УНИВЕРЗИТЕТ „ГОЦЕ ДЕЛЧЕВ“ – ШТИП

ФАКУЛТЕТ ЗА ИНФОРМАТИКА

ИНФОРМАЦИОНИ СИСТЕМИ И ТЕХНОЛОГИИ

Штип

ГОРАН ВИТАНОВ

ПРОЦЕС НА КРЕИРАЊЕ И УПОТРЕБА НА СИСТЕМ ЗА РУДАРЕЊЕ НА ПОДАТОЦИ ВО ГОЛЕМОПРОДАЖБА

МАГИСТЕРСКИ ТРУД

Штип, Јули 2015

UNIVERSITY "GOCE DELCEV" - STIP

FACULTY OF COMPUTER SCIENCE

INFORMATION SYSTEMS AND TECHNOLOGIES

Stip

GORAN VITANOV

PROCESS OF CREATING AND USING DATA MINNING SYSTEM IN WHOLESALE

MASTER'S THESIS

Stip, July 2015

КОМИСИЈА ЗА ОЦЕНКА И ОДБРАНА

ПРЕТСЕДАТЕЛ: Професор д-р. Александра Милева

Универзитет „Гоце Делчев“ – Штип,

Факултет за информатика

ЧЛЕН: Доцент д-р. Благој Делипетрев

Универзитет „Гоце Делчев“ – Штип,

Факултет за информатика

ЧЛЕН - МЕНТОР: Доцент д-р. Зоран Здравев,

Универзитет „Гоце Делчев“ – Штип,

Факултет за информатика

Дата на одбрана: _____

БЛАГОДАРНОСТ

Голема благодарност до моето семејство за неизмерната љубов, разбирање и поттик во работата на магистерскиот труд.

Чест ми е да искажам благодарност и до мојот ментор, доцент д-р Зоран Здравев, за големата соработка во изработката на магистерскиот труд.

ПРОЦЕС НА КРЕИРАЊЕ И УПОТРЕБА НА СИСТЕМ ЗА РУДАРЕЊЕ НА ПОДАТОЦИ ВО ГОЛЕМОПРОДАЖБА

PROCESS OF CREATING AND USING DATA MINNING SYSTEM IN WHOLESALE

Рецензирани и објавени/прифатени за објавување трудови:

1. Process of Creating and Using Data Warehouse in a Wholesale, Goran Vitanov, Cveta Martinovska, Zoran Zdravev. In: Proceedings of the Ninth International Conference on Informatics and Information Technology CIIT 2012, April 19-22, 2012, Molika, Bitola, Macedonia.

2. Goran Vitanov, Igor Stojanovik, Zoran Zdravev, Improving the Wholesales Trough Using the Data Mining Techniques. In: ICTI 2012: ICT Innovations 2012 - Secure and Intelligent Systems, September 12-15, 2012, Ohrid, Republic of Macedonia.

ПРОЦЕС НА КРЕИРАЊЕ И УПОТРЕБА НА СИСТЕМ ЗА РУДАРЕЊЕ НА ПОДАТОЦИ ВО ГОЛЕМОПРОДАЖБА

Краток извадок

Овој труд го опишува процесот на креирање податоци, информации и знаење преку пример со оперативна база на податоци. Согласно праксата, деловните апликации во денешниот корпоративен свет се креирани да складираат податоци во различни форми. Податоците се тесно поврзани со функционалниот процес на организацијата и може да потекнуваат од повеќе извори, влезови и системи. Купувачи, фактури, порачки, вработени и производствени податоци се само некои примери на собирање, обработка, складирање и акумулирање на голема маса на податоци. Податочниот склад е релациона база на податоци, но е дизајнирана за анализа и пребарување наместо за трансакциско процесирање. Нејзиниот централно интегриран податочен склад е дизајниран за известување и одржување на историјата на податоците. Екстракцијата, трансформацијата и полнењето овозможува податочниот склад да стане оперативен. Ова овозможува анализира на мултидимензинални податоци од повеќе перспективи со користење консолидација, drill-down, slicing и dicing. Со OLAP податочните коцки, податоците се прикажуваат во еден едноставен формат, агрегирани и пресметани во повеќе димензии. За деловните корисници OLAP податочните коцки претставуваат алатка за анализа, но не претставуваат откривање знаење. Математички и статистички методи се користат за идентификување скриени трендови и невообичаени шеми во податоците со цел предвидување на иднината. Податочното рударење се користи за извлекување на претходно непознати шеми како групи на податоци (кластер анализа), невообичаени податоци (откривање на аномалии) и зависности (асоцијативно правило) преку автоматска и полуавтоматска анализа на големи податочни множества. Последен чекор е користење на знаењето со цел креирање на мудрост.

Клучни зборови: Податочен склад, податочно рударење, кластер анализа, OLAP, големопродажба.

PROCESS OF CREATING AND USING DATA MINNING SYSTEM IN WHOLESALE

Abstract

This paper presents the process of creating data, information, and knowledge through a real live database example. According to the practice, business applications in today's corporate world are created to gather data in many forms. Data is closely associated with functional process of the organization and can originate from multiple sources, inputs, and systems. Customers, invoices, orders, employees, and manufacturing data are some examples of collecting, processing, storing, and accumulating an extensive amount of data. Data warehouse is a relational database, but it is designed for analysis and query rather than transactional processing. Its central integrated data storage is designed for reporting and maintaining data history. Extraction, transformation and loading makes operational the data warehouse. This enables to analyze multidimensional data from multiple perspectives using consolidation, drill-down, slicing and dicing. With OLAP cubes data is published in a user friendly form, already aggregated and computed in multiple dimensions. For business users an OLAP cube presents an analytical tool, but not represent knowledge discovery. Mathematical statistics methods are used to identify hidden trends and unusual patterns into data in order to predict the future. Data mining is used to extract previously unknown patterns such as groups of data records (cluster analysis), unusual records (anomaly detection), and dependencies (association rule mining), through automatic or semi-automatic analysis of large datasets. The final step is utilizing the knowledge to create wisdom.

Keywords: Data warehouse, data mining, cluster analysis, OLAP, wholesale.

Содржина

Листа на слики

Листа на табели

ВОВЕД

Континуираниот и брз развој на компјутерската технологија во последните три декади доведе до појава на моќни и достапни компјутери како и опрема за собирање и складирање информации. Технологијата даде огромен поттик за развој на индустријата за бази на податоци и информации, при што се креираа огромен број бази на податоци кои се користат за трансакциски менаџмент, пребарување и пронаоѓање на информации како и анализа на податоци.

Податоците се складираат во различни видови на бази на податоци и складови за информации. Спојувањето на хетерогените извори на податоци наметнува појава на податочни складови, места каде на организиран и унифициран начин се чуваат истите.

Надвор од базите на податоци и податочните складови, исто така, се акумулира огромна маса на податоци. Типичен пример за тоа се World Wide Web, тековите на податоци (Data stream), телекомуникации, мрежи со сензори итн. Ваквата состојба на изобилството на податоци може да се опише како „богати со податоци, но сиромашни со информации“, каде посебен предизвик претставува ефективно и ефикасно анализирање на податоци од така различни форми.

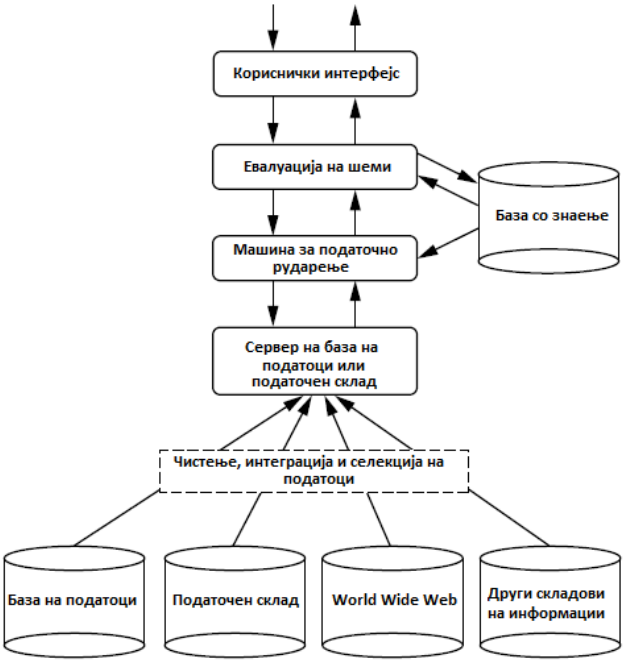
Вака собраните и складираните огромни маси на податоци и информации ја надминаа нашата можност како човечки суштества за следење без користење на моќни алатки. Резултат на сето тоа е создавање на гробници на податоци или дата архиви кои многу ретко се посетуваат, а донесувањето на важни одлуки не се базира на информациите кои се складирани во информационите складови, туку почесто на интуицијата на донесувачите на одлуки. Бидејќи донесувачите на одлуки не располагаат или ретко користат алатки кои ќе овозможат искористување на податоците од податочните складови, друг вообичаен начин да се дојде до податоците е користење на експерти за рачно извлекување на знаење, процес кој генерира многу грешки и одзема многу време.

За разлика од тоа, алатките за рударење на податоци извршуваат анализа и можат да откријат важни податочни шеми при што овозможуваат

донесување на важни деловни одлуки, помагаат во медицински и научни истражувања, откривање на криминал итн.

Па така податочното рударење би можело да се преформулира како рударење на знаење од огромни маси на податоци или споредбено, кога се бара злато, не се вели дека се пребаруваат карпите. Откривањето на знаење е процес кој може да се опише во следните чекори:

- утврдување на изворите на податоци;
- чистење на податоци;
- интегрирање на податоци;
- селектирање на податоци;
- трансформација на податоци;
- рударење на податоци;
- евалуација на шема и
- презентирање на знаењето.



Слика 1. Процес на податочно рударење
Figure 1. Data mining process

Цели на научно-истражувачкиот труд

Основна цел на научно-истражувачкиот труд е создавање методологија за креирање на податочен склад и систем за податочно рударење. Креирањето на методологијата ќе се имплементира преку следниве потцели:

1. Трансформирање на историските податоци во знаење;

   Тековниот систем е структуриран во годишни бази на податоци кои се користат за креирање на извештаи за работењето во тековната година. Користењето и споредувањето со историските податоци е комплексно и бавно. Како прв и неминовен чекор се наметна организирање и чистење на тековните податоци, а исто така и на податоците од претходните години во податочен склад. Преку процесот на екстракција, трансформација и полнење потребно е складот да се синхронизира на дневно ниво. Добиениот податочен склад е основа на креирање на податочни коцки и употреба на техники за податочно рударење преку кои се доаѓа до скриеното знаење.
2. Нумерички и графички приказ на добиеното знаење на менаџерскиот тим;

   Подигнување на свесноста за можностите за искористување на скриените информации од страна на менаџерските тимови е приоритет на ова истражување.

   Средното ниво на менаџмент, преку податочните коцки на еден интегрален начин, би можело да го анализира работењето на компанијата во тековната и изминатите години. Овде можноста за креирање кориснички извештаи би била неограничена преку користење на пивот табели. За врвниот менаџмент, преку техниките за податочно рударење, планирано е да се овозможува автоматско креирање на извештаи, откривање на скриеното знаење со цел донесување на стратешки одлуки.
3. Создавање на инфраструктура за понатамошен развој на системот за податочно рударење.

   Системот е замислен како целосно автоматизиран процес на креирање податочен склад, податочни коцки, употреба на техники за податочно рударење. Структурата на системот неопходно е да овозможува

понатамошен развој без надворешни интервенции. За таа цел во градењето на извештаите, планирано е да се вклучи и лицето одговорно за ИТ поддршката во компанијата, кое ќе биде обучено за поддршка и креирање извештаи.

Структура на магистерскиот труд

Магистерскиот труд е составен од шест глави:

Глава 1. Извор на податоци

Глава 2. Подготовка на податоците

Глава 3. Податочен склад

Глава 4. Податочни коцки

Глава 5. Податочно рударење

Глава 6. Креирање и употреба на систем за рударење на податоци (практична примена)

Заклучоци, препораки и предизвици.

Во првата глава даден е теоретски осврт на видовите извори на податоци кои би можеле да се користат при полнење еден податочен склад. Посебно се разгледани релационите бази на податоци, податочните складови, трансакционите бази на податоци и напредните податочни складови.

Во втората глава е прикажан процесот и техниките за подготовка на податоците, односно ETL (Extraction, Transformation and Loading).

Во третата глава содржи теоретски приказ за креирање на еден податочен склад. Освртот овде е на архитектурата како и чекорите за дизајнирање и архитектура на податочни складови.

Во четвртата глава се дава теоретски приказ за креирање на податочна коцка. Опишани се шеми за мултидимензионални бази, категоризација и

генерализација на мерки, хиерархиски концепти и OLAP операции во мултидимензионален модел.

Во петтата глава даден е осврт на целите и техниките на податочно рударење. Прикажани се асоцијативното правило и кластер анализата. Кластер анализата е анализирана подетално преку хиерархиски кластери, центроидно базирани кластери и кластерите базирани на дистрибуција.

Во шестата глава практично е имплементирана целокупната теорија која што е прикажана погоре. Утврдувањето на изворите на податоци, креирањето на податочен склад, процесот на ETL, податочните коцки, техниките на податочно рударење и сето тоа во форма која е употреблива и лесно разбирлива за корисниците е прикажано чекор по чекор.

1. ИЗВОРИ НА ПОДАТОЦИ

Вообичаено е да имаме различни видови на информации и податоци врз основа на кои ќе се изврши рударењето. Така типовите на извори на податоци вклучуваат релациони бази на податоци, трансакциони бази на податоци, напредни податочни системи, податочни складови, текстуални фајлови, табели за пресметување, аудио-фајлови, World wide web, видео фајлови итн. Техниките и предизвиците за рударење се разликуваат за секој складишен систем поодделно.

1.1. РЕЛАЦИОНИ БАЗИ НА ПОДАТОЦИ

Системот на бази на податоци, исто така познат како database management system (DBMS), се состои од колекција на меѓусебно поврзани податоци познати како база на податоци и софтвер за менаџирање и пристап до податоците. Софтверот овозможува дефинирање на структурата на базата за складирање, правата за пристап и осигурување за конзистентност на податоците. Релационата база е колекција на табели, каде секоја табела има:

- единствено име;
- свои атрибути, колони или полиња во кои е дефинирано каков тип на податоци ќе се складираат ;
- податоци складирани во многу редови, при што секој ред се идентификува преку единствен клуч. (1)

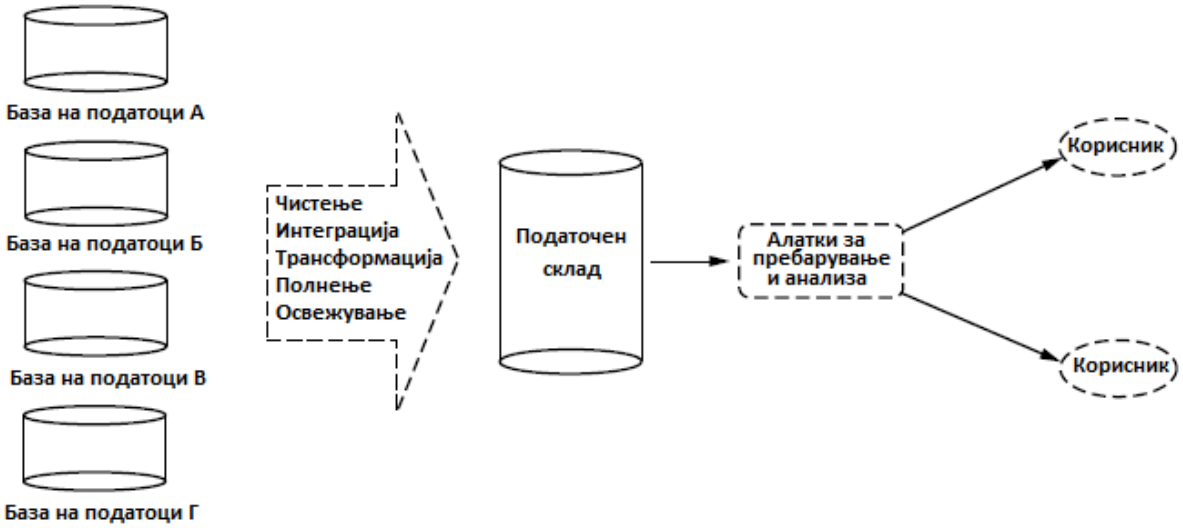
Пристапот до податоците е преку прашалници пишувани во релационен јазик за прашалници, како што е SQL (Structured Query Language). На пример, преку прашалниците може да пребаруваме: збирот на продажбите по дати, недели, број на вработени итн. Значи, овде може да се изврши обично селектирање или може да се вршат посложени операции како групирање, просек, број, максимум, минимум итн. Исто така, овде може да се задаваат и

филтри при прикажување на податоците. Пример за тоа може да биде продажба за месец јануари, продажба за одреден комерцијалист итн.

Кога се применува техника на рударење на податоци на релациона база на податоци, тогаш се бара тренд или податочна шема, што е сосема различно од претходно наведеното. Така, на пример, може да го анализираме ризикот за продолжување на соработката со одложено плаќање за одредена категорија на купувачи. Релационите бази на податоци се едни од најчесто достапните и користени во светот на податочното рударење.

1.2. ПОДАТОЧЕН СКЛАД

Податочниот склад како извор на податоци за рударење претставува збир на информации собрани од повеќе извори и складирани според една унифицирана шема кои вообичаено се наоѓаат на едно место. Полнењето на податочниот склад со податоци е процес кој се состои од чистење, интегрирање, трансформирање, полнење и планирано периодично освежување на податоците. Истиот е познат како ETL (Extract, Transform, Load).



Слика 2. Процес на креирање податочен склад
Figure 2. Data warehouse creating process *(2)*

Со цел да се овозможи подобро донесување на одлуки податоците во податочниот склад се организирани по носители како, на пример: купувачи, производи, добавувачи итн. Целта на складирањето е да овозможи историска перспектива на групирани и сумирани податоци за период, најчесто, од 5 до 10 години. Планирањето на податочните складови е да овозможи мултидимензионални податочни коцки, каде податочните коцки овозможуваат мултидимензионален преглед на податоци кои овозможуваат брз пристап до претходно сумирани и обработени податоци.

1.3. ТРАНСАКЦИОНИ БАЗИ НА ПОДАТОЦИ

Трансакционите бази на податоци се датотеки каде секој податок претставува трансакција, а трансакцијата вклучува единствен трансакционен идентификационен број и листа на полиња што ја сочинуваат трансакцијата. Трансакционата база на податоци може да вклучува и дополнителни табели кои содржат и други податоци поврзани со трансакцијата. За разлика од релационите бази на податоци, податоците во трансакционите бази на податоци се запишуваат во обична датотека. (3)

1.4. НАПРЕДНИ ПОДАТОЧНИ ИФОРМАЦИОНИ СИСТЕМИ И НАПРЕДНИ АПЛИКАЦИИ

Во светот на бизнисот најмногу се користат релационите бази на податоци. Прогресот на технологијата на база на податоци, појавата на нови видови на напредни податоци и информациони системи наметнува развој на нови апликации.

Новите апликации со бази на податоци вклучуваат ракување со просторни податоци како мапи, податоци за инженерски дизајн, системски компоненти, интегрални кола, хипертекст и мултимедијални податоци кои вклучуваат текст, видео, слика, аудио, сензорски податоци, world wide web итн. Овие апликации

бараат ефикасни податочни структури и методи за ракување со сите горенаведени податоци.

За да се одговори на тие потреби, креирани се напредни податочни системи и апликации кои вклучуваат:

- објектно-релациони бази на податоци;
- просторни бази на податоци;
- текст-бази на податоци и мултимедијални бази на податоци;
- хетерогени бази на податоци;
- податочни текови и
- World Wide Web. (4)

2. ПОДГОТОВКА НА ПОДАТОЦИТЕ

Се поставува прашањето, како да го подобриме квалитетот на податоците со цел резултатите од рударењето да бидат релевантни. Прашањето се поставува бидејќи базите на податоци од реалниот свет изобилуваат со недостатоци, а со оглед на тоа што техниките на рударење на податоците се применуваат на бази на податоци со огромни големини понекогаш и по неколку терабајти, лошиот квалитет може значително да влијае на крајниот резултат.

Во пракса се среќаваат повеќе техники за подготовка на податоците и тоа:

- чистење на податоците - отстранување на неконзистентни и непотполни податоци;
- интегрирање на податоците - спојување на податоци од хетерогени извори и формирање на еден заеднички податочен склад;
- трансформација на податоците - нормализација за да се зголеми ефикасноста и точноста на алгоритамите за рударење на податоци;
- намалување на податоците - групирање и сумирање, отстранување на непотребни карактеристики. (5)

Овие техники се користат поединечно или заедно во зависност од ситуацијата и значително го подобруваат времето и крајниот резултат од рударењето на податоци.

2.1. ИНТЕГРАЦИЈА И ТРАНСФОРМАЦИЈА НА ПОДАТОЦИ

2.1.1. ИНТЕГРАЦИЈА НА ПОДАТОЦИ

Комбинирањето на податоци од различни извори на едно место во податочен склад наметнува потреба од процес на интегрирање на податоци. Изворите на податоци можат да бидат од бази на податоци, текст датотеки, обични датотеки итн. Со тоа се наметнуваат многу прашања кои се поврзани со процесот на интеграцијата.

Интеграционата шема на објекти може да биде многу комплицирана. При поврзувањето на податоци од различни извори во секојдневието се судруваме со проблемот на идентификација на влезовите. Пример за идентификациски проблем е, како да знаеме дека АртиклИд од еден извор е исто како ШифраАртикл од друг извор, односно ИмеАртикл дали е исто со Артикл итн.

Исто така важно прашање претставува вишокот на податоци. Сума на продажба може да биде вишок атрибут под услов ако може да се изведе од еден или од други атрибути. Некои вишок атрибути можат да бидат откриени со корелациона анализа со што го мериме влијанието на еден врз друг атрибут. За два нумерички атрибути A, B можеме да ја мериме јачината на корелацијата врз основа на пресметка на коефициентот на корелација.

$$r_{A,B} = \frac{\sum_{i=1}^{N}(a_i - \overline{A})(b_i - \overline{B}\,\frac{\sum_{i=1}^{N}(a_i b_i - N\overline{AB})}{N\sigma_A\sigma_B})}{N\sigma_A\sigma_B}$$

Така N претставува бројот на редови, a_i, b_i се вредностите на A и B во редот i, $\overline{A}$, $\overline{B}$ се просечните вредности на A и B, σ_A и σ_B се стандардните девијации на A и B и $\sum(a_i b_i)$ претставува сума на производот на вредностите на A и B за секој ред. Треба да се забележи дека, ако $-1 \leq r_{A,B} \leq 1$. Ако $r_{A,B}$>0 тогаш помеѓу A и B постои позитивна корелација, што значи дека со зголемување на вредноста на A се зголемува и вредноста на B. Поголема вредност значи појака корелација или поголемо влијание на A со B и може да индицира дека A или B може да се отстрани како вишок атрибут. Ако резултатот е еднаков на 0, тогаш A и B се независни, па не постои корелација помеѓу нив. Обратниот случај негативна вредност значи дека со зголемување на едното се намалува другото и обратно.

Овде треба да напоменеме дека ако има корелација меѓу A и B тоа автоматски не значи дека A го предизвикува B или обратно. На пример, ако се прави демографска анализа, поголем број на болници и поголем број на крадци на коли не значи дека едното го предизвикува другото и обратно. Двата фактора не се поврзани и зависат од трет фактор популација.

Трето важно прашање при интеграција на податоците е откривање и решавање на конфликтите со вредностите на податоците. Во реалниот свет вредностите на атрибутите од различни извори можат да се разликуваат како резултат на различно претставување, мерење и кодирање. На пример, во еден ланец хотели цената на собата во различни градови и држави може да се разликува не само според валутата, туку и според видот на услугите како бесплатен појадок, даноци итн. Водејќи сметка за ваквите можни разлики за исти податоци кои потекнуваат од различни извори, се оневозможува крајниот резултат од рударењето на податоци да биде нерелевантен.

