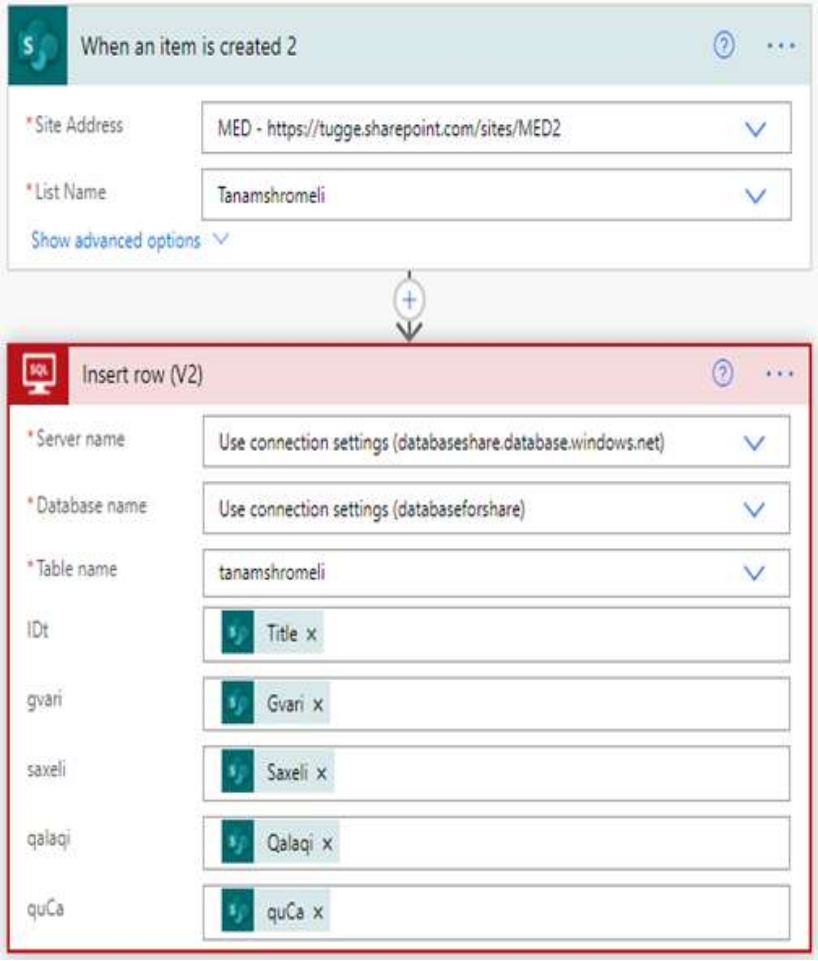
ნახ.4.43

1) Power Automate-ის საშუალებით გავიაროთ ავტორიზაცია;
2) ავირჩიოთ

Data -> Connections -> New connection;

3) ავირჩიოთ **SharePoint**

მოვახდინოთ სამუშაო ნაკადის კონფიგურაცია. 4.44 ნახაზზე ასახული ნაკადი დეტალურად აღწერს Azure SQL-ის tanamshromeli-ის ცხრილის ველების შევსების პროცესს.


When an item is created 2
*Site Address
MED - https://tugge.sharepoint.com/sites/MED2
*List Name
Tanamshromeli
Show advanced options
Insert row (V2)
*Server name
Use connection settings (databaseshare.database.windows.net)
*Database name
Use connection settings (databaseforshare)
*Table name
tanamshromeli
IDt
Title
gvari
Gvari
saxeli
Saxeli
qalaqi
Qalaqi
quCa
quCa

ნახ.4.44

4.7.2. ღრუბლოვანი NoSQL ბაზა MongoDB Atlas

MongoDB Atlas პირველი ღრუბლოვანი ვერსიაა ამ ტიპის მონაცემთა ბაზებიდან. იგი მომხმარებელს აძლევს საშუალებას წარადგინოს ერთდროულად რამდენიმე აპლიკაცია მსხვილ ღრუბლოვან პროვაიდერებთან. MongoDB-ს მონაცემთა ბაზა შესაძლებელია განთავსდეს ისეთ ღრუბლოვან სერვერებზე, როგორიცაა AWS (Amazon Web Services), Azure ან GCP (Google Cloud Platform) [46,48,49].

ღრუბლოვანი MongoDB Atlas, კლასტერიზაციის საშუალებით დეველოპერებს სთავაზობს ისარგებლონ, ყოველი პროვაიდერის მიერ წარმოდგენილი საუკეთესო აპლიკაციით, რაც უმარტივებთ და ეფექტურს ხდის მათ მუშაობას.

მულტიღრუბლოვანი კლასტერები კომპანიებს სთავაზობს მონაცემთა მიწოდების უპრეცედენტოდ მოქნილ და ეფექტურ სერვისს [50]. ღრუბლოვან პროვაიდერებს შორის მონაცემთა გაცვლა და ბიზნეს მოთხოვნების დაკმაყოფილება საკმაოდ მნიშვნელოვანი და ძვირად ღირებული პროცესია, რასაც MongoDB Atlas პლატფორმა ეფექტურად ართმევს თავს.

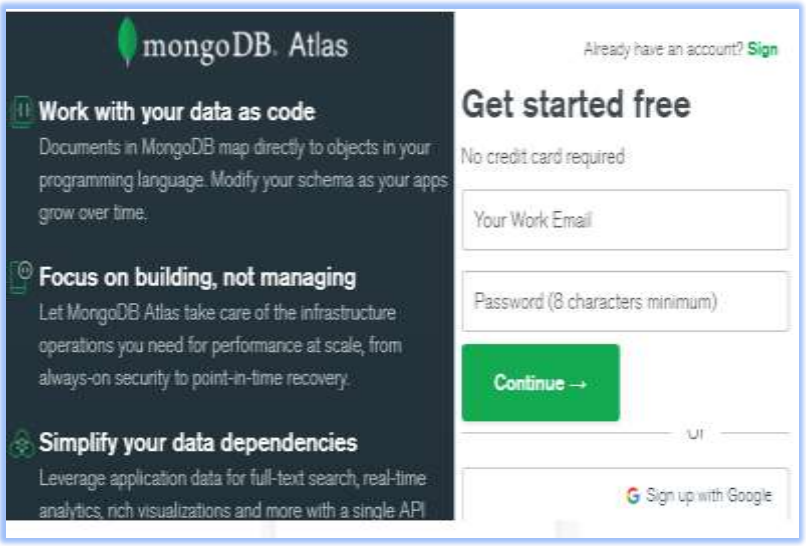
MongoDB Atlas – ეფექტურად გამოიყენება დანართებთან / აპლიკაციებთან სამუშაოდ. მისი დახმარებით მონაცემთა ბაზები იქმნება ძალიან სწრაფად, მათი დამუშავების და მართვისთვის საჭიროა მცირე დრო და, რაც მთავარია, დაცულია უსაფრთხოების მაღალი სტანდარტით.

MongoDB Atlas პლატფორმა არის ე.წ. MongoDB მონაცემთა ბაზის სერვისი, რაც იმას ნიშნავს, რომ სერვისი ავტომატურად აკონფიგურირებს მონაცემთა ბაზას და განათავსებს მას როგორც უნიკალურს. აღნიშნული სერვისი მომხმარებელს ათავისუფლებს

NoSQL მონაცემთა ბაზის დაპროექტების და მართვისთვის საჭირო სამუშაოების შესრულებისგან და საშუალებას აძლევს კონცენტრირდეს უშუალოდ აპლიკაციის მართვაზე.

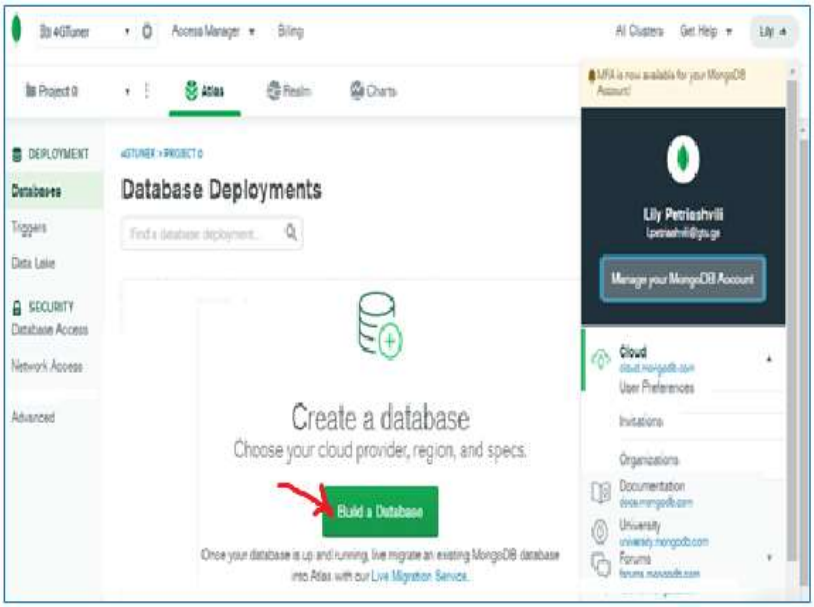
MongoDB Atlas პლატფორმის გამოყენებისთვის მომხმარებელი პირველ რიგში უნდა დარეგისტრირდეს თავისი ელ_ფოსტის მისამართით და შეავსოს სარეგისტრაციო ველები (ნახ.4.45).

(მისამართი: https://www.mongodb.com/cloud/atlas).



ნახ.4.45

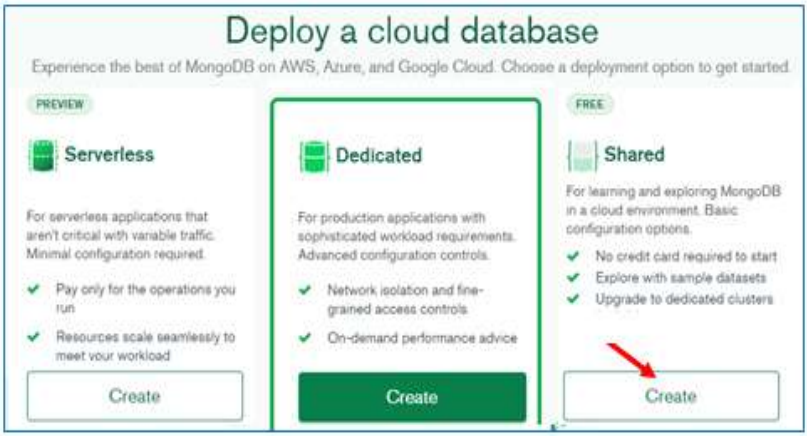
რეგისტრაციის შემდეგ, როგორც 4.46 ნახაზზეა ნაჩვენები, მიიღება ფანჯარა საიდანაც ხდება მონაცემთა ბაზის აგება



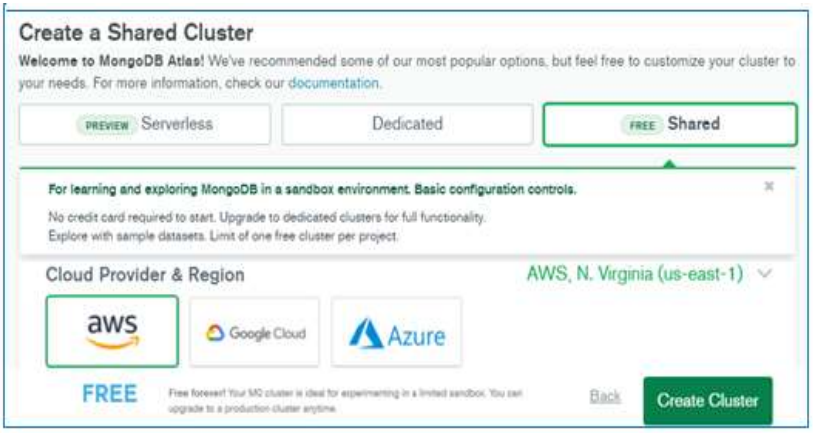
ნახ.4.46

Build a Database ღილაკით გადავდივართ გარემოში, სადაც შესაძლებელია შევარჩიოთ პლატფორმის გამოყენების სხვადასხვა სერვისი. ჩვენ შემთხვევაში ვაკეთებთ თავისუფალ არჩევანს, რომელიც განკუთვნილია არაკომერციული სასწავლო პროცესისთვის და ნიმუშის შესაბამისად არჩევანს ვადასტურებთ ღილაკით Create (ნახ.4.47).

შემდეგ გადავდივართ კლასტერის შექმნის და ფორმირების ეტაპზე. მიღებულ ფანჯარას აქვს შემდეგი სახე (ნახ.4.48).

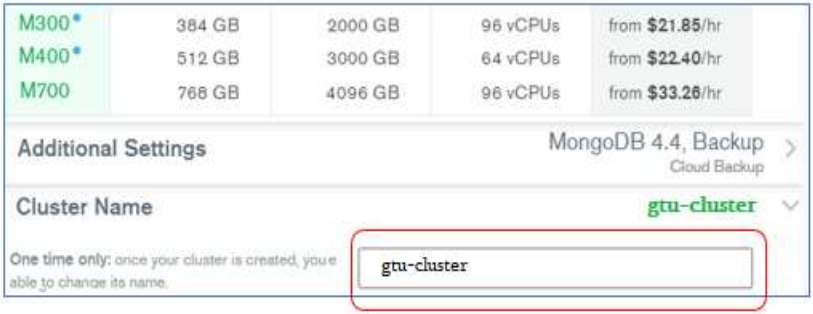


ნახ.4.47



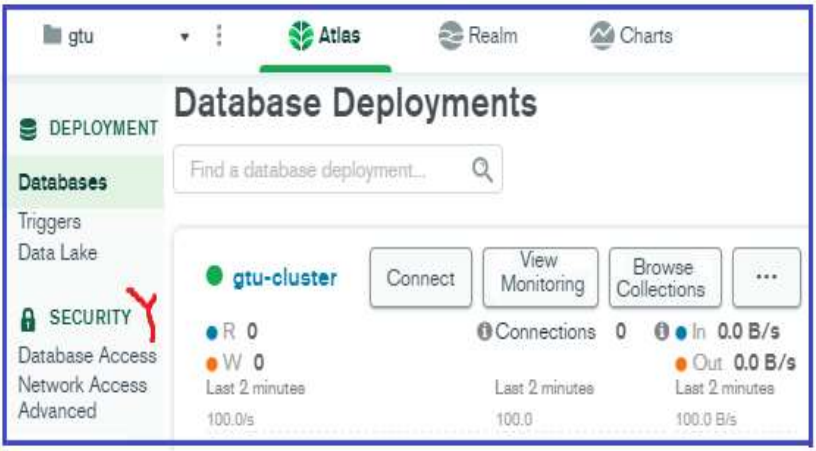
ნახ.4.48

ვირჩევთ ღრუბლოვან პროვაიდერს (AWS, Google ან Azur). ჩვენ ავირჩიეთ AWS. შემდეგ ფანჯრის ბოლოს ველში Cluster Name ჩავწეროთ კლასტერის დასახელება, მაგ., gtu-claster (უნდა გამოვიყენოთ ქვედა რეგისტრის სიმბოლოები და რიცხვები, „ჰარის“ გარეშე) (ნახ.4.49).



ნახ.4.49

Create Claster ღილაკით გადავდივართ ფანჯარაში, სადაც გამოჩნდება ჩვენს მიერ ფორმირებული კლასტერი, სახელით „gtu-cluster" (ნახ.4.50).



ნახ.4.50

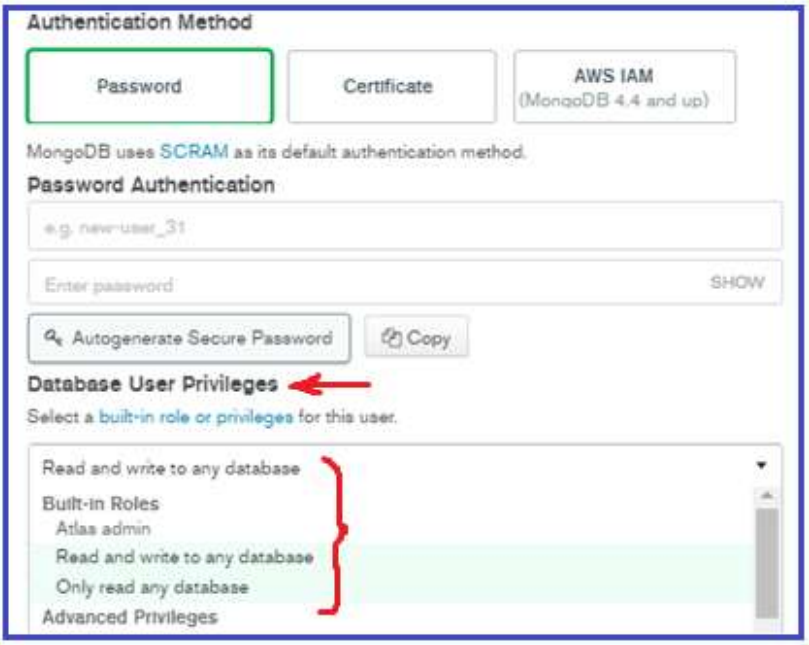
კლასტერის შექმნის შემდეგ ფანჯრის მარცხენა მხარეს მოთავსებული **SECURITY** – **Database Access** ჩანართიდან უნდა განისაზღვროს სისტემის მომხმარებელთა ჩამონათვალი, ვისაც შეეძლებათ ინფორმაციის მიღება და ჩაწერა. აღნიშნული ჩანართის

გააქტიურებისას ქვემოთ მოცემულ ფანჯარაში ვამატებთ Database Users – მომხმარებლებს (ნახ.4.51).

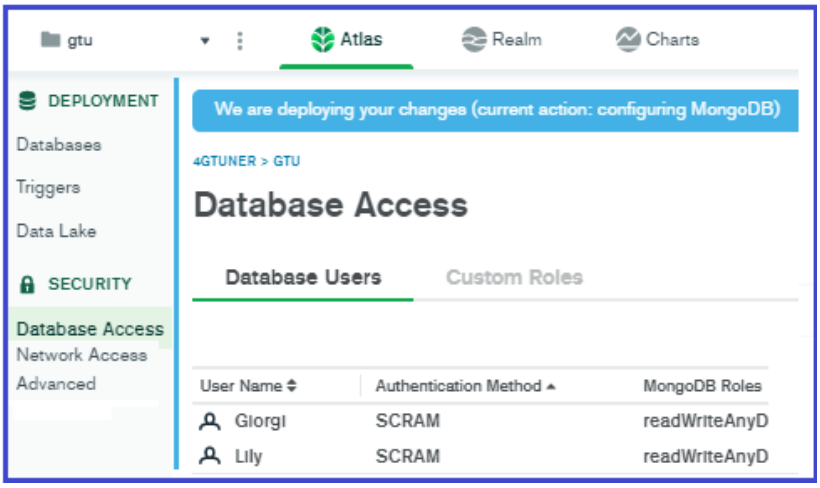
ნახ.4.51

ახალი მომხმარებლების შესახებ მონაცემთა განსაზღვრისას უნდა გავააქტიუროთ ფანჯრის ქვედა მარჯვენა ნაწილში მოთავსებული ღილაკი Add New Database User. შედეგად მიიღება ფანჯარა, რომლის ველებშიც შეგვაქვს მომხმარებლის სახელი და პაროლი (ნახ.4.52).

ამ ფანჯრის ჩანართიდან **Database User Privileges** ყოველ მომხმარებელს შესაძლებელია მივანიჭოთ ან შევუზღუდოთ უფლებები, აღნიშნულ ჩანართში მოცემული შეთავაზებების შესაბამისად. მაგალითად, 4.53 ნახაზზე წარმოდგენილია ორი მომხმარებელი, რომელთაც წვდომა ექნება ჩვენს ღრუბლოვან მონაცემთა ბაზაზე.

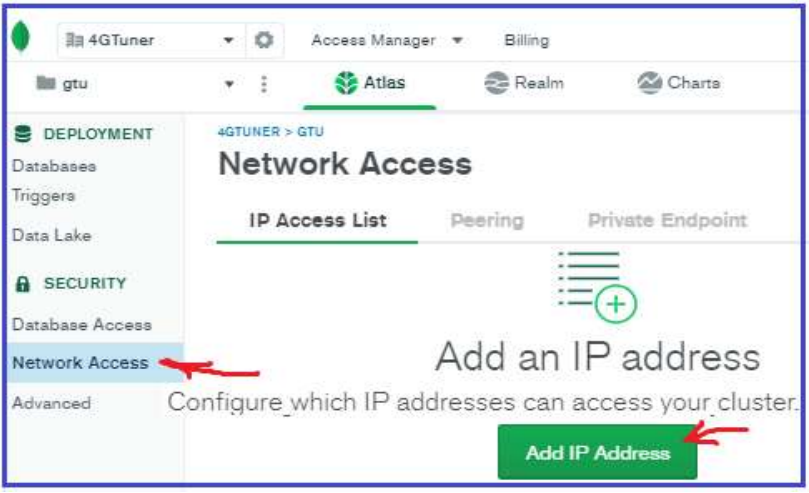


ნახ.4.52



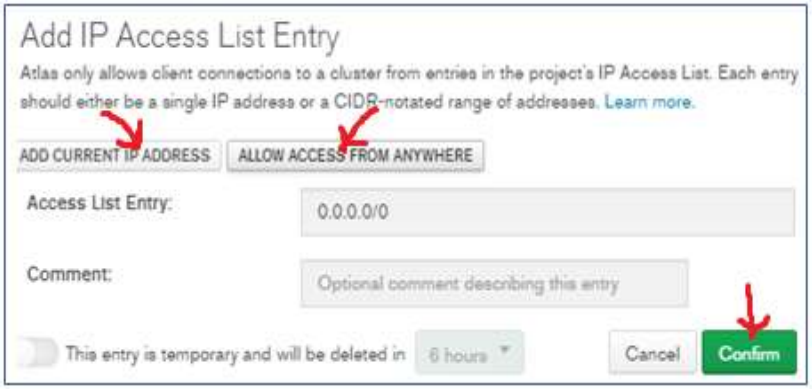
ნახ.4.53

ინტერფეისზე მომდევნო ჩანართი არის **Network Access,** სადაც უნდა განისაზღვროს მომხმარებლის IP მისამართები ჩანართის გააქტიურების შედეგად მიიღება ფანჯარა (ნახ.4.54).



ნახ.4.54

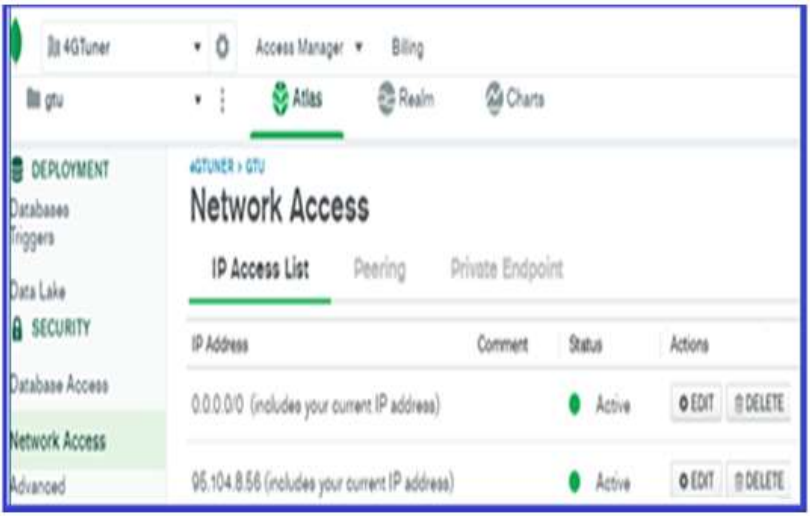
Add IP Addresse ღილაკის გააქტიურებით მიიღება ფანჯარა, სადაც ემატება ის IP მისამართები, რომელთაც Atlas-ი აძლევს სისტემასთან წვდომის უფლებას (ნახ.4.55).



ნახ.4.55

ამისათვის კონკრეტულ მომხმარებელთა მისამართებს ვამატებთ ღილაკით ADD CURRENT IP ADDRESS (დაემატოს მიმდინარე IP მისამართი), ხოლო თუ არ არის ამის აუცილებლობა, მაშინ ვააქტიურებთ ღილაკს ALLOW ACCESS FROM ANYWERE (დასაშვებია წვდომა ნებისმიერი ადგილიდან). ბოლოს ვადასტურებთ **Confirm** ღილაკით.

დავამატეთ ორი IP მისამართი, რომელიც არის აქტიური და აქვს წვდომა Atlas პლატფორმაზე. შესაბამისი საილუსტრაციო მაგალითი წარმოდგენილია 4.56 ნახაზზე.



ნახ.4.56

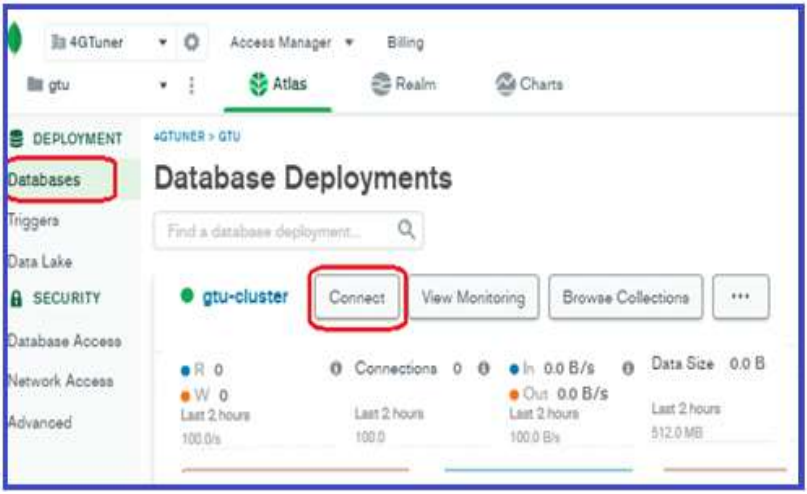
დასკვნა_1:

ჩვენ შევქმენით კლასტერი, დავამატეთ მომხმარებლები და განვსაზღვრეთ IP მისამართები.

შემდეგი ეტაპი არის MongoDB - ბაზასთან დაკავშირება და მონაცემებთან წვდომა.

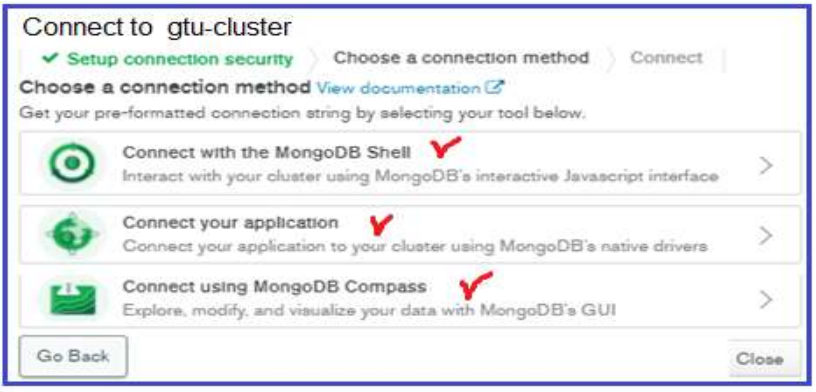
➢ **დაკავშირება მონაცემთა ბაზასთან.**

მონაცემთა ბაზასთან დასაკავშირებლად ვააქტიურებთ ჩანართს Databases და შემდეგ ვირჩევთ Connect (ნახ.4.57).



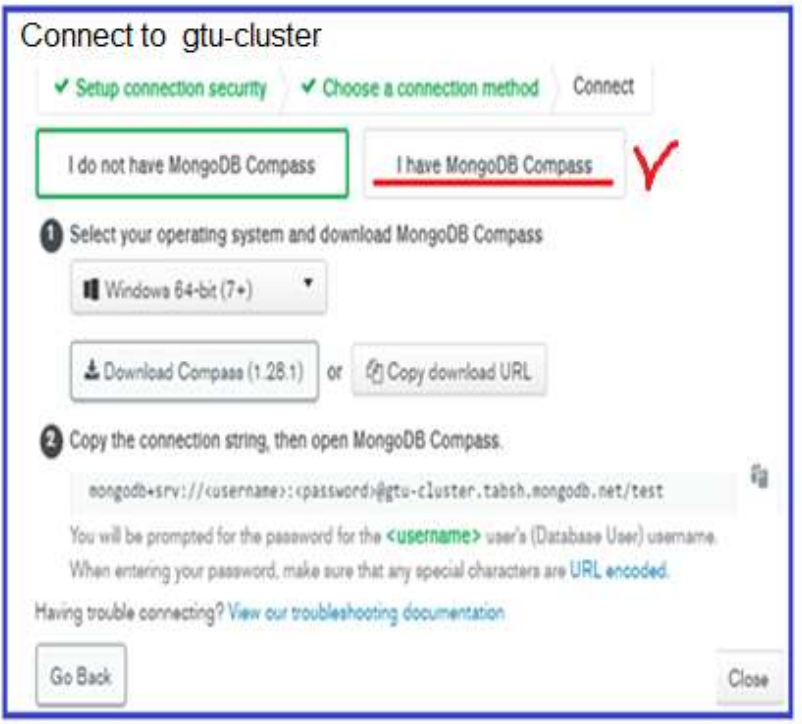
ნახ.4.57

მიიღება ფანჯარა, საიდანაც კლასტერის დაკავშირება შესაძლებელია, როგორც MongoDB Shell-ის, ასევე MongoDB Compass-გარემოსთან (ნახ.4.58).



ნახ.4.58

კლასტერი დავაკავშიროთ MongoDB Compass-თან, ამისათვის, როგორც სისტემაშია მოცემული გავააქტიუროთ ღილაკი - **Connect using MongoDB Compass**, შედეგად მიიღება ფანჯარა, ორი ღილაკით, პირველი, როდესაც მომხმარებელს არა აქვს MongoDB Compass და შეუძლია ჩამოტვირთოს და ლოკალურად დააყენოს თავის პერსონალურ კომპიუტერში და მეორე, როდესაც გვაქვს და შესაძლებელია MongoDB Compass-ით სარგებლობა, ჩვენ შემთხვევაში მოვნიშნოთ ღილაკი I have MongoDB Compass (ნახ.4.59).

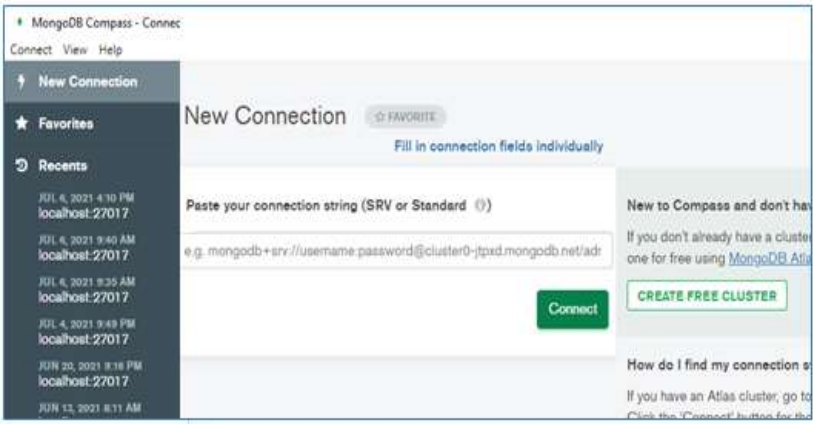


ნახ. 4.59

შედეგად მიიღება ფანჯარა, საიდანაც ვაკოპირებთ სტრიქონს MongoDB Compass-ისთვის (ნახ.4.60).

ნახ. 4.60

პარალელურად ლოკალურად ვხსნით ჩვენს MongoDB Compass გარემოს (ნახ.4.61).



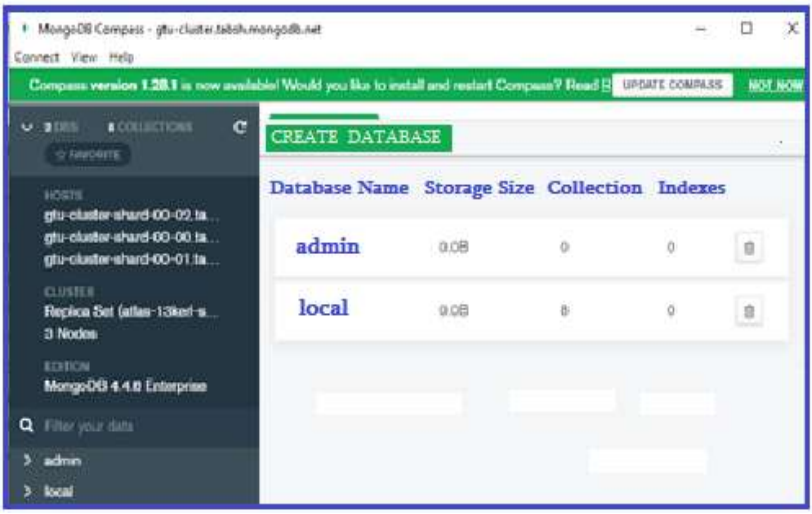
ნახ.4.61

დაკოპორებულ სტრიქონს ჩავსვამთ სპეციალურად გამოყოფილ არეში (ნახ.4.62).

ნახ.4.62

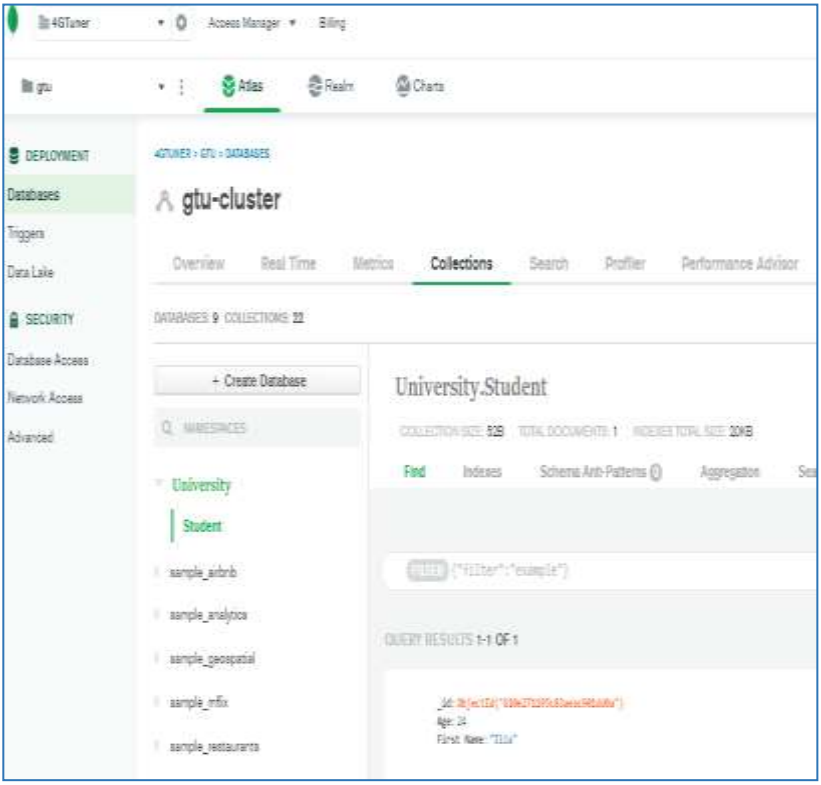
დავაკორექტირებთ მომხმარებლის სახელს და ჩავწერთ შესაბამის პაროლს, რასაც ვადასტურებთ Connect ღილაკით.

შედეგად მოხდება დაკავშირება, მიიღება ფანჯარა, სადაც ვამატებთ ახალ ბაზას. ვქმნით კოლექციებს (ნახ.4.63).



ნახ.4.63

MongoDB Atlas გარემოში განახლების ღილაკის გააქტიურების შემდეგ აისახება ჩვენს მიერ შექმნილი, მაგალითად, Universty ბაზა, რომლის კოლექციის დასახელებაა Student (ნახ.4.64).



ნახ.4.64

დასკვნა_2:

მიღებული შედეგის თანახმად ლოკალური MongoDB Compass გარემოდან ავსახეთ ღრუბლოვან MongoDB Atlas მონაცემთა ბაზაში ჩვენ მიერ ფორმირებული მონაცემთა ბაზა, რომლის შემდგომი ანალიზისა და დამუშავებისათვის შესაძლებელია გამოყენებულ იქნას ყველა თანამედროვე ტექნოლოგია და ინსტრუმენტული საშუალება.

V თავი
მონაცემთა საცავის მოდელირება და კვლევა

ელექტრონული ბიზნესისა და კომერციის სისტემები განაწილებული მართვის ობიექტების ერთობლიობაა. მაგალითად დიდი სავაჭრო ცენტრები, სუპერტმარკეტები და ა.შ. მათ აქვს (ან სასურველია ჰქონდეს მენეჯმენტის სრულყოფის მიზნით) განაწილებული მონაცემთა საცავები და ინფორმაციის დამუშავების (ინტელექტუალური ანალიზის) ტექნოლოგიები [43, 72].

კლიენტ-სერვერული არქიტექტურის კონფიგურაცია, ზოგადად, ჰომოგენური (ან ჰეტეროგენული) კომპიუტერული რესურსებით ხასიათდება. ასეთი სისტემის მიზანია მომხმარებელთა მოთხოვნების მაქსიმალური დაკმაყოფილება (უშუალოდ მარკეტში ან ინტერნეტიდან), რაც უზრუნველყოფს მოგების სტაბილურობას.

მასობრივი მომსახურების (ან რიგების) თეორიის მიხედვით, პირობითად, „მომსახურე ორგანოს“ როლს ასრულებს სუპერმარკეტის თანამშრომელი (კლიენტებთან უშუალო კონტაქტი) ან კომპიუტერული ქსელის სერვერი (ელექტრონული კონტაქტი). ორივე შემთხვევაში პროცესი მსგავსი მოდელით აიგება (მოთხოვნების ნაკადი, მომსახურების დრო, რიგების სიგრძე და ა.შ.), თვით ამ მაჩვენებელთა მნიშვნელობები კი იქნება განსხვავებული.

ასეთი მულტიპროცესორული ქსელური კონფიგურაციის სისტემების დაპროექტებისას საჭიროა მრავალი მახასიათებლის გათვალისწინება, მათი ოპტიმალური მნიშვნელობების შერჩევა ძალზე მნიშვნელოვანია და ამვე დროს რთულიც. ამ სიდიდეთა ოპტიმიზაცია არა მარტო გაზრდის კომპიუტერული ქსელის წარმადობას, არამედ შეამცირებს მის შესაქმნელად საჭირო ხარჯებსაც. საჭიროა ისეთი მაჩვენებლების გათვალისწინება, როგორიცაა საწარმოო სიმძლავრეები, ადამიანური და საერთო რესურსები და ა.შ. და მათი ეფექტიანი გამოყენება.

კორპორაციულ კომპიუტერულ ქსელებსა და ელექტრონული ბიზნესის ობიექტებზე მიმდინარე მოვლენების (დინამიკური პროცესების) მოდელირებისათვის მოსახერხებელია გრაფო-ანალიზური სისტემის, კერძოდ, პეტრის ქსელების გამოყენება, რაოდენობრივი მახასიათებლების ანალიზისათვის კი - მასობრივი მომსახურების სისტემების თეორია [9].

წინამდებარე თავში გადმოცემულია ჩვენ მიერ სტუ-ში შემუშავებული ალგორითმული სქემები და შესაბამისი პროგრამუ-ლი პაკეტები, რომელთა დანიშნულებაცაა კომპიუტერული ქსელის სიმძლავრეების ანალიზი და მათი ოპტიმალური განაწილება [110,111].

5.1. განაწილებული სისტემის რესურსების მართვის პროცესის მასობრივი მომსახურების მოდელი სტაციონარული რეჟიმისათვის

განვიხილოთ კომპიუტერული ქსელი, სადაც არის რამდენიმე მომხმარებელი და რამდენიმე სერვერი (მომსახურე). დავუშვათ, რომ სერვერთაგან ერთ-ერთი ასრულებს გამანაწილებლის ფუნქციას, ე.ი. იღებს მომხმარებლისაგან მოთხოვნას და უგზავნის მას მომსახურებისათვის იმ სერვერს, რომელიც თავისუფალია. თუ ყველა სერვერი დაკავებულია, მოთხოვნა დგება რიგში და ელოდება ერთ-ერთი მათგანის განთავისუფლებას.

სერვერი, მიიღებს რა მოთხოვნას გამანაწილებელი სერვერიდან, ემსახურება მას და უბრუნებს ისევ გამანაწილებელ სერვერს, რომელიც, თავის მხრივ პასუხს უბრუნებს მომხმარებელს.

უნდა ვიგულისხმოთ, რომ მოთხოვნები მომხმარებლებისგან მოდის უწყვეტად, გარკვეული სიხშირით. თითოეული სერვერი ერთეული მოთხოვნის მომსახურებას ანდომებს გარკვეულ დროს. იმ შემთხვევაში როდესაც, მოთხოვნათა ფორმირების სიხშირე დიდია, გამანაწილებელ სერვერთან წარმოიქმნება რიგი. თუკი მოთხოვნათა ფორმირების სიხშირე ძალზე დიდია ქსელი შეიძლება

გადაიტვირთოს და ვეღარ შეძლოს ფუნქციონირება.

ჩვენი მიზანია ქსელის არსებული პარამეტრების მეშვეობით დავადგინოთ მისი მუშაობის კრიტიკული წერტილი, შევარჩიოთ ისეთი მახასიათებლები, რომლებიც უზრუნველყოფს მის ნორმალურ ფუნქციონირებას და შევქმნათ პროგრამული პროდუქტი, რომელიც ყოველივე ამას განახორციელებს. რიგების თეორიის თვალსაზრისით ზემოაღწერილი სისტემა არის M/M/m ტიპის [5, 113].

განვიხილოთ მახასიათებლები და მათ შორის კავშირები, რომლებიც გააჩნია ქსელს. საჭიროა აღინიშნოს, რომ ქსელის ფუნქციონირებას განვიხილავთ სტაციონარულ რეჟიმში. ამ შემთხვევაში, როგორც ცნობილია, გარკვეულ იდეალიზაციასთან გვაქვს საქმე.