2.1.2. ТРАНСФОРМАЦИЈА НА ПОДАТОЦИ

Кај трансформацијата на податоците, податоците се трансформираат или консолидираат во форма која што е адекватна за рударење. Трансформацијата на податоци може да вклучува:

- Израмнување (smoothing), подразбира отстранување на шумот од податоците. Ваквата техника вклучува регресија, кластерирање, групирање.
- Агрегација е техника каде се применуваат операции за сумирање, групирање итн. Пример е агрегирање на дневните продажби за пресмета на месечна или годишна сума. Овој чекор обично се користи при конструкција на податочни коцки.
- Генерализација на податоци, каде податоци од ниско ниво или „примитивни“ се заменуваат со концепти од високо ниво преку користење на концептуална хиерархија. На пример, категорискиот атрибут улица може да биде генерализиран со концепт од повисоко ниво како град или држава. Слично, нумеричките вредности како старост може да бидат мапирани со концепт од повисоко ниво како млади, средновечни и стари.
- Нормализација, е намалување на податочните атрибути во многу мал ранг како од -1.0 до 1.0 или од 0.0 до 1.0. Нормализацијата е посебно корисна за алгоритамите за класификација вклучувајќи неврални мрежи или мерење на растојание како најблизок сосед (nearest neighbor), класификација и

кластерирање. Постојат min-max нормализација, z-score нормализација, нормализација со децимално намалување итн.

- Конструирање на атрибути или конструирање на карактеристики. За да го помогнат процесот на рударење новите атрибути се конструираат или додаваат од дадено множество на атрибути. Дадените атрибути се додаваат со цел да ја зголемат точноста и разбирањето на структурата кај повеќедимензионалните податоци. Пример за тоа е кога сакаме да го додадеме атрибутот површина кој се базира на атрибутите ширина и должина. Со комбинирање на ваквите атрибути добиените конструкции можат да овозможат пронаоѓање на скриени информации за врските помеѓу податочните атрибути кои можат да бидат корисни при откривање на знаењето.

2.2. РЕДУКЦИЈА НА ПОДАТОЦИТЕ

Со оглед на тоа што при креирањето на податочните складови се користат огромна маса на податоци, комплексните анализи и рударење на податоци одземаат многу време и ресурси. Со тоа се наметнува потребата за користење техники за намалување на бројот на податоци од една страна, а сепак да се задржи интегритетот на оригиналните податоци. Така, рударењето на редуцираните податоци треба да биде ефикасно, а во исто време треба да даде речиси исти аналитички резултати. Затоа стратегиите за редуцирање треба да вклучуваат:

- агрегација за податочни коцки, каде операциите за агрегации се прават со цел конструкција на податочна коцка;
- селекција на атрибути од подмножество, каде ирелевантните, малку релевантните или излишните атрибути од димензиите се откриваат и отстрануваат;
- димензионална редукција користи механизми за кодирање за да се намали големината на множеството податоци;

- бројна редукција, каде податоците се заменуваат или се врши проценка. Се креира помала податочна застапеност со користење на модели со параметри или методи без параметри како кластерирање, хистограми и земање мостри;
- дискретизација и концепт на хиерархиско генерирање, каде податоците се заменуваат со рангови или повисоки концептуални нивоа. Дискретизацијата на податоци е форма на бројно намалување која е многу корисна за автоматско генерирање на концепт хиерархија што овозможува повеќе нивоа на апстракција при рударењето на податоци.

2.3. ЧИСТЕЊЕ НА ПОДАТОЦИТЕ

Многу често во секојдневните бази и податочни складови се среќаваат некомплетни, неконзистентни и ирелевантни податоци. Тие настануваат како резултат на кориснички апликации со слабо дизајнирани форми за внесување на податоци (најчесто со многу опционални полиња), човечки грешки при внесување на податок и намерни грешки како нецелосно, делумно или неточно внесување. Исто така, може да се резултат на грешки во опремата која снима, системски грешки итн.

Квалитетните податоци потребно е да задоволуваат одредени критериуми и тоа:

- Униформност, сите податоци кои означуваат одредени мерки треба да бидат од исти мерен систем, како на пример: метри или инчи, килограми или фунти и мора да бидат конвертирани во иста мерка користејќи аритметичка трансформација.

- Валидност е степенот на усогласеност со одредени деловни правила или ограничувања. Кога се користат модерните технологии за бази на податоци многу лесно се остварува валидноста на податоците преку правилен дизајн. Неправилен дизајн на кориснички апликации, бази на податоци, неадекватна

технологија за собирање на податоци генерираат невалидност. Најчести пропусти за податочна валидност и ограничувања се:

- ограничување на податочен тип. Секое поле во таблите треба да складира одреден тип на податоци (текст, дата, цел број итн.);
- ограничување на опсег, обично минимум или максимум на одредени броеви или дати;
- задолжителни ограничувања, што значи дека одредени колони не смеат да бидат празни;
- единствени ограничувања, што означува дека одредени полиња или комбинации на полиња мора да бидат единствени во табелите;
- ограничување со множество од вредности. Пример за тоа се повикувачките броеви на градовите во Македонија;
- ограничување со надворешен клуч, се мисли дека вредностите во колоната мора да имаат врска со единствените вредности од друга табела;
- вкрстена валидација. Ваков случај имаме кога збирот на учеството на сите региони треба да биде 100%.

- Точност и прецизност е степенот на точност во однос на одредени стандардни мерки. Случај е кога ќе ги споредиме поштенските броеви на купувачите од нашата база со поштенските броеви од надворешен извор.
- Конзистентност е степенот во кој одредени податоци се исти и се споредуваат внатре во системот. Постои неконзистентност кога еден исти купувач е внесен двапати со мала грешка во името или со две различни адреси. Овде треба да се утврди кој податок е точен и да се изврши промена.

Се врши ревизија за да се утврдат аномалиите, при што се користат методи на бази на податоци и статистички методи. Низата на операции што се употребуваат се утврдува после процесот на ревизија и е клучна за остварување на висок квалитет на податоците. Процесот на чистење, односно извршувањето на низата операции, треба да биде ефикасен, брз и евтин. По извршувањето на операциите на чистење, се врши проверка на резултатите и ако има потреба се интервенира рачно.

Бидејќи овој процес е доста комплициран, одзема многу време и можностите за грешки се огромни, па доброто планирање и автоматизација се неопходен чекор за успешно завршување на процесот. Во овој процес се користат сопствени или готови софтверски решенија.

3. ПОДАТОЧЕН СКЛАД

Податочен склад претставува постојано складиште на податоци кое служи како физичка имплементација на податочен модел, место каде се чуваат информации кои се потребни за донесување на стратешки одлуки. Податочниот склад често се гледа како архитектура конструирана преку интегрирање на податоци од многу хетерогени извори за да поддржи извршување на прашалници, аналитички извештаи и донесување одлуки. Врз основа на претходното, може да се каже дека податочното складирање претставува процес на конструирање и користење на податочни складови.

За полесно да се разбере што е тоа податочен склад најдобро е да се споредат концептот на OLTP и OLAP.

Табела 1. Разлика помеѓу OLTP и OLAP концепт
Table 1. OLTP and OLAP concept differences (6)

Особина	OLTP	OLAP
Карактеристика	оперативно процесирање	процесирање на информации
Ориентација	трансакции	анализа
Корисник	оператор, администратор на база	менаџер, аналитичар...
Функција	дневни операции	поддршка на одлуки, долгорочни информациски барања
Дизајн на базата	етнитетен, апликационо ориентиран	ѕвезда, снегулка, предметна ориентација
Податоци	ажурирани до момент	историска, точност се одржува со тек на време
Сумирање	примитивни, виско детални	сумирани, консолидирани

Поглед	детални, реалциони	сумирани, мултидимензионирани
Единица на работа	кратка, обична трансакција	комплексни прашалници
Пристап	читање / пишување	претежно читање
Фокус	податоците	излез на информации
Операции	индекс / дисперзија на примарен клуч	многу скенирања
Пристапени податоци	десетици	милиони
Број на корисници	илјадници	стотици
Големина на база	100 MB до GB	100 GB до TB
Приоритет	високи перформанси и достапност	голема флексибилност, корисничка автономија
Метрика	трансакциска продуктивност	време на извршување, продуктивност на прашалници

Главна цел на on-line операционите системи на бази на податоци е извршување on-line трансакции и прашалници. Овие системи се викаат on-line transaction processing (OLTP системи). Тие ги покриваат дневните операции на организациите како: порачки, залиха, продажби, плаќања, сметководство итн. Од друга страна, системите за податочни складови им служат на корисниците или менаџерите со цел анализирање на податоци и донесување одлуки. Ваквите системи можат да организираат и презентираат податоци во различна форма во зависност од потребите на различни типови корисници. Овие системи се познати како on-line analytical processing (OLAP системи). Главните разлики меѓу OLTP и OLAP прикажани во Табела 1:

- корисничка и систем ориентација: OLTP системите се корисничко ориентирани и се користат за трансакции и процесирање на прашалници од оператори, клиенти и професионалци за информациони технологии. OLAP системите се пазарно ориентирани и се користат за анализирање на податоци од менаџери, раководители и аналитичари.

- содржината на податоци: OLTP системите работат воглавно со тековни податоци кои се премногу детални и се користат за донесување тековни одлуки. OLAP системите управуваат со голема маса на историски податоци кои се сумирани, агрегирани на различни нивоа што ги прави многу лесни за донесување одлуки.
- дизајн на базата на податоци: OLTP системите се главно апликациски ориентирани, додека кај OLAP системите дизајнот е субјектно ориентиран.
- пребарување, OLTP системите се фокусирани на тековните податоци во организацијата или одделот без повикување на историски податоци, односно податоци од други оддели. За разлика од тоа, OLAP системите спојуваат различни верзии на бази на податоци, односно го следат еволуциониот процес на организацијата. Тие, исто така, вклучуваат и податоци од различни организации или оддели.
- шемите за пристап во OLTP системите се состојат од кратки и единечни трансакции. Ваквиот систем бара симултана контрола на конекции и механизми за обновување. Пристапот до OLAP системите е главно со операции за читање на многу комплексни прашалници.

3.1. ЧЕКОРИ ВО ДИЗАЈНИРАЊЕ И КОНСТРУКЦИЈА НА ПОДАТОЧЕН СКЛАД

За да се дизајнира ефективен податочен склад потребно е да се разберат и анализираат потребите на бизнисот, и врз основа на тоа се конструира рамка за бизнис анализа. Конструкцијата на големи и комплексни информациони системи може да се разгледува како конструкција на голема и комплексна зграда за која сопственикот, архитектот и градежниците имаат различни потреби. Сите потреби се комбинираат и се формира комплексна рамка, при што четири различни прашања во однос на дизајнот на податочниот склад мора да бидат земени предвид:

- од врвот надолу, овозможува селекција на релевантните информации потребни за изградба на податочниот склад. Овие информации се совпаѓаат со тековните и идните бизнис потреби.
- извор на информациите, ги изложува информациите кои се снимени, складирани и управувани од оперативниот систем. Овие информации можат да бидат документирани со различни нивоа на детали и прецизност, од индивидуални табели до интегрирани табели како извори на податоци. Изворите на податоци најчесто се моделираат со традиционални техники за моделирање.
- податочниот склад вклучува табели со факти и табели со димензии. Ги претставува информациите кои се складирани во податочниот склад вклучувајќи пред калкулирани суми, бројачи, а исто така и информации во однос на извор, дата, време, потекло кои се дадени да обезбедат историски контекст.
- бизнис прашалници е перспектива на податоците во податочниот склад од гледна точка на крајниот корисник.

Градењето и користењето на податочен склад е комплексна задача бидејќи бара деловни вештини, технолошки вештини и вештини за управување со софтвер.

Деловните вештини е аспект кој вклучува разбирање како таквиот систем ќе ги складира и управува податоците, како да се изградат екстрактори кои ќе ги трансферираат податоците во податочниот склад и како да се направи софтвер за освежување на податочниот склад со што ќе се овозможи едно разумно ниво на ажурирање со тековните податоци. Користењето на податочниот склад исто така подразбира разбирање и превод на деловните барања во прашалници кои ќе бидат задоволени од податочниот склад.

Во однос на технолошките вештини, од аналитичарот на податоци се бара вештина за оценка или мислење врз основа на квантитативните информации, извлекување факти кои се базираат на историските податоци кои се складирани во податочниот склад. Овие вештини вклучуваат можност да се откријат шеми и

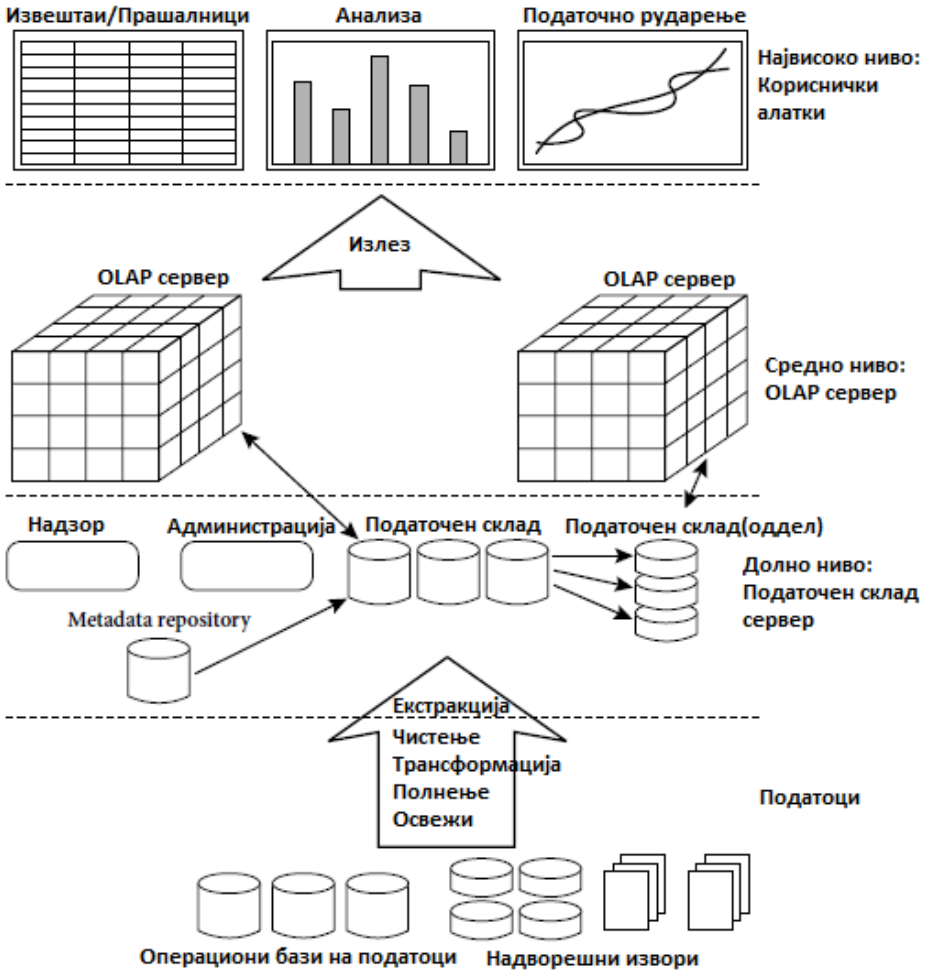
трендови базирани на историјата, аномалии, промени и врз основа на таа анализа да се дадат препораки до менаџерите.

Вештините за управување со програма вклучуваат потреба од разбирање на многу технологии, продавачи на опрема и софтвер како и крајните корисници со цел да се достават резултатите навреме и со ниски трошоци.

Пристапот за градење на податочен склад може да биде од врвот надолу, од дното нагоре или комбиниран метод.

- Од врвот надолу пристапот започнува со целокупно дизајнирање и планирање. Се користи во случаи каде технологијата е добро позната како и деловните проблеми кои треба да се решат се чисти и лесно разбирливи.
- Од дното нагоре пристапот започнува со експерименти и прототип. Се користи во рани фази на бизнис моделирање и развој на технологијата со што овозможува организацијата да се движи напред разумно, со помалку трошоци и користење на придобивките од новата технологија.
- Кај комбинираниот пристап, за проблеми од стратешка природа се користи пристапот од врвот надолу, додека за истражување на нови можности се применува од дното нагоре.

3.2. АРХИТЕКТУРА НА ПОДАТОЧНИ СКЛАДОВИ



Слика 3. Архитектура на податочен склад
Figure 3. Data warehouse architecture

Архитектурата на податочните складови обично се состои од три нивоа:

- прво ниво е серверот на базата на податоци на податочниот склад и секогаш претставува релационен систем на база на податоци. Во позадина се користат алатки за полнење на складот од операциони бази на податоци или други надворешни извори. Овие алатки и услуги извршуваат екстракција на податоци, чистење и

трансформација, а исто така го полнат, освежуваат и обновуваат податочниот склад. За таа цел се користат разни апликациско програмски интерфејси (API) познати како портали. Ваквите портали овозможат клиентските програми да генерираат SQL (Structured Query Language) код кој се извршува на серверот.

- средно ниво е OLAP сервер и се имплементира како релационен OLAP (ROLAP) модел што претставува проширена релациона база на податоци која мапира операции на мултидимензионални податоци на стандардни релациони операции или мултидимензионален OLAP (MOLAP) модел кој претставува сервер со посебна намена каде директно се имплементираат мултидимензионални податоци и операции.
- највисоко ниво е клиентско ниво кое е составено од алатки за пребарување и известување, алатки за анализа и алатки за рударење на податоците.

Од архитектонска гледна точка постојат три модели на податочни складови: податочен склад на претпријатија, податочни складови на оддели и виртуелен податочен склад.

- Податочен склад на претпријатие ги собира сите информации кои се разделени во целата организација. Овозможува широка интеграција на податоци од различни операциони системи или провајдери, обично содржи детални податоци, а исто така и сумирани податоци при што големината може да биде од неколку гигабајти, терабајти, па и повеќе. Може да биде имплементиран на традиционални сервери, супер сервери или платформи со паралелна архитектура. Бара екстензивно деловно моделирање и дизајнирањето и изградбата може да трае со години.
- Податочни складови на оддели имаат вредност за одредена група на корисници и содржат подмножество од вкупните податоци на едно претпријатие. Примери за такви податочни складови се податочен склад за маркетинг, продажба, производство. Овде

податоците воглавно се сумирани. Најчесто е имплементација на евтини сервери кои се дел од одделот.

- Виртуелните податочни складови се множество на прашалници над операционите бази на податоци. За ефикасно процесирање на прашалниците само еден дел од нив се материјализираат. Виртуелните податочни складови лесно се градат, но бараат дополнителен капацитет на операционите сервери на бази на податоци.

4. ПОДАТОЧНИ КОЦКИ

Коцките се главни објекти во on-line аналитичкото процесирање (OLAP), технологија која овозможува брз пристап до податоците на еден податочен склад. Коцка е множество од податоци која обично е конструирана од подмножество од податочен склад и е организирана и сумирана во мултидимензионална структура дефинирана од множества од димензии и мерки.

Земено генерално димензиите се перспективи или ентитети креирани врз основа на плановите на организацијата за чување на податоците. На пример, може да се формира податочен склад за продажба со цел да се чуваат податоците за продажба во однос на димензиите: време, артикли, продажни единици, региони, групи на производи итн. Овие димензии овозможуваат да се следат продажни податоци како месечна продажба за категорија на производи и артикли и региони во кој артиклите се продадени. Секоја димензија може да има табела која е поврзана со неа, позната како димензионална табела. Димензионалната табела време може да има атрибути: ден, недела, месец, квартал и година. Димензионалните табели може да бидат специфицирани од корисници, експерти или да бидат автоматски генерирани и приспособени на дистрибуцијата на податоци.

Мултидимензионалните податочни модели главно се организирани околу некоја централна тема како: продажба, маркетинг, финансии. Оваа тема е претставена од табела со факти. Фактите се нумерички мерки кои може да се гледаат како количини преку кои сакаме да ги анализираме врските помеѓу димензиите. Пример за димензии во еден податочен склад за продажба се промет во денари, продадени количини во бројки, продадени количини во килограми, одобрен рабат итн. Табелата со факти ги содржи имињата на фактите или мерките, како и клучевите со кој се прави врска со табелите со димензии.

Кога се размислува за податочни коцки обично се мисли на 3-D геометриски структури, но во податочните складови коцките се n-

димензионални. За да се добие почиста слика и полесно да се разберат податочните коцки и мултидимензионалните податочни модели, најдобро е да се објасни 2-D податочна коцка.

Табела 2. 2-D податочна табела
Table 2. 2-D data table

Локација	Скопје				
Количина.					
	Безбедност	**Домашни уреди**	**Компјутери**	**Телефони**	**Вкупно**
Q1	400	605	825	14	1844
Q2	512	680	952	31	2175
Q3	501	812	1023	30	2366
Q4	580	927	1038	38	2583
Grand Total	**1993**	**3024**	**3838**	**113**	**8968**

Како што се гледа од сликата, тоа би било продажни податоци за продажба по квартал за одреден тип на производи во даден регион. Овде во 2-D презентацијата податоците се претставени од аспект на временската димензија прикажана како квартал и димензијата вид на производи. Фактите или мерките се претставени како продажба во денари.

Сега да претпоставиме дека сакаме да ги видиме продажните податоци тродимензионално.

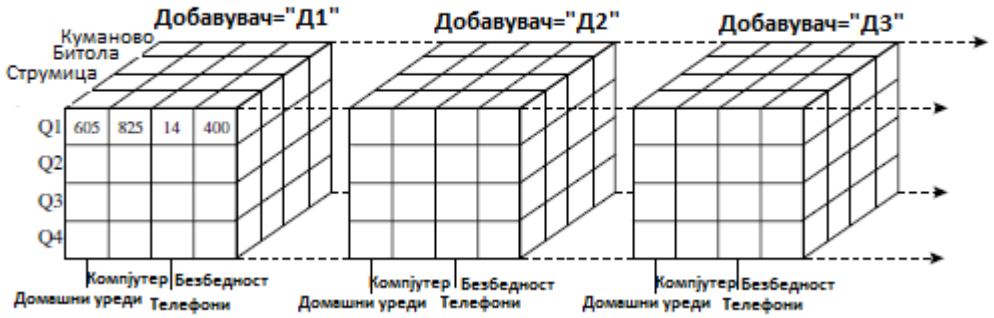
Табела 3. 3-D податочна табела
Table 3. 3-D data table

Количина. Column Labels	Битола				Битола Total	Куманово				Куманово Total	Скопје				Скопје Total	Струмица				Струмица Total
w Labels	Безбедност	Домашни уреди	Компјутери	Телефони		Безбедност	Домашни уреди	Компјутери	Телефони		Безбедност	Домашни уреди	Компјутери	Телефони		Безбедност	Домашни уреди	Компјутери	Телефони	
Q1	591	818	746	43	2198	872	1087	968	38	2965	400	605	825	14	1844	623	854	882	89	2448
Q2	682	894	769	52	2397	925	1130	1024	41	3120	512	680	952	31	2175	698	943	890	64	2595
Q3	728	940	795	58	2521	1002	1034	1048	45	3129	501	812	1023	30	2366	789	1032	924	59	2804
Q4	784	978	864	59	2685	984	1142	1091	54	3271	580	927	1038	38	2583	870	1129	992	63	3054
Grand Total	**2785**	**3630**	**3174**	**212**	**9801**	**3783**	**4393**	**4131**	**178**	**12485**	**1993**	**3024**	**3838**	**113**	**8968**	**2980**	**3958**	**3688**	**275**	**10901**

Како што се гледа од табелата, освен претходните две димензии време и вид на производи, како трета димензија овде ќе го додадеме и регионот.

Слика 4. 3-D податочна коцка
Figure 4. 3-D data cube (6)

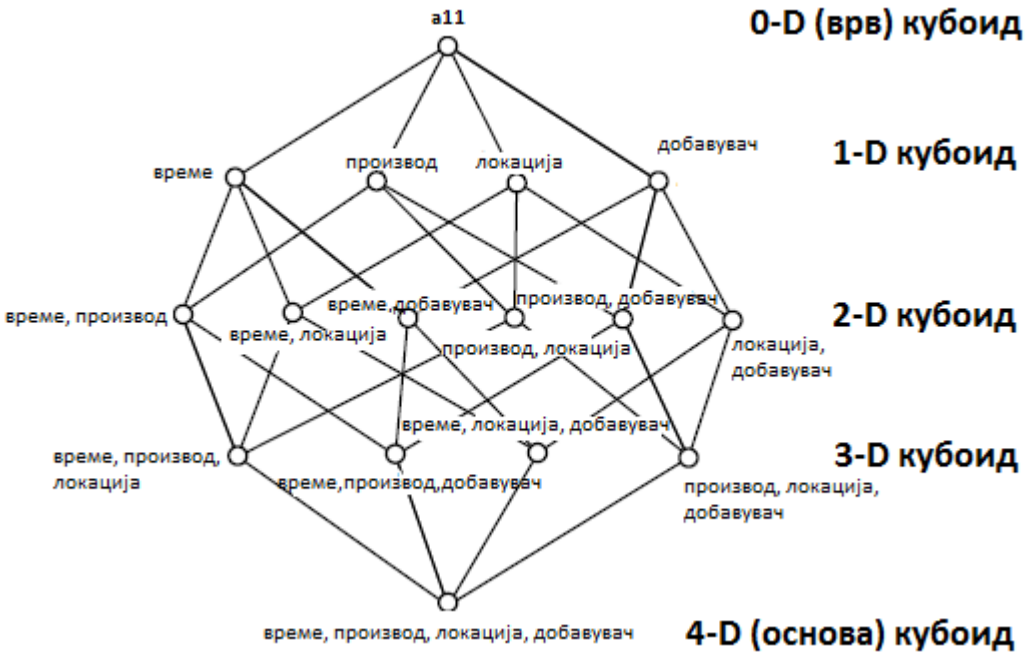
Може да претпоставиме дека сакаме да ги видиме продажните податоци со уште една дополнителна димензија, како на пример добавувач. Гледањето на податоците во 4-D е комплицирано, но како и да е може да ја разгледаме 4-D податочната коцка како серии од 3-D коцки.



Слика 5. 4-D податочна коцка
Figure 5. 4-D data cube

Ако продолжиме на овој начин, можеме да прикажеме n-D податоци како серии од (n-1)-D коцки. Овде битно е да се напомене дека коцките се метафора за мултидимензионално податочно складирање.

Горенаведените табели прикажуваат податоци со различен степен на сумирање. Во литературата за податочното складирање податочните коцки, како што се прикажани погоре, се нарекуваат и кубоиди. Од дадено множество на димензии можеме да генерираме кубоид за секое можно подмножество од дадените димензии. Резултатот треба да формира латица од кубоидите, секоја прикажувајќи податоци со различно ниво на сумирање или групирање. Латиците од кубоидот се претставени како податочни коцки.



Слика 6. Латица од кубоиди
Figure 6. Lattice of cuboids

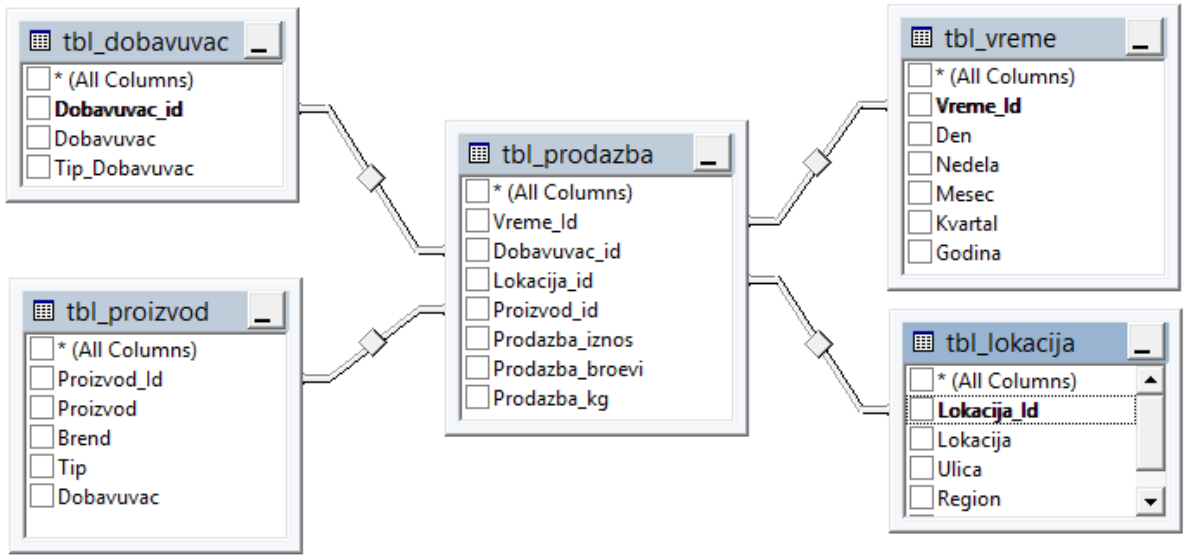
Кубоидот кој има најниско ниво на сумирање се вика базичен кубоид, односно 4-D кубоидот е базичен кубоид за дадените димензии време, група на производ, регион и добавувач. 3-D кубоидот прикажан на сликата за димензиите време, вид на производ и регион е небазичен кубоид. 0-D кубоидот кој има највисок степен на сумирање се вика врв кубоид. Во нашиот случај врв кубоид е продажба во денари за сите димензии.

4.1. ШЕМИ ЗА МУЛТИДИМЕНЗИОНАЛНИ БАЗИ

При дизајнот на релационите бази на податоци најчесто користен податочен модел е субјектно-релационен, каде шемата на базата на податоци се состои од множество на субјекти и врски меѓу нив. Ваквиот податочен модел одговара за on-line трансакциско процесирање. Од друга страна податочниот склад бара прецизна субјектно ориентирана шема која одговара на on-line податочна анализа.

Најпопуларниот модел за податочни складови е мултидимензионалниот модел. Ваков модел може да постои во форма на шема во вид на ѕвезда, снегулка или факт констелациона шема.

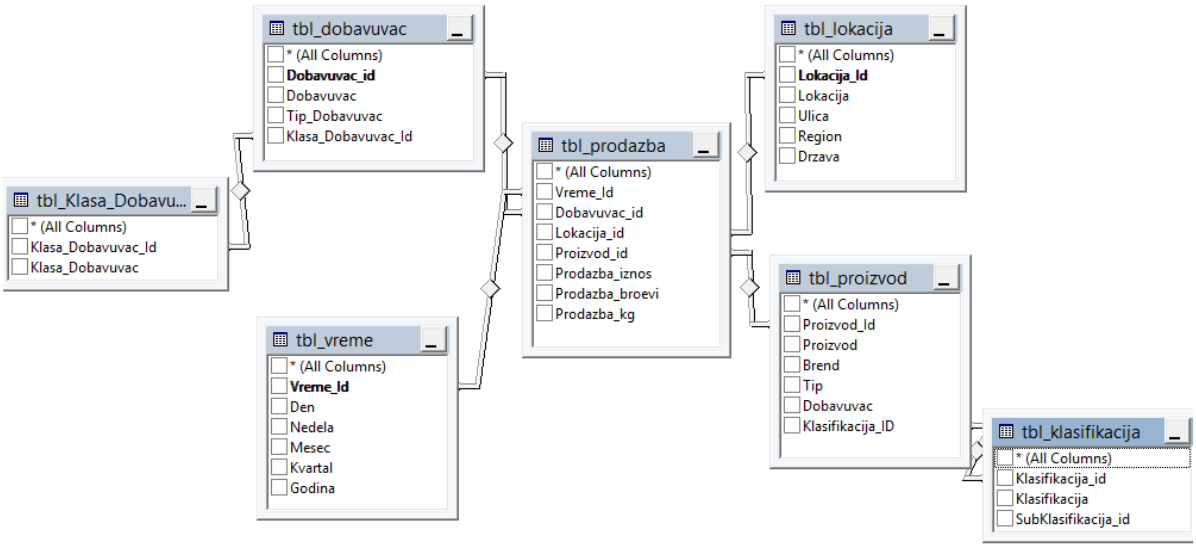
Шемата ѕвезда е најчесто користен модел во податочните складови. Содржи една голема централна табела со факти или мерки и множество на помали табели при што секоја од нив претставува посебна димензија. Графикот на оваа податочна шема има форма на ѕвезда со една централна и повеќе мали табели наоколу.



Слика 7. Шема ѕвезда
Figure 7. Star schema

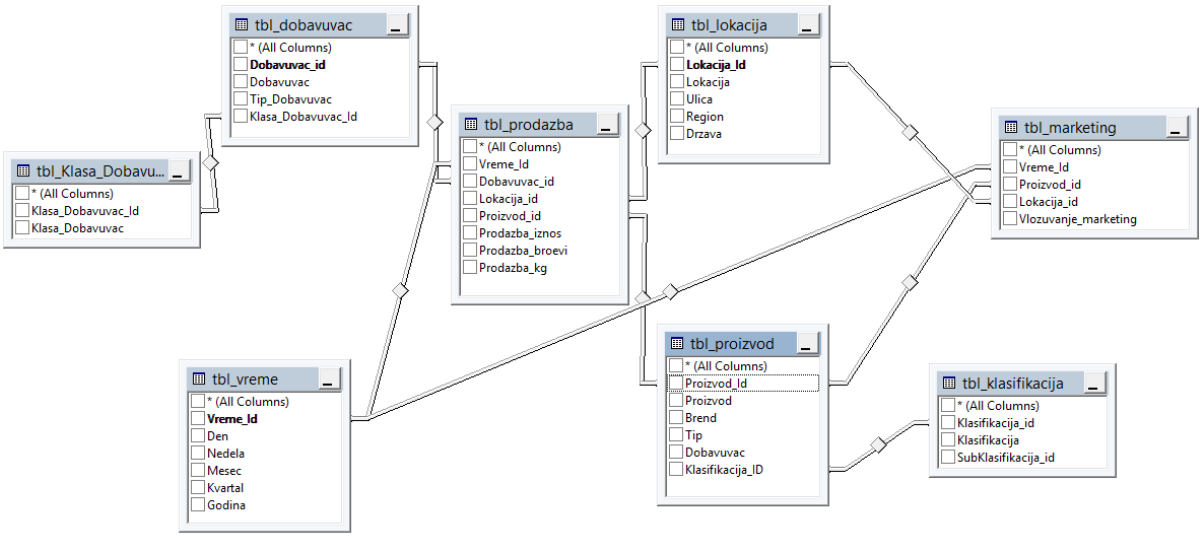
Шемата снегулка е варијанта на шемата ѕвезда каде одредени табели со димензии се нормализирани, што значи делење на податоците и во додатни табели. Главната разлика помеѓу снегулка и ѕвезда шема-моделите е тоа што табелите со димензии кај снегулка моделот можат да се чуваат во нормализирана форма за да се намали вишокот. Ваква табела лесно се одржува и зачувува простор за складирање. Од друга страна, структурата на снегулка може да ја намали ефективноста на пребарувањето бидејќи се потребни повеќе врски за да се изврши прашалникот. И покрај фактот што податочната шема

снегулка врши намалување на вишокот на податоци, сепак значително влијае на перформансите и не е популарна при креирање на податочни складови.



Слика 8. Шема снегулка
Figure 8. Snowflake schema

За да се одговори на потребите на софистицираните апликации каде се бара повеќе факт табели да делат табели со димензии неопходна е употребата на факт констелација шемата. Овој вид на шема може да се гледа како група на ѕвезди и врз основа на тоа се вика галаксија шема или факт констелација шема.



Слика 9. Шема факт констелација (галаксија)
Figure 9. Fact constellation schema (galaxy)

4.2. КАТЕГОРИЗАЦИЈА И ГЕНЕРАЛИЗАЦИЈА НА МЕРКИ

За да се одговори на прашањето како се пресметуваат мерките, прво треба да се разгледа начинот на нивната категоризација. Од мултидимензионална гледна точка просторот на податочните коцки може да се дефинира како множество на парови димензии – вредности. Мерка во податочна коцка е нумеричка функција која може да се оцени во секоја точка на податочната коцка. Вредноста на мерката се пресметува за дадена точка преку агрегирање на податоците кои кореспондираат со соодветните димензија - мерка парови кои ја дефинираат дадената точка.

Врз основа на типот на агрегатните функции кои се користат, мерките може да бидат организирани во три категории и тоа: дистрибутивни, алгебарски и холистички. (7)

За една агрегатна функција велиме дека е дистрибутивна само ако може да биде пресметана на дистрибутивен начин. Да претпоставиме дека податоците се поделени во n множества при што ја применуваме функцијата на секој дел што резултира со n агрегатни вредности. Ако резултатот што произлегува со примена на функцијата на n агрегатни вредности е ист со

резултатот со примена на функцијата на целото множество на податоци, значи функцијата може да биде пресметана на дистрибутивен начин. На пример, функцијата sum() може да биде пресметана за една податочна коцка прво со поделба на коцката на множество од подкоцки, пресметка на sum() за секоја подкоцка и на сумирање на сумите што се добиени за секоја подкоцка. Од овде, sum() е дистрибутивно агрегатна функција. На ист начин min(), max(), count() се дистрибутивни агрегатни функции.

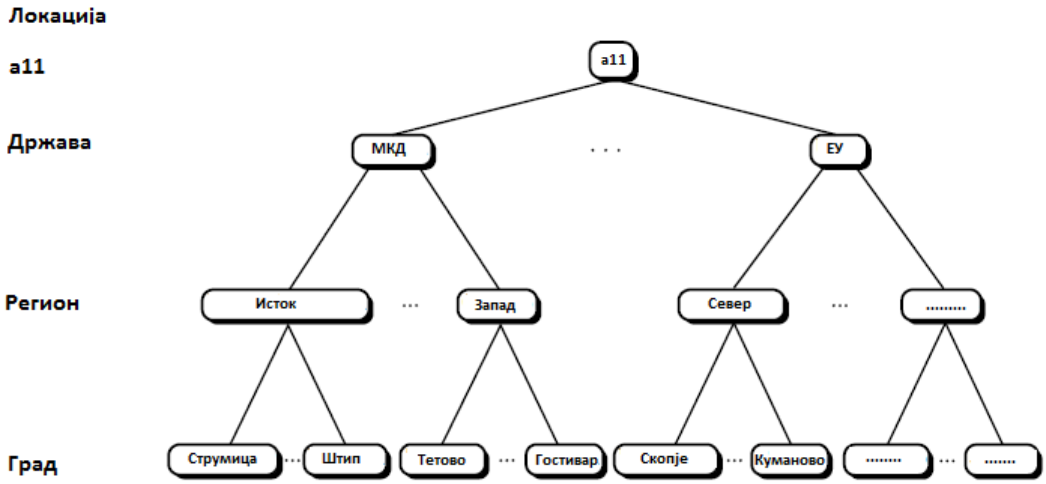
Алгебарската функција е агрегатна функција што може да биде пресметана со M аргументи (каде M е позитивен цел број), при што секој од тие аргументи се добива со примена на дистрибутивна агрегатна функција. На пример, avg()- просек може да се пресмета како sum()/count(), каде sum() и count() се дистрибутивни агрегатни функции. Исто така, може да се покаже дека min_N() и max_N() (каде се наоѓаат N минимум и N максимум вредности за даденото множество), standard_deviation() итн. Мерката е алгебарска ако се добива со примена на алгебарско агрегатни функции.

Холистичка агрегатна функција е ако нема константна граница на големината на складот потребен за опишување на субагрегатот. Тоа значи дека таму не постои алгебарска функција со M аргументи (каде M е константа) што ја карактеризира пресметката. Примери на холистички функции се median(), mode() и rang(). Најчесто апликациите кои користат големи податочни коцки бараат ефикасна пресметка на дистрибутивни и алгебарски мерки за што постојат и многу ефикасни техники. За разлика од тоа, ефикасна пресметка на холистички мерки е многу тешко. Ефикасни техники за приближна пресметка на некои холистички мерки постојат и тие се користат како замена на егзактни пресметки на холистичките мерки.

4.3. ХИЕРАРХИСКИ КОНЦЕПТИ

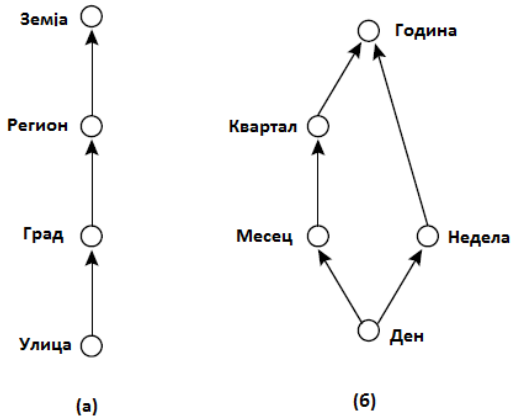
Хиерархискиот концепт дефинира секвенца на мапирање од множество на концепти од ниско ниво кон повисоко ниво, односно кон поопшти концепти. Ако се разгледува концептот хиерархијата за димензија локација, вредностите

градови вклучуваат Струмица, Скопје, Тетово итн. Секој град може да биде мапиран за одреден регион или држава на која припаѓа. Струмица југоисток, Скопје север итн. Регионите можат да бидат мапирани на држави на коишто припаѓаат или во нашиот случај Македонија. Ваквите мапирања формираат хиерархиски концепт за димензијата локација преку мапирање на множество на концепти со ниско ниво (градови) кон повисоко ниво, повеќе општи концепти (држави).



Слика 10. Хиерархиски концепт за димензијата локација
Figure 10. Hierarchical concept for dimension location

Многу хиерархиски концепти се имплицитни во шемата на базите на податоци. Да претпоставиме дека димензијата локација е опишана со атрибутите: број, улица, град, регион, поштенски број и земја. Овие атрибути се поврзани со вкупен ред, формирајќи хиерархиски концепт како што е прикажано на Слика 10.



Слика 11. Хиерархиски концепт за димензии локација и време
Figure 11. Hieratical concept for dimension location and time

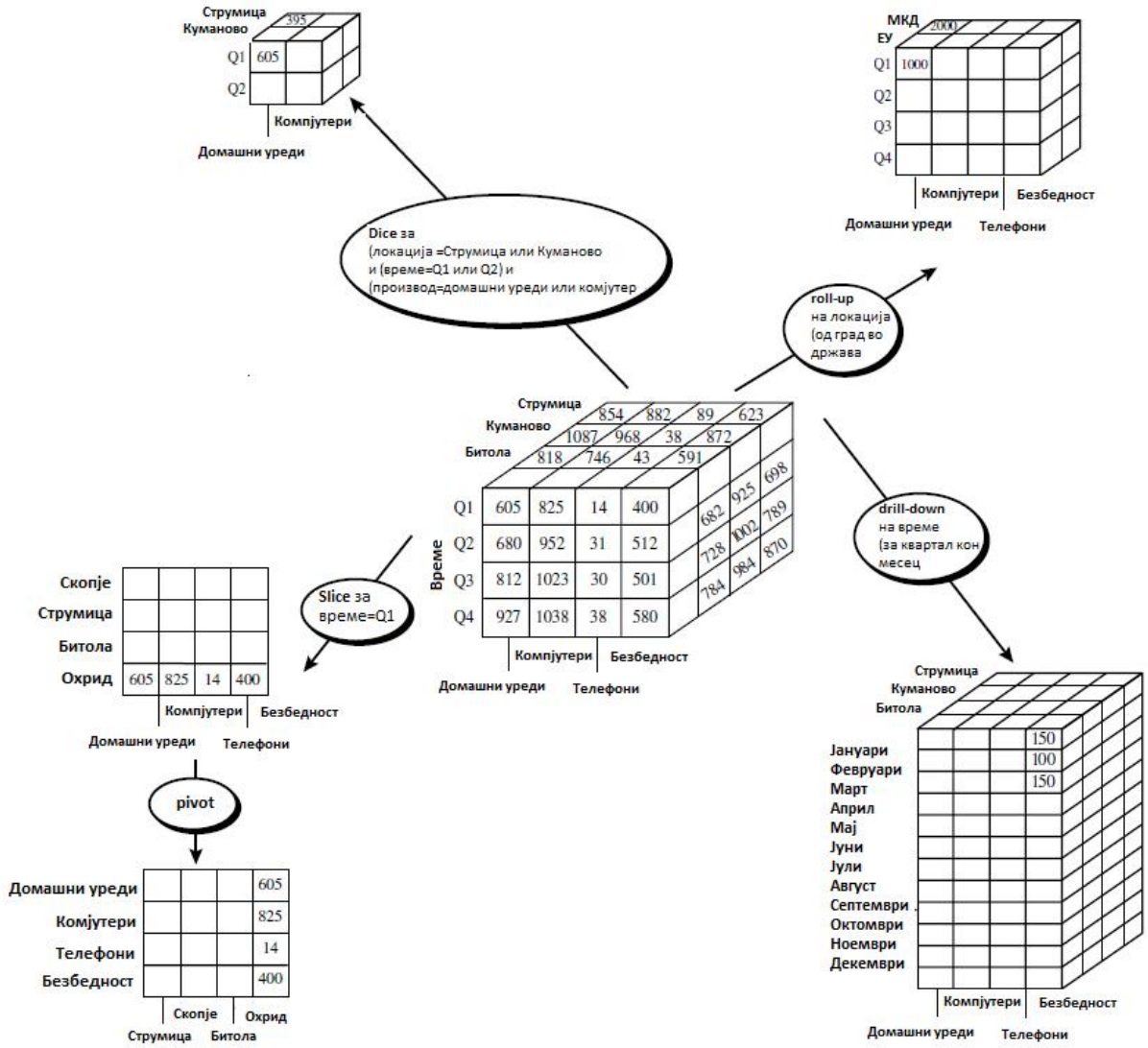
Алтернативно атрибутите на димензиите може да бидат организирани во делумен ред формирајќи латици. Делумен ред за димензијата време врз основа на атрибутите ден, недела, месец, квартал и година е опишана во Слика 11. Хиерархискиот концепт кој е вкупен или делумен ред помеѓу атрибутите во базата на податоци се вика хиерархиска шема. Хиерархиските концепти што се вообичаени за многу концепти можат да бидат предефинирани во системот за податочно рударење, како што е хиерархискиот концепт за време. Системот за податочно рударење треба да им овозможи на корисниците флексибилност за креирање на предефинирани хиерархии во согласност со нивните потреби. Корисниците можат да дефинираат фискалната година да почнува од 1. април, а академската од 1. септември. Хиерархискиот концепт, исто така, може да биде дефиниран преку групирање или дискретизација на вредности за дадена димензија или атрибут, што резултира со хиерархија за групирање множества. Групен или делумен ред може да биде дефиниран помеѓу група на вредности за димензијата цена групирана во интервал (X…Y) го означува рангот од X (ексклузивно) до Y (инклузивно). Исто така, може да има повеќе од еден хиерархиски концепт за даден атрибут или димензија кој се темели на различни точки на гледање на корисникот. Во случајот корисникот може да сака да ги организира цените со дефинирање на рангови за евтина, средна и скапа цена.

Хиерархиските концепти може да се дефинираат рачно од системски корисници, експерти или може автоматски да бидат генерирани врз основа на статистичка анализа на дистрибуцијата на податоци.

4.4. ВИДОВИ НА OLAP ОПЕРАЦИИ ВО МУЛТИДИМЕНЗИОНАЛЕН ПОДАТОЧЕН МОДЕЛ

Сега се поставува прашањето како да се искористат хиерархиските концепти во OLAP. Во мултидимензионалните модели податоците се организирани во повеќе димензии каде секоја димензија содржи повеќе нивоа на апстракција дефинирани преку хиерархиски концепти. Ова им овозможува на корисниците флексибилност при гледањето на податоците од различни перспективи. Материјализирањето на различните потреби на корисниците овозможува интерактивно пребарување и анализа на податоци при што се користат повеќе операции над OLAP податочните коцки, односно OLAP овозможува корисничко пријателска средина за интерактивна анализа на податоци.