რეალურად დროის ყოველ t მომენტში სისტემაში არსებობს მოთხოვნათა რაღაც k რაოდენობა. ალბათობა იმისა, რომ დროის მოცემულ t მომენტში სისტემაში იმყოფება k მოთხოვნა, აღვნიშნოთ $P_k(t)$-თი. ჩვენ უნდა ვიგულისხმოთ, რომ t-ს ზრდასთან ერთად ალბათობა $P_k(t)$ თანდათან მუდმივი ხდება. ამ შემთხვავაში $P_k(t)$-ს ნაცვლად შეიძლება გამოვიყენოთ P_k, რომელიც უკვე აღარ არის დროის ფუნქცია.

ეს დაშვება არ გულისხმობს იმას, რომ სისტემა არ გადადის ერთი მდგომარეობიდან მეორეში, რა თქმა უნდა დროის მიხედვით იცვლება ქსელში არსებული მოთხოვნების რაოდენობა, მაგრამ ალბათობა იმისა, რომ სისტემაში საკმარისად დიდი დროის გასვლის შემდეგ იმყოფება k მოთხოვნა, გამოიხატება P_k-თი.

პროგრამაში საანგარიშო ფუნქციების დასაპროგრამებლად გამოვიყენოთ ზემოაღნიშნული კლასიკური მოდელები. ამგვარად, სერვერების რაოდენობით, შემოსულ

მოთხვნათა ინტენსივობით და დროით, რომელსაც ანდომებს სერვერი თითოეული მოთხოვნის მომსახურებას, შეგვეძლება დავადგინოთ ქსელის მახასიათებლები.

აღვნიშნოთ მოთხოვნათა მოსვლის ინტენსივობა λ-თი, ხოლო თითოეული მოთხოვნის მომსახურების დრო Ts-ით. ამ შემთხვევაში ერგოდიულობის პირობა არის:

$$\lambda * Ts < 1 .$$

ქსელს აქვს შემდეგი მახასიათებლები:

1) მოძრაობის ინტენსივობა: $u = \lambda * Ts$
2) სერვერის დატვირთვა: $\rho = u/m$.

იმისათვის, რომ სისტემა იყოს სტაბილური, სერვერს უნდა შეეძლოს თავი გაართვას მოთხოვნათა მოსვლის საშუალო ინტენსივობას, ეს კი ნიშნავს, რომ მოძრაობის ინტენსივობა უნდა იყოს სერვერთა რაოდენობაზე ნაკლები, ან რაც იგივეა, სერვერის დატვირთვა უნდა იყოს ერთზე ნაკლები, ე.ი.

$$u < m \text{ ან } \rho < 1.$$

M/M/m სახის სისტემების კვლევისას მნიშვნელოვანი ადგილი უკავია *ერლანგის ფუნქციას*. ეს ფუნქცია განსაზღვრავს იმის ალბათობას, რომ ყველა სერვერი დაკავებულია, და იმავდროულად იმის ალბათობასაც, რომ მოსულ მოთხოვნას მოცდა მოუწევს. ერლანგის ფუნქციისთვის გამოვიყენებთ გამოსახულებას:

$$Ec(m,u) = (u^m / m!) / (u^m / m! + (1-\rho) \sum_{k=0}^{m-1} (u^m / m!))$$

მომხმარებლისთვის დიდი მნიშვნელობა აქვს მოთხოვნის რიგში დგომის (მოცდის) საშუალო დროს, იგი გამოითვლება ფორმულით:

$$Tw=\frac{Ec(m,u)Ts}{m(1-\rho)}$$

აუცილებელია განვსაზღვროთ მოთხოვნის სისტემაში ყოფნის საშუალო დრო:

Tq=Tw+Ts.

ალბათობა იმისა, რომ მოთხოვნის სისტემაში ყოფნის დრო ნაკლებია T-ზე, დამოკიდებულია u = m-1 თუ არა.

თუ ეს პირობა სრულდება, მაშინ ადგილი აქვს შემდეგ ტოლობას:

$$P(\text{სისტემაში ყოფნის დრო}<t)=1-(1+\frac{t}{Ts}Ec(m,u))e^{\frac{-t}{Ts}}$$

წინააღმდეგ შემთხვევაში:

$$P(\text{სისტემაში ყოფნის დრო}<t)=$$
$$=1+\frac{B+Ec(m,u)}{B}e^{\frac{-t}{Ts}}+\frac{Ec(m,u)}{B}e^{-(m-u)\frac{t}{Ts}}$$

სადაც B = m-1-u.

დროის ყოველ მომენტში ქსელში იარსებებს მოთხოვნათა გარკვეული რაოდენობა. რაც ნაკლები მოთხოვნაა ქსელში, მით უკეთ ფუნქციონირებს იგი. ალბათობა იმისა, რომ ქსელში არის k მოთხოვნა არის P_k სადაც

$$P_k=\frac{u}{k!}P_0$$

როცა k <= m და

$$P_k = \frac{u^k}{m! m^{k-m}} P_0 \quad \text{როცა} \quad k>=m$$

P_0 არის ალბათობა იმისა, რომ ქსელში საერთოდ არაა მოთხოვნა.

ეს რაც შეეხებოდა ალბათობებს. თვით სისტემაში არსებულ მოთხოვნათა რაოდენობა კი არის Lq, სადაც

$$Lq = u + \frac{pEc(m,u)}{1-\rho}$$

თუკი ქსელში არის m ან m-ზე ნაკლები მოთხოვნა, მაშინ იმ მოთხოვნების რაოდენობა, რომლებიც რიგში დგანან 0-ის ტოლია, ხოლო თუ ვიცით, რომ x მოთხოვნა რიგში დგას, მაშინ მთლიანად სისტემაში იქნება x+m მოთხოვნა. ასე, რომ გვაქვს შემდეგი მახასიათებლები:

ალბათობა იმისა, რომ არცერთი მოთხოვნა არ იცდის:

$$\text{P(არცერთი მოთხოვნა არ იცდის)}= \sum_{k=0}^{m} P_k$$

ალბათობა იმისა, რომ x მოთხოვნა დგას რიგში:

$$\text{P(x მოთხოვნა იცდის)}= P_{x+m} \quad \text{სადაც} \quad x>m$$

მომლოდინე მოთხოვნათა საშუალო რიცხვი:

$$Lw = \frac{pEc(m,u)}{1-\rho}$$

ობიექტ-ორიენტირებული დაპროგრამების საფუძველზე, კომპიუტერული ქსელისათვის მიზანშეწონილია შეიქმნას კლასი, რომლის დახურული პარამეტრები იქნება მოთხოვნათა მოსვლის სიხშირე, მომსახურების დრო,

სერვერების რაოდენობა და ა.შ. ფუნქცია – წევრების სახით კი რეალიზებული იქნება ყველა ზემოთ ჩამოთვლილი მახასითებლების გამოთვლა, შესაბამისი ფორმულების გამოყენებით.

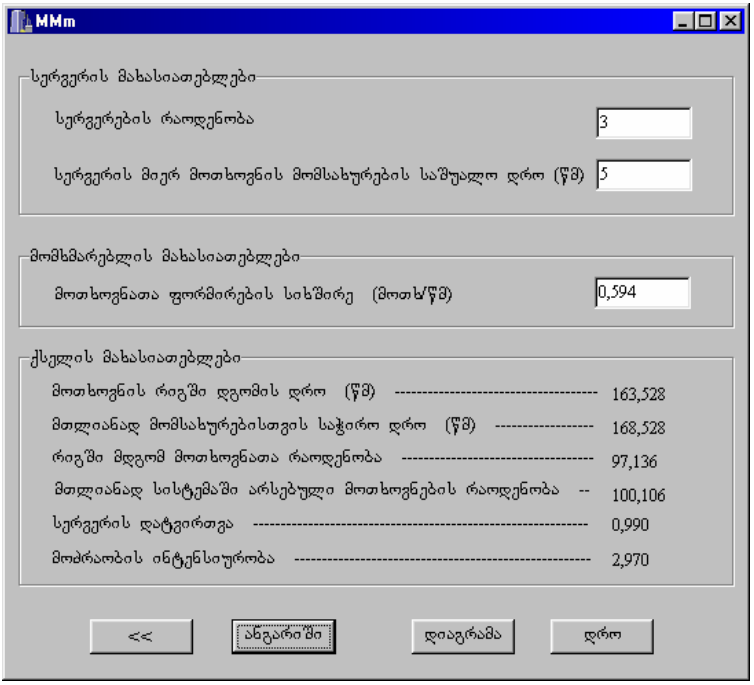
ცალკე კლასებად იქნება რეალიზებული მომხმარებლის ინტერფეისი და პარამეტრთა შორის დამოკიდებულებათა გრაფიკების აგებისა და ვიზუალიზაციის ფუნქციები.

ზემოთ განხილული სიდიდეები სრულად ახასიათებს კომპიუტერული ქსელის მუშაობის სტაციონალურ რეჟიმს. ჩვენ მიერ შექმნილი პროგრამული საშუალება სწორედ ამ სიდიდეებს და ფორმულებს იყენებს ქსელის პარამეტრების ანალიზისათვის და მათი ოპტიმალური მნიშვნელობის შერჩევისათვის.

იგი, იღებს რა ინფორმაციას ქსელში მოთხოვნების მოსვლის სიხშირეზე, სერვერთა რაოდენობასა და თითოეული მოთხოვნის მომსახურების დროზე, ანგარიშობს ისეთ პარამეტრებს როგორიცაა მოთხოვნის რიგში დგომის დრო, ბუფერში მოთავსებული მომლოდინე მოთხოვნათა რაოდენობა, სერვერის დატვირთვა და მოძრაობის ინტენსივობა, სხვადასხვა ალბათობები და ა.შ.

გარდა ამისა, გამოითვლის მოცემულ პირობებში ოპტიმალური მუშაობისათვის საჭირო პარამეტრებს და აგებს მათ შორის დამოკიდებულებათა გრაფიკებს.

5.1 ნახაზზე მოცემულია C++ ენაზე (Borland_C++Builder ინსტრუმენტით) აგებული მომხმარებლის ინტერფეისი, რომელშიც მუშაობა წარიმართება ვიზუალური და ტრადიციული დაპროგრამების კომპონენტების რევერსული ტექნოლოგიით [9].



ნახ.5.1.

მომხმარებელს შეუძლია შეიტანოს (და ცვალოს) სამი პარამეტრის მნიშვნელობა:

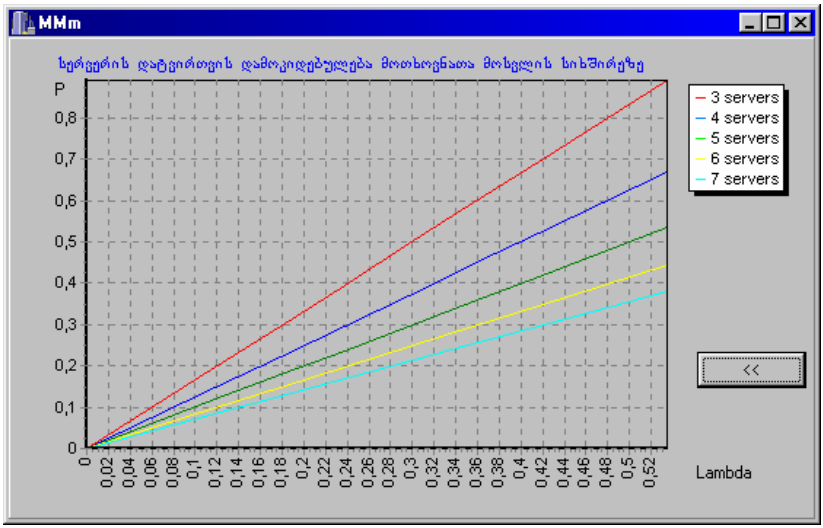
- სერვერების რაოდენობა;
- მომსახურების საშუალო დრო და
- მოთხოვნათა რაოდენობის ინტენსიურობა.

ღილაკით „**ანგარიში**“ სისტემა გაიანგარიშებს ქსელის ძირითად მახასიათებლებს, კერძოდ:

- სერვერის დატვირთვა;
- მოძრაობის ინტენსიურობა;
- მოთხოვნის რიგში დგომის დრო;
- მთლიანად მომსახურებისთვის საჭირო დრო;

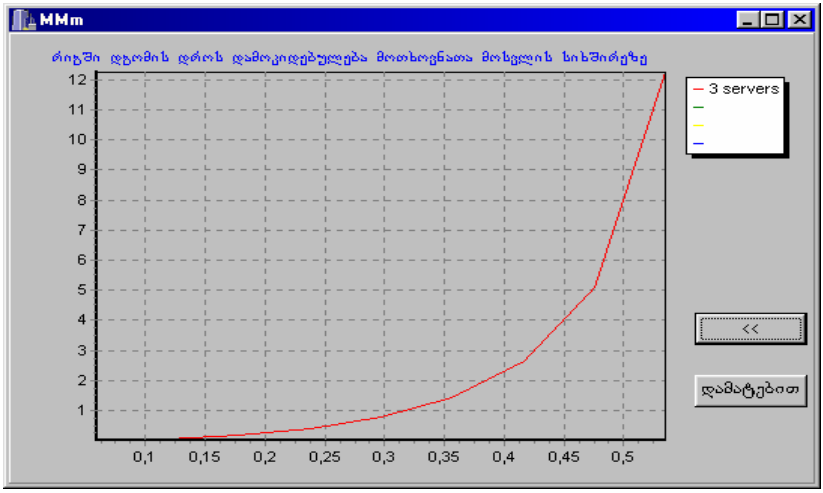
- რიგში მდგომ მოთხოვნათა რაოდენობა;
- სისტემაში მყოფ მოთხოვნათა საერთო რაოდენობა.

ღილაკით „**დიაგრამა**“ გამოიტანება გაანგარიშების შედეგად მიღებული გრაფიკები. 5.2 ნახაზზე მოცემულია პროგრამულად მიღებული დიაგრამა სერვერის დატვირთვის დამოკიდებულებისა მოთხოვნათა მოსვლის სიხშირეზე, სერვერთა სხვადასხვა რიცხვისთვის (მაგ., 3-:-7).

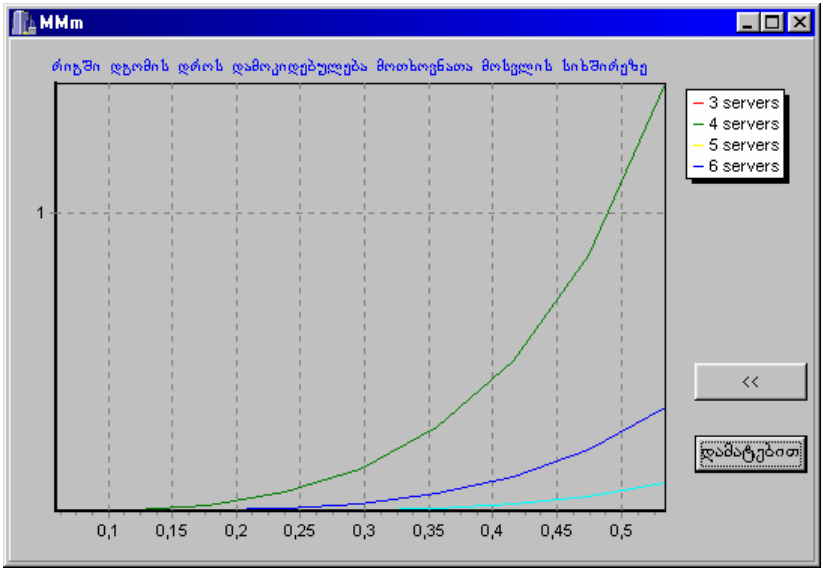


ნახ.5.2.

5.3 და 5.4 ნახაზებზე კი მოცემულია დიაგრამები მოთხოვნათა რიგში დგომის დროის დამოკიდებულებისა მოთხოვნათა მოსვლის სიხშირეზე სერვერების სხვადასხვა რაოდენობის (მაგალითად, 3-:-6) შემთხვევაში. ბოლო დიაგრამებიდან კარგად ჩანს, თუ როგორ იკლებს მოთხოვნათა რიგში დგომის დრო მომსახურე არხების მომატებით.


MMm
რიგში დგომის დროს დამოკიდებულება მოთხოვნათა მოსვლის სიხშირეზე
12
11
10
9
8
7
6
5
4
3
2
1
0,1
0,15
0,2
0,25
0,3
0,35
0,4
0,45
0,5
3 servers
<<
დამატებით

ნახ.5.3.


MMm
რიგში დგომის დროს დამოკიდებულება მოთხოვნათა მოსვლის სიხშირეზე
1
0,1
0,15
0,2
0,25
0,3
0,35
0,4
0,45
0,5
3 servers
4 servers
5 servers
6 servers
<<
დამატებით

ნახ.5.4.

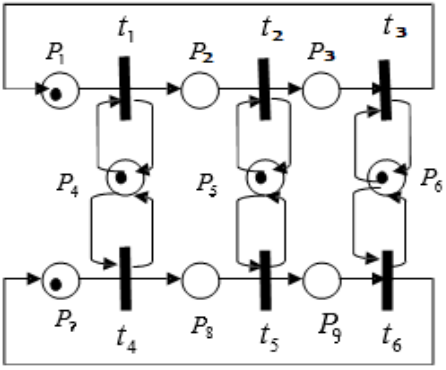
5.2. განაწილებული სისტემის რესურსების მართვის პროცესის კვლევა პეტრის ქსელის გრაფით დინამიკური რეჟიმისათვის

გავაანალიზოთ კომპიუტერული ქსელის პროცესები დინამიკურ რეჟიმში. სერვერების მიერ კლიენტთა მოთხოვნების დამუშავების პროცესი შეიძლება მოდელირებულ იქნას ტრანზიტული დროითი პეტრის ქსელის (Timed Transition Petri Net) საშუალებით [60, 114].

პეტრის ქსელის დროითი გაფართოება ამ შემთხვევაში იქნება სტოქასტური. ასეთი ქსელის ანალიზი შესაძლებელია მარკოვის პროცესების მეთოდების გამოყენებით, რომელშიც დრო ექსპონენციალურადაა განაწილებული [5].

სტოქასტური პეტრის ქსელის მისაღებად საჭიროა „პოზიცია-გადასასვლელების ქსელს“ დაემატოს გადასასვლელთა გაშვების (დაყოვნების, მოლოდინის) დროთა მომენტები (მაგალითად, μ1, μ2, . . . , μn).

განვიხილოთ კერძო მაგალითი კომპიუტერების ქსელისათვის, ორი სერვერითა და სამი კლიენტით. 5.5 ნახაზზე მოცემულია შესაბამისი სტოქასტური პეტრის ქსელის საწყისი მდგომარეობა.



ნახ.5.5. პეტრის ქსელის გრაფი:
$S_1(P_1,P_2,P_3)$, $S_2(P_7,P_8,P_9)$ - 2_სერვერი
$C(P_4,P_5,P_6)$ კლიენტი;
P - პოზიცია (სერვერის მდგომარეობა);
T - გადასასვლელი;
მარკერი - შავი წერტილი;
P_1 და P_7 - აქტიურებია;
t_1 ან t_4 გაიხსნება P_4-ის (მოთხოვნისა) მარკერით

მარკერის არსებობა S1(p1, p2, p3) და S2(p7, p8, p9) სერვერებში მიუთითებს მათ მზადყოფნაზე კლიენტების მომსახურებისათვის.

დავუშვათ, რომ C(p4, p5, p6) კლიენტის პოზიციებში მარკერები მუდმივადაა, ე.ი. მოთხოვნები არსებობს და ისისნი ელოდება სერვერის მომსახურებას.

Tj გადასასვლელის გახსნის დროა (ანუ მომსახურების დაყოვნების დრო). Tj-ური გადასასვლელის გახსნის საშუალო დრო იქნება 1/μ, სადაც μ გადასასვლელის გახსნის ინტენსივობაა.

სისტემის მდგომარეობები, ანუ მარკერების სიმრავლე შეიძლება ასე ჩაიწეროს:

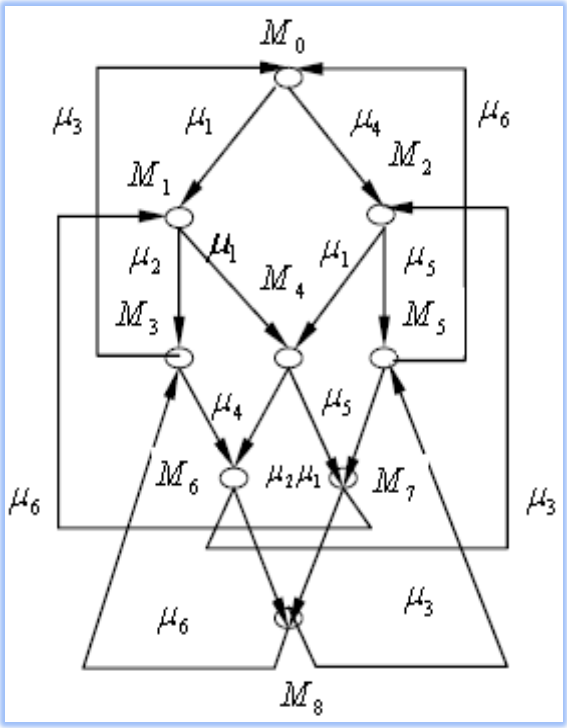
p პოზიციები და t გადასასვლელები:

M1 — 1 0 0 1 1 1 0 0 1
M2 — 0 1 0 1 1 1 0 0 1
M3 — 1 0 0 1 1 1 1 0 0
M4 — 0 1 0 1 1 1 0 0 1
M5 — 1 0 0 1 1 1 0 1 0
M6 — 0 0 1 1 1 1 1 0 0
M7 — 0 1 0 1 1 1 0 1 0
M8 — 0 0 1 1 1 1 0 1 0

სადაც:

- Mi, 0 <= i <= K მდგომარეობებია (მარკირებები);
- Tj, 1 <= j <= L – გადასასვლელები;
- μ_j, 1 <= j <= L – დაყოვნების დრო გადასასვლელის გასაღებად. ჩვენ შემთხვევაში m=3 და n=2, ამიტომ კომბინაცია იქნება $m^n = 9$.