Pivot (ротација) е визуелна операција која ја ротира податочната оска со цел да овозможи алтернативна презентација на податоците. Слика 12 покажува pivot операција, каде група на производи и локација оските во дводимензионалниот пресек се ротираат. Другите примери вклучуваат ротирање на оските во 3-D оски или трансформирање на една 3-D коцка во серија на 2-D рамнини.



Слика 12. Операции со податочни коцки
Figure 12. Data cubes operations

Slice and dice: Slice (парче) операцијата извршува селекција на една димензија од дадената коцка што дава резултат подкоцка. Слика 12 покажува операција на сечење каде податоците за продажба се селектираат од централната коцка за димензијата време користејќи го критериумот време=Q1. Dice (сечење) операцијата дефинира подкоцка преку извршување на селекција на две или повеќе димензии. На Слика 12 е прикажана dice операцијата на централната коцка преку селекција која вклучува три димензии (локација, време и група на производи).

Roll-up операцијата извршува агрегација на податочната коцка преку качување горе на хиерархискиот концепт за дадената димензија или преку редуцирање на димензијата. Слика 12 го прикажува резултатот на roll-up операција извршена на централната коцка преку качување горе на хиерархискиот концепт за димензијата локација. Со други зборови, наместо да се групираат податоците по место, резултатот на групирањето во коцката е преку податоците за земја.

Drill-down е обратна операција од roll-up и врши навигација од помалку детални податоци кон повеќе детални податоци. Drill-down може да се реализира или преку слегнување подолу на хиерархискиот концепт за дадената димензија или преку вклучување на дополнителна димензија. Слика 12 го прикажува резултатот на drill-down операција извршена врз централната коцка преку спуштање подолу на хиерархискиот концепт време, што резултира резултатите во податочната коцка да бидат прикажани по месец наместо по квартал. (8)

5. ПОДАТОЧНО РУДАРЕЊЕ

Податочното рударење е пресметковен процес за откривање на шеми во големи множества на податоци вклучувајќи методи од области на вештачка интелигенција, машинско учење, статистика и системи на бази на податоци. Целта на процесот на податочното рударење е да се извлечат информации од податочните множества и нивно трансформирање во разбирлива структура за понатамошна употреба. За разлика од чекорот на обичната анализа, податочното рударење вклучува бази на податоци и управување со податоци, пред процесирање на податоци, модели и разгледување на заклучоци, разгледување на комплексност, постпроцесирање на откриените структури, визуелизација и on-line ажурирање. Имаме и несоодветна употреба на терминот податочно рударење бидејќи целта е извлекување или пронаоѓање на шеми и знаење од голема маса на податоци, а не извлекување на податоци само по себе.

Вистинската цел на податочното рударење е автоматска или полуавтоматска анализа на голема количина на податоци со цел извлекување на претходно непознати и интересни шеми како група на податоци (кластер анализа), невообичаени податоци (откривање на аномалии) и зависности (рударење со асоцијативно правило). Шемите можат да се гледаат како еден вид на збирни влезни податоци и можат да се користат за понатамошни анализи во машинското учење и предвидувачки анализи. Чекор во податочното рударење може да открие повеќе групи во податоците, кои понатаму можат да се користат да се добие попрецизно предвидување со системот за донесување одлуки. Собирањата на податоци, подготовката на податоци, презентациите и известувањата не се чекор во податочното рударење, но припаѓаат во целиот процес на откривање на знаење како дополнителни чекори.

Краен чекор во откривањето на знаењето од податоци е да се потврди дека шемата што произлегува од алгоритамот за податочно рударење се појавува во пошироко множество податоци. Сите откриени шеми со алгоритамите за податочно рударење не се секогаш валидни. Вообичаено е

алгоритамите за податочно рударење да најдат шеми во множеството за тренирање кои не се присутни во вкупното множество на податоци. Ова се вика overfitting. Да се надмине ова, при проверката се користи тест-множество на податоци на коишто алгоритамот за податочно рударење не е трениран со што добиените шеми се применуваат на ова тест-множество и резултатите се споредуваат со посакуваниот излез. Ако добиените шеми не ги задоволуваат стандардите потребна е промена на предпроцесирањето и чекорите за податочното рударење. Ако добиените шеми се во согласност со посакуваните стандарди, тогаш конечен чекор е да се претстават научените шеми и нивно претворање во знаење.

Податочното рударење, за жал, може да се злоупотреби. Добиените резултати за кои се мисли дека се значајни, може да не го предвидуваат идното однесување и не може да бидат репродуцирани на нов примерок на податоци при што имаат минимална употреба. Обично тоа е резултат на вклучување на многу хипотези и неизвршување на валидно статистичко тестирање на хипотезите.

5.1. ЦЕЛИ НА ПОДАТОЧНОТО РУДАРЕЊЕ

Најчесто се наведуваат шест цели на податочното рударење:

- откривање на аномалии е идентификација на невообичаени податоци кои можат да бидат интересни или податочни грешки кои бараат понатамошно истражување;
- асоцијативно правило (моделирање на зависности) е пребарување на врски помеѓу варијаблите;
- кластерирање има за цел откривање на групи и структури на податоци кои на некој начин се слични, без користење на познати структури во податоците;
- класификација е кога имаме генерализација на познати структури и ги применуваме на нови податоци. Пример за тоа е кога e-mail програма се обидува да класифицира електронски пораки како спам и нормални;

- регресијата се обидува да најде функција која ги моделира податоците со најмалку грешки;
- сумирање овозможува покомпактна презентација на множеството на податоци, вклучувајќи визуелизација и генерирање на извештај. (9)

5.2. ОТКРИВАЊЕ НА АНОМАЛИИ

Во податочното рударење откривањето на аномалии или откривањето на отстапувања е процес на откривање на ставки, настани или набљудувања кои не се во согласност со очекуваната шема или другите ставки во податочното множество. Обично ставките со аномалии може да претставуваат проблеми како банкарски измами, структурни недостатоци, медицински проблеми или пронаоѓање на грешки во текст. Аномалиите може да бидат отстапувања, бучава, новини, девијации и исклучоци.

Во контекст со злоупотреби на мрежа или откривање на упад, од интерес не се ретките објекти, туку неочекуван излив на активности. Оваа шема не е вообичаена за статистичката дефиниција на отстапување како редок објект и многу методи за откривање на отстапувања (воглавно ненадгледувани методи) ќе потфрлат на вакви податоци, освен ако не се агрегирани правилно. За разлика од тоа алгоритамот на кластер анализа може да открие микрокластери формирани од тие шеми.

Постојат три категории на техники за откривање на аномалии.

- техниката за ненабљудувано откривање на аномалии, открива аномалии во непроверето множество на податоци за тестирање, под претпоставка дека главнината на инстанциите во податочното множество се нормални преку проверка на инстанции за кои се смета дека се поклопуваат со остатокот на податочното множество;
- техника за набљудувано откривање на аномалии бара податочното множество кое е одбележано како нормално или ненормално и вклучува тренирање врз основа на претходно наведеното;

- техниката за делумно надгледување на откривање на аномалии конструира модел кој го претставува нормалното однесување за дадено нормално множество на податоци кое се тренира и потоа се тестира веројатноста на тест-инстанцијата. (10)

Како најчести техники за откривање на аномалии се:

- Техники базирани на густината - k-nearest neighbor, local outlier factor;
- Support vector machines;
- Neural networks;
- Cluster analysis.

5.2.1. АСОЦИЈАТИВНО ПРАВИЛО

Правилото на асоцијативно учење е популарен и добро истражуван метод за откривање на интересни врски помеѓу варијабли во големи бази на податоци. Насочена е кон откривање на силни правила во базите на податоци со користење различни мерки. Следејќи ја основната идеа на Rakesh Agrawal проблемот на рударење со асоцијативно правило е дефиниран како: Нека $I = \{i_1, i_2, \ldots\ldots, i_n\}$ е множество на n бинарни атрибути наречени елементи. Нека $D = \{t_1, t_2, \ldots\ldots, t_n\}$ е збир на трансакции кое се нарекува база на податоци. Секоја трансакција во D има единствен трансакциски идентификатор ID и содржи подмножество на елементи во I. Правилото се дефинира како импликација на формата $X \Rightarrow Y$, каде $X, Y \subseteq I$ и $X \cap Y = \emptyset$. Множеството од елементи за X и Y се нарекуваат претходници (од лева страна) и последователни (од десна страна).

За да се претстави концептот може да се искористи пример од супермаркет. Збирот на елементите $I = \{\text{прашок}, \text{омекнувач}, \text{јогурт}, \text{млеко}\}$

Табела 4. Табела со трансакции
Table 4. Transactions table

ID	прашок	омекнувач	јогурт	млеко
1	1	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	1	1	0
5	0	1	0	0

и мала табела која содржи елементи со кодови 1 за присуство и 0 за отсуство на елементот во трансакцијата. Овде правилото за супермаркетот би било $\{\text{јогурт}, \text{омекнувач}\} \Rightarrow \{\text{прашок}\}$. Значи ако купи јогурт и омекнувач купувачот исто така купува и прашок. За селектирање на интересните правила од множеството врз сите можни правила може да се користат ограничувања на различни мерења на значење и интерес.

Најпознати ограничувања се минимален праг на поддршка и доверливост.

- Поддршката supp(X) на множество елементи X е дефинирана како пропорција од трансакциите во множеството на податоци кои го содржат елементот. Во табелата множеството на елементи, $\{\text{прашок}, \text{јогурт}, \text{омекнувач}\}$ има поддршка од 1/5 што е еднакво на 0.2, бидејќи се појавува во 20% од сите трансакции.
- Доверба на правилото се дефинира $conf(X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}$. Правилото $\{\text{јогурт}, \text{омекнувач}\} \Rightarrow \{\text{прашок}\}$ има доверливост од 0.2/0.2=1.0, што значи дека за 100% од трансакциите кои содржат омекнувач и јогурт правилото е точно (кога купувач купува јогурт и омекнувач 100% купува и прашок). Треба да се внимава кога се чита изразот $supp(X \cup Y)$, значи поддршка на појавување на трансакции каде X и Y заедно се појавуваат, а не поддршка на појавување на трансакции каде X или Y се појавуваат. (11)

- Доверливоста може да биде претставена како проценка на веројатност P(Y|X), веројатност за пронаоѓање на правилото во трансакцијата од десната страна, под услов дека истите тие трансакции се наоѓаат и од левата страна.
- Подигање на правилото се дефинира како $lift\ (X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X) * supp(Y)}$ или коефициент на анализираната поддршка со очекуваната ако X или Y беа независни. Правилото $\{\text{прашок, омекнувач}\} \Rightarrow \{\text{јогурт}\}$ има лифт од $\frac{0.2}{0.4*0.4} = 1.25.$
- Убедливост на правилото се дефинира како $conv\ (X \Rightarrow Y) = \frac{1-supp(Y)}{1-conv\ (X \Rightarrow Y)}$. Правилото $\{\text{прашок, омекнувач}\} \Rightarrow \{\text{јогурт}\}$ има убедливост од $\frac{1-0.4}{1-0.5} = 1.2$ и може да се толкува како коефициент од очекуваната фреквенција на појавување на X без да се појави Y (што може да се каже, фреквенција дека правилото ќе направи погрешно предвидување) ако X или Y беа независни поделени со набљудуваната фреквенција на погрешни предвидувања. Така вредноста од 1.2 покажува дека правилото $\{\text{прашок, омекнувач}\} \Rightarrow \{\text{јогурт}\}$ ќе биде погрешно за 20% ако асоцијацијата помеѓу X и Y е врз основа на случаен избор. (12)

Псевдокодот на асоцијативното правило е:

Нека n е број на посакувани кластери
Нека S е множество на карактеристични вектори (|S| е големина на множество)
Нека A е множество на поврзани кластери за секој карактеристичен вектор
Нека sim(x,y) е функција на сличности
Нека c[n] се вектори за нашите кластери

Иницијализирај:

```
Нека S' = S
  //избери n произволни вектори за почеток на кластерите
  за i=1 до n
     j = rand(|S'|)
     c[n] = S'[j]
     S' = S' - {c[n]} //отстрани го овој вектор од S' за да може да се избере повторно
  крај за

  //додели почетни кластери
  за i=1 до |S|
     A[i] = argmax(j = 1 до n) { sim(S[i], c[j]) }
```

```
крај за

Старт:

  Нека промена = точно
    додека промена
      промена = неточно //претпоставуваме дека нема промена
                        //додели повторно карактеристични вектори на
                        //кластерите

  за i = 1 до |S|
    a = argmax(j = 1 до n) { sim(S[i], c[j]) }
     ако a != A[i]
       A[i] = a
       промена = точно //вектор ја променува припадноста – треба да ги
                       //пресметаме повторно кластер вектори и
                       //стартуваме повторно
     крај ако
   крај за

 //пресметај повторно кластер локации ако се случи промена
 ако промена
   за i = 1 до n
     очекувана вредност, бројач = 0
      за j = 1 до |S|
        ако A[j] == i
          очекувана вредност = очекувана вредност + S[j]
          бројач = бројач + 1
       крај ако
      крај за

        c[i] = очекувана вредност /бројач
   крај за
 крај ако (13)
```

5.3. КЛАСТЕР АНАЛИЗА

Кластер анализата или кластерирање има за цел групирање на множество на објекти на начин каде објектите во иста група се повеќе слични помеѓу себе отколку со другите од други групи. Групите коишто се формираат се нарекуваат кластери. Кластер анализата сама по себе не е некој специфичен алгоритам и воглавно проблемот се решава преку различни алгоритами кои се разликуваат значајно во сфаќањето на тоа што го сочинува кластерот и ефикасноста во пронаоѓање на неговите елементи. Според тоа, не може да се даде една точна дефиниција за кластер што е една од причините за постоење на многу алгоритами за кластер. Сепак сите тие дефиниции имаат еден заеднички

именител, а тоа е група на податочни објекти. Со оглед на тоа што различни истражувачи користат различни модели и алгоритами за кластери, разбирањето на кластерот се разликува значително по неговите особености. Разбирањето на овие кластер модели е клучно за разбирање на разликите помеѓу различни алгоритами. Вообичаените кластер модели вклучуваат:

- модел за поврзаност, на пример хиерархиско кластерирање гради модел врз основа на далечинско поврзување;
- центроид модел, на пример k-means алгоритам го претставува секој кластер како единичен среден вектор;
- дистрибутивни модели, кластерите се моделираат со користење на статистичка дистрибуција како мултиваријантна нормална дистрибуција користена од алгоритам за максимизација на очекувањето;
- модели за густина, на пример dbscan и OPTICS ги дефинираат кластерите како поврзани региони со густина во податочниот простор.

5.3.1. ХИЕРАРХИСКИ КЛАСТЕРИ

Основна идеа на хиерархиското кластерирање е дека блиските објекти се повеќе поврзани отколку оддалечените. Овие алгоритами ги поврзуваат објектите да формираат кластери врз основа на нивното растојание. Овде кластерот може да се опише како максимално растојание потребно да ги поврзеш деловите на еден кластер, при што на различни растојанија се формираат различни кластери коишто можат да се претстават со користење на дендрограм. Во дендрограмот у-оската го означува растојанието на кое кластерите се поврзуваат, додека објектите што се сместени покрај х-оската не се мешаат во кластерот.

Стратегиите за хиерархиско кластерирање генерално се делат на два типа:

- Агломеративни имаме каде пристапот е „од доле – нагоре“ и секое набљудување започнува во сопствениот кластер и парови на кластери се спојуваат кога се движи нагоре во хиерархијата.
- Поделен (devisive) имаме кога пристапот е „одгоре – надолу“ и сите набљудувања започнуваат во еден кластер, а поделбите се извршуваат рекурзивно кога се движи надолу во хиерархијата.

Во општ случај комплексноста на агломеративните кластери е $O(n^3)$, што кај големи множества на податоци ги прави многу бавни. Divisive (делбено) кластерирање со сеопфатно пребарување е $O(2^n)$, што е дури полошо. За да се одлучи кои кластери ќе се комбинираат за агломеративно или каде кластерот треба да се подели потребно е да се измери различноста помеѓу множествата кои се набљудуваат. Кај повеќето методи од хиерархиското кластирање тоа се постигнува со употреба на соодветни мерки за мерење на растојанијата помеѓу паровите кои се набљудуваат и критериумот за поврзување кој ја покажува различноста на множествата како функција од раздалеченост на парови на множествата кои се набљудуваат.

Изборот на соодветни метрики ќе влијае на формата на кластерот така што некои елементи ќе бидат поблиску, а други подалеку едни од други. На пример во 2-D простор растојанието помеѓу точката (1,0) и почетокот (0,0) е секогаш 1 во согласност со вообичаените норми, но растојанието помеѓу (1,1) и почетната (0,0) може да биде 2 во согласност со Manhattan растојание, $\sqrt{2}$ според Евклидовото растојание или 1 според максималното растојание. Најчесто користени метрики за хиерархиско кластерирање се:

$\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$ - Евклидово растојание

$\|a-b\|_2^2 = \sum_i (a_i - b_i)^2$ - Квадратно Евклидово растојание

$\|a-b\|_1 = \sum_i |a_i - b_i|$ - Менхетеново растојание

$\|a-b\|_\infty = \max_i |a_i - b_i|$ - максимална растојание $\sqrt{(a-b)^T S^{-1}(a-b)}$ - Махаланобисова растојание, каде S е коваријансна матрица (14)

Критериумот за поврзување го утврдува растојанието помеѓу множествата кои се набљудуваат како функција на растојанија на парови помеѓу набљудувањата. Како најчесто користени критериуми за поврзување помеѓу две множества на набљудување A и B се:

$\max\{d(a,b) : a \in A, b \in B\}$ - максимална или кластерирање на целосно поврзување;

$\min\{d(a,b) : a \in A, b \in B\}$ - минимална или кластерирање на поединечно поврзување;

$\frac{1}{|A||B|}\sum_{a\in A}\sum_{b\in B} d(a,b)$ - просек или кластерирање на просечно поврзување или UPGMA;

$\|c_s - c_t\|$ каде c_{sand} c_{tare} се центроиди на кластери s и t соодветно – центроидно поврзување на кластери UPGMC.

Каде d е избраната метрика, другите критериуми за поврзување вклучуваат:

- сума на сите интра-кластер варијанси;
- намалување на варијансата за кластер којшто се спојува;
- веројатноста дека кандидатите кластери произлегуваат од истата дистрибутивна функција (V- поврзување);
- производот од влезните и излезните врски на K-nearest neighbor график;
- зголемувањето на некои опишувачи на кластери после спојувањето на два кластери. (15)

5.3.2. ЦЕНТРОИДНО БАЗИРАНИ КЛАСТЕРИ

Кластерите кај центроидно базираните кластери се претставуваат преку централен вектор кој не мора секогаш да биде елемент на податочното множество. Кога бројот на кластери е фиксен, k-means кластерирање е основа за решавање на проблемот. Тоа е метод за векторска квантификација кој има за цел поделба на n опсервации во k кластери при што секоја опсервација припаѓа на кластер со најблиска средна вредност, служејќи како прототип за кластерот. Резултатот е поделба на податочниот простор во Voronoi клетки.

За дадено множество на набљудување $(x_1, x_2, \dots, x_n)$, каде секое набљудување е d-димензионален реален вектор, k-means кластерирањето има за цел да ги подели n набљудувањето во $k(\leq n)$ множества $S = \{S_1, S_2, \dots, S_k\}$ за да го минимизира квадратот на сумите внатре во кластерот или со други зборови потребно е да се најде

$$\arg\min_{s} \sum_{i=1}^{k} \sum_{x \in S_i} d(x, \mu_i) = \arg\min_{s} \sum_{i=1}^{k} \sum_{x \in S_i} \| x - \mu_i \|^2$$

Каде μ_i е средина на точките во S_i. (16)

Лојдовиот алгоритам, како најпознат k-means алгоритам, се користи за решавање на проблеми од k-means кластерирање и работи на следниот начин:

1. Утврдување на број на кластери k	
2. Иницијализирање на центар на кластерите	$\mu_i = \text{некоја вредност}, i = 1, \dots, k$
3. Утврдување на најблизок кластер за секоја податочна точка	$c_i = \{j: d(x_j, \mu_i) \leq d(x_j, m_I), I \neq i, j = 1, \dots n\}$

4. Поставување на позицијата на секој кластер до средината на сите податочни точки што припаѓаат на тој кластер	$\mu_i = 1\|c_i\| \sum j \in c_i x_i, \forall_i$
5. Повторување на чекорите 2-3 до конвергенција	
каде	|c|= број на елементи во c

5.3.3. КЛАСТЕРИ БАЗИРАНИ НА ДИСТРИБУЦИЈА

Кластерите наједноставно може да се дефинираат како објекти кои најверојатно припаѓаат на иста дистрибуција. Својство на овој метод е што потсетува на вештачко креирање податочни множества преку земање случајни објекти од дистрибуцијата.

Иако теоретската основа на овие методи е совршена, тие страдаат од еден клучен проблем познат како overfitting, освен ако не се ставени ограничувања на сложените модели. Посложените модели обично се во состојба да ги објаснат податоците подобро, што од друга страна го прави изборот на соодветен модел многу комплексно.

Еден од најпознатите методи е Гаусов модел на комбинација кој го користи алгоритам за максимизација на очекувањето. Овој алгоритам користи параметри со максимална веројатност на статистичкиот модел во случаи каде равенките не можат да се решат директно. Овие модели вклучуваат латентни варијабли како дополнување на непознати параметри и познати опсервации на податоци. Тоа се изгубени вредности во податоците или моделот може да се формулира поедноставно преку претпоставка за постоење на дополнителни не опсервирани податочни точки.

Даден статистички модел кој генерира множество X на опсервирани податоци, множество на неопсервирани латентни податоци или исчезнати вредности Z и вектор од непознати параметри θ заедно со функција на веројатност $L(\theta; X, Z) = p(X, Z|\theta)$ максималната проценка на веројатноста на непознати параметри се утврдува преку маргинална веројатност од опсервираните податоци

$$L(\theta; X) = p(X|\theta) = \sum_{Z} p(X, Z|\theta)$$

Овде квантификацијата е често нерешлива, односно ако *Z* е секвенца на настани, при што бројот на вредности расте експоненцијално со должината на секвенцата, ја прави пресметката на сумата екстремно тешка.

Овој алгоритам бара максимална проценка на маргиналната веројатност преку постојано извршување на следниве два чекори:

Чекор на очекување (Е чекор): Пресметува очекувана вредност од функцијата логаритам на веројатноста, со почитување на условната дистрибуција на Z на даден X под дадената проценка на параметрите $\theta^{(t)}$:

$$Q\left(\theta\middle|\theta^{(t)}\right) = E_{Z|X,\theta^{(t)}}[\log L\,(\theta; X, Z)]$$

Чекор за максимизација (М чекор): наоѓа параметар кој ја максимизира таа количина:

$$\theta^{(t+1)} = arg_{\theta} \max Q(\theta|\theta^{(t)})$$

Треба се има предвид дека во типичните модели каде се применува алгоритамот на проценка – веројатност имаме:

1. Опсервираните податочни точки X може да бидат дискретни (земајќи вредности во конечно или пребројиво бесконечно множество) или континуирано (земајќи вредности во непребројиво бесконечно множество). Може да биде вектор на опсервација поврзана со секоја податочна точка.

2. Изгубени вредности или латентни варијабли Z се дискретни, изведени од фиксен број на вредности и има една латентна вредност за опсервираната податочна точка.
3. Параметрите се континуирани и ги има два вида. Параметри кои се поврзани со сите податочни точки и параметри поврзани со одредена вредност на латентна варијабла односно поврзани со сите податочни точки чија соодветна латентна варијабла има одредена вредност.

Проценка – максимизирање можно е, исто така, да се примени и на други видови модели. Ако ја знаеме вредноста на параметрите θ , вообичаено можеме да ја најдеме и вредноста на латентните варијабли Z преку максимизирање на логаритамската веројатност над сите можни вредности од Z или едноставно преку процесирање над Z или преку алгоритам како Ветерби или алгоритам за скриени Маркови модели. Обратно, ако ја знаеме вредноста на латентните варијабли Z, многу лесно можеме да најдеме проценка на параметрите θ преку групирање на опсервираните податочни точки според вредноста на поврзаните латентни варијабли и просек на вредностите или некои функции од вредностите за точките во секоја група.

Тоа укажува на повторлив алгоритам во случаи каде заедно θ и Z се непознати:

1. Прво се инцијализираат параметрите θ на некој случајни вредности.
2. Се пресметуваат најдобрите вредности на Z за дадените параметри.
3. Потоа се користат тукушто пресметаните вредности на Z за пресметка на подобра проценка за параметрите θ. Параметрите поврзани со одредена вредност на Z ќе ги користат само оние податочни точки чии поврзани латентни варијабли ја имаат таа вредност.
4. Повторување на чекорите 2 и 3 до конвергенција.

Но, исто така, можеме да направиме нешто подобро од правење на тежок избор за Z со оглед на тековните вредности на параметарот и просеците само врз множество од податочни точки поврзани со одредена вредност на Z, наместо утврдување на веројатноста на секоја можна вредност на Z за секоја податочна

точка и потоа користење на веројатностите поврзани со одредената вредност на Z за пресметка на пондериран просек над вкупното множество на податочни точки. Како резултат се добива алгоритам познат како алгоритам за максимизација на очекувањето. Точките користени за пресметка на пондерираната средина се викат меки точки (спротивно на тврдите точки користени во алгоритам од типот k-means). Пресметаните веројатности за Z се задни веројатности и е тоа што се пресметува во E чекорот. Меките точки користени за пресметка на нов вредности на параметарот е тоа што се пресметува во M чекорот.

6. КРЕИРАЊЕ И УПОТРЕБА НА СИСТЕМ ЗА РУДАРЕЊЕ НА ПОДАТОЦИ (ПРАКТИЧНА ИМПЛЕМЕНТАЦИЈА)

Врз основа на праксата, денешните деловни апликации се креирани да собираат информации во најразлични форми. Собраните податоци се тесно поврзани со функционалниот процес на организациите и може да потекнуваат од различни извори, влезови и системи. Купувачи, артикли, фактури, порачки, вработени, нормативи на готови производи, следење на технолошкиот процес се само некои примери за собирање, обработка, складирање и акумулирање на огромни количини на податоци. Складирањето на податоците во бази на податоци е со цел да се одговори на дневните операции без извлекување на некоја дополнителна вредност. Системите воглавно се креирани за OLTP (On Line Transaction Processing), за да може да се справат со огромен број на online трансакции како вметнување, ажурирање и бришење на трансакции, како и одржување на интегритетот на податоците во повеќе корисничко опкружување.

Тековните системи, покрај собирањето на податоци, имаат можности за извлекување на податоците од базите на податоци и нивно трансформирање во информации. Ваквата можност за пребарување и креирање на извештаи како залихи на производи, биланси, топ-купувачи или артикли, продажба по категории на производи, одјави за нарачки во малопродажба итн. им овозможува на корисниците следење на тековното работење и донесување на краткорочни одлуки.

Зголемената конкуренција, динамиката на пазарите и потребата да се предвидат идните текови со голема веројатност, го наметнува прашањето дали можеме да извлечеме уште некоја додатна корист од постоечките податоци, нормално не нарушувајќи го постоечкиот информационен систем. Со примена на техники на податочно рударење кои се базирани на алгоритами и статистика се овозможува трансформирање на информацијата во знаење, односно овозможува идентификување на скриените трендови и невообичаените примероци во податоците. Само организациите кои ја разбираат вистинската

вредност на податоците и технологијата напредуваат и имаат конкурентска предност која им овозможува сигурна иднина.

Ваквата состојба беше причина и поттик да се спроведе практична имплементација на еден систем кој ќе биде во склад со најновите технологии. Практичниот дел беше изведен врз основа на реални податоци од работата на една компанија во периодот од 2009 до јануари 2015 година. Во анализата беа опфатени податоците за комплетното работење на компанијата.

Зошто беше имплементиран системот?

- За да се обезбеди искористување на историските податоци во периодот од 2009-2015.
- Да се прикаже и образложи комплетниот процес на ИТ секторот на компанијата.
- Да се запознаат менаџерите со една сосема нова димензија на искористување на податоците во менаџирањето.
- Да се даде една основа за понатамошен развој и автоматизација на целиот процес.

Како беше спроведена имплементацијата?

Базите на податоци беа анализирани, при што се констатира дека постојат разлики во однос на бројот на табелите и структурата на самите табели од година во година. Анализираните бази на податоци се задржуваат на сегашните сервери и се направи план како да се искористат податоците без да се наруши интегритетот на тековното работење. Воедно, се водеше сметка да не се нарушат перформансите на OLTP процесот како примарна функција на тековниот систем.

Користени технологии

Цел на имплементацијата на системот беше да се искористат најновите технологии за бази на податоци преку креирање на извештаи и примена на техники на податочно рударење. За таа цел се користени:

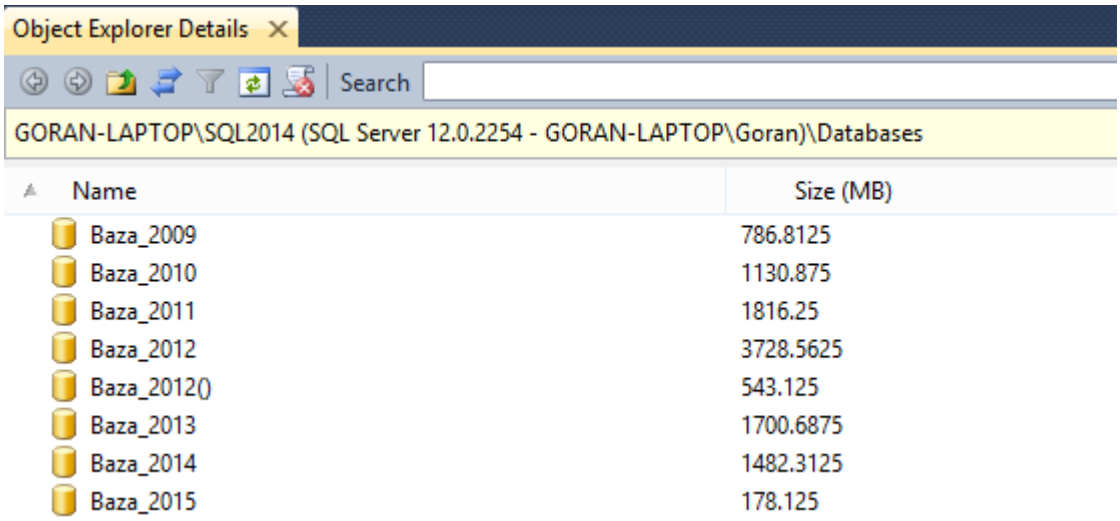
- SQL Server 2014 trial како систем за менаџирање релациони бази на податоци. Системот ќе ги управува историските бази на податоци и новоформираниот податочен склад. Се користи најновата технологија од 2014 година што може да се заклучи и од самиот назив на серверот.
  Visual Studio Ultimate 2013 trial се користи за изработка на два проекти:
  - Проект за процес на екстракција, трансформација и полнење (ETL-Extraction Transformation Loading);
  - Проект за креирање на податочна коцка и модели за податочно рударење.
- Microsoft Office Professional Plus 2013 trial се користи за прикажување на добиените резултати во форма која лесно е читлива и разбирлива од страна на корисниците.
- SQL Server Data Mining Add-ins е алатка која овозможува креирање на модели за податочно рударење и визуелно прикажување на добиените резултати.
- Orange е софтвер со отворен код кој служи за визуелизација на податоци, примена на статистички техники и техники на податочно рударење.

Утврдување на извори на податоци

Како извори на податоци се користат Microsoft SQL Server релациони бази на податоци и тоа:

Табела 5. Користени бази на податоци
Table 5. Used databases

База на податоци	Големина (MB)	Број на табели	Податоци
Baza_2009	786.81	91	4,454,151
Baza_2010	1,130.88	164	6,584,645
Baza_2011	1,816.25	176	9,817,423
Baza_2012	3,728.56	219	11,037,926
Baza_(Pocket PC)	543.13	27	3,472,343
Baza_2013	1,700.69	245	5,516,942
Baza_2014	1,482.31	255	7,917,991
Baza_2015	178.13	255	1,152,813
Вкупно	**11,366.75**	**1,432**	**49,954,234**



Слика 13. Користени бази на податоци SQL Server
Figure 13. Used SQL Server databases

Секоја од овие релациони бази на податоци има:

- единствено име кое што ја означува годината за која се однесува базата на податоци;
- табели чиј број варира од година во година. Бројот на табелите како што може да се види од Табела 5 се зголемува од година во година со цел да одговори на зголемените барања на информациониот систем;
- Податочни полиња кои дефинираат типови на податоци кои ќе се складираат во истите. Кај одредени полиња има промена во типот на податоци согласно еволуцијата на системот и најчести модификации се:
    - smallint во int – зголемување на капацитетот од -2^15 (-32,768) до 2^15-1 (32,767) на -2^31 (-2,147,483,648) до 2^31-1 (2,147,483,647).
    - varchar() во nvarchar() – можност да се запишуваат текстуални податоци со локална поддршка.
    - nvarchar (помал капацитет) во nvarchar(поголем капацитет). Пример nvarchar(30) во nvarchar(50).

- Број на податочни редови.

Како што може да се види во Табела 5, како извор на податоци се користат осум релациони бази на податоци со големина од 11,366.75MB, 1432 податочни табели и 49,954,234 податочни записи.

6.1. КРЕИРАЊЕ НА ПОДАТОЧЕН СКЛАД

Податочниот склад служи за физичка имплементација на податочен модел, место каде ќе се чуваат информации на организиран начин подготвени за понатамошна употреба. Тргнувајќи од тоа, како прво потребно е да се анализираат и разберат потребите на бизнисот. Самата рамка за бизнис анализа наметнува одговор на четири основни прашања во однос на дизајнот на податочниот склад:

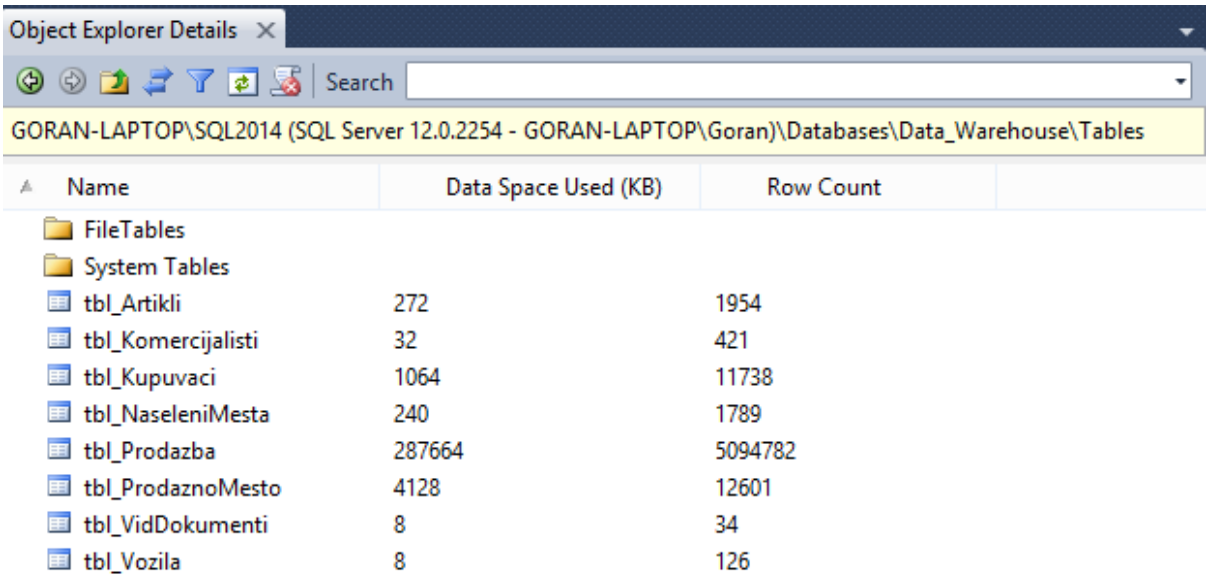
- Дефинирање на сегашните и идните деловни потреби на компанијата.
- Дефинирање на деловните прашалници или перспективата на податоците од аспект на крајниот корисник
- Табели со факти и димензии.
- Извори на податоци кои се користат и нивната обработка од аспект на компатибилност со планираниот податочен склад.

Од аспект на сегашното работење потребите се фокусираат кон анализа на продажбата. Анализата ќе биде достапна до сите структури вклучени во процесот на продажба. Идните планови вклучуваат надополнување на системот за потребите на маркетингот, човечките ресурси и финансиите. Планот е системот да го користат најмалку 80 корисници преку извршување на комплексни операции без да се нарушат перформансите на тековниот систем за OLTP. За да се одговори на овие потреби потребно е креирање на нова база на податоци Data_Warehouse.mdf.

```
CREATE DATABASE [Data_Warehouse]
 CONTAINMENT = NONE
 ON  PRIMARY
```

```
( NAME = N'Data_Warehouse', FILENAME = N'D:\Bazi Magisterska\Data_Warehouse.mdf' ,
SIZE = 5120KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB )
 LOG ON
( NAME = N'Data_Warehouse_log', FILENAME = N'D:\Bazi
Magisterska\Data_Warehouse_log.ldf' , SIZE = 1024KB , MAXSIZE = 2048GB , FILEGROWTH =
10%)
GO
ALTER DATABASE [Data_Warehouse] SET COMPATIBILITY_LEVEL = 120
GO
```

Следниот чекор е креирање на табели, процес во кој се дефинираат мерките и димензиите кои се користат понатаму во анализите. Покрај складирање на податоци, структурата на табелите треба да одговори на потребите за креирање на податочни коцки и модели за податочно рударење.



Слика 14. Табели во податочен склад
Figure 14. Tables in data warehouse

Место каде се дефинираат и зачувуваат мерките е табелата tbl_Prodazba. Вообичаено мерките се нумерички вредности и во нашиот случај тоа се податочните полиња ProdazbaIznos, ProdazbaKg, ProdazbaBroevi и ProdazbaM3. Останатите полиња ги претставуваат врските со димензиите артикли, продажни места, видови документи, комерцијалисти, возила, купувачи и дата (димензија време).

```
CREATE TABLE [dbo].[tbl_Prodazba](
	[ProdazbaId] [int] IDENTITY(1,1) NOT NULL,
```

```
	[ArtikalId] [int] NULL,
	[ProdaznoMestoID] [int] NULL,
	[VidDokumentID] [int] NULL,
	[KomercijalistID] [int] NULL,
	[VoziloID] [int] NULL,
	[ProdazbaIznos] [real] NULL,
	[ProdazbaKg] [real] NULL,
	[ProdazbaBroevi] [real] NULL,
	[ProdazbaM3] [real] NULL,
	[Kupuvac_ID] [int] NULL,
	[Data] [date] NULL,
 CONSTRAINT [PK_tbl_Prodazba] PRIMARY KEY CLUSTERED
(
	[ProdazbaId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

GORAN-LAPTOP\SQ...dbo.tbl_Prodazba

Column Name	Data Type	Allow Nulls
ProdazbaId	int	☐
ArtikalId	int	☑
ProdaznoMestoID	int	☑
VidDokumentID	int	☑
KomercijalistID	int	☑
VoziloID	int	☑
ProdazbaIznos	real	☑
ProdazbaKg	real	☑
ProdazbaBroevi	real	☑
ProdazbaM3	real	☑
Kupuvac_ID	int	☑
Data	date	☑

Слика 15. Табела tbl_Prodazba
Figure 15. Table tbl_Prodazba

Откако е креирана основната табелата со мерки tbl_Prodazba, се преминува на дефинирање и креирање на табели во кои ќе се зачувуваат податоците за димензиите.

Табелата tbl_Artikli ја претставува димензијата артикли. За да можеме да вршиме понатамошна анализа, врз основа на сите аспекти поврзани со хиерархијата артикл, потребно е на едно место да се зачувуваат податоци за

артикл, програма и добавувач. Клучот ArtikalID е врската со табелата tbl_Prodazba, односно овозможува поврзување со мерките на димензијата артикли. За разлика од изворните бази на податоци каде податоците се нормализирани, овде е извршено групирање со цел подобрување на перформансите. Како извор на податоци за оваа табела се табелите Artikli, Programi и Dobavuvac и ова ќе го видиме понатаму во процесот на екстракција трансформација и полнење ETL. Овде, како што е прикажано со SQL кодот, податочните полиња се дефинирани за да овозможат компатибилност со историските бази на податоци.

```
CREATE TABLE [dbo].[tbl_Artikli](
	[Artikal_ID] [int] NOT NULL,
	[Artikal] [nvarchar](50) NULL,
	[Edmera] [varchar](5) NULL,
	[Programa] [nvarchar](50) NULL,
	[Dobavuvac] [nvarchar](50) NULL,
 CONSTRAINT [PK_tbl_Artikli] PRIMARY KEY CLUSTERED
(
	[Artikal_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

GORAN-LAPTOP\SQL2...- dbo.tbl_Artikli | Object Explorer Details

Column Name	Data Type	Allow Nulls
Artikal_ID	int	☐
Artikal	nvarchar(50)	☑
Edmera	varchar(5)	☑
Programa	nvarchar(50)	☑
Dobavuvac	nvarchar(50)	☑

Слика 16 Табела tbl_Artikli
Figure 16 Table tbl_Artikli

Табелата tbl_Komercijalisti се користи за дефинирање и зачувување на димензијата комерцијалисти. Оваа димензија овозможува анализа врз основа на два параметри комерцијалист и продажна дивизија. Структурирана е да овозможи компатибилност со изворните табели. KomercijalistID е примарниот клуч кој ја поврзува димензијата комерцијалисти со табелата со мерки tbl_Prodazba.

```
CREATE TABLE [dbo].[tbl_Komercijalisti](
	[KomercijalistID] [int] NOT NULL,
	[Komercijalist] [nvarchar](50) NULL,
	[ProdaznaDivizija] [nvarchar](20) NULL,
 CONSTRAINT [PK_tbl_Komercijalisti] PRIMARY KEY CLUSTERED
(
	[KomercijalistID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

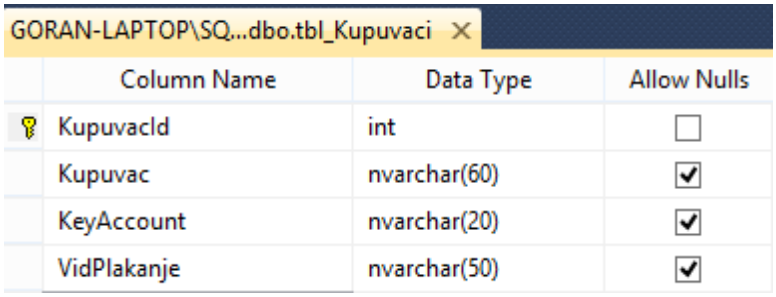
GORAN-LAPTOP\SQL...bl_Komercijalisti

Column Name	Data Type	Allow Nulls
KomercijalistID	int	☐
Komercijalist	nvarchar(50)	☑
ProdaznaDivizija	nvarchar(20)	☑

Слика 17. Табела tbl_Komercijalisti
Figure 17. Table tbl_Komercijalisti

Димензијата купувачи е дефинирана и зачувана во табелата tbl_Kupuvaci. При креирањето на димензијата купувачи, идејата беше да се земе предвид што е тоа што најрепрезентативно го објаснува купувачот. Затоа освен името, при креирањето на димензијата, вклучени се видот на плаќање и категоризацијата според големина, односно селектирани се податоци од три изворни табели. Клучот KupuvacId е врската со tbl_Prodazba.

```
CREATE TABLE [dbo].[tbl_Kupuvaci](
	[KupuvacId] [int] NOT NULL,
	[Kupuvac] [nvarchar](60) NULL,
	[KeyAccount] [nvarchar](20) NULL,
	[VidPlakanje] [nvarchar](50) NULL,
 CONSTRAINT [PK_tbl_Kupuvaci] PRIMARY KEY CLUSTERED
(
	[KupuvacId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```


GORAN-LAPTOP\SQ...dbo.tbl_Kupuvaci

Column Name	Data Type	Allow Nulls
KupuvacId	int	☐
Kupuvac	nvarchar(60)	☑
KeyAccount	nvarchar(20)	☑
VidPlakanje	nvarchar(50)	☑

Слика 18. Табела tbl_Kupuvaci
Figure 18. Table tbl_Kupuvaci

За да можеме да вршиме подетални анализи, како потреба се наметна креирање на посебна димензија продажни места. За разлика од купувачи, која претставува повисока хиерархија, продажното место е поврзано со местото на испорака односно еден купувач може да има повеќе продажни места. Креирањето на оваа димензија како посебна се наметна од потребата за анализа на продажното место врз основа на атрибутите тип, површина и локација. Врската со табелата tbl_Prodazba е преку клучот prodaznoMesto_Id, а во исто време преку оваа димензија вршиме поврзување на географската димензија tbl_NaseleniMesta со основната табела tbl_Prodazba преку полето NaselenoMestoID. Со издвојувањето на оваа димензија како посебна, сакаме да креираме модел во вид на снегулка која претставува посложена форма на мултидимензионален модел на податочен склад.

```
CREATE TABLE [dbo].[tbl_ProdaznoMesto](
	[ProdaznoMesto_Id] [int] NOT NULL,
	[ProdaznoMesto] [nvarchar](120) NULL,
	[Tip] [nvarchar](50) NULL,
	[Povrsina] [nchar](20) NULL,
	[Lokacija] [nvarchar](50) NULL,
	[NaselenoMestoID] [int] NULL,
	[KupuvacID] [int] NULL,
 CONSTRAINT [PK_tbl_ProdaznoMesto] PRIMARY KEY CLUSTERED
(
	[ProdaznoMesto_Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

GORAN-LAPTOP\SQ...I_ProdaznoMesto

Column Name	Data Type	Allow Nulls
ProdaznoMesto_Id	int	☐
ProdaznoMesto	nvarchar(120)	☑
Tip	nvarchar(50)	☑
Povrsina	nchar(20)	☑
Lokacija	nvarchar(50)	☑
NaselenoMestoID	int	☑
KupuvacID	int	☑

Слика 19. Табела tbl_ProdaznoMesto
Figure 19. Table tbl_ProdaznoMesto

Преку tbl_NaseleniMesta овозможуваме анализа на продажбата од географски аспект. Димензијата населени места креира хиерархија која вклучува населено место, општина, подрегион, регион, област и држава. Оваа димензија е поврзана со димензијата продажни места преку клучот NaselenoMestoID и притоа овде немаме директна врска со табелата со мерки tbl_Prodazba. Ваквото поврзување не влијае на перформансите на податочниот склад.