გადასასვლელების გახსნის ორგანიზება, როცა სისტემა ყველა მდგომარეობას გადის ნაჩვენებია 5.6 ნახაზზე, რომელიც პეტრის ქსელის მიღწევადობის გრაფია.



ნახ.5.6

ასეთი სტოქასტური პეტრის ქსელის რაოდენობრივი ანალიზი შეიძლება განხორციელდეს შესაბამისი მარკოვის პროცესებით. განვიხილოთ მარკოვის ჯაჭვის მაგალითი, რისთვისაც ამ ნახაზზე გრაფის რკალებზე მივამაგროთ გადასასვლელთა გაშვების μ კოეფიციენტები.

ჩვენთვის საინტერესოა დავადგინოთ სისტემის თითოეულ მდგომარეობაში გადასვლის ალბათობები, ამისათვის საჭიროა შევადგინოთ კოლმოგოროვის განტოლებათა სისტემა [60]:

$$P5*\mu 6+P3*\mu 3-P0*(\mu 1+\mu 4)=0$$

$$P7*\mu 6+P0*\mu 1-P1*(\mu 2+\mu 4)=0$$

$$P0*\mu 4+P6*\mu 3-P2*(\mu 1+\mu 5)=0$$

$$P6*\mu 3+P0*\mu 4-P3*(\mu 1+\mu 5)=0$$

$$P1*\mu 4+P2*\mu 1-P4*(\mu 2+\mu 5)=0$$

$$P2*\mu 5+P8*\mu 3-P5*(\mu 1+\mu 6)=0$$

$$P3*\mu 4+P4*\mu 2-P6*(\mu 3+\mu 5)=0$$

$$P4*\mu 5+P5*\mu 1-P7*(\mu 2+\mu 6)=0$$

$$P6*\mu 5+P7*\mu 2-P8*(\mu 3+\mu 6)=0$$

$$P0+P1+P2+P3+P4+P5+P6+P7+P8=1$$

მაგალითად, თუ დავუშვებთ, რომ:

$\mu 1=3,\ \mu 2=5,\ \mu 3=2,\ \mu 4=3,\ \mu 5=1,\ \mu 6=7,$

მაშინ ალბათობათა მნიშვნელობები, შესაბამისად იქნება:

$$P0=0.11;\ P1=0.05;\ P2=0.07;$$

$$P3=0.26;\ P4=0.06;\ P5=0.02;$$

$$P6=0.37;\ P7=0.01;\ P8=0.05.$$

უნდა აღინიშნოს, რომ კომპიუტერული ქსელისათვის ერთი მდგმარეობიდან მეორეში გადასვლის ინტენსივობა **μ** არის სერვერის მიერ შესაბამისი კლიენტის მოთხოვნის მომსახურებისათვის საჭირო დროის შებრუნებული სიდიდე, ე.ი. **1 / Ts**.

ამ განტოლებათა სისტემის გაუსის მეთოდით ამოხსნით მივიღებთ სისტემის ერთი მდგომარეობიდან მეორეში გადასვლის ალბათობებს $P_0, P_1, P_2, \ldots, P_8$.

5.3. ბიზნეს პროცესების მოდელირება რიგების თეორიის ჩაკეტილი სისტემებით

საბაზრო ეკონომიკის კონკურენციულ გარემოში ბიზნეს-პროცესების ეფექტური მართვის ერთ-ერთი ძირითადი მეთოდია მათემატიკური მოდელირება. ფაქტობრივად, სისტემური ანალიზის და ოპერაციათა კვლევის მათემატიკური მეთოდები ხდება ეკონომიკურ გამოკვლევათა განუყოფელი ნაწილი.

უნდა აღინიშნოს, რომ ეკონომიკურ-მათემატიკური მოდელირების ერთ-ერთი მნიშვნელოვანი ნაწილია რიგების თეორია, კერძოდ, რიგების თეორიის ჩაკეტილი სისტემები. სწორედ მათი მეშვეობით აღიწერება მრავალი ეკონომიკური სისტემა, რომელთა ფუნქციობაში გადამწყვეტ როლს ასრულებს შემთხვევითი (სტოქასტური) ფაქტორები.

მეორეს მხრივ, ასეთი სისტემების მართვის ერთ-ერთი არსებითი ატრიბუტია მათი ფუნქციობის ეფექტიანობის ეკონომიკური მაჩვენებლის (კრიტერიუმის) შერჩევა, რაც თავის მხრივ, ოპტიმალური სამოქმედო გადაწყვეტილებათა მიღების საფუძველია. ეს იმას ნიშნავს, რომ ალტერნატიული სამოქმედო ვარიანტებიდან აირჩევა ის, რომელსაც შეესაბამება ეფექტიანობის ეკონომიკური მაჩვენებლის მეტი ან ნაკლები მნიშვნელობა.

ეფექტიანობის კრიტერიუმში გათვალისწინებული უნდა იყოს სისტემაში არსებული დანახარჯები, შემოსავლები, მოგება, საჯარიმო სანქციებით გამოწვეული ზარალი და სხვა ეკონომიკური ასპექტები, რომლებიც ობიექტურად ასახავს სისტემის საბაზრო მდგომარეობას.

აღნიშნულ გარემოებათა გონივრული შეხამება წარმატებული ბიზნესის ძირითადი საწინდარია. ჩვენი ქვეყნის მძიმე სოციალურ-ეკონომიური მდგომარეობისა და არსებული საგადასახადო კანონმდებლობის გათვალისწინებით, საკმაოდ რთულია თავი

დავაღწიოთ გაკოტრებას და ვაწარმოოთ წარმატებული კონკურენტუნარიანი ბიზნესი.

ჩვენი მიზანია ბიზნესის მართვის სისტემისთვის შევქმნათ გადაწყვეტილებათა მიღების ხელშემწყობი, ისეთი ეფექტიანი მოდელი, რომლის მთავარი ორიენტირი იქნება წარმატებისათვის აუცილებელი სამუშაოების შესრულების ოპტიმალური პირობების შექმნა.

ბიზნესის ერთ-ერთ საყოველთაოდ გავრცელებულ სახეობაა ე.წ. „სავაჭრო ობიექტთა ბიზნესი“, სადაც საქმე გვაქვს კლიენტების მომსახურებასთან. ერთი შეხედვით ამ მარტივ საქმიანობასთან დაკავშირებულია საკმაოდ დიდი სირთულეები.

კონკრეტულად განვიხილოთ მაღაზიათა მუშაობის პრინციპი. მაღაზიია ყოველთვის დროულად უნდა იყოს უზრუნველყოფილი კლიენტის მოთხოვნილების შესაბამისი მაღალი ხარისხის პროდუქციით, რადგან არარეალიზებადი საქონელი იგივეა, რაც მაღაზიაში არსებული ცარიელი დახლები.

არსებობს კლიენტთა მოზიდვის მრავალი ფაქტორი, რომელთაც აქ არ განვიხილავთ. კლიენტთა მოთხოვნილების დაკმაყოფილების პირობის უკან დგას ერთ-ერთი მნიშვნელოვანი ფაქტორი, როგორიცაა მაღაზიის სერვისების (მომსახურების) დონე.

თუ ჩვენ სათანადო ანალიზის გარეშე გავზრდით მომსახურე პერსონალს, რათა დროულად და მაღალ დონეზე მოხდეს შესაბამისი მომსახურება, შესაძლებელია მიღებული ხარჯები იმდენად დიდი აღმოჩნდეს, რომ ბევრად გადააჭარბოს არსებულ მოგებას, რაც საბოლოოდ მიგვიყვანს გაკოტრებამდე.

ამიტომ უნდა შევარჩიოთ საქმიანობის ისეთი მოდელი, რომელიც ზუსტად განსაზღვრავს, ჩვენ მიერ შერჩეულ ობიექტის მომგებიან მუშაობას. ეკონომიკის სფეროში თეორიულ და განსაკუთრებით პრაქტიკულ დონეზე ხშირად ისმის ისეთი ამოცანები სადაც, საქმე ეხება დიდი რაოდენობით მომხმარებელთა

მომსახურების პროცესს. სწორედ ასეთი ტიპის ამაცანათა გადასაჭრელად ყველაზე ეფექტური საშუალებაა მასობრივი მომსახურების თეორიის (მმთ) ანუ რიგების თეორიის გამოყენება.

ამ თეორიის გამოყენებით საშუალება გვეძლევა გამოვიკვლიოთ და შევისწავლოთ სისტემის მართვის პროცესები, დავადგინოთ სათანადო კანონზომიერება ამ სისტემის ცალკეულ ელემენტებს შორის და მიღებული შედეგები გამოვიყენოთ აღნიშნულ პროცესთა სრულყოფაში. ჩვენი შემდგომი მიზნებისათვის განვიხილავთ შემდეგი სახის სისტემას.

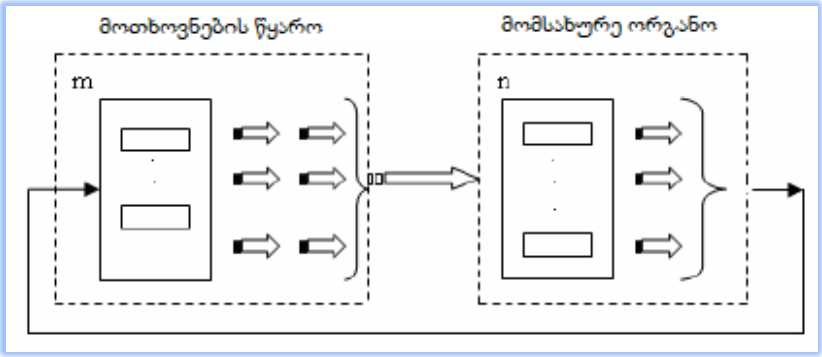
მასობრივი მომსახურების სისტემა შედგება ორი სახეობის ელემენტისაგან:

1) *მომხმარებელი ობიექტები*, რომლებიც საჭიროებს სათანადო მომსახურებას და

2) *მომსახურე ობიექტები*, რომლებიც ახორციელებს მომხმარებლის მომსახურება.

ჩვენ განვიხილავთ ასეთ ამოცანას: მოცემული გვაქვს სავაჭრო ფირმა m რაოდენობის მაღაზიით. ამ მაღაზიათა ქსელს, საქონლის მომარაგებაზე ემსახურება n ბრიგადა. როგორც კი რომელიმე მაღაზიაში გაჩნდება მოთხოვნა რომელიმე სახის საქონელზე, მაღაზიის ხელმძღვანელი უკავშირდება საწყობს და უკვეთავს საქონელს. ყოველ ბრიგადას ერთსადაიმავე დროს შეუძლია მოემსახუროს მხოლოდ ერთ მაღაზიას, ხოლო თუკი შეკვეთის მოსვლის დროს ყველა ბრიგადა დაკავებულია, მაშინ საქონლის მიტანა გადაიდება მანამდე, სანამ არ განთავისუფლდება რომელიმე ბრიგადა. თუ მოთხოვნათა რაოდენობა გადააჭარბებს ბრიგადების რაოდენობას, მაშინ წარმოიქმნება რიგი მომსახურებაზე.

მომხმარებელი ობიექტი არის მაღაზიათა m რაოდენობა, რაც შეადგენს მოთხოვნათა m მოცულობის სასრულ წყაროს. მომსახურეობის ობიექტს კი შეადგენს n რაოდენობის მომარაგების ბრიგადა (ნახ.5.7).



ნახ.5.7.

შემოვიტანოთ აღნიშვნები:

• ერთი მაღაზიიდან შემოსულ მოთხოვნათა საშუალო რაოდენობა დროის ერთეულში – λ;

• ერთი მოთხოვნის მომსახურების საშუალო დრო – 1/μ;

• მომსახურებული მოთხოვნების საშუალო რაოდენობა დროის ერთეულში – α;

• ერთი მოთხოვნის მომსახურებიდან შემოსული საშუალო შემოსავალი დროის ერთეულში – c_1;

• მომსახურების ერთი ორგანოს შენახვა დროის ერთეულში ჯდება – c_2 ლარი

• კლიენტის ლოდინი დროის ერთეულში იწვევს ჯარიმას – c_3 ლარი.

სისტემაში შემოსული მოთხოვნა გადაეცემა მომსახურე ორგანოს და იწყება მოთხოვნის დაკმაყოფილება, იმ შემთხვევაში, თუ რომელიმე მომსახურე ორგანო არის თავისუფალი. წინააღმდეგ შემთხვევაში მოთხოვნა დგება რიგში და ელოდება, სანამ არ გათავისუფლდება რომელიმე მომსახურე ორგანო და მისი დაკმაყოფილება მოხდება

მხოლოდ იმ შემთხვევაში, როცა დაკმაყოფილდება მის წინ მდგომი ყველა მოთხოვნა. მაშასადამე მოქმედებს მომსახურების არაპრიორიტეტული დისციპლინა FCFS (First Come, First Served) – „მოვიდა პირველი, მომსახურდა პირველი“).

მომსახურების სისტემაში სრულდება ორი სახის ოპერაცია:

1) მოთხოვნის „წარმოშობა“;

2) მოთხოვნის მომსახურება.

პირველი არის ფიქტიური ოპერაცია, მეორე რეალური. დავუშვათ, რომ ორივე სახის ოპერაციის ხანგრძლივობები ექსპონენტურად განაწილებული შემთხვევითი სიდიდეებია. ასეთ პირობებში სისტემის ფუნქციობა აღიწერება მარკოვის პროცესით მდგომარეობათა დისკრეტული სივრცითა და უწყვეტი დროით.

სისტემის ყოფაქცევა დროში აღიწერება ფუნქციებით:

$$P(i,t)= P\{t \text{ მომენტში მომსახურებაზე და რიგში მყოფი მოთხოვნათა რაოდენობა არის } i\}.\ i=\overline{0,m}.$$

სტანდარტული ალბათური მსჯელობის საფუძველზე მიღებულია დიფერენციალურ განტოლებატა სისტემა P(i,t) ფუნქციების მიმართ (კოლმოგოროვის განტოლებები).

სისტემის სტაციონალური მდგომარეობა აღიწერება ალგებრულ განტოლებათა შემდეგი სისტემით:

$$\begin{cases} -m\lambda P(0)+\mu P(1)=0 \\ -((m-i)\lambda+i\mu)P(i)+(i+1)\mu P(i+1)+\lambda(m-i+1)P(i-1)=0,\ 1\le i<n \\ -((m-i)\lambda+n\mu)P(i)+n\mu P(i+1)+\lambda(m-i+1)P(i-1)=0,\ n\le i<m \\ -n\mu P(i)+(m-i+1)\lambda P(i-1)=0 \end{cases}$$

სადაც

$$\lim_{t \to \infty} P(i,t) = P(i)$$

და ამასთანავე

$$\sum_{i=1}^{\infty} P(i) = 1$$

- სისტემის ეკონომიკური ეფექტიანობის მაჩვენებელი აღვნიშნოთ F-ით. იგი არის დროის ერთეულში არსებული სუფთა მოგება: F=F(m, n, λ, μ) ;

- დროის ერთეულში სისტემის შემოსავალია - $\mathbf{c_1\alpha}$;

- სისტემის დანახარჯები შეადგენს – $\mathbf{c_2 n}$;

- კლიენტის ლოდინით გამოწვეული ჯარიმა იქნება $\mathbf{c_1\delta}$, სადაც δ არის დროის ნებისმიერ მომენტში მომსახურებაში მყოფი მოთხოვნების საშუალო რაოდენობა.

ალბათობის თეორიიდან ცნობილია, რომ

$$\delta = \sum_{i=0}^{\infty} iP(i) \qquad (5.1)$$

ცხადია, რომ δ არის m, n, λ, μ პარამეტრების ფუნქცია:

$$\delta = \delta(m,n,\lambda,\mu)$$

ასევე, α არის იგივე პარამეტრების ფუნქცია:

$$\alpha = \alpha(m, n, \lambda, \mu).$$

საბოლოოდ მივიღებთ სისტემის მოგების ფუნქციას:

$$\mathbf{F=F(m,n,\lambda,\mu) = c_1\alpha(m,n,\lambda,\mu) - c_2 n - c_3\delta(m,n,\lambda,\mu)}.$$

ამ ფუნქციის გამოყენებით შეიძლება დაისვას და ამოიხსნას მათემატიკური დაპროგრამების შემდეგი ამოცანები:

1) m, λ, μ – პარამეტრები ფიქსირებულია, ვიპოვოთ n-ის მნიშვნელობა, რომელიც F-ფუნქციას მიანიჭებს მაქსიმალურ მნიშვნელობას;

2) n, λ, μ – პარამეტრები ფიქსირებულია, ვიპოვოთ m - ის მნიშვნელობა, რომელიც F-ფუნქციას მიანიჭებს მაქსიმალურ მნიშვნელობას.

№	m, n	P(0)	P(1)	P(2)	P(3)	P(4)	P(5)
1	5, 1	0.3*10-6	0.008	0.0007	0.014	0.19	0.79
2	5, 2	0.25*10-6	0.00002	0.0006	0.012	0.15	0.08
3	5, 3	0.003*10-5	0.0004	0.00004	0.00012	0,2	0,8
4	5, 4	0.260.10-5	0.00015	0.0046	0.06	0.3	0.55
5	5, 5	0.24*10-5	0.0002	0.0052	0.07	0.42	0.50

(5.1) ფორმულის გათვალისწინებით შეგვიძლია დავწეროთ, როცა:

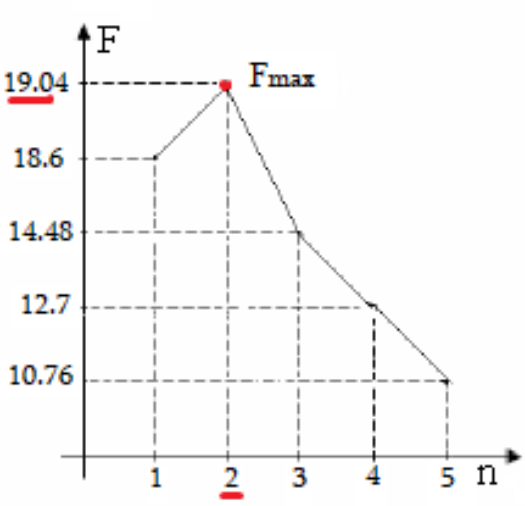
1) n=1, m=5, δ=4,7;
2) n=2, m=5, δ= 1,1;
3) n=3, m=5, δ= 4,8;
4) n=4, m=5, δ= 4,5;
5) n=5, m=5, δ= 4,4;

საბოლოოდ: F მნიშვნელობისათვის მივიღებთ ცხრილს:

№	c_1	c_2	c_3	n	α	δ	F
1	3.4	2.01	0.68	1	5	4.7	18.6
2	3.4	2.01	0.68	2	5	1.1	19.04
3	3.4	2.01	0.68	3	5	4.8	14.48
4	3.4	2.01	0.68	4	5	4.5	12.7
5	3.4	2.01	0.68	5	5	4.4	10.76

განხილული ხუთი შემთხვევიდან ოპტიმალური ვარიანტია მე-2, როცა m=5, n=2;

ვიზუალურად შეიძლება ეს პროცესი გამოვსახოთ გრაფიკის საშუალებით (ნახ.5.8).



ნახ.5.8.

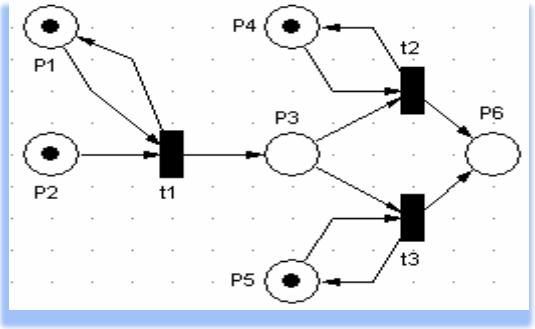
აბსცისათა ღერძი (n) შეესაბამება მომსახურების ობიექტთა მნიშვნელობებს, ხოლო ორდინატთა ღერძი (F) კი - მოგების კოეფიციენტებს. მივიღებთ: $F_{მოგ.}$= max.

ფაქტობრივად, ეს არის მთელრიცხვა პროგრამირების ამოცანა, რომლის ამოხსნა m-ის და n-ის რეალური მნიშვნელობისათვის ($n \leq m \leq 30$) შესაძლებელია კომპიუტერული გადარჩევის მეთოდით.