```
CREATE TABLE [dbo].[tbl_NaseleniMesta](
	[NaselenoMestoID] [int] NOT NULL,
	[NaselenoMesto] [nvarchar](50) NULL,
	[Opstina] [nvarchar](50) NULL,
	[Region] [nvarchar](50) NULL,
	[SubRegion] [nvarchar](50) NULL,
	[Area] [nvarchar](50) NULL,
	[Drzava] [nvarchar](50) NULL,
	[Naselenie] [real] NULL,
 CONSTRAINT [PK_tbl_Mesta] PRIMARY KEY CLUSTERED
(
	[NaselenoMestoID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

GORAN-LAPTOP\SQ...bl_NaseleniMesta

Column Name	Data Type	Allow Nulls
NaselenoMestoID	int	☐
NaselenoMesto	nvarchar(50)	☑
Opstina	nvarchar(50)	☑
Region	nvarchar(50)	☑
SubRegion	nvarchar(50)	☑
Area	nvarchar(50)	☑
Drzava	nvarchar(50)	☑
Naselenie	real	☑

Слика 20. Табела tbl_NaseleniMesta
Figure 20. Table tbl_NaseleniMesta

За да се одговори на сите прашања поврзани со продажбата потребно е да се креира и димензијата видови документи. Целата на оваа димензија е да овозможи анализа на продажба од аспект на начинот на плаќање готовинска, фактура, консигнација итн. Клучот VidDokumentId овозможува поврзување со табелата со мерки tbl_Prodazba. Иако оваа димензија можеше да се интегрира во табелата tbl_Prodazba, анализата покажа дека е неопходно креирање на истата врз основа на визијата за развој на податочниот склад и негово користење од страна на финансискиот оддел.

```
CREATE TABLE [dbo].[tbl_VidDokumenti](
	[VidDokumentId] [int] NOT NULL,
	[VidDokument] [nvarchar](50) NULL,
 CONSTRAINT [PK_tbl_VidDokumenti] PRIMARY KEY CLUSTERED
(
	[VidDokumentId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

GORAN-LAPTOP\SQ...tbl_VidDokumenti

Column Name	Data Type	Allow Nulls
VidDokumentId	int	☐
VidDokument	nvarchar(50)	☑

Слика 21. Табела tbl_VidDokumenti
Figure 21. Table tbl_VidDokumenti

Креирањето на димензијата возила не е директно поврзана со примарната намена на податочниот склад. Како и претходната димензија и оваа димензија е поврзана со визијата за развој на податочниот склад, односно овде вршиме подготовка на складот за потребите на логистичкиот оддел. Преку креирање на хиерархија возило и волумен се овозможува анализа на искористувањето на возниот парк и логистичките центри. Овде имаме директно поврзување со табелата tbl_Prodazba преку клучот VoziloId.

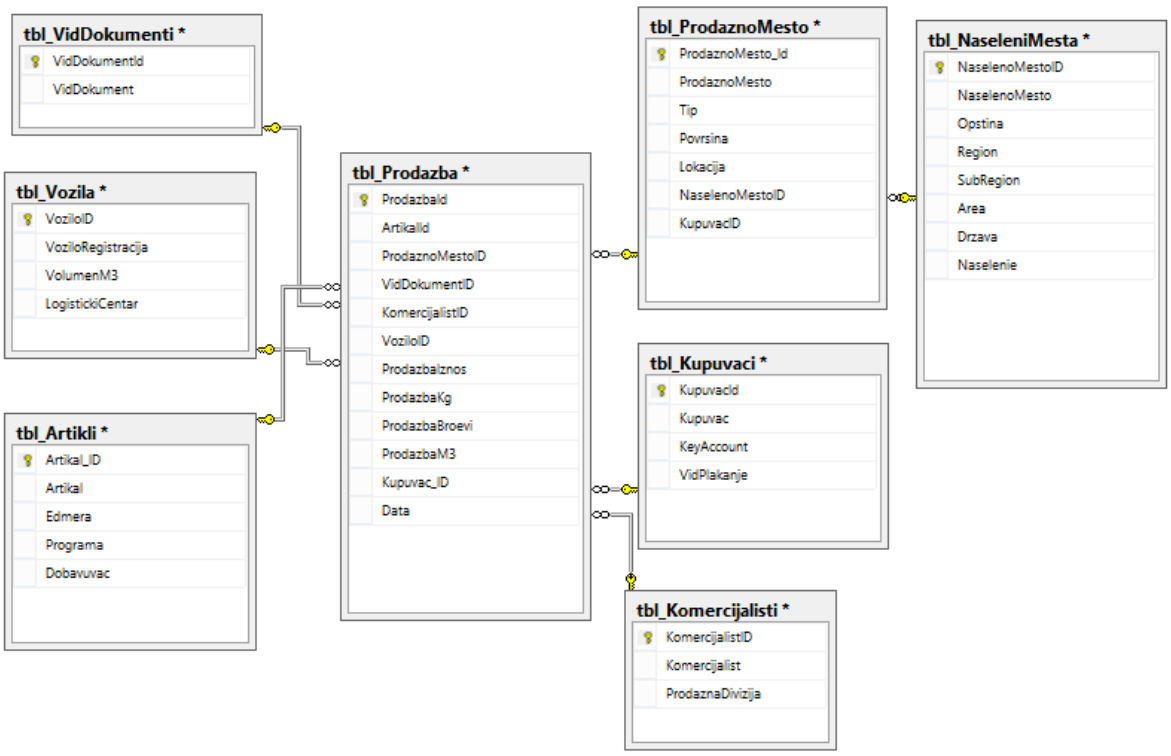
```
CREATE TABLE [dbo].[tbl_Vozila](
	[VoziloID] [int] NOT NULL,
	[VoziloRegistracija] [char](10) NULL,
	[VolumenM3] [real] NULL,
	[LogistickiCentar] [nvarchar](30) NULL,
 CONSTRAINT [PK_tbl_Vozila] PRIMARY KEY CLUSTERED
(
	[VoziloID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

GORAN-LAPTOP\SQL2...- dbo.tbl_Vozila

Column Name	Data Type	Allow Nulls
VoziloID	int	☐
VoziloRegistracija	char(10)	☑
VolumenM3	real	☑
LogistickiCentar	nvarchar(30)	☑

Слика 22. Табела tbl_Vozila
Figure 22. Table tbl_Vozila

За да имаме целосна анализа, неопходно е креирање на димензија време. Димензијата време е креирана во понатамошниот процес, односно при процесот на креирањето на податочната коцка. Хиерархијата е детална и вклучува дата, ден од недела, недела, ден од месец, месец, месец од година, месец од половина година, месец од квартал, квартал итн. Со тоа се заокружува процесот на креирање на податочен склад како што е прикажано на Слика 23.



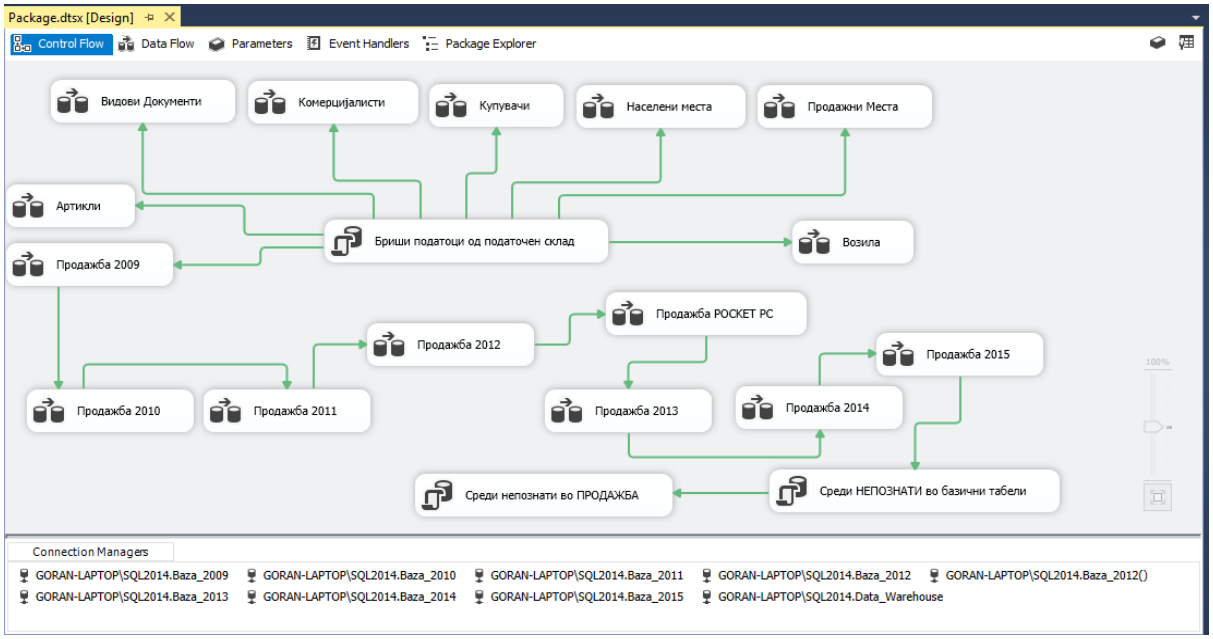
Слика 23. База на податоци податочен склад (Data_Warehouse)
Figure 23. Database data warehouse (Data_Warehouse)

6.2. ЕКСТАРКЦИЈА, ТРАНСФОРМАЦИЈА И ПОЛНЕЊЕ НА ПОДАТОЧНИОТ СКЛАД

Откако ќе се утврдат изворите на податоци, дефинираат и креираат податочните табели, потребно е креирање на план за полнење на податочниот склад. Процесот е комплексен и се состои од повеќе поврзани последователни чекори. Како што може да се види од Слика 24, во нашиот случај чекорите на екстракција, трансформација и полнење се поделени во повеќе подчекори, односно паралелно се извршуваат за секоја табела поодделно.

Сега полнењето на податочниот склад е во тест фаза и притоа имаме процеси кои нема да се извршуваат во фазата на користење. Процесите како бришење на податоци, продажба 2009, 2010, 2011, 2012, 2013, 2014 се исфрлат во фазата на користење, но и покрај тоа целокупниот процес трае 3 минути. Во фаза на користење, ќе имаме само надополнување на складот со најновите

податоци на дневно ниво, со тоа што времето за полнење ќе се намали на 5 секунди.



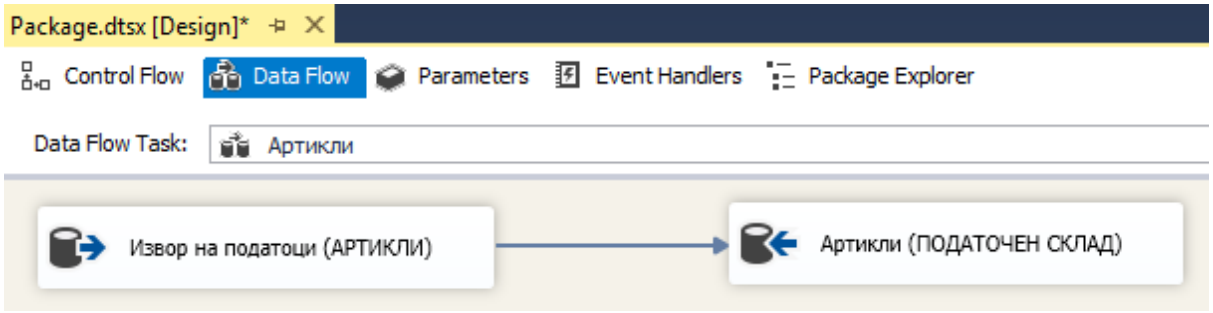
Слика 24. Процес на екстракција, трансформација и полнење
Figure 24. Extraction, transformation and loading process

Прв чекор е бришењето на податоците од податочниот склад. Овој чекор не е вообичаен во пракса, но бидејќи складот е во тест фаза и трпи промени во структурата на табелите, бришење на сите податоци е неопходно. За таа цел се извршуваат следниве SQL команди.

```
TRUNCATE TABLE TBL_ARTIKLI
TRUNCATE TABLE TBL_KOMERCIJALISTI
TRUNCATE TABLE TBL_KUPUVACI
TRUNCATE TABLE TBL_NASELENIMESTA
TRUNCATE TABLE TBL_PRODAZBA
TRUNCATE TABLE TBL_PRODAZNOMESTO
TRUNCATE TABLE TBL_VIDDOKUMENTI
TRUNCATE TABLE TBL_VOZILA
```

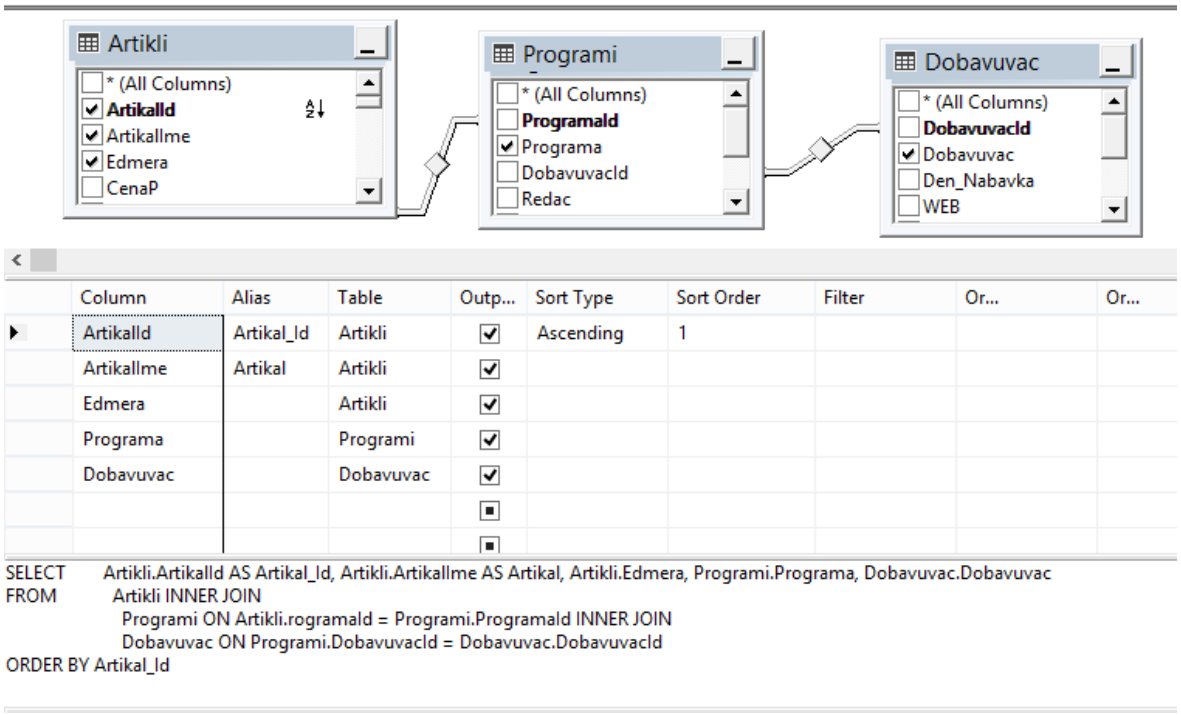
Архитектурата на податочниот склад наметува поделба на табелите во две логички целини и тоа мерки и димензии. Димензиите како посебен дел од податочниот склад бараат посебен пристап при полнењето. Врз основа на направената анализа констатиран е континуитет на податоците кои ги претставуваат димензиите и тоа од 2009 година до денес. Како резултат на тоа како извор на податоци за полнење на табелите со димензии се земат податоците од последната година, односно е земена Baza_2015.

Како што може да се види од Слика 25, процесот на полнење на табелата за димензијата артикли се состои од две фази.



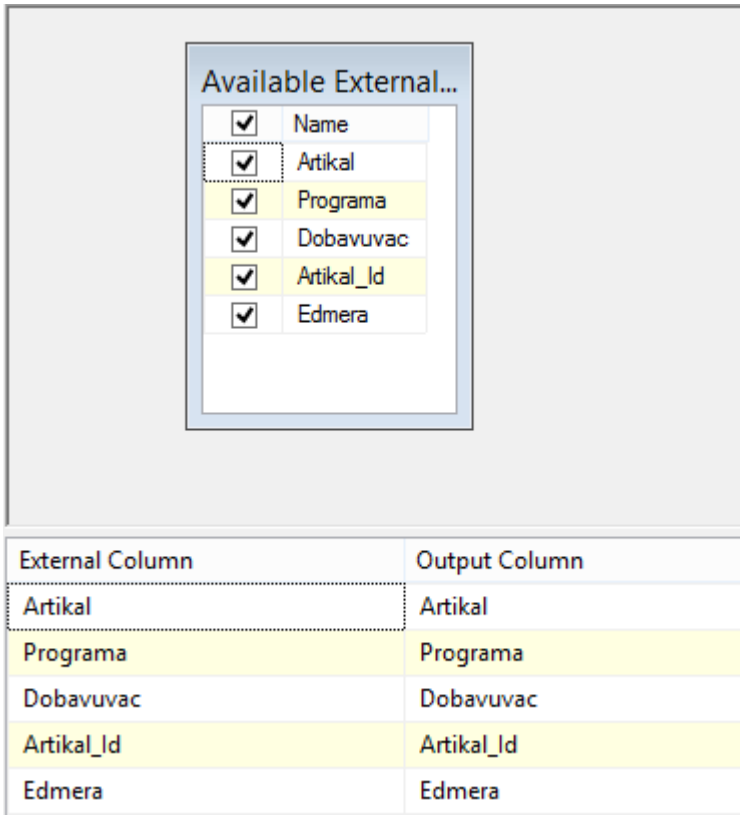
Слика 25. Тек на податоци за табела tbl_Artikli
Figure 25. Data flow table tbl_Artikli

Првата фаза или извор на податоци, е процес на комбинација на податоци од три табели како што е прикажано на Слика 26.



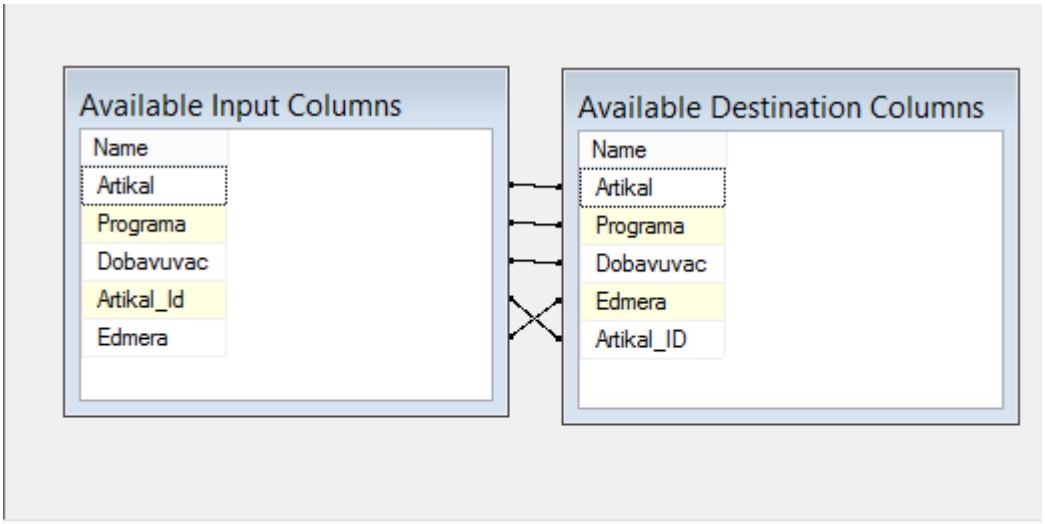
Слика 26. SQL извор на податоци tbl_Artikli
Figure 26. SQL Data source tbl_Artikli

Како што може да се види од SQL кодот, овде извор на податоци се табелите Артикли, Програма и Добавувач. Со користење на SQL вршиме ограничување со надворешен клуч, односно вредностите во колоната мораат да имаат врска со уникатните вредности од друга табела. Исто така имаме и единствени ограничувања или во нашиот случај ArtikalId има единствена вредност. Податоци кои се селектираат се ArtikalId, Artikal, EdMera, Programa и Dobavuvac и со тоа е дефиниран изворот на податоци и податоците кои ќе се екстахираат како што е прикажано на Слика 27.



Слика 27. Селектиран извор на податоци tbl_Artikli
Figure 27. Selected data source tbl_Artikli

Откако е дефиниран процесот на екстракција и трансформација, следен чекор е дефинирање на процесот на полнење, односно дефинирање на табелата и полињата и нивната врска со изворот на податоци. Овде потребно е да имаме компатибилност на типовите на податоци кај изворот и целата со цел да се избегнат грешки и стопирање на целиот процес. Во нашиот случај, како што е прикажано на Слика 28, воспоставена е врска помеѓу податочните полиња и имаме компатибилност со што е избегнато настанување на грешки во процесот на екстракција, трансформација и полнење на димензијата артикли.



Input Column	Destination Column
Artikal	Artikal
Programa	Programa
Dobavuvac	Dobavuvac
Edmera	Edmera
Artikal_Id	Artikal_ID

Слика 28. Релации за полнење на табела tbl_Artikli
Figure 28. Relations for loading tbl_Artikli

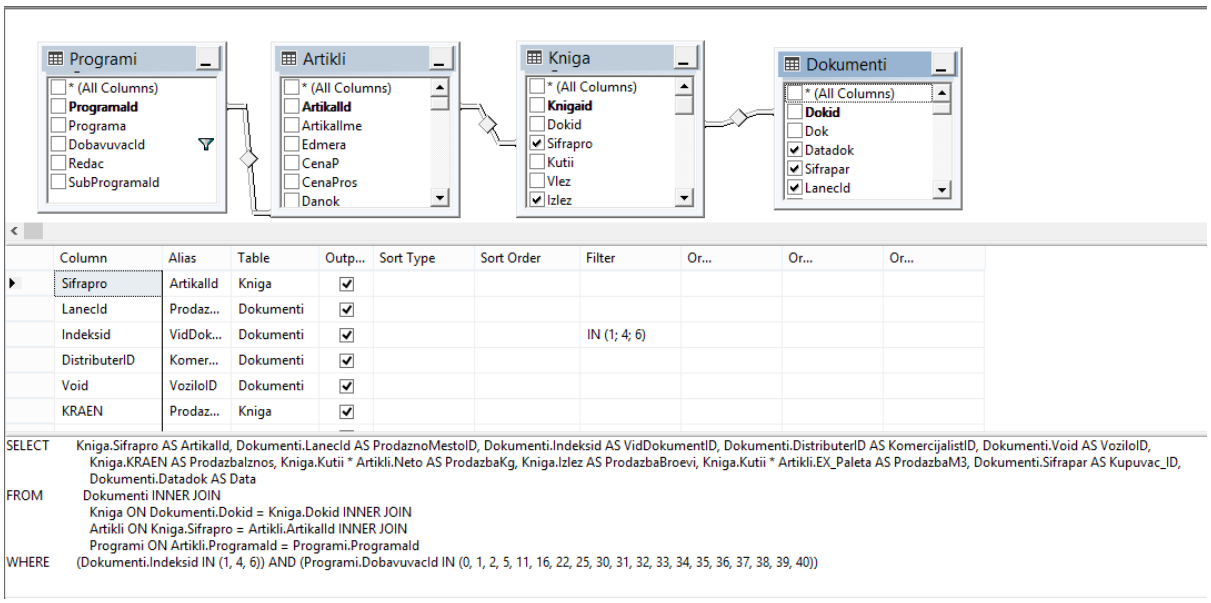
Процесот на екстракција, трансформација и полнење на табелите со димензии е идентичен на процесот што е прикажан со димензијата артикли.

Процесот на полнење со податоци на табелата tbl_Prodazba се состои од подпроцеси прикажани на Слика 29 кои се извршуваат последователно.

Слика 29. Тек на податоци за останати табели
Figure 29. Data flow other tables

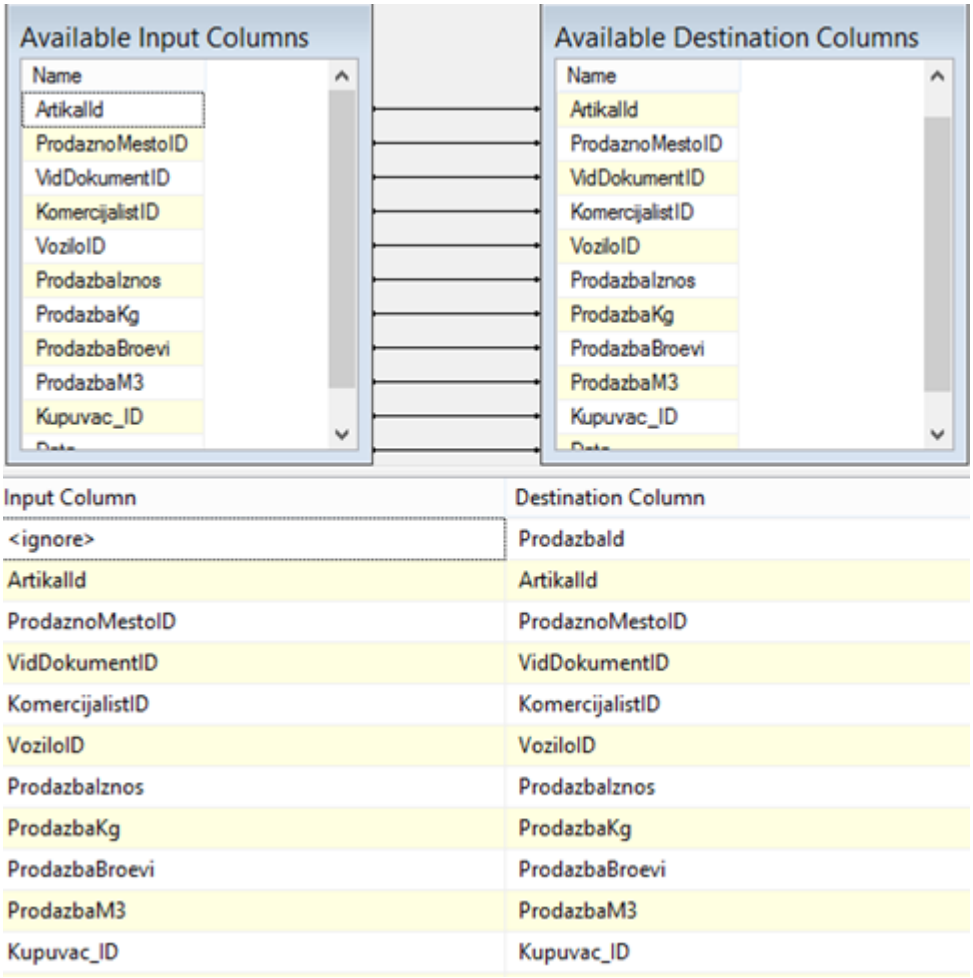
Кај секој од овие подпроцеси, дефинираме извор на податоци за секоја година посебно како што е прикажано на Слика 29. При креирањето на изворот на податоци, се креираат и нови атрибути како ProdazbaKg и ProdazbaM3 со цел да помогнат во понатамошниот процес на податочно рударење. Со комбинирање на овие атрибути се овозможува пронаоѓање на скриени информации за врските на податочните атрибути во понатамошниот процес. Во процесот на екстракција користиме ограничување со надворешен клуч, ограничување на податочен тип, како и филтри за редукција на податоците. Со користењето на филтрите селектираме податоци кои се поврзани со продажба

што е и примарна цел на овој податочен склад, односно се отстрануваат податоци кои се поврзани со набавка, поврат на роба итн. Со користењето на овие техники за чистење, редукција и трансформација на податоци се овозможува ефикасно и точно податочно рударење.



Слика 30. SQL извор на податоци tbl_Prodazba
Figure 30. SQL data source tbl_Prodazba

Процесот понатаму продолжува со дефинирање, односно поврзување на податочните полиња од изворот со целата и проверка на нивната компатибилност. Овој процес е прикажан на Слика 31. Како што може да се види од сликата, воспоставена е врска помеѓу сите полиња со исклучок на ProdaznaId. Оваа податочно поле се наоѓа само во податочниот склад и е креирано со цел да овозможи уникатност, односно единственост на секој податочен ред во табелата tbl_Prodazba.



Слика 31. Релации за полнење на табела tbl_Prodazba
Figure 31. Relations for loading tbl_Prodazba

Откако заврши процесот на полнење на податочниот склад, се врши проверка на податоците и ако е потребно се применува чистење. Во нашиот случај имаме потреба од чистење на податоци и затоа во процесот имаме имплементирани два чекори кои извршуваат повеќе SQL команди.

Првиот чекор или како што е означен во дијаграмот Среди НЕПОЗНАТИ во базични табели ги извршува следниве SQL команди:

1. Бришење на податочен ред, под услов да недостасува податок за купувач.

```
DELETE FROM tbl_Prodazba
WHERE       (Kupuvac_ID IS NULL)
```

2. Ажурирање на продажно место со вредност Null под услов ако не постои вредност за даденото продажно место во табелата tbl_ProdazniMesta. Ова всушност е подготовка на полето ProdaznoMestoId за ажурирање кое ќе го извршиме во наредната фаза.

```
UPDATE      tbl_Prodazba
SET          ProdaznoMestoID = NULL
FROM         tbl_Prodazba LEFT OUTER JOIN
                tbl_ProdaznoMesto ON tbl_Prodazba.ProdaznoMestoID =
tbl_ProdaznoMesto.ProdaznoMesto_Id
WHERE       (tbl_ProdaznoMesto.ProdaznoMesto_Id IS NULL)
```

3. Ажурирање на полето ProdazbaM3 со вредност 0, односно отстранување на Null вредности од табела tbl_Prodazba за податочно поле ProdazbaM3. Кај одредени артикли немаме зададено вредност за волумен, а значењето на овој атрибут во понатамошната анализа за овие категории на производи е незначителна.

```
UPDATE      tbl_Prodazba
SET          ProdazbaM3 = 0
WHERE       (ProdazbaM3 IS NULL)
```

Вториот чекор е означен како Среди непознати во продажба и ги извршува следниве SQL команди:

1. Бришење на сите редови во табела tbl_Prodazba каде вредноста на полето ArtikalID нема релација со табела tbl_Artikli. Оваа команда проверува дали има продажба која нема релација со димензијата артикли и врз основа на тоа го брише редот.

```
DELETE FROM tbl_Prodazba
FROM         tbl_Prodazba LEFT OUTER JOIN
                tbl_Artikli ON tbl_Prodazba.ArtikalId = tbl_Artikli.Artikal_ID
```

```
WHERE       (tbl_Artikli.Artikal_ID IS NULL)
```

2. Ажурирање на податочно поле KomercijalistID од табела tbl_Prodazba во вредност 6 под услов KomercijalistID да е Null. Претпоставката е, секаде каде ова поле нема вредност продажбата да е извршена од главниот магацин кој, пак, има единствена вредност 6.

```
UPDATE      tbl_Prodazba
SET          KomercijalistID = 6
FROM         tbl_Komercijalisti RIGHT OUTER JOIN
                tbl_Prodazba ON tbl_Komercijalisti.KomercijalistID =
tbl_Prodazba.KomercijalistID
WHERE       (tbl_Komercijalisti.KomercijalistID IS NULL)
```

3. Продолжување на започнатото во претходниот чекор, чекорот во кој податочното поле ProdaznoMesto_Id беше ажурирано во вредност Null. Креирање релација помеѓу табелите tbl_Prodazba, tbl_ProdaznoMesto и tbl_Kupuvaci врз основа на клуч Kupuvac_ID, селектирање на првата вредност на ProdaznoMesto_Id од tbl_ProdaznoMesto и ажурирање на податочно поле ProdaznoMesto_Id во табела tbl_Prodazba, каде вредноста на ProdaznoMesto_Id е 0 или Null. Ажурирањето на оваа податочно поле овозможува релација со димензиите продажни места и населени места.

```
UPDATE      tbl_Prodazba
SET          ProdaznoMestoID = tbl_ProdaznoMesto.ProdaznoMesto_Id
FROM         tbl_Prodazba INNER JOIN
                tbl_Kupuvaci ON tbl_Prodazba.Kupuvac_ID =
tbl_Kupuvaci.KupuvacId INNER JOIN
                tbl_ProdaznoMesto ON tbl_Kupuvaci.KupuvacId =
tbl_ProdaznoMesto.KupuvacID
WHERE       (tbl_Prodazba.ProdaznoMestoID IS NULL) OR
                (tbl_Prodazba.ProdaznoMestoID = 0)
```

4. Ажурирање на податочно поле VoziloID во табела tbl_Prodazba каде вредноста на VoziloID е Null. Анализата покажа дека имаме вредност Null само при испорака од централен магацин и затоа ажурираме со вредност 13.

```
UPDATE      tbl_Prodazba
SET          VoziloID =13
FROM        tbl_Vozila RIGHT OUTER JOIN
              tbl_Prodazba ON tbl_Vozila.VoziloID = tbl_Prodazba.VoziloID
WHERE      (tbl_Vozila.VoziloID IS NULL)
```

5. Откако е ажурирано полето ProdaznoMestoID следен чекор е бришење на редовите од табела tbl_Prodazba каде немаме вредност за ProdaznoMestoID.

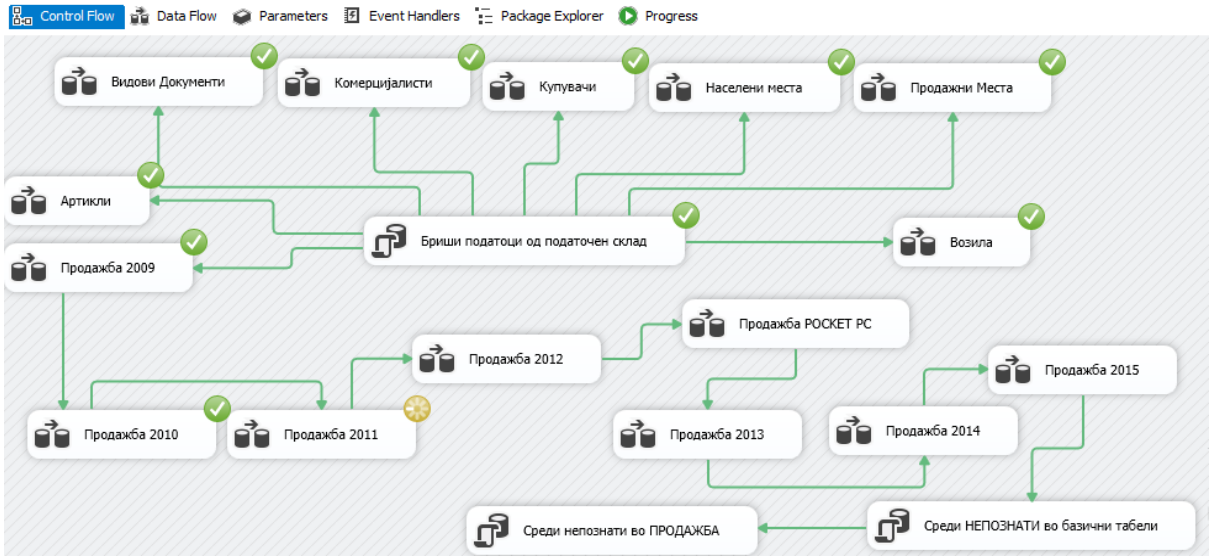
```
DELETE FROM tbl_Prodazba
WHERE      (ProdaznoMestoID IS NULL)
```

6. Бришење на податочни редови од табела tbl_Prodazba, каде податочното поле Kupuvac_ID нема релација со табелата tbl_Kupuvaci.

```
DELETE FROM tbl_Prodazba
FROM        tbl_Prodazba LEFT OUTER JOIN
              tbl_Kupuvaci ON tbl_Prodazba.Kupuvac_ID =
tbl_Kupuvaci.KupuvacId
WHERE      (tbl_Kupuvaci.KupuvacId IS NULL)
```

6.2.1. ПЕРФОРМАНСИ НА ПРОЦЕСОТ НА ЕКСТРАКЦИЈА, ТРАНСФОРМАЦИЈА И ПОЛНЕЊЕ НА ПОДАТОЧНИОТ СКЛАД

Следен чекор во процесот на имплементација на податочниот склад е проверка на неговата функционалност и перформанси. За почеток целокупниот процес се тестираше само на еден сервер, при што овде се отстранија сите грешки во однос на компатибилноста на податоците и SQL командите. Откако беа отстранети сите грешки и успешно се изврши процесот (Слика 32) се премина кон тестирање на перформансите на три различни сервери.



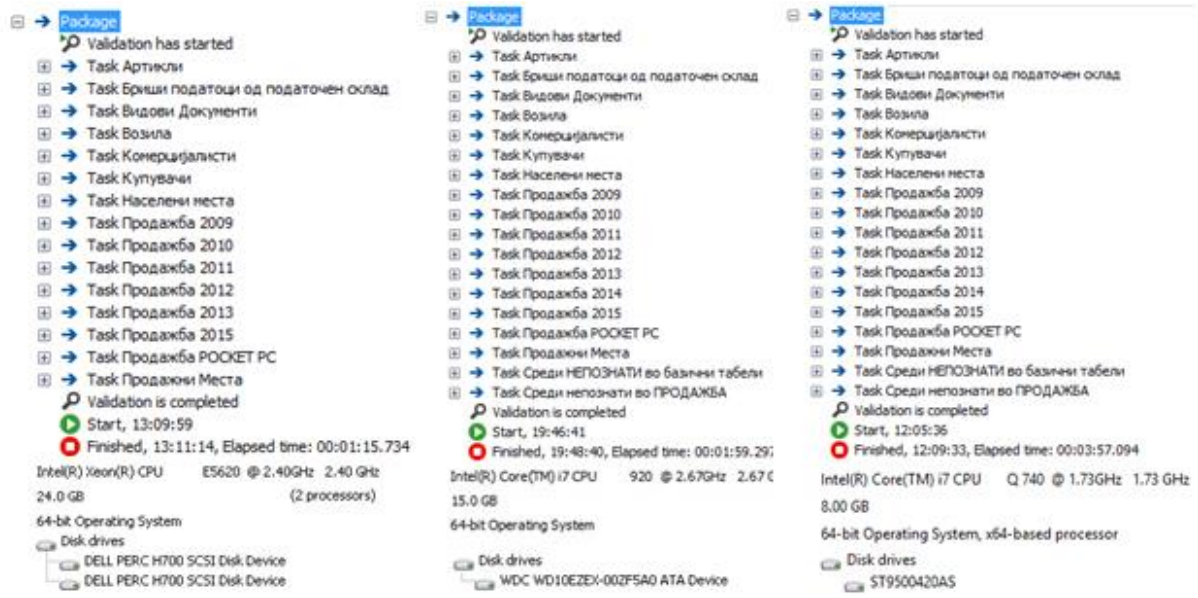
Слика 32. Извршување на ETL процес
Figure 32. Execution ETL process

За да можеме да извлечеме точен заклучок во однос на тоа каква серверска машина ни е потребна, извршивме тестирања на сервери кои се разликуваа во однос на процесор, меморија и тврди дискови како што е наведено во Табела 6.

Табела 6. Спецификација на сервери
Table 6. Servers specification

Рбр	Процесор	Меморија (RAM)	Хард дискови
1	Intel Xeon E5620 (2 Processors) 2.4GHz 16 cores	24GB	3xSCSI 15k / 2xSATA 7200rpm RAID1/RAID5 6Gb/s
2	Intel I7 920 2.67GHz 8cores	16GB	1x SATA 7200rpm 64MB Cache 1TB 6Gb/s
3	Intel I7 Q740 1.73GHz 8cores	8GB	1x SATA 7200rpm 16MB Cache 500GB 3Gb/s

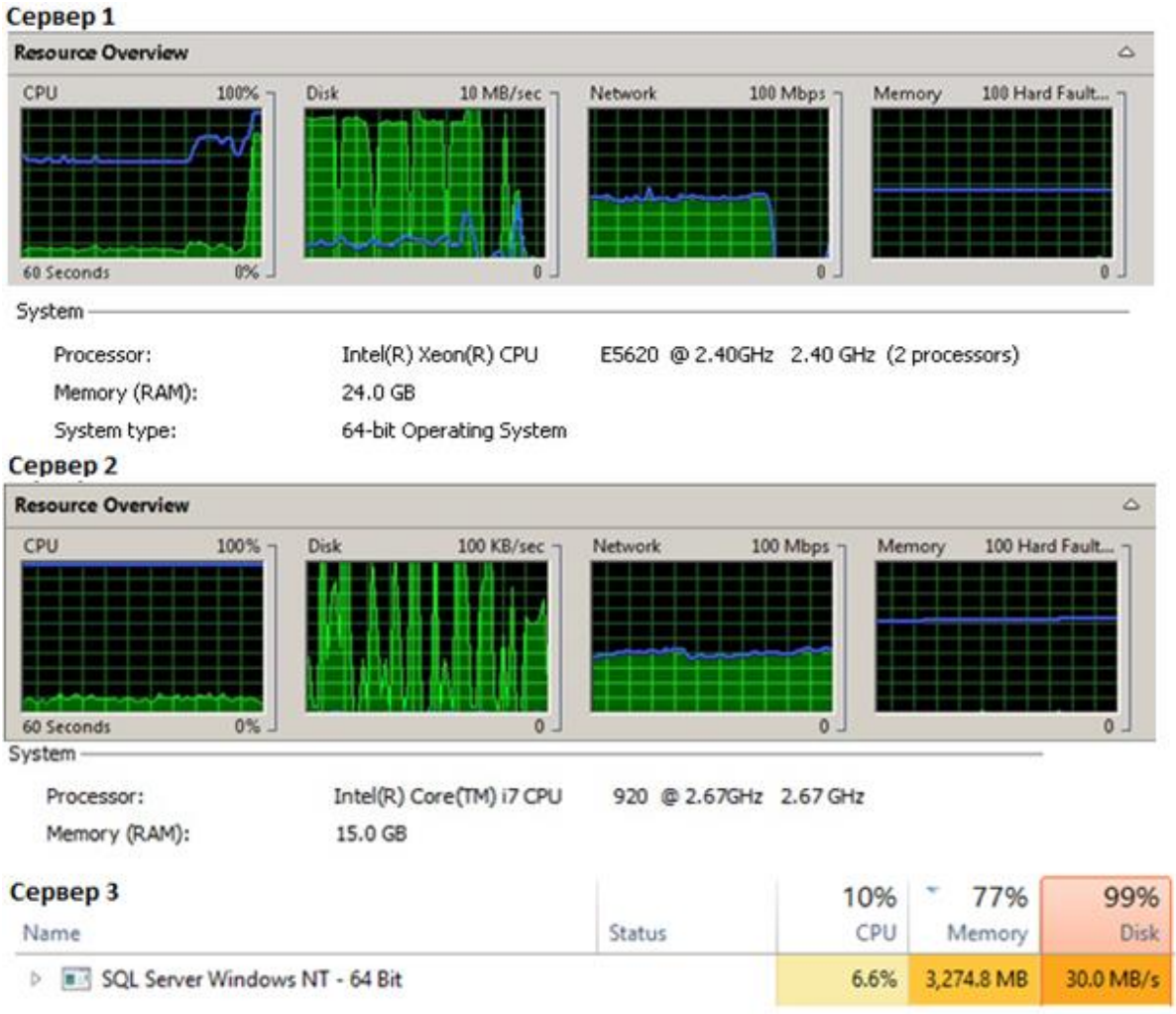
При тестирањето се мереше времето потребно за извршување на процесот на екстракција, трансформација и полнење, како што е прикажано на Слика 33.



Слика 33. Време за извршување на ETL процес
Figure 33. Time to perform ETL process

Како што може да се види од сликата, најдобри перформанси покажа серверот реден број 1, што е и нормално со оглед на конфигурацијата. Од друга страна, перформансите на серверот број 3 се значително послаби во однос на претходните што не наведе на подетална анализа на ваквата ситуација. Сега при извишувањето на процесот се мереа посебно перформансите на процесорот, меморијата и тврдите дискови. Како што може да се види од слика

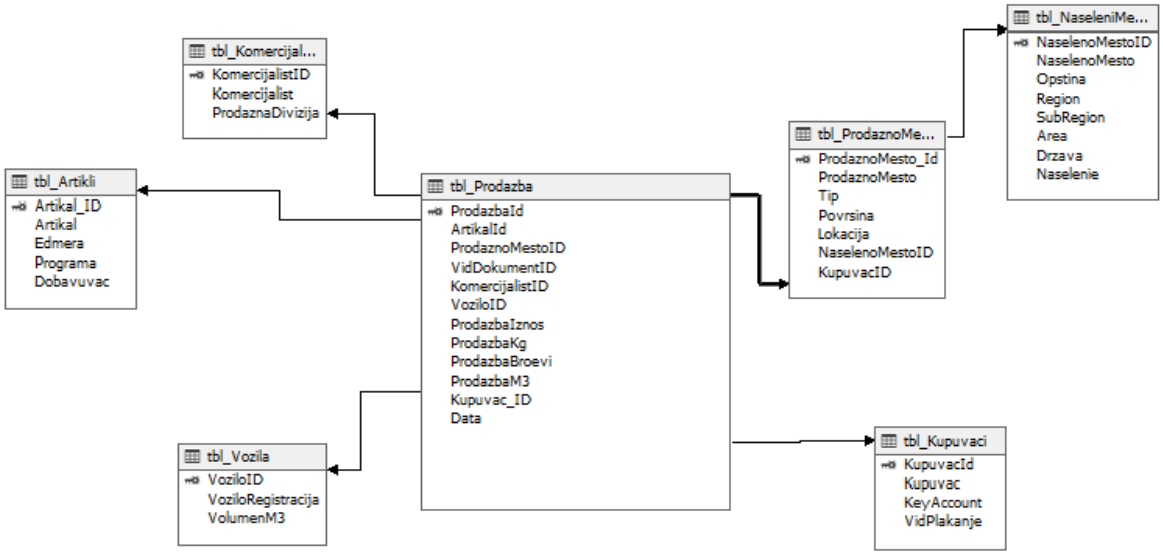
6.20 користењето на процесорот и работната меморија е минимално и врз основа на тоа може да се заклучи дека при избор на сервер за оваа фаза немаме потреба од екстремно моќни процесори и многу работна меморија. За разлика од тоа користењето на тврдите дискови е максимално во текот на целиот процес. Ако се погледне табела 6, во колоната во која се опишани тврдите дискови, уште на прв поглед се согледува дека првиот сервер користи SCSI дискови што се одразува и времето на извршувањето на процесот. Серверот број 2, за разлика од серверот број 3, има понапреден тврд диск што се одразува и врз времето на извршување на процесот. Врз основа на мерењето на перформансите може да заклучиме дека во оваа фаза тврдите дискови се од примарно значење.



Слика 34. Користени ресурси при извршување на ETL процесот
Figure 34. Resources used in the execution of ETL process

6.3. КРЕИРАЊЕ НА ПОДАТОЧНА КОЦКА

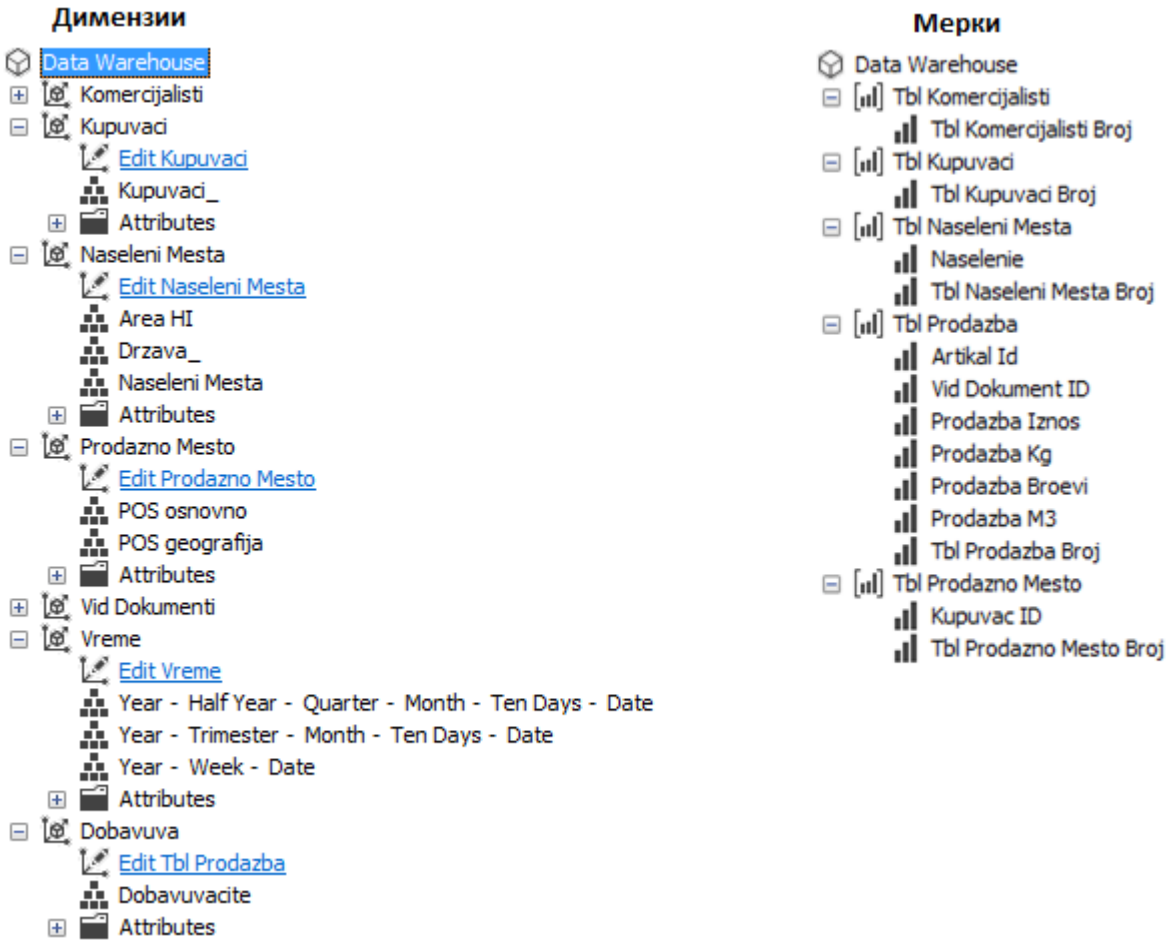
Формираниот податочен склад е организиран да одговори на потребите за креирање на мултидимензионален податочен модел поврзан со анализа на продажбата. Мултидимензионалниот модел е основа за on-line аналитичко процесирање (OLAP). Податочните коцки се главни објекти каде е организирана и сумирана мултидимензионалната структура дефинирана од множества од димензии и мерки. Уште при градењето на податочниот склад планираме нумеричките вредности односно мерките да бидат зачувани во табелата tbl_Prodazba, а во останатите табели да се наоѓаат податоците за димензиите, како што е прикажано на Слика 35.