5.4. კომპიუტერული ქსელის რესურსების სინქრონიზების პროცესის მოდელირება პეტრის ქსელებით მრავალმომხმარებლურ რეჟიმში

განიხილება პროცესების, გამოთვლითი სისტემების ტექნიკური და პროგრამული უზრუნველყოფის სხვადასხვა სახის ამოცანების მოდელირების საკითხები. მათ შორის პარალელური პროცესების ეფექტური მართვისათვისაც.

5.9 ნახაზზე მოცემულია პეტრის ქსელის გრაფით კომპიუტერული სისტემის ძირითადი რესურსების წარმოდგენის ტიპური მაგალითი [115].



ნახ.5.9

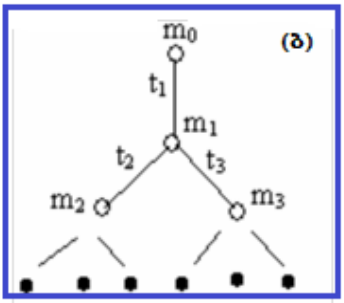
P_1 აღწერს მოვლენას, რომ *ცენტრალური პროცესორი თავისუფალია*; P_2 – მოთხოვნის შემოსვლა და t1-მოლოდინის გახსნა (გადასვლა); P_3 – შეტანა-გამოტანის მოთხოვნა მოლოდინის მდგომარეობაშია; P_4 – პირველი შ/გ-1 მოწყობილობა თავისუფალია; P_5 – შ/გ-2 თავისუფალია; P_6 – შედეგის მოლოდინი (მარკერის მოსვლა აქ t2 ან t3 – იდან ნიშნავს მიმდინარე მოთხოვნის შესრულებას.

პეტრის ქსელის გრაფის გადასასვლელები ასახავს

შემდეგ პროცესებს: t_1 – ცენტრალური პროცესორი ასრულებს მოთხოვნას; t_2 – მუშაობს შ/გ-1 ან t_3 – შ/გ-2 მოწყობილობა. გამოთვლითი სისტემის მუშაობის პროცესის მოდელირება მდგომარეობათა სივრცეში გამოისახება შემდეგ მარკირებათა ვექტორერებით და მიღწევადობის ხის ფრაგმენტით (ნახ.5.10).

	P_1	P_2	P_3	P_4	P_5	P_6
m_0	1	1	0	1	1	0
m_1	1	0	1	1	1	0
m_2	1	0	0	1	1	1

(ა)



ნახ.5.10. პეტრის ქსელის (ნახ.5.9) შესაბამისი პროცესის გადასვლების მატრიცა (ა) და მიღწევადობის ხის ფრაგმენტი (ბ)

მაგალითიდან კარგად ჩანს, რომ t_1-ის გაშვების შემდეგ (იხ. სტრიქონი m_1) მარკერი გადაადგილდება P_2-დან P_3-ში და ამ დროს წარმოიშობა კონფლიქტური სიტუაცია: შეიძლება t_2-ის ან t_3-ის გაშვება, მაგრამ ერთის გაშვება ბლოკავს მეორეს. m_1 მდგომარეობაში შესაძლებელია ახალი მოთხოვნის მოსვლა (P_2-ში ჩნდება მარკერი) და ამ სიტუაციაში შესაძლებელია პარალელურად t_1 და t_2 ან t_1 და t_3-ის შესრულება.

ახლა განვიხილოთ ჩვენი ერთ-ერთი ამოცანისათვის

პეტრის ქსელის ხელსაწყოს გამოყენების მაგალითი, ლოკალურ ქსელში განაწილებული ბაზების სინქრონიზებული მართვა მრავალმომხმარებლურ რეჟიმში.

ქსელის ტოპოლოგია შევირჩიოთ საერთო სალტით (ან ვარსკლავური) ერთი სერვერით. დავუშვათ, რომ მონაცემთა ბაზები განაწილებულია კვანძებში შერეული სტრატეგიის შესაბამისად (როგორც ყველაზე რთული მოდელი), ხოლო მათი კატალოგი მოთავსებულია ცენტრალიზებულ სერვერ-მონაცემთა ბაზაში.

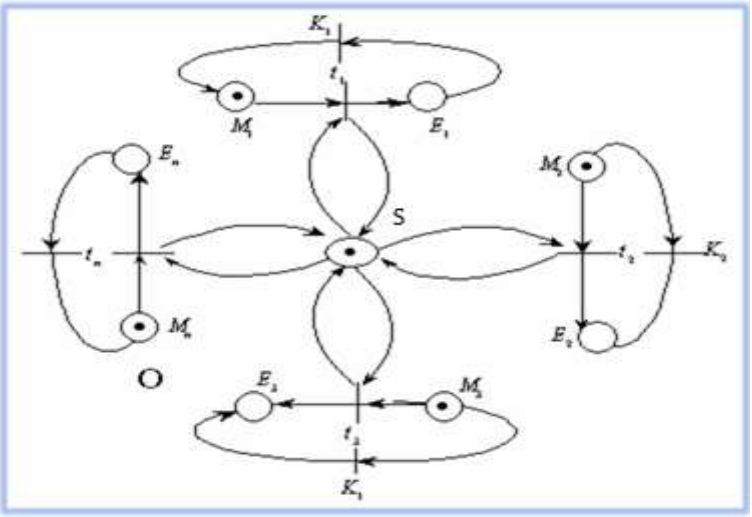
კვანძებში დასაშვებია მათი ლოკალური ბაზების მონაცემთა კატალოგების ქვესიმრავლეების არსებობა. მომხმარებლები სარგებლობენ თანაბარი პრიორიტეტით. მათი მოთხოვნების (ტრანზაქციების) მოსვლის ალბათობა განიხილება განაწილების პუასონის კანონით [113].

მოთხოვნები მუშავდება ლოკალურ კვანძებში თუ მათ სჭირდება სხვა ბაზების მონაცემები, წინააღმდეგ შემთხვევაში ისინი მიმართავენ სერვერს, და თუ შესაბამისი გადაცემის არხი და ბაზის ფაილები თავისუფალია, მიიღებენ გარკვეული პრედიკატით დამუშავებულ ინფორმაციას. თუ მონაცემები ბლოკირებულია სხვა ტრანზაქციებით, მაშინ ეს მოთხოვნა დგება შესრულების რიგში ან ხელმეორედ მიეწოდება გარკვეული ინტერვალის შემდეგ.

პეტრის ქსელების გამოყენებით მსგავსი პროცესების მოდელირება შესაძლებელია სინქრონიზაციის უნარის მქონე პროცედურებით [9,115]. ასეთ ამოცანათა კლასს მიეკუთვნება პროცესების ურთიერთგამორიცხვის, ჩიხების რეგულირების („ბრძენი ჩინელების შესახებ“), p და v - ოპერაციები სემაფორზე და სხვ. [116].

5.11 ნახაზზე მოცემულია მრავალმომხმარებლურ

რეჟიმში, ზოგადად *n*-კვანძისა და ერთი სერვერ-მანქანის შეთანხმებული ფუნქციონირების პროცესის ფრაგმენტი პეტრის ქსელის გრაფით, კერძოდ, მოთხოვნების (ტრანზაქციების) დასაკმაყოფილებლად.



ნახ. 5,11

სქემის პოზიციებია: M – მოთხოვნა ელოდება შესრულებას (პოზიციაში არის მინიმუმ ერთი მარკერი), ან არ ელოდება (არაა მარკერი); E –დამუშავებული მოთხოვნა (მარკერი ≥1), ან დაუმუშავებელი (=0); S – სინქრონიზაციის პოზიციაა და მისი საშუალებით აქ მოდელირდება საერთო რესურსი (მაგალითად, მონაცემები, გადაცემის არხი და ა.შ.). მისი მნიშვნელობაა – რესურსი თავისუფალია გამოსაყენებლად (მარკერი ≥1), ან არ არის თავისუფალი (=0). სქემის გადასასვლელებია: t – მოთხოვნა

მუშავდება; $k-$ მოთხოვნა დამუშავდა. საწყის მდგომარეობაში შეიძლება დავუშვათ, რომ მოთხოვნა ყველა კვანძში შემოსულია და ელოდება შესრულებას, S სერვერ-მანქანის რესურსი თავისუფალია (S = 1). მოცემულ სიტუაციაში შესაძლებელია ნებისმიერი ერთი გადასასვლელის გახსნა. აქ ადგილი აქვს კონფლიქტურ სიტუაციას, რადგან ერთის გახსნა აბლოკირებს დანარჩენებს.

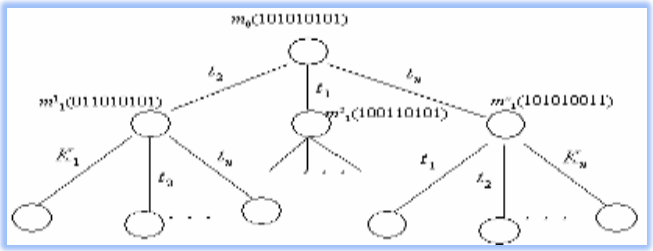
ვთქვათ, შესრულდა t_1, მაშინ მარკერი M_1-დან გადაადგილდება E_1-ში და მარკერი S-ში იგივე რჩება (გაიცემა 1 და ემატება 1). მეორე ბიჯზე შესაძლოა ორი პარალელური პროცედურის შესრულება:

1) K_1-ის გაშვება და ახალი მოთხოვნის მომზადება პირველ კვანძში;

2) ერთი რომელიმე გადასასვლელის გახსნა.

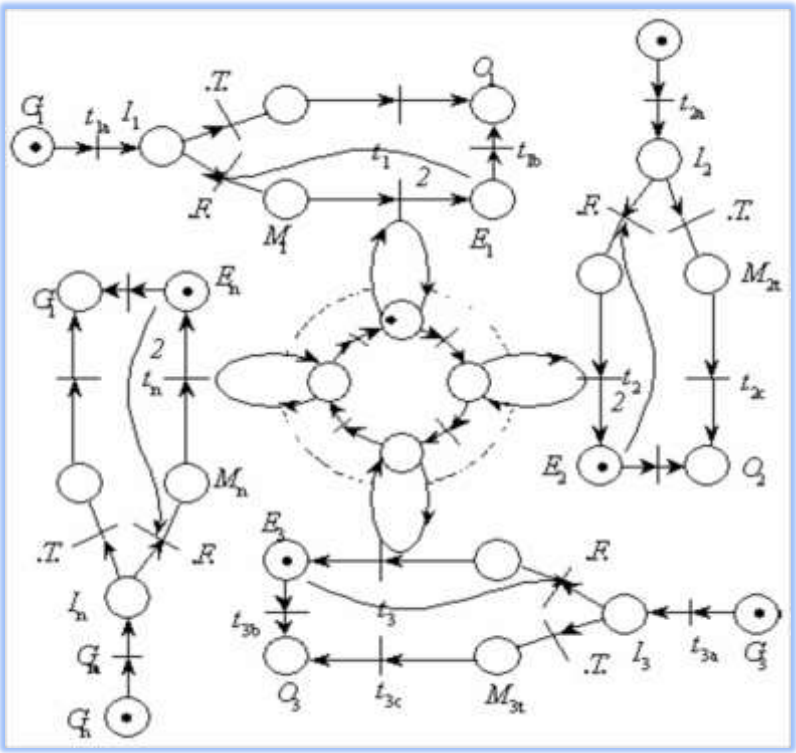
ამგვარად, მოთხოვნები მიმდევრობით გადამუშავდება ყველა კვანძში, თუმცა შესაძლოა პროცესი ისე წარიმართოს, რომ 1 და 2-ის მონაცვლეობით მივიღოთ უსასრულო მოლოდინის ციკლი [116].

5.12 ნახაზზე მოცემულია განხილული მაგალითის პეტრის ქსელის შესაბამისი მიღწევადობის ხის ფრაგმენტი (შესაძლოა უსასრულოც).



ნახ.5.12

ამოცანის დასმის გართულების საფუძველზე, ასაგები მოდელის დაზუსტების მიზნით, განვიხილოთ მოდელირების ასეთი პროცესი: სინქრონიზაციის პოზიციის (*S*) დეტალიზებული აღწერა, მოთხოვნების ფორმირების გენერატორის შემოტანა (*G*), მოთხოვნის ანალიზის (I) და მისი დამუშავების შედეგების (*O*) გაცემით. 5.13 ნახაზზე მოცემულია ამ პროცესის პეტრის ქსელის ფრაგმენტი.



ნახ.5.13.

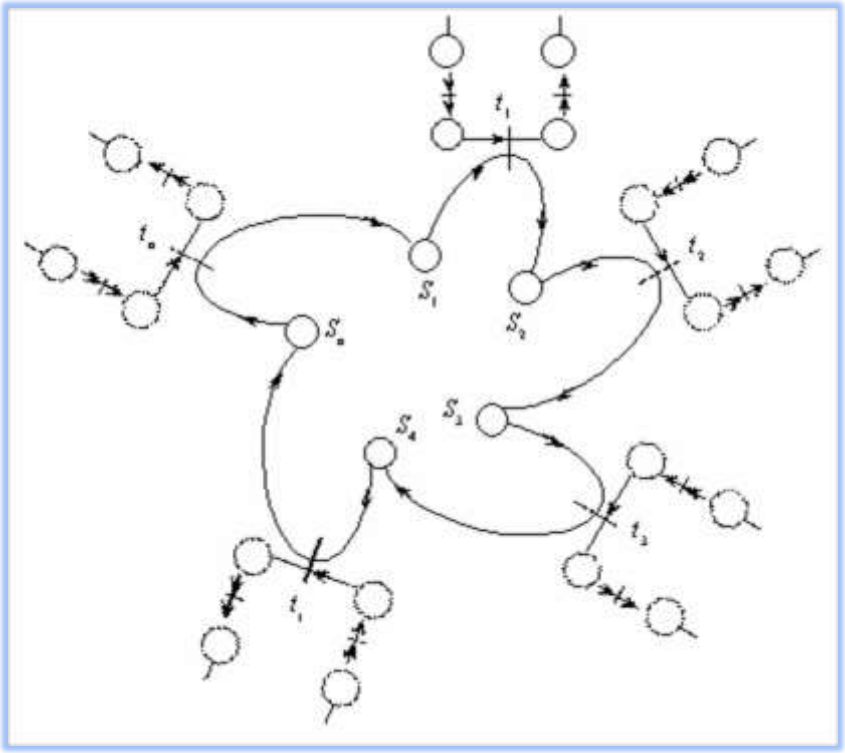
აქ არის მოთხოვნის ანალიზის ჩატარება რესურსების შესახებ (ლოკალურ კატალოგში). თუ მოთხოვნისათვის ყველა საჭირო რესურსი აქვეა, მაშინ გადაწყვეტილების (–I–.T./.F.) ბლოკი განსაზღვრავს მდგომარეობაში გადასვლას (ლოკალურ ბაზასთან მიმართვა), რომლის შემდეგაც გაიშვება გადასასვლელი (შედეგის გაცემა) *Oi*-ში.

თუ გადაწყვეტილების ბლოკი იძლევა .F.-ს, მაშინ მოთხოვნა ფორმირდება საერთო რესურსების გამოსაყენებლად და იგი მიმართავს სერვერ-მბ-ს (უფრო ზუსტად მოთხოვნა გადადის დამუშავების მოლოდინის მდგომარეობაში), M_i -ში თავსდება მარკერი და იგი ელოდება სინქრონიზაციას -1 პოზიციიდან.

ზოგადად შეიძლება მივიღოთ, რომ სინქრონიზაციის ***s*** ვექტორის ელემენტები გადასასვლელების მიმდევრობითი (სინქრონული გაშვებით) ღებულობს მარკერს, როდესაც კვანძში მოთხოვნაა და პოზიციაში მოვიდა მარკერი, მაშინ იხსნება გადასასვლელი.

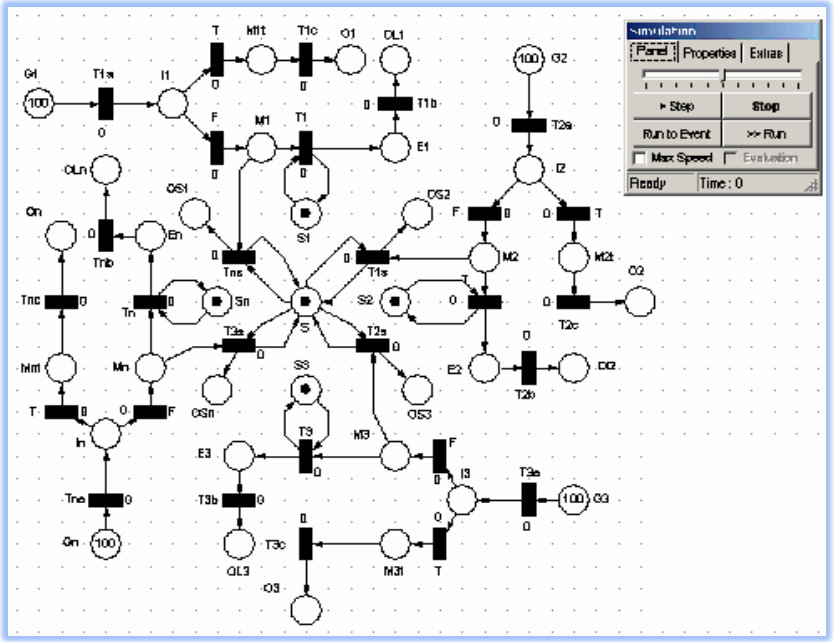
მოთხოვნა ღებულობს არხით საჭირო მონაცემებს და მარკერი გადადის E_i პოზიციაში. S_i-დან მარკერი გაიცემა t_j - ის გაშვების დროს, ამიტომ გადასასვლელი ბლოკირებულია. t_j-ის გაშვებით S_i-ში ბრუნდება მარკერი, რის შემდეგაც შესაძლოა მარკერის გადაგზავნა შემდეგი კვანძისათვის.

აღნიშნული ამოცანისათვის შესაძლებელია ალტერნატიული პეტრის ქსელის განხილვა, რომლის ფრაგმენტიც მოცემულია 5.14 ნახაზზე.



ნახ. 5.14

5.15 ნახაზზე ნაჩვენებია კლიენტ-სერვერ არქიტექტურით ორგანიზებული განაწილებული სისტემის პეტრის ქსელის მოდელი. იგი აგებულია Visual Object Net ++ გრაფო-ანალიზური რედაქტორის გარემოში, რაც იძლევა მისი შემდგომი ანალიზის საშუალებას პროცესების იმიტაციის გზით [117].



ნახ. 5.15

ამგვარად, შესაძლებელია კვლევის ობიექტის შემდგომი დაზუსტება, რაც ასახული იქნება პეტრის ქსელის შესაბამის დეტალიზებულ ფრაგმენტში. ამიტომ დამპროექტებელი თვითონ წყვეტს, თუ ობიექტის ქცევის რომელი დონის დეტალიზაციის მოდელირებას აპირებს.

პეტრის ქსელის ანალიზისთვის შეიძლება აგრეთვე გადასასვლელებისათვის დაყოვნების დროითი პარამეტრების განხილვა. ამ მხრივ საყურადღებოა პეტრის ქსელის დეტერმინირებული და სტოქასტური მოდელირების ფუნქციონირების ანალიზი [114].

5.5. პეტრის ქსელების მიზეზ-შედეგობრივი პროცესების პრედიკატულ ფორმაში ასახვა

ბერლინის ჰუმბოლდტის უნივერსიტეტის პროფესორის, ვოლფგანგ რეისიგის მიერ („დაპროგრამების თეორიის" კათედრის გამგე, კარლ-ადამ პეტრის მოწაფე) ღრმად იქნა შესწავლილი და წარმოდგენილი პეტრის ქსელების გამოყენების შედეგები ობიექტების ქცევის მოდელირებისა და ანალიზისათვის [118, 119].

ვ. რეისიგმა, გ. სურგულაძემ და დოქტორანტმა დ. გულუამ ერთობლივი კვლევის საფუძველზე შეიმუშავეს პეტრის ქსელების მოდელების, მონაცემთა რელაციური ბაზებისა და ობიექტ-ორიენტირებული პროგრამირების თეორიების კავშირი (იზომორფიზმის თვალსაზრისით) ნაწილობრივ მოწესრიგებულ სისტემებთან, ასახვის პრედიკატულ და რელაციურ ფორმებთან და ა.შ. [120-122].

ამჯერად, ჩვენი მიზანია ერთ-ერთი შედეგის კონკრეტული მაგალითით წარმოდგენა, განაწილებული რესურსების სინქრონიზაციის მოდელის შესაბამისი პეტრის ქსელის ანალიზის და პრედიკატული ფორმის საფუძველზე.

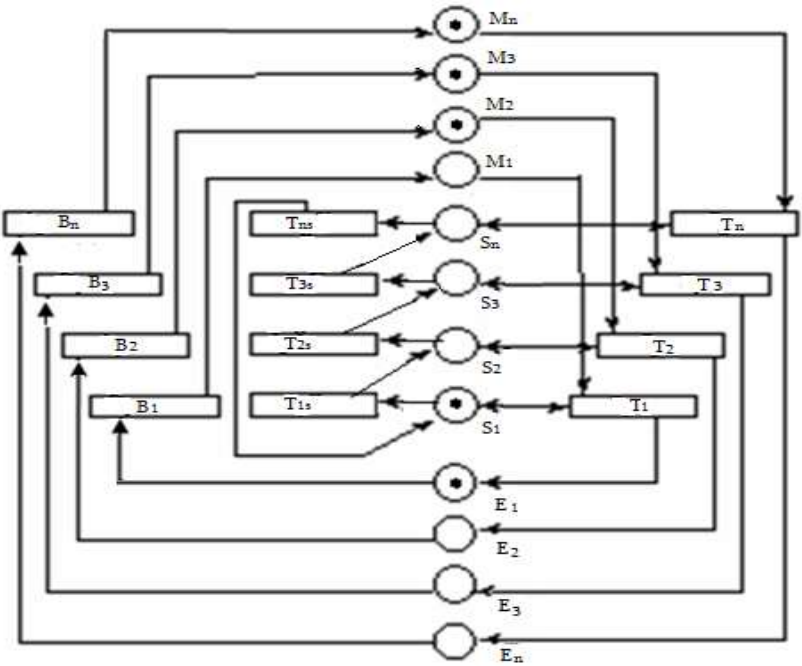
5.15-ა,ბ ნახაზების შესაბამისად განვიხილოთ მიზეზ-შედეგობრივი დამოკიდებულების მოდელის ფრაგმენტი. აქ m და e პრედიკატები განიხილება როგორც პოზიციები, ხოლო M_i და E_i- მოთხოვნები – როგორც მარკერები.

პეტრის ქსელის პოზიციების განზოგადებით მივიღებთ პრედიკატებს (ნახ.5.16):

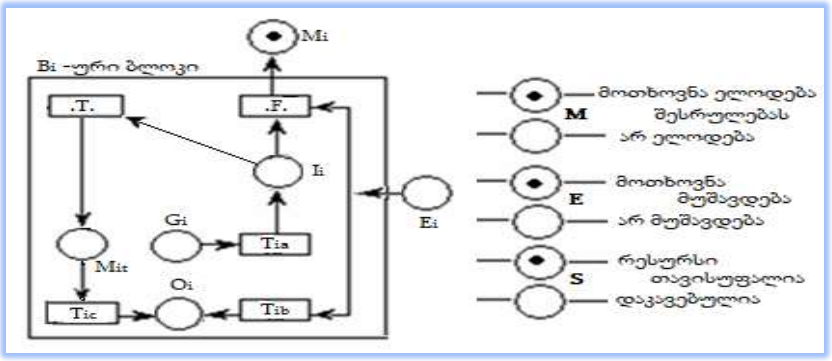
m(M_1,M_2, . . . , M_n) – „მომლოდინე მოთხოვნები" (m(P_i,i=2,n);

e(E_1,E_2, . . . , E_n) – „დამუშავების მოთხოვნები" (e(P_1);

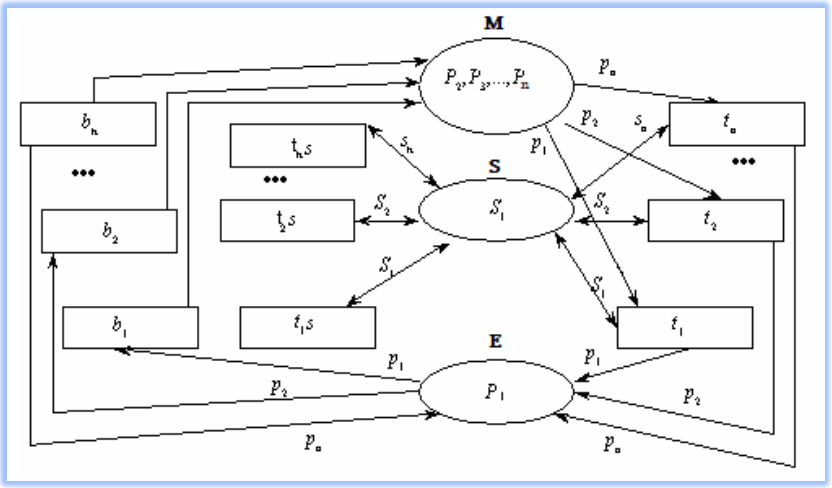
s(S_1,S_2, . . . , S_n) – „თავისუფალი რესურსები" (s(S_1) და ა.შ.



ნახ.5.15-ა

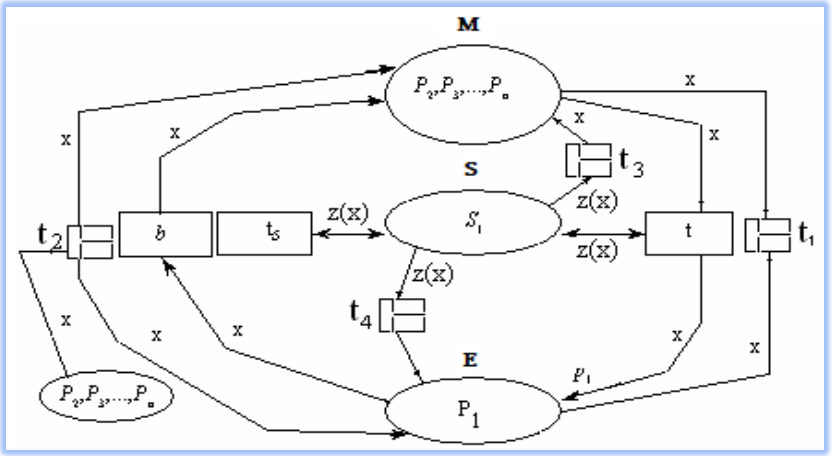


ნახ.5.15-ბ



ნახ. 5.16

შემდეგ ბიჯზე განვაზოგადებთ გადასასვლელებს და მივიღებთ 5.17 ნახაზზე წარმოდგენილ სურათს.



ნახ. 5.17

აქ x ცვლადია, რომელიც იღებს კონკრეტულ მნიშვნელობებს, ხოლო z(x)–ით აღიწერება სემანტიკური კანონი, მაგალითად, რესურსების განაწილების შესახებ:

z: P->S, სადაც P_i->S_{i+1}, if i=1,n-1 და P_n->S_1

პრედიკატებს შორის არსებული ფაქტორების ასახვის სქემა მოცემულია აქვე. ფაქტებს აქვს შემდეგი ლოგიკური გამოსახვა და სემანტიკა:

t_1: ~(m(x) ^ e(x)): მოთხოვნა (ტრანზაქცია), რომელიც მუშავდება, არ ელოდება შესრულებას და მოთხოვნა, რომელიც ელოდება შესრულებას, არ მუშავდება;

t_2: p(x) -> m(x) V e(x): ყოველი მოთხოვნა მუშავდება ან ელოდება შესრულებას, სხვა მოქმედება გამორიცხულია;

t_3: s(z(x)) -> m(x): თუ რესურსი თავისუფალია, მაშინ მოთხოვნა არ მუშავდება;

t_4: ~(e(x) ^ s(z(x))): თუ მოთხოვნა მუშავდება, მაშინ რესურსი დაკავებულია, და თუ რესურსი არაა დაკავებული, მაშინ მოთხოვნა არ მუშავდება.

5.6. მონაცემთა საცავის ინფორმაციული ბლოკების დამუშავების პროცესების ასახვა პეტრის ქსელებით

ინფორმაციული სისტემების სპექტრი დღეისათვის ძალზე ფართოა. მათი გამოყენების მნიშვნელობა და აქტუალურობა მთელ მსოფლიოში განსაკუთრებით გაიზარდა „კოვიდ-19“ პანდემიის ფონზე. ინფორმაციული სისტემების კლასიკური და ახალი ტიპები შემდეგია:

- *MIS – Management Information System (მენეჯმენტის საინფორმაციო სისტემები);*
- *ERP – Enterprise Resource Planing (საწარმოო რესურსების დაგეგმვა (მართვა);*
- *DSS – Decision Support System (გადაწყვეტილების მხარდამჭერი სისტემები);*
- *EXP – Expert Systems (AI, ექსპერტული სისტემები (ხელოვნური ინტელექტი);*
- *KMS – Knowledge Management System (ცოდნის მართვის სისტემები);*
- ***WHS – Warehousing System (OLAP, Data Mining, BigData) (მონაცემთა საცავების სისტემები);***
- *OAS – Office Automation System (ოფისის ავტომატიზაციის სისტემები, მაგ., Office-365)*
- *და სხვ.*

ასეთი სისტემების დაპროექტება და რეალიზაცია ხორციელდება უახლესი ინფორმაციული ტექნოლოგიებით, რომელთა საფუძველიც ობიექტ-, პროცეს- და სერვის-ორიენტირებული მიდგომებია [25].

ორგანიზაციული სისტემების მართვის პრობლემების და ამოცანების გადაწყვეტის თანამედროვე კომპიუტერული ტექნიკის

და ინფორმაციული ტექნოლოგიების გამოყენების ერთ-ერთი აქტუალური და მნიშვნელოვანი მიმართულებაა *მონაცემთა საცავები (რეპოზიტორიები)*, რაც უკავშირდება მრავალდონიანი განაწილებული საინფორმაციო სისტემების შექმნას [9,63].

მონაცემთა საცავების დამკვიდრება გამოწვეული იყო იმ აუცილებლობით, რომელიც განაპირობა მართვის დიდ და რთულ სისტემებში განაწილებული ინფორმაციული ბაზების მონაცემთა „საწყობში“ თავმოყრამ.

ეს უკანასკნელი შესაძლებელს ხდის მომხმარებელს მიეცეს მონაცემთა ინტერაქტიული კრიტერიუმებით განსაზღვროს ტექნიკური საშუალება, საჭირო მონაცემთა სტრუქტურის ინტენსიურობის დასადგენად. ამის გამო ანალიზური პროცესების სისტემა ეყრდნობა მრავალგანზომილებიან მონაცემთა სტრუქტურას, რომელიც მონაცემთა ადეკვატურ პრეზენტაციას უწყობს ხელს.

ბოლო პერიოდის განმავლობაში პროგრამული ინდუსტრიის სწრაფმა განვითარებამ მნიშვნელოვან წარმატებას მიაღწია. მათ შორის უნდა აღინიშნოს მონაცემთა ბაზების მართვის სისტემების პროგრამული პაკეტების უდიდესი წილი მსოფლიო ბაზარზე, რომელთა შორის ღირსეული ადგილი უკავია ORACLE, Ms SQL Server, My SQL და სხვ. [64].

როგორც აღვნიშნეთ, მრავალდონიანი განაწილებული საინფორმაციო სისტემების მონაცემთა საწყობებში თავს იყრის ამ ობიექტისათვის მნიშვნელოვანი ინფორმაციული ბლოკები, რომელთა კლასიფიკაცია შეიძლება კონტექსტური ასპექტებით განხორციელდეს.

მაგალითად, თუ განვიხილავთ *დიდი სავაჭრო ცანტრების* ინფორმაციული ობიექტების ისტორიებს, მათი პროდუქციის

კონიუნქტურით, ფასებით, საქონელბრუნვით, თანამშრომლებით, სავაჭრო გეგმებითა და ფაქტობრივი ფინანსური შემოსავლებით, პარტნიორებითა და კონკურენტებით, კლიენტებითა და საერთო ბაზრის მოთხოვნილებებით - საქმე გვექნება დიდი მოცულობის ინფორმაციასთან. შეიძლება ითქვას, რომ მონაცემთა საცავები იყო ერთ-ერთი პირველწყარო და მიზეზი დიდ მონაცემთა სისტემების წარმოშობისა და შემდგომი განვითარების, ძლიერი ანალიტიკური და სერვისული პროგრამული ნაწილით, სადაც აქტიურად ჩაერთო ხელოვნური ინტელექტი და მანქანური დასწავლის ალგორითმები და მეთოდები.

ინფორმაციული ბლოკები, რომლებიც მონაცემთა საცავებშია განაწილებული, მიზანმიმართულად თავსდება ინტერნეტ–გვერდებზე და ხელმისაწვდომია ფართო მომხმარებლისთვის.

ობიექტ-ორიენტირებული ანალიზისა და დაპროექტების მიდგომა გულისხმობს არა მხოლოდ საინფორმაციო საცავების შექმნას, არამედ მისი ინფორმაციული ბლოკების მიზნობრივად დამუშავების პროგრამების პაკეტების შექმნას და გამოყენებას.

კლასისთვის, როგორც მართვის საინფორმაციო სისტემის ძირითადი კომპონენტისთვის, დამახასიათებელია ინკაფსულაციის, მემკვიდრეობითობის და პოლიმორფიზმის თვისებები. ინკაფსულაცია კი გულისხმობს კონტენტ-შესაბამისი მეთოდების შემუშავებას ამ ინფორმაციული ბლოკების დასამუშავებლად.

მაგალითად, დიდ სავაჭრო ცენტრებისთვის ასეთი მეთოდები შეიძლება იყოს მომხმარებელთა მოთხოვნის განსაზღვრა (მარკეტული გამოკვლევის ინფორმაციული ბლოკი), სავაჭრო გეგმებისა და საქონელბრუნვის მართვა (ისტორიული, ტრადიციული, რეგიონალური ასპექტების გათვალისწინებით), მომხმარებელთა (კლიენტთა) მომსახურების ფორმებისა და პროცესის

გაუმჯობესება (რიგების შემცირება, ყურადღებიანი მომსახურება და ა.შ.) პროგნოზის, ანალიზის, ოპტიმიზაციის მეთოდების გამოყენება და ა.შ.

მაგალითად, მასობრივი მომსახურების პროგრამული პაკეტი GPSS_World დამუშავებულია Minuteman Software (აშშ) კომპანიის მიერ [123]. იგი გამოიყენება დისკრეტული და უწყვეტი პროცესების კომპიუტერული, იმიტაციური მოდელირებისათვის. მაღალ დონეზე ასახავს ინტერაქტიულობას და ინფორმაციის ვიზუალურ წარმოდგენას.

განვიხილოთ კონკრეტული ამოცანა „სავაჭრო ცენტრში მოლარის მომსახურება კლიენტებთან“. პროცესი ასეთია: კლიენტები შემოდიან მარკეტში, შეარჩევენ მათთვის სასურველ პროდუქციას და ელოდებიან მომსახურებას. სიტუაცია შეიძლება იყოს ს ხ ვ ა დ ა ს ხ ვ ა:

ა) მოლარე დაკავებულია და კლიენტი დგება რიგში;

ბ) მოლარე თავისუფალია და ემსახურება კლიენტს.

დავუშვათ ყოველი კლიენტის გამოჩენის მომენტი და მომსახურების დრო მოლარისთვის ცნობილია.

ჩვენი მიზანია შევადგინოთ ისეთი სისტემის ფუნქციონირების იმიტაცია, რომლის დროსაც ცნობილი იქნება მოლარე დროის რამდენი პროცენტის განმავლობაში იქნება თავისუფალი და დროის რა შუალედი დასჭირდება კლიენტს მარკეტში ყოფნისათვის, რათა არ მოხდეს დიდი რიგების წარმოქმნა და კლიენტის დაკარგვა.

სისტემის მდგომარეობის ცვლილების დინამიკა დროში პირველ რიგში უნდა განისაზღვროს სამოდელირებელი ობიექტის მდგომარეობით. ჩვენ შემთხვევაში მოლარის მდგომარეობით (დაკავებულია ან თავისუფალი) და

კლიენტთა რაოდენობით. სისტემის მდგომარეობა შეიცვლება ა) მარკეტში კლიენტთა შემოსვლით ბ) მოლარის მომსახურების დამთავრებით და მომდევნო კლიენტის მიღებით. იმიტაციის ილუსტრირებისათვის სისტემის მდგომარეობას ჩვენ განვსაზღვრავთ ხდომილებათა (მოვლენების) დროითი მოწესრიგებით, რაც შეესაბამება მარკეტში კლიენტთა შემოსვლას და გასვლას.

იმიტაციის შედეგი გამომავალი მონაცემების შესაბამისად წარმოდგენილია 5.1 ცხრილში.

შემოსვლის მომენტი და კლიენტთან მომსახურეობის დრო ცხრ.5.1

კლიენტის ნომერი	შემოსვლის მომენტი	მომსახურების დრო
1	4,6	4,9
2	12,5	3,6
3	17,2	4,5
4	17,9	2,4

ამ ცხრილის გათვალისწინებით შევადგინოთ 5.2 ცხრილი, რომლისთვისაც საჭიროა ჩავთვალოთ, რომ დროის საწყის მომენტში სისტემაში არ არის კლიენტი, მოლარე თავისუფალია და პირველი კლიენტი შემოდის, როცა ის 4,6-ის ტოლია.

მე-2 ცხრილში 1-ელი და მე-2 სვეტები არსებულია პირველიდან, მომსახურეობის საწყისი დრო არის მე-3 სვეტში, რომელიც დამოკიდებულია იმაზე, მომდევნო კლიენტმა დატოვა თუ არა მარკეტი. მე-4 სვეტში წასვლის დრო გამოითვლება, როგორც მესამე სვეტის შესაბამისი

ელემენტების ჯამი და მოცემული კლიენტის მომსახურების დრო, რომელიც გამოითვლება 5.1 ცხრილის დახმარებით. ყოველი კლიენტის ყოფნის დრო რიგში და მარკეტში გამოითვლება 5.2 ცხრილის დახმარებით. ამ გარდაქმნათა საშუალო მნიშვნელობა შესაბამისად ტოლია 2,27 და 5,97.

ცხრილი 5.2. შეიცავს შემაჯამებელ ინფორმაციას კლიენტთან მიმართებაში, მაგრამ არ შეიცავს ცნობებს მოლარეზე და რიგის სიგრძეზე. ასეთი ინფორმაციის მისაღებად საჭიროა გამოვიკვლიოთ ამ სიტუაციასთან დაკავშირებული მოვლენები.

„კლიენტის შემოსვლა“ – ხდომილების დადგომისას შემდგომი სიტუაცია მარკეტში განისაზღვრება მოლარის მდგომარეობით. თუ მოლარე თავისუფალია, ის გადადის მდგომარეობაში „დაკავებულია“ და ემსახურება კლიენტს. ამით იგეგმება ხდომილება – „მოცემული კლიენტის წასვლა“, რომელიც დროის მომენტში ტოლია მიმდინარე დროს, დამატებული მომსახურეობის დრო.

ცხრ.5.2

კლიენტის ნომერი (1)	შემოსვლის მომენტი (2)	მომსახურ. დაწყების მომენტი (3)	წასვლის მომენტი (4)	რიგში დგომის დრო (5)=(3)-(2)	მარკეტში დაყოფილი დრო (6)=(4)-(2)
1	4,6	4,6	9,5	0	4,9
2	12,5	12,5	16,1	0	3,6
3	13,6	17,2	26,2	3,6	12,6
4	14,8	17,9	19,3	3,1	4,5

თუ მოლარე დაკავებულია, კლიენტთა მომსახურება არ შეიძლება დაიწყოს და, შესაბამისად, კლიენტი დგება რიგში (რიგის სიგრძე იზრდება ერთით).

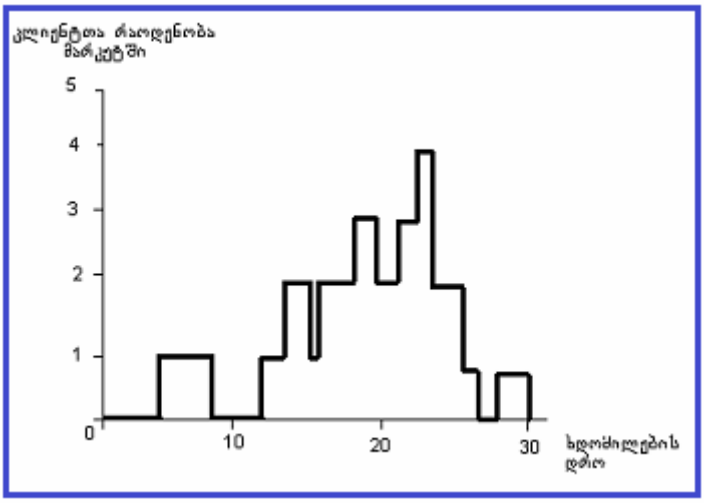
მოვლენათა დამუშავების ლოგიკა „კლიენტი ტოვებს მარკეტს“ დამოკიდებულია რიგის სიგრძეზე. თუ რიგში დგას ერთი კლიენტი მაინც, მაშინ მოლარე ითვლება დაკავებულად, თუ კლიენტი ვერ შეძლებს დალოდებას, ტოვებს რიგს (რიგის სიგრძე შემცირდება ერთით). თუ რიგი თავისუფალია მოლარე ჩაითვლება თავისუფლად.

5.3 ცხრილში აღწერილია მოლარის მდგომარეობის ხდომილება, ორიენტირებული აღწერა და მარკეტში კლიენტთა რაოდენობა.

ცხრ.5.3

შემოსვლის მოვლენა	კლიენტის ნომერი	ხდომილების სახელი	რიგის სიგრძე	კლიენტთა რაოდენობა	მოლარის მდგომარეობა	მოლარის დაუკავებელი დრო
0,0	-	დასაწყისი	0	0	თავისუფ.	-
4,6	1	შემოსვლა	0	1	დაკავებ.	4,6
5,2	1	გასვლა	0	0	თავისუფ.	-
5,9	2	შემოსვლა	1	2	დაკავებ.	
7,3	3	შემოსვლა	2	3	დაკავებ.	
8,2	4	გასვლა	0	0	თავისუფ.	
12,5	5	შემოსვლა	1	1	დაკავებ.	3,6
13,6	3	შემოსვლა	1	4	დაკავებ.	8,3
13,9	6	გასვლა	1	1	დაკავებ.	
14,8	7	გასვლა	2	3	დაკავებ.	2,1
18,7	4	შემოსვლა	0	0	თავისუფ.	
19,4	5	გასვლა	1	1	დაკავებ.	
20,5	8	გასვლა	0	0	თავისუფ.	
26,8	9	შემოსვლა	2	2	დაკავებ.	
28,7	10	შემოსვლა	0	0	თავისუფ.	
30	10	გასვლა	0	0	თავისუფ.	

ამ სიდიდეთა ცვლილება დროში შეიძლება გამოვსახოთ გრაფიკულად (ნახ.5.18)



ნახ.5.18

იმიტაციის შედეგი გვიჩვენებს რომ პირველი 30 წუთის განმავლობაში მარკეტში საშუალოდ ერთდროულად იმყოფებოდა 0,9 კლიენტი, ხოლო მოლარე თავისუფალი იყო დროის 18,6 %-ით.

ქრონოლოგიურად, რომ დავალაგოთ შემოსვლის და გასვლის მოვლენები, საჭიროა შემოვიტანოთ ხდომილებათა ჩანაწერები, რაც ექვემდებარება შემდგომ დამუშავებას. უნდა მოხდეს მომდევნო ხდომილების დადგომის მომენტის დაფიქსირება და აგრეთვე გასვლის. ამ მომენტების შედარებით განისაზღვრება თუ რომელი მოვლენა უნდა იქნას შერჩეული.

მოვლენის დადგომის დროს სისტემის მდგომარეობა

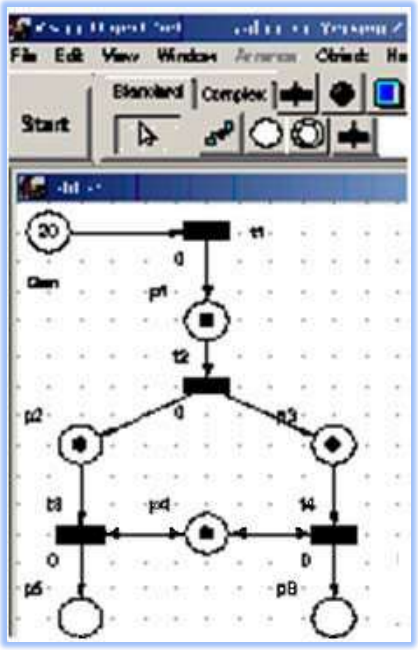
იცვლება შესაბამისად ლოგიკურ-მათემატიკური დამოკიდებულებით, რაც უკავშირდება ამ ხდომილებებს.

სისტემის მდგომარეობის ცვლილება შეიძლება განვიხილოთ ორ ასპექტში:

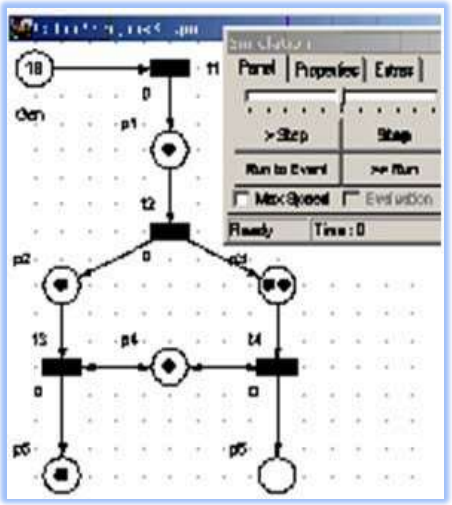
1) როგორც პროცესი, რომელშიც კლიენტს ემსახურებიან (კლიენტის თვალსაზრისი) და

2) როგორც მოვლენათა თანამიმდევრობა, რომელიც იწვევს მოლარის მდგომარეობის ცვლილებას (მოლარის თვალსაზრისი).

აღნიშნული პროცესებისთვის აგებულ იქნა პეტრის ქსელის გრაფი, რომელიც 5.19 ნახაზზეა წარმოდგენილი.



ნახ.5.19-ა



ნახ.5.19-ბ

იგი საშუალებას იძლევა შევარჩიოთ მომსახურების ისეთი ოპტიმალური მდგომარეობა, როდესაც კლიენტთა მომსახურება შესაძლებელია მინიმალური დროის განმავლობაში

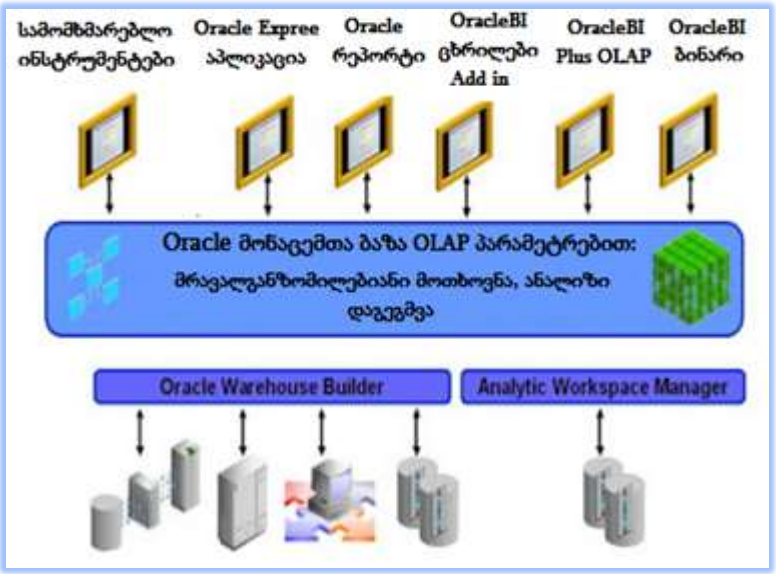
5.7. მონაცემთა საცავის რეალიზაცია Oracle ბაზაში

როგორც ზემთ აღვნიშნეთ, OLAP კუბი აღწერს მონაცემებს და შესაძლებელს ხდის მათ მარტივად მოპოვებას, ამასთან მონაცემები შესაძლებელია ერთ ან რამდენიმე კუბის ღერძებზე წარმოვადგინოთ და ჩავუტაროთ მრავალგანზომილებიანი ანალიზი [64, 125].

ბოლო პერიოდში OLAP ტექნოლოგიის დანერგვა და გამოყენება ღებულობს ფართო მასშტაბებს. ერთ-ერთ ევოლუციურ

ნაბიჯად შეიძლება ჩაითვალოს Oracle 12g ვერსია, სადაც ორგანიზებული და მატერიალიზებულია OLAP კუბი, რომელიც ეხმარება მონაცემთა საცავებში მოთხოვნების დაყენებასა და მათ ფორმულირებში, ასევე მომსახურების პროცესში, მაგალითად, როგორიცაა მონაცემთა აქტუალიზაცია, განახლება, დამუშავება და ა.შ. [64, 124].

Oracle OLAP არის მრავალგანზომილებიანი ანალიზური მექანიზმი, რომელიც აგებულია Oracle12g მონაცემთა ბაზის საფუძველზე. იგი უზრუნველყოფს რთულ გამოთვლებს, სადაც გამოიყენება მარტივი SQL მოთხოვნის ენა და ხორციელდება მონაცემთა ცენტრალიზებული მართვა. Oracle OLAP-ის არქიტექურას აქვს 5.20 ნახაზზე მოცემული სახე.



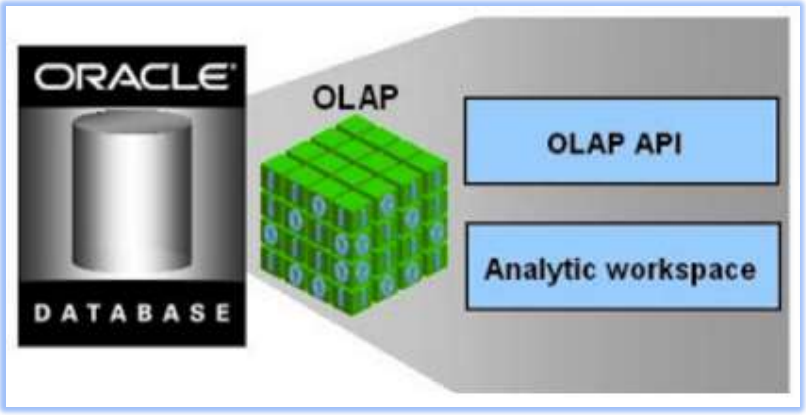
ნახ.5.20. Oracle OLAP -ის არქიტექურა

Oracle OLAP სისტემა ერთიანი ფუნქციონალური პლატფორმაა მონაცემთა საცავის და ბიზნესანალიზური სისტემის ასაგებად. ერთიანი პლატფორმა ემყარება საიმედო და იაფ Grid ინფრასტრუქტურას, რომელიც ამავე დროს არის უნიკალური, ინტეგრირებული პლატფორმა ანალიზისთვის.

Oracle OLAP მომხმარებელს აძლევს საშუალებას გამოიყენოს ანალიზური საშუალებები, მაგალითად, ფინანსურ მოდელებში, პროგნოზირებაში, დროითი მწკრივის გამოთვლაში, რეგრესიასა და ა.შ. ასობით ანალიზური ფუნქცია შესაძლებელია მარტივად გაერთიანდეს სამომხმარებლო ფუნქციაში, რომელიც გადაჭრის ნებისმიერი სახის ანალიზურ მოთხოვნას.

OLAP Oracle კუბი ბაზირებულია „ვარსკვლავის“ სქემაზე, განზომილებები ფორმულირებულია ფაქტების ცხრილის გარშემო.

Oracle OLAP კომპონენტები შედგება ორი ძირითადი პარამეტრისგან: OLAP API და ანალიზური სამუშაო გარემო (AW) (ნახ.5.21).



ნახ.5.21. Oracle OLAP - კომპონენტები

- **OLAP API ფუნქციონალი**

OLAP API – არის Java ენაზე ბაზირებული აპლიკაცია, პროგრამული ინტერფეისი, რომელიც უზრუნველყოფს ონლაინ რეჟიმში მონაცემებზე წვდომას და მათ ანალიზურ დამუშავებას, ხოლო OLAP API ფუნქციონალი ახდენს ონლაინ ანალიზური პროცესების იმპლემენტაციას და სამუშაო გარემოს მოწყობას [126].

მრავალგანზომილებიან მონაცემებთან წვდომა შესაძლებელია არა მხოლოდ Java დანართის საშუალებით, არამედ აპლიკაციითაც, რომელიც ორიენტირებულია SQL-ენაზე.

იმ შემთხვევაში თუ მონაცემები ფიზიკურად მდებარეობს რელაციურ ცხრილებში, მაშინ მათთან შესაძლებელია ნებისმიერ სტრუქტურირებულ მონაცემთა მოთხოვის ენის გამოყენება. ანალიზურ სივრცეში არის შესაძლებლობა შეიქმნას რელაციურ წარმოდგენათა გარემო (View), სადაც ობიექტებს ანალიზურ სივრცესთან ექნება წვდომა. ასეთი წარმოდგენა ავტომატურად გენერირდება სპეციალური შენახვის პროცედურების დახმარებით, სადაც გამოიყენება ობიექტ-ორიენტირებული მონაცემთა ბაზები Oracle და ცხრილური ფუნქციების ტექნოლოგიები.

Oracle Express server მრავალი წლის განმავლობაში იყო OLAP ტექნოლოგიის ძირითადი სერვერი, თავის უპრეცედენტო ფუნქციონალური შესაძლებლობების გამო, მაგრამ Oracle კორპორაციამ წარმოადგინა ახალი ვერსია Oracle OLAP, სადაც Oracle Express-ის ყველაზე მძლავრი მხარეებია და ჩააშენა მასში მონაცემთა ბაზის ბირთვი [124, 125].

Oracle OLAP სრულად ინტეგრირებულია რელაციურ მონაცემთა ბაზასთან, სადაც ყველა მონაცემი და მეტამონაცემი

იმართება და ინახება Oracle9i-ს საშუალებით, იგი გამოირჩევა სისტემის მონაცემთა დიდი მასშტაბების უზრუნველყოფით, მართვისას არის კონტროლირებადი და უსაფრთხო და ასევე არის ხელმისაწვდომი კომერციული თვალსაზრისით.

რევოლუციური თავისებურება, რაც Oracle და OLAP -ის ინტეგრირებაში აისახება არის ის, რომ შესაძლებელი გახდა მრავალგანზომილებიან მონაცემებში სტანდარტულ SQL ენაზე დაიწეროს მოთხოვნები, მონაცემთა შენახვის და ანალიზური დამუშავებისათვის.

სხვა შესაძლებლობებთან ერთად, Oracle OLAP-ს აქვს OLAP კატალოგი მეტამონაცემთა დონეზე, რომელშიც აღიწერება როგორც რელაციური, ასევე მრავალგანზომილებიანი მონაცემები, სადაც გამოიყენება ე.წ. მრავალგანზომილებიანი კუბები, Java და OLAP API.

აღნიშნული ინსტრუმენტებით შესაძლებელი ხდება აიგოს დანართები, რომელთანაც OLAP ქმნის საერთო ინტერფეისს.

ამგვარად, შეიძლება ითქვას, რომ Oracle OLAP არის მრავალგანზომილებიანი მონაცემთა ბაზა, სადაც მონაცემთა ბაზა Oracle პარალელურად მუშაობს OLAP-ის ფუნქციონალურ შესაძლებლობებთან.

გამოყენებული ლიტერატურა:

1. ჩოგოვაძე გ., გოგიჩაიშვილი გ., სურგულაძე გ. მართვის ავტომატიზებული სისტემები - გამოყენებითი მეცნიერება და მისი როლი ინფორმაციული საზოგადოების ფორმირებაში. *II-საერთაშ. სამეცნ.-ტექნ. კონფ. „საინფორმაციო საზოგადოება და განათლების ინტენსიფიკაციის ტექნოლოგიები".* თეზის.კრ., ISBN 978-9941-8-2620-7. სტუ, „IT კონსალტ. სამეცნ. ცენტრი". თბ., 2021. გვ.22.

2. ჩოგოვაძე გ., ფრანგიშვილი ა., ჯაგოდნიშვილი თ., სურგულაძე გ. ინფორმაციული საზოგადოება - მულტიდისციპლინური განათლების თანამედროვე გამოწვევა. სტუ-ს შრ.კრებ. „მას", N2(26), თბ., 2018, გვ. 19-24

3. ჩოგოვაძე გ., სურგულაძე გ., თოფურია ნ., ხარიტონაშვილი მ. ინფორმაციული საზოგადოება და ინტერდისციპლინური სწავლება ციფრული ტექნოლოგიების ბაზაზე. ISBN 978-9941-8-3338-0. მონოგრაფია. სტუ. „IT-კონსალტ. სამეცნ. ცენტრი", თბ., 2021. -360 გვ.

4. ვედეკინდი ჰარტმუტ (გერმანია), სურგულაძე გ., თოფურია ნ. განაწილებული ოფის-სისტემების მონაცემთა ბაზების დაპროექტება და რეალიზაცია UML-ტექნოლოგიით. ISBN 99940-57-17-0. სტუ. „ტექნიკ.უნივ.", თბ., 2006 -237 გვ.

5. ბოლხი გუნტერ (გერმანია), გოგიჩაიშვილი გ.,, სურგულაძე გ., პეტრიაშვილი ლ. მართვის ავტომატიზებული სისტემების ობიექტ-ორიენტირებული დაპროექტების და მოდელირების ინსტრუმენტები (MsVisio, WinPepsy, PetNet, CPN). ISBN 99940-56-77-8. სტუ. „ტექნ.უნივ.", თბ., 2013. -232გვ. gtu.ge/book/ims/GogichaiSurgul.pdf

6. მეიერ-ვეგენერი კლაუს (გერმანია), სურგულაძე გ., ბასილაძე გ. საინფორმაციო სისტემების აგება მულტიმედიური მონაცემთა

ბაზებით. მონოგრ., ISBN 978-9941-20-468-5. სტუ. „ტექნიკ.უნივ.“, თბ., 2014 -345 გვ. https://gtu.ge/book/gia_sueguladze/2giasurg_1.pdf

7. Bothe Klaus (Germany), Surguladze G. Objektorientierte Modellierung und Programmierung mit der UML. ISBN 99940-56-01-8. Berlin-Tbilisi. Humboldt Univ., GTU. 2006. 60 p.

8. რეისიგი ვოლფგანგ (გერმანია), სურგულაძე გ., გულუა დ. ვიზუალური ობიექტ-ორიენტირებული დაპროგრამების მეთოდები (BorlandC++Builder, PetriNet). ISBN 99928-943-9-3. სტუ. „ტექნ. უნივ.", თბ., 2001, 200 გვ.

9. სურგულაძე გ., პეტრიაშვილი ლ. მონაცემთა საცავის აგების ტექნოლოგია ინტერნეტული ბიზნესის სისტემებისათვის. მონოგრ., ISBN 99940-40-36-7 . სტუ, „ტექნ. უნივ.“. თბ., 2005. -200 გვ.

10. Data warehouse. Internet resource: https://en.wikipedia.org/wiki/Data_warehouse (15.12.21)

11. What is a Data Warehouse and Why Does It Matter To Your Business? Internet resource: https://www.talend.com/resources/what-is-data-warehouse/ (15.05.2021)

12. Chhabra M. Understanding EDW (Enterprise Data Warehouse) Simplified. 2021. Internet resource: https://hevodata.com/learn/enterprise-data-warehouse-edw-simplified/

13. Chan Cl. Data-warehouse in Banking industry 2017. https://medium.com/@clementchan/data-warehouse-in-banking-industry-7820319-41547. (10.11.21)

14. Finnegan P., Sammon D. Foundations of an Organisational Prerequisites Model for Data Warehousing, Proceedings of the 7th European Conf. on Information Systems (ECIS 2016), Copenhagen, June

15. Inmon W.H. Building the Data Warehouse, Second Edition, New York: John Wiley & Sons. 1998

16. Kachur, R. (2018) Data Warehouse Management Handbook, Paramus: Prentice Hall.

17. Sethi M. Data Warehousing and OLAP Technology. International Journal of Engineering Research and Applications (IJERA). ISSN: 2248-9622 www.ijera.com Vol. 2, Issue 2,Mar-Apr 2012, pp.955-960

18. Mueller-Stewens, G. (2003) Die Organisation als Gegenstand von Veränderungsprozessen, in H. Oesterle & R. Winter (Eds.), Business Engineering: Auf dem Weg zum Unternehmen des Informationszeitalters (pp. 133-145), Berlin: Springer (in German)

19. Meyer M., Winter, R. Organization of Data Warehousing in Large Service Companies: A Matrix Approach Based on Data Ownership and Competence Centers, Journal of Data Warehousing, 6 (4), 2018

20. Codd E.F., Codd S.B., Salley C.T. Providing OLAP (On-Line Analytical Processing) to User Analyst: An IT Mandate." Available from Arbor Software's. 1993.

21. Inmon, W.H. Building the Data Warehouse (Third Edition), New York: John Wiley & Sons, 2002

22. Abramson I. Data Warehouse: the Choice of Inmon versus Kimball. IAS Inc. https://www.ismll.uni-hildesheim.de/lehre/bi-10s/script/Inmon-vs-Kimball.pdf

23. Vassiliadis P., Quix Ch., Vassiliou Y., Jarke M. A Model for Data Warehouse Operational Processes. Conference Paper · March 2000. DOI: 10.1007/3-540-45140-4_30 · Source: CiteSeer

24. Kimball, R. The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses, New York: John Wiley & Sons. 2006

25. სურგულაძე გ. კომპიუტერული პროგრამირების მეთოდები და მეთოდოლოგიები (SP, OOP, VP, Agile, UML). ISBN 978-9941-

1900-1. სტუ. „IT-კონსალტინგ ცენტრი“. თბ., 2019. -200 გვ. https://gtu.ge/book/Surg_ProgMethod_2019.pdf

26. Гогичаишвили Г.Г., Сургуладзе Г.Г. Разработка прикладного программного обеспечения интегрированных информационных систем управления на основе UML. Georgian Electronic Scientific Journal, 2002, N1, 42-48. http://gesj.internet-academy.org.ge

27. სურგულაძე გ., შონია ო., ყვავაძე ლ. მონაცემთა განაწილებული ბაზების მართვის სისტემები (MsSQL Server, Access, InterBase, JDBC, Oracle). ISBN 99940-35-18-5. სტუ, „ტექნ.უნივ.", თბ., -230 გვ.

28. სურგულაძე გ., პეტრიაშვილი ლ. ვიზუალური დაპროგრამება C# ენის ბაზაზე ინფორმაციულ სისტემებისათვის (Visual StudioNET 2019 პლატფორმაზე). ISBN 978-9941-8-1708-3. სტუ. „IT-კონსალტინგ ცენტრი“. თბ., 2019. B5, -200 გვ.

29. სურგულაძე გ., ქრისტესიაშვილი ხ., სურგულაძე გიორგი. საწარმოო რესურსების მენეჯმენტის ბიზნეს-პროცესების მოდელირება და კვლევა. ISBN 978-9941-20-557-6. სტუ. „ტექნიკური უნივერსიტეტი“, თბ., 2015 -216 გვ. https://gtu.ge/book/gia_sueguladze/GiaSurgBPMN+ERP.pdf

30. სურგულაძე გ., გულუა დ. ქსელური არქიტექტურები ბიზნესისათვის. ISBN 978-9941-0-9842-0. სტუ. „IT-კონსალტინგ ცენტრი“. თბ., 2017. -269 გვ. https://gtu.ge/book/gia_sueguladze/SurGul_ServTechBusiness.pdf

31. ვაჩნაძე რ., თურქია გ., ლომსაძე პ. ბიზნესი. ტ.1,(ტ.2). „ESM“, თბილისი, 2005, (2007). 269 გვ. (272 გვ.)

32. ჯენსონი ე., მოვსესიანი ა., ოგნივცევი ს. მსოფლიო ეკონომიკა. (თარგ. რუს.), ISBN 99928-0-637-0. თბ., (მოსკოვი). 2003.

33. სურგულაძე გ., გულუა დ., კახელი ბ. პროგრამული აპლიკაციების აგება ვირტუალიზაციის პირობებში. ISBN 978-9941-8-0627-

4. სტუ. „IT-კონსალტინგის სამეცნ. ცენტრი“. თბილისი, 2019. -159 გვ. https://gtu.ge/book/SurGuluaKakheli_Virtualization.pdf

34. ბოტჰე კ., სურგულაძე გ., დოლიძე თ., შონია ო., სურგულაძე გ. თანამედროვე პროგრამული პლატფორმები და ენები. ISBN 99940-14-11-0. სტუ, „ტექნიკური უნივ.", თბ., 2003. -250 გვ.

35. სურგულაძე გ., დოლიძე ს. მომხმარებლის ინტერფეისის დაპროგრამება (AngularJS, ReactJS). ISBN 978-9941-8-0625-4. სტუ. „IT-კონსალტინგ ცენტრი“. თბ., 2019. -106 გვ.

36. სურგულაძე გ., კივილაძე გ. შესავალი NoSQL მონაცემთა ბაზებში. ISBN978-9941-0-9642-6. სტუ. „IT-კონსალტინგ ცენტრი“. თბ., 2017. -152 გვ.

37. Node.js. https://ru.wikipedia.org/wiki/Node.js (14.11.2021)

38. Secure Electronic Transaction. Internet resource: https://en.wikipedia.org/wiki/Secure_Electronic_Transaction (14.11.2021)

39. Digital economy. https://en.wikipedia.org/wiki/ Digital_economy

40. Taylor D. What is OLAP? Cube, Analytical Operations in Data Warehouse. Internet resource: https://www.guru99.com/online-analytical-processing.html (15.05.21)

41. Java Metadata Interface. Internet resource: https://en.wikipedia .org/wiki/Java_Metadata_Interface (14.11.2021)

42. What is HOLAP ? Internet resource: https://www.tutorialspoint .com/what-is-holap

43. IBM DB2 Warehouse Manager for AS/400 — Foundation for Business Intelligence Applications. Software Announcement 200-176. IBM United States. 2000. https://www.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=897/ENUS200-176&infotype=AN&subtype=CA&app - name=skmwww (10.12.21)

44. Cloud Data Warehouse Guide. Internet resource: https://www.qlik.com/us/cloud-data-migration/cloud-data-warehouse#:~:text=A%20cloud%20data%20warehouse%20is,for%20scalable%20BI%20and%20analytics

45. Portal.azure.com (10.11.21)

46. Azure SQL. Migrate, modernize, and innovate on the modern SQL family of cloud databases. https://azure.microsoft.com/en-us/products/azure-sql/#product-overview (20.12.21)

47. What Is Microsoft Power Automate ? in "Create Cross-Program Workflows Easily with the Modern Microsoft 365 Service Microsoft Power Automate". https://www.theprojectgroup.com/en/office-365-microsoft-flow (30.11.21)

48. About AWS (Amazon Web Services). Internet resource: https://aws.amazon.com/about-aws/ (25.10.21)

49. Rajnish Kumar Sharma. (2021). What is Google Cloud Platform (GCP) and Why Should You Choose it ? https://www.netsolutions.com/insights/what-is-google-cloud-its-advantages-and-why-you-should-adopt-it/#why-use-gcp (25.10.21)

50. Database. Deploy a multi-cloud database. Internet resource: https://www.mongodb.com/atlas/database (25.10.21)

51. ჩოგოვაძე გ., სურგულაძე გ., შონია ო. მონაცემთა და ცოდნის ბაზების საფუძვლები. თსუ, „განათლება“, თბ., 375 გვ.

52. What is a Data Warehouse and Why Does It Matter to Your Business? Internet resource: https://www.talend.com/resources/what-is-data-warehouse/ (1.01.22)

53. Albrecht J. Anfrage optimierung in Data-Warehause-Systemen auf grundlage des multidimensionalen Datenmodele. Fridrich-Alexander-Universitet, Erlangen-Nurnberg. 2002

54. სურგულაძე გ. ვიზუალური დაპროგრამება C#_2010 ენის ბაზაზე. ISBN 978-9941-20-021-2. სახელმძღვ., სტუ. „ტექნიკური უნივერსიტეტი". 2011, -445 გვ. https://gtu.ge/book/GiaSurg_C_2010.pdf

55. Data Mining - Cluster Analysis. https://www.tutorialspoint.com/data_mining/dm_cluster_analysis.htm (16.01.22)

56. ჩოგოვაძე გ., სურგულაძე გ., გულიტაშვილი მ., დოლიძე ს. პროგრამული აპლიკაციების ხარისხის მართვა: ტესტირება და ოპტიმიზაცია. ISBN 978-9941-20-629-2. სტუ. „IT-კონსალტინგ ცენტრი“. თბ., 2020. -363 გვ. https://gtu.ge/book/Surgu_Software-Quality.pdf

57. Codd's 12 Rules for Relational Database Management. Internet resource: https://olap.com/learn-bi-olap/codds-paper/ (14.02.22)

58. Чоговадзе Г.Г., Качибая В.В., Сургуладзе Г.Г. Теория реляционных зависимостей и проектирование логической схемы баз данных. ISBN 5-511-00072-8. Изд. Тбилисский Гос. Универс. 1988, -230 ст.

59. Lafler K.P. Querying the Data Warehouse with the SQL Procedure SELECT Statement. Software Intelligence Corporation. Spring Valley, CA 91979-1390. https://support.sas.com/resources/papers/proceedings/proceedings/sugi23/Advtutor/p41.pdf (17.10.21)

60. სურგულაძე გ., გულუა დ., თურქია ე. ბიზნეს-პროცესების მოდელირება პეტრის ქსელებით. ISBN 978-9941-14-125-6. სტუ. „ტექნიკ.უნივერს.“, თბ., 2008 -128 გვ. https://gtu.ge/book/gia_sueguladze/SurgEka_PetNet1.pdf

61. Bauer A., Günzel H. Data-Warehouse Systeme: Architektur, Entwicklung, Anwendung. ISBN 9783527813032. German. 2013. - 712 S.

62. Gerken W. Data-Warehouse-Systeme für Dummies. ISBN 9783527813032. Publisher Wiley-VCH. 2018.

63. ჩოგოვაძე გ., ფრანგიშვილი ა., სურგულაძე გ., მართვის საინფორმაციო სისტემების დაპროგრამების ჰიბრიდული ტექნოლოგიები და მონაცემთა მენეჯმენტი. ISBN 978-9941-20-790-7. სტუ. „ტექნიკ.უნივერსიტეტი“, თბ., 2017. -1001 გვ.

64. პეტრიაშვილი ლ., სურგულაძე გ. მონაცემთა მენეჯმენტის თანამედროვე ტექნოლოგიები (Oracle, MySQL, MongoDB,Hadoop). ISBN 978-9941-27-176-2. სტუ. „IT-კონსალტინგ ცენტრი“. თბ., 2017. 202 გვ. https://gtu.ge/book/PetriSurgu_DataManagmTechn.pdf

65. Wixom B.H., Watson H.J. An Empirical Investigation of the Factors Affecting Data Warehouse Success, MIS Quarterly, 25 (1), 2017, pp.17-41

66. Gane-Sarson Data Flow Diagram Tutorial. Internet resource: https://online.visual-paradigm.com/knowledge/software-design/gane-sarson-dfd-tutorial

67. Rumbaugh, Booch and Jacobson Methodologies. Internet resource: https://iq.opengenus.org/rumbaugh-booch-and-jacobson-methodologies/

68. Blayo F., Demartines P. Data analysis: How to compare Kohonen neural networks to other techniques ? Microcomputing Laboratory. INF - Ecublens, Switzerland. 2005

69. Agrawal G.L., Gupta H. Optimization of C4.5 Decision Tree Algorithm for Data Mining Application. Intern. Journal of Emerging Technology and Advanced Engineering. ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 3, Issue 3, March 2013. 341-345. Website: www.ijetae.com

70. C4.5 Algorithm. Internet resource: https://en.wikipedia.org/ wiki/C4.5_algorithm

71. Systems of reasoning based on analogous cases. Internet resource: https://studbooks.net/2104522/informatika/sistemy_rassuzhdeniy_osnove_analogichnyh_sluchaev (20.02.22)

72. Wang P., Zhang Ch., Cui H. Clementine Server: A data mining software for business solution. Internet resource. ppt-file. Internet resource. (20.12.22)

73. Machine Learning and Predictive Analytics Solutions - Altair. www.altair.com/knowledge-studio/knowledge-seeker-apache-spark

74. Scalable and vibrant community. http://people.apache.org/~isabel/content/users/basics/algorithms.html

75. K-Means Clustering in Julia. 2020. https://www.geeksforgeeks.org/k-means-clustering-in-julia/

76. SciPy K-Means – Syntax & Examples. https://www.tutorialkart.com/python/scipy/scipy-kmeans/

77. WEKA Data Mining. https://www.javatpoint.com/weka-data-mining

78. Data Mining Using SAS Enterprise Miner. https://www.wiley.com/en-us/Data+Mining+Using+SAS+Enterprise+Miner-p-9780470149010

79. Frolick M.N., Lindsey K. Critical Factors for Data Warehouse Failure, Journal of Data Warehousing, vol.8, N 1. 2013

80. Rui, Y., Carmona, V.I.S., Pourvali, M. et al. Knowledge Mining: A Cross-disciplinary Survey. Mach. Intell. Res. 19, 89–114 (2022). https://doi.org/10.1007/s11633-022-1323-6

81. Brownlee J. A Gentle Introduction to Expectation-Maximization (EM Algorithm). 2019. https://machinelearningmastery.com/expectation-maximization-em-algorithm/

82. PageRank. Internet resource: https://en.wikipedia.org/wiki /Page Rank (25.11.21)

83. Varagouli E. Everything You Need to Know About Google PageRank (Why It Still Matters). 2020. Internet resource: https://www.semrush.com/blog/pagerank/

84. Hwang, H.-G., Kua, C.-Y., Yenb, D. C., & Chenga, C.-C. (2012). Critical factors influencing the adoption of data warehouse technology: a study of the banking industry in Taiwan, Decision Support Systems, In Press, Corrected Proof.

85. Kazi Imran M., Qazi Baseer A. Use of Data Mining in Banking. Intern. Journ. of Engineering Research and Applications (IJERA) ISSN: 2248-9622, vol. 2, Issue 2, Mar-Apr 2012, pp.738-742

86. Data Mining For Financial Data Analysis Last Updated : 23 Feb, 2022. Internet resource: https://www.geeksforgeeks.org/data-mining-for-financial-data-analysis/

87. Weiss G.M. Data Mining in the Telecommunications Industry. Fordham University. USA. 2009

88. What is the role of data mining for the Telecommunication Industry? 2021. https://www.tutorialspoint.com/what-is-the-role-of-data-mining-for-the-telecommunication-industry (19.01.22)

89. Umamaheswari K., Janakiraman S. Role of Data mining in Insurance Industry. Publ. 2014. Economics. Internet resource: www. semanticscholar.org/paper/Role-of-Data-mining-in-Insurance-Industry-Umamaheswari-
Janakiraman/33093523a43c0fddb1b175fc9d45ed1740768583

90. Faheem Shakeel. Insurance Data Analytics: The Gold Mine for Insurers. 2022. https://www.damcogroup.com/blogs/insurance-data-analytics-for-insurers/

91. Zheng, Zijian. Application of Data Mining Techniques in Transportation Safety Study. text/dissertation. 2018. https://library. ndsu.edu/ir/handle/10365/28877

92. Mitroshin P., Shitova Y., Shitov Y., Vlasov D., Mitroshin A. Big Data and Data Mining Technologies Application at Road Transport Logistics. Transportation Research Procedia. Vol. 61, 2022, pp. 462-466

93. სურგულაძე გ.,სურგულაძე გ.,ქარქაშაძე ი., მჭედლიშვილი ა. ლოგისტიკის მენეჯმენტის მხარდამჭერი საინფორმაციო სისტემის აგება. ISBN 978-9941-8-2487-6. სტუ. „IT-კონსალტინგ ცენტრი“. თბ., 2020. -380 გვ. https://gtu.ge/book/Sugu_Logistics_IS.pdf

94. Rani Kumari, Kavita Saini. Automobile Industries using Data Mining and Predictive Analytics: An Industry 4.0 Approach. Jour. of Xi'an Univ.of Architecture & Technology. ISSN 1006-7930. Vol. XII, Issue V, 2020. 2977-2986 (30.04.21)

95. Dunskiy I. Data Mining in the Healthcare Industry: Key Benefits & Examples. https://demigos.com/blog-post/data-mining-in-the-healthcare-industry-benefits-examples/ (20.05.22)

96. Chogovadze G., Kiviladze G., Surguladze G. Modeling of Emergency Medical Service in Flu Season: Algorithm for Dispatching Ambulance Units. Bulletin "Moambe" of Georgian National Academy of Sciences, vol. 12, N.4. Tbilisi, 2018, pp. 47-52

97. Tyson J. How Firewalls Work. Internet resource: https://compu ter.howstuffworks.com/firewall.htm (19.01.22)

98. What is DMZ Network ? Internet resource: https://intelli paat.com/blog/what-is-dmz-network/ (25.01.22)

99. Proxy Server. https://www.tutorialspoint.com/internet_technolo gies/proxy_servers.htm (10.01.22)

100. Braun T. Virtual Private Network Architecture, 2000. Internet resource: https://www.researchgate.net/publication/2405348_Virtual_Private_Network_Architecture

101. IPsec (Internet Protocol Security). Cisco Lessons. https://networklessons.com/cisco/ccie-routing-switching/ipsec-internet-protocol-security (10.12.21)

102. What is Point-to-Point Tunneling Protocol (PPTP)? https://networkencyclopedia.com/point-to-point-tunneling-protocol-pptp/ (10.12.21)

103. About the IETF. Internet resource: https://www.internetsociety.org/about-the-ietf/ (20.11.22)

104. Petkovic D. Microsoft SQL Server 2019: A Beginner's Guide, 7th Ed., Copyright © 2020 by McGraw-Hill Education. https://dl1.newoutlook.it/book/2020/03/Microsoft-SQL-Server-2019-A-Beginners-Guide.pdf

105. Snaidero B. SQL Server Architecture Overview. Intern. resource: https://www.mssqltips.com/sqlservertutorial/9220/sql-server-architecture-overview/ (10.02.22)

106. Kadlec J. SQL Server. CTO @ Edgewood Solutions. 2021. https://www.mssqltips.com/sqlserverauthor/38/jeremy-kadlec/ (1.11.21)

107. Peterson R. SQL Server Architecture (Explained). 2022. Intern. resource:www.guru99.com/sql-server-architecture.html#8 (1.12.21)

108. სურგულაძე გ., ბულია დ., თურქია ე. Web-აპლიკაციების დამუშავება მონაცემთა ბაზების საფუძველზე (ADO.NET, ASP.NET, C#). ISBN 978-9941-14-289-5. სტუ. „ტექნიკური უნივერსიტეტი“, თბ., 2009. -189 გვ.

109. სურგულაძე გ., ბულია ი. კორპორაციულ Web-აპლიკაციათა ინტეგრაცია და დაპროექტება. მონოგრ., ISBN 978-9941-20-165-3. სტუ. „ტექნიკური უნივერ-სიტეტი“, თბ., 2012. – 324 გვ.

110. სურგულაძე გ., კაშიბაძე მ. ორგანიზაციულ სისტემებში ინფორმაციული რესურსების მართვა. ISBN 978-9941-14-447-6. სტუ. „ტექნიკ.უნივერს.“, თბ., 2009 -128 გვ.

111. სურგულაძე გ., ოხანაშვილი მ., სურგულაძე გიორგი. მარკეტინგის ბიზნეს-პროცესების უნიფიცირებული და იმიტაციური მოდელირება. ISBN 978-9941-14-377-9. სტუ. „ტექნიკ.უნივერს.“, თბ., 2009 -170 გვ.

112. Hal Flynn. Designing and Building Enterprise DMZs. cisco dmz network tutorial. 2006. -714 p. https://books.google.ge/books?id=wkgef1FuGzwC&pg=PA714&dq=cisco+dmz+network+tutorial&hl=en&sa=X&ved=2ahUKEwipvIHLjLX8AhWP7KQKHfmRC6EQ6AF6BAgIEAI#v=onepage&q=cisco%20dmz%20network%20tutorial&f=false

113. Bolch G., Greiner S., DeMeer H., Travedi K. Queueing Networks and Markov Chains. John Wiley&Sons.Inc., 2000.

114. სურგულაძე გ., გულუა დ. განაწილებული სისტემების ობიექტ-ორიენტირებული მოდელირება უნიფიცირებული პეტრის ქსელებით. 99940-48-07-4. მონოგრ., სტუ. თბილისი. 2005. -210 გვ.

115. სურგულაძე გ., შონია ო., პეტრიაშვილი ლ. კომპიუტერული ქსელის რესურსების სინქრონიზების პროცესის მოდელირება პეტრის ქსელებით მრავალმომხმარებლურ რეჟიმში. პერიოდ.სამეც. ჟურნ. „ ინტელექტი“, N1, თბ., 1997

116. სურგულაძე გ., ქაშიბაძე მ. ოპერაციულ სისტემებში პროცესების მართვის კვლევა პეტრის ქსელების თეორიის გამოყენებით, სტუ, „ტექნ. უნივ.“, თბ., 1993. 32 გვ.

117. Surguladze G., Petriashvili L., Shonia O., Surguladze Gr. Visual modeling and dynamic analysis of distributed business-processes on the basis of net-technologies and Petri networks. Georg.Scien.Acad.- “Moambe”, No172-2, 2005.

118. Reisig W. Petrinetze - eine Einfuhrung. Springer-Verlag. Heidelberg, 1982

119. Reisig W. Petrinetze: Modellierungstechnik, Analisemethoden, Fallstudien. ISBN 978-3-8348-1290-2. Vieweg+Teubner Verlag. Springer Fachmedien Wiesbaden GmbH. 2010. - 246 S.

120. რეისიგი ვ., სურგულაძე გ., გულუა დ. ვიზუალური ობიექტ-ორიენტირებული დაპროგრამების მეთოდები (BorlandC++ Builder, PetriNet). ISBN 99928-943-9-3. სტუ, თბ., 2002. -200 გვ.

121. რეისიგი ვ., სურგულაძე გ., გულუა დ. პროგრამული სერვერების ორგანიზაცია კორპორაციულ ქსელებში. ISSN 1512-3979. სტუ-ს შრ.კრებ. „მას", N1(2), თბ., 2007, გვ. 100-104

122. რეისიგი ვ., , სურგულაძე გ., გულუა დ. MODELLIERUNG-2008: ახალი ხიდი მეცნიერებასა და წარმოებას შორის. ბერლინის ჰუმბოლდტის უნივერსიტეტი (გერმანია). ISSN 1512-3979. სტუ-ს შრ.კრებ. „მას", N1(4), თბ., 2008, გვ. 9-15

123. GPSS World Tutorial Manual. http://www.minutemansoftware .com/tutorial/tutorial_manual.htm (18.01.22)

124. Creating an Oracle Data Warehouse. Oracle® Warehouse Builder User's Guide 10g Release 2 (10.2.0.2). Part Number B28223-05. https://docs.oracle.com/cd/B31080_01/doc/owb.102/b28223/concept_basics.htm.

125. Rittman M. Understanding Oracle OLAP Dimensions And Cubes. Burleson Consulting. The Oracle Corporation. http://www.dba-oracle.com/t_olap_dimensions_cubes.htm

126. Oracle® OLAP Java API Reference. 12c Release 2 (12.2). E83795-01. https://docs.oracle.com/cd/E82638_01/olapi/index.html

გადაეცა წარმოებას 20.02 2022. ოფსეტური ქაღალდის ზომა 60X84 1/16. პირობითი ნაბეჭდი თაბახი 16,5. ტირაჟი 50 ეგზ.

Verba volant, scripta manent

სტუ-ს „IT კონსალტინგის ცენტრი~,
თბილისი, მ.კოსტავას 77

ISBN 978-9941-8-0623-0