Слика 35. Извор на податоци за податочна коцка
Figure 35. Data source for data cube

За да можеме да ги користиме предностите на податочните коцки креирани се мерки и димензии кои се прикажани на Слика 36. Од табелите во кои се зачувани димензиите, за креирање мерки се користи агрегатна функција count(). Целта на креирањето на овие мерки е да овозможиме пребројување на матичните податоци на секоја димензија и споредба со активните податоци во табелата продажба. Потоа, од табелата tbl_Prodazba креирани се главните мерки кои се користат при понатамошните анализи и тоа: износ на продажба,

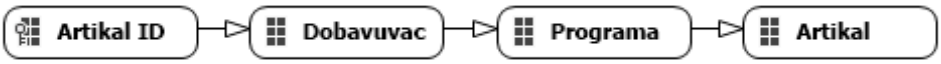
продажба во килограми, продажба во метри кубни, број на документи, број на артикли. Овде ги користиме агрегатни функции sum() и count(). Според нашата анализа со креирањето на овие мерки се овозможува исполнување на примарната функција на системот, а тоа е анализа на продажба.



Слика 36. Мерки и димензии во податочна коцка
Figure 36. Measures and dimensions in data cube

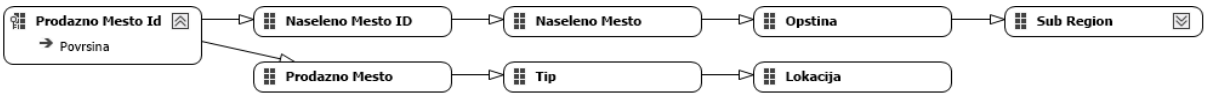
Главна цел, при креирањето на димензиите, беше хиерархиски концепти да се во согласност со реалните потреби на бизнисот. Согласно тоа, формирани се следниве димензии и хиерархиски концепти:

- Димензија артикли, димензија со атрибути добавувач, програма и артикл.



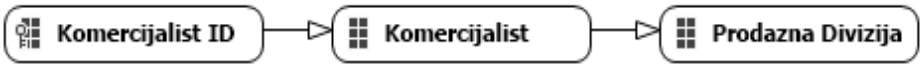
Слика 37. Димензија артикли
Figure 37. Dimension products

- Димензија продажни места со атрибути населено место, општина, субрегион, тип на објект и локација.



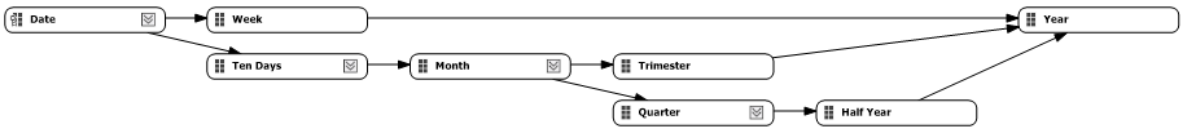
Слика 38 Димензија продажно место
Figure 38 Dimension point of sale

- Димензија комерцијалисти со атрибути комерцијалист и продажна дивизија.



Слика 39. Димензија комерцијалисти
Figure 39. Dimension sales men

- Димензија време со атрибути дата, недела, декада, месец, квартал, половина година и година.



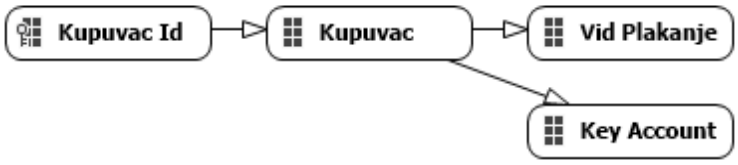
Слика 40. Димензија време
Figure 40. Time dimension

- Димензија населени места со атрибути населено место, општина, подрегион, регион, област и држава.

Слика 41. Димензија населено места

Figure 41. Dimension places

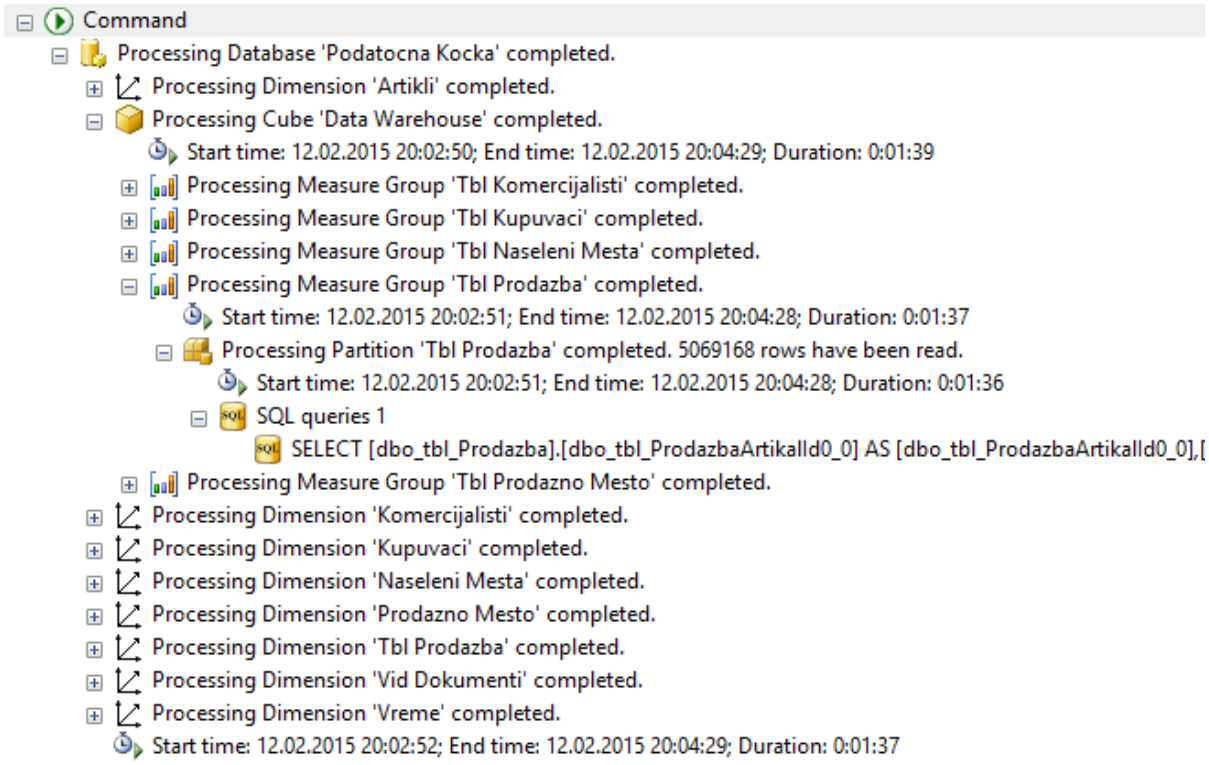
- Димензија купувачи со атрибути купувач, вид на плаќање и класификација на купувач.



Слика 42. Димензија купувачи
Figure 42. Customers dimension

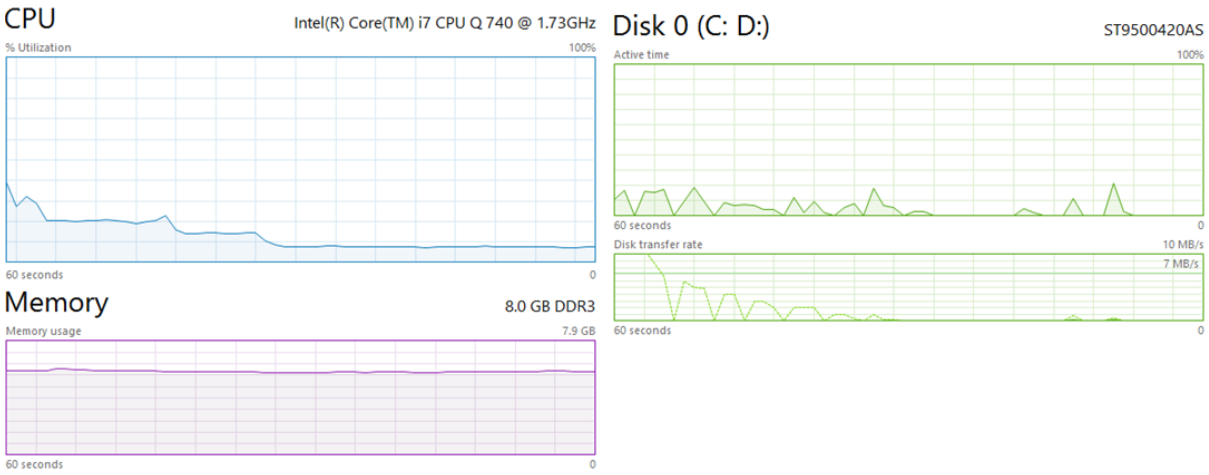
6.3.1. ПЕРФОРМАНСИ НА ПОДАТОЧНА КОЦКА

Откако е креирана податочна коцка, се пристапува кон ажурирање со цел да се проверат перформансите и откријат потенцијалните грешки. Отстранувањето на грешки е неопходно од аспект на непречено ажурирање и оневозможување на потенцијално стопирање на процесот на ажурирање. По отстранувањето на грешките се преминува кон тестирање на перформансите како што е прикажано на Слика 43.



Слика 43. Тестирање перформанси на процесирање на податочна коцка
Figure 43. Testing performance of data cube processing

Врз основа на искуството од претходното тестирање, сега се изврши тестирање на серверот со најслаби перформанси. Како што е прикажано на Слика 43, процесот на ажурирање се изврши за една минута и триесет и седум секунди, а бидејќи процесот е планиран да се извршува на секој четири часа времето на ажурирање е прифатливо. Сепак, како и во претходниот случај, перформансите се анализираат врз основа користење на процесор, работна меморија (RAM) и тврд диск. Како што може да се види од Слика 44 користењето на процесорот и тврдиот диск е минимално. Единство што може да се заклучи е дека работната меморија треба да биде повеќе од 8GB, односно препорачливо е минимум 16GB.



Слика 44. Користени ресурси при процесирање на податочна коцка
Figure 44. Used resources when processing data cube

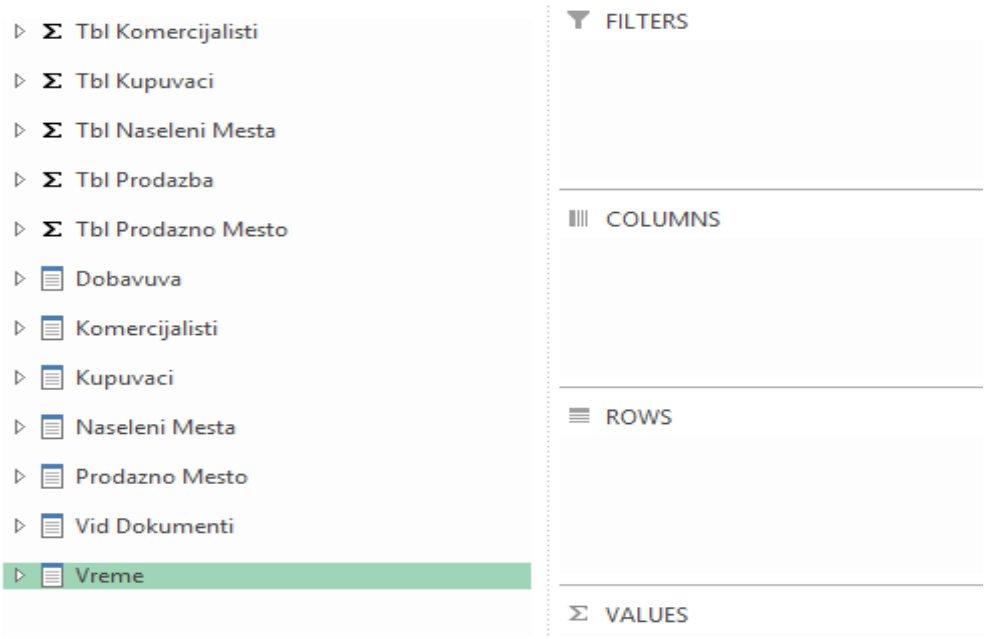
6.3.2. КОРИСТЕЊЕ НА ПОДАТОЧНАТА КОЦКА

Откако е креирана и ажурирана податочната коцка потребно е да се направи кориснички интерфејс кој ќе биде лесно разбирлив за крајникот корисник. За да можеме да ги видиме можностите на OLAP и хиерархиските концепти, креиран е проект во Microsoft Excel за да се овозможи интерактивно пребарување и анализа на податоци. Бидејќи целата на ова истражување не е да ги анализираме податоците туку да ги покажеме можностите на мултидимензионалните модели на податоци ќе преминеме на објаснување на операциите кои кое користат при OLAP на мултидимензионалните податочни модели.

Основно е да направиме поврзување на корисничката програма со податочната коцка. Самото поврзување овозможува изложување на креираните мерки и димензии и нивно искористување од страна на корисникот.

Како што е прикажано на Слика 45, од десна страна се изложуваат мерките и димензиите, додека од десна страна е овозможено корисникот сам да формира филтри, редови, колони и вредности. Овозможено е креирање на n димензии и извршување на операции како pivot (ротација), slice и dice (парчиња

и сечење), roll-up (виткање) и drill-down (дупчење надолу), операции кои ќе бидат објаснети понатаму.



Слика 45. Кориснички интерфејс за мерки и димензии
Figure 45. User interface of measures and dimensions

Како прво ќе креираме податочна коцка со димензиите населени места, типови објекти и време, за мерката износ на продажба како што е прикажано на Слика 46.

Prodazba Iznos	Column Labels						
	Center	East	North		North Total	West	Grand Total
Row Labels			Скопје	Тетово - Гостивар			
HORECA	57,514,912	43,135,668	212,484,112	14,082,765	226,567,136	27,841,248	355,058,656
LKA	430,693,408	313,587,744	442,744,832	155,397,712	598,143,808	446,772,448	1,789,191,040
Calendar 2009	30,153,206	19,931,538	26,198,330	14,750,455	40,948,768	24,521,542	115,555,120
Calendar 2010	36,109,808	31,805,894	33,140,748	17,633,312	50,774,080	37,947,068	156,636,992
January 2010	1,264,228	1,001,634	1,262,155	835,973	2,098,128	1,669,215	6,033,207
February 2010	1,847,571	2,445,486	2,200,862	923,114	3,123,974	1,670,886	9,087,916
March 2010	2,976,924	2,614,149	2,754,217	1,629,400	4,383,616	3,581,200	13,555,888
April 2010	3,191,348	3,170,810	2,514,186	1,329,754	3,843,939	2,892,332	13,098,428
May 2010	3,106,083	2,867,678	2,681,008	1,135,268	3,816,275	3,115,647	12,905,696
June 2010	2,693,141	2,368,673	3,056,357	1,169,177	4,225,534	2,387,863	11,675,217
July 2010	3,114,083	2,563,274	2,390,308	2,113,195	4,503,502	3,585,624	13,766,480
August 2010	3,077,352	2,914,830	3,050,777	2,182,955	5,233,730	3,895,176	15,121,096
September 2010	3,340,285	2,364,481	3,458,457	1,275,941	4,734,395	3,020,031	13,459,190
October 2010	3,425,613	2,717,932	2,769,089	1,411,146	4,180,233	3,505,542	13,829,325
November 2010	3,972,166	3,050,205	3,126,995	1,737,243	4,864,238	3,919,082	15,805,689
December 2010	4,101,019	3,726,741	3,876,339	1,890,144	5,766,484	4,704,468	18,298,692
Calendar 2011	77,279,432	56,677,896	83,081,216	30,220,706	113,301,944	84,246,680	331,506,528
Calendar 2012	65,897,928	49,976,944	69,711,432	22,185,022	91,896,432	79,290,560	287,061,760
Calendar 2013	119,791,760	84,914,320	123,913,616	37,008,008	160,921,632	122,117,912	487,745,728
Calendar 2014	90,711,696	64,469,148	94,766,016	28,788,428	123,554,368	89,447,296	368,181,856
Calendar 2015	10,748,807	5,811,586	11,933,514	4,811,771	16,745,284	9,201,656	42,507,328
NKA	414,005,280	306,997,696	1,649,048,832	111,969,112	1,761,016,704	232,522,896	2,714,540,032
OTHER	2,339,308	2,817,047	26,549,530	9,783,990	36,333,520	1,130,329	42,620,204
TT	806,889,600	439,615,008	768,497,856	277,519,008	1,046,012,800	675,669,888	2,968,226,304
WHOLESALER	337,224,864	156,192,784	258,426,272	119,560,456	377,987,168	194,824,464	1,066,228,096
Grand Total	2,048,667,264	1,262,345,856	3,357,751,296	688,313,024	4,046,061,312	1,578,761,344	8,935,864,320

Слика 46. Податочна коцка
Figure 46. Data cube

На вака креираната податочна коцка прва операција што ќе ја примениме е операција pivot, односно промена на местата на димензиите населени места, типови објекти и време прикажано на Слика 47. Со промена на редовите во колони и обратно можеме да ги читаме податоците од една друга перспектива при што агрегатната функција sum() пресметува иста вкупна вредност.

Prodazba Iznos	Column Labels						
	HORECA	LKA	NKA	OTHER	TT	WHOLESALER	Grand Total
Row Labels							
Center	**57,514,912**	**430,693,408**	**414,005,280**	**2,339,308**	**806,889,600**	**337,224,864**	**2,048,667,264**
Велес	3,269,847	57,132,756	95,296,576	9,913	41,123,496	62,901,088	259,733,664
Гевгелија -Валандово	15,135,450	98,030,576	33,225,914	191,311	184,806,512	6,371,271	337,761,056
Кавадарци - Неготино	7,766,071	129,089,600	59,310,192	329,157	103,270,400	37,493,832	337,259,264
Прилеп	3,053,909	41,162,804	55,346,028	912,474	157,652,832	97,746,216	355,874,272
Радовиш	2,181,294	6,306,377	30,648,250	319,217	49,828,368	13,746,606	103,030,112
Струмица	26,108,330	98,970,824	140,178,752	577,236	270,207,456	118,965,648	655,008,320
East	**43,135,668**	**313,587,744**	**306,997,696**	**2,817,047**	**439,615,008**	**156,192,784**	**1,262,345,856**
Кочани - Исток	8,748,143	50,865,564	90,036,816	1,521,518	141,177,424	38,992,040	331,341,504
Куманово	9,583,461	121,010,992	128,500,944	876,095	179,474,368	77,407,928	516,853,792
Штип - Св.Николе	24,804,032	141,710,896	88,459,704	419,434	118,962,504	39,792,696	414,149,248
North	**226,567,136**	**598,143,808**	**1,761,016,704**	**36,333,520**	**1,046,012,800**	**377,987,168**	**4,046,061,312**
Скопје	212,484,112	442,744,832	1,649,048,832	26,549,530	768,497,856	258,426,272	3,357,751,296
Тетово - Гостивар	14,082,765	155,397,712	111,969,112	9,783,990	277,519,008	119,560,456	688,313,024
West	**27,841,248**	**446,772,448**	**232,522,896**	**1,130,329**	**675,669,888**	**194,824,464**	**1,578,761,344**
Битола - Ресен	12,774,092	180,061,456	119,558,544	731,713	270,584,032	59,538,480	643,248,256
Кичево - Дебар	7,893,499	96,283,080	19,436,066	127,184	129,886,528	34,349,116	287,975,488
Охрид - Струга	7,173,658	170,428,272	93,528,192	271,432	275,202,400	100,936,896	647,540,800
Grand Total	**355,058,656**	**1,789,191,040**	**2,714,540,032**	**42,620,204**	**2,968,226,304**	**1,066,228,096**	**8,935,864,320**

Слика 47. Операција пивот на податочна коцка
Figure 47. Pivot operation on data cube

Со операцијата slice ќе избереме една единствена вредност од димензијата област за вредноста East. Добиваме една подкоцка за вредноста East и сите податоци од другите димензии прикажано на Слика 48.

Prodazba Iznos	Column Labels						
	HORECA	LKA	NKA	OTHER	TT	WHOLESALER	Grand Total
Row Labels							
East	**43,135,668**	**313,587,744**	**306,997,696**	**2,817,047**	**439,615,008**	**156,192,784**	**1,262,345,856**
Кочани - Исток	8,748,143	50,865,564	90,036,816	1,521,518	141,177,424	38,992,040	331,341,504
Куманово	9,583,461	121,010,992	128,500,944	876,095	179,474,368	77,407,928	516,853,792
Штип - Св.Николе	24,804,032	141,710,896	88,459,704	419,434	118,962,504	39,792,696	414,149,248
Grand Total	**43,135,668**	**313,587,744**	**306,997,696**	**2,817,047**	**439,615,008**	**156,192,784**	**1,262,345,856**

Слика 48. Операција парче на податочна коцка
Figure 48. Slice operation on data cube

Операцијата dice е слична со операцијата slice и се разликува во поголемиот број на селектирани вредности од една димензија. Во нашиот случај ќе избереме повеќе вредности од димензијата област и сите податоци од останатите димензии прикажано на Слика 49.

Prodazba Iznos	Column Labels						
	HORECA	LKA	NKA	OTHER	TT	WHOLESALER	Grand Total
Row Labels							
East	43,135,668	313,587,744	306,997,696	2,817,047	439,615,008	156,192,784	1,262,345,856
Кочани - Исток	8,748,143	50,865,564	90,036,816	1,521,518	141,177,424	38,992,040	331,341,504
Куманово	9,583,461	121,010,992	128,500,944	876,095	179,474,368	77,407,928	516,853,792
Штип - Св.Николе	24,804,032	141,710,896	88,459,704	419,434	118,962,504	39,792,696	414,149,248
North	226,567,136	598,143,808	1,761,016,704	36,333,520	1,046,012,800	377,987,168	4,046,061,312
Скопје	212,484,112	442,744,832	1,649,048,832	26,549,530	768,497,856	258,426,272	3,357,751,296
Тетово - Гостивар	14,082,765	155,397,712	111,969,112	9,783,990	277,519,008	119,560,456	688,313,024
Grand Total	269,702,688	911,729,472	2,068,011,776	39,150,576	1,485,626,112	534,179,296	5,308,400,128

Слика 49. Операција сечење на податочна коцка
Figure 49. Dice operation on data cube

Операцијата drill-down овозможува навигација помеѓу нивоата, од ниво на повеќе сумирани кон помалку сумирани податоци. Во нашиот случај ќе извршиме навигација од ниво подрегион, општина, населено место, продажно место прикажано на Слика 50.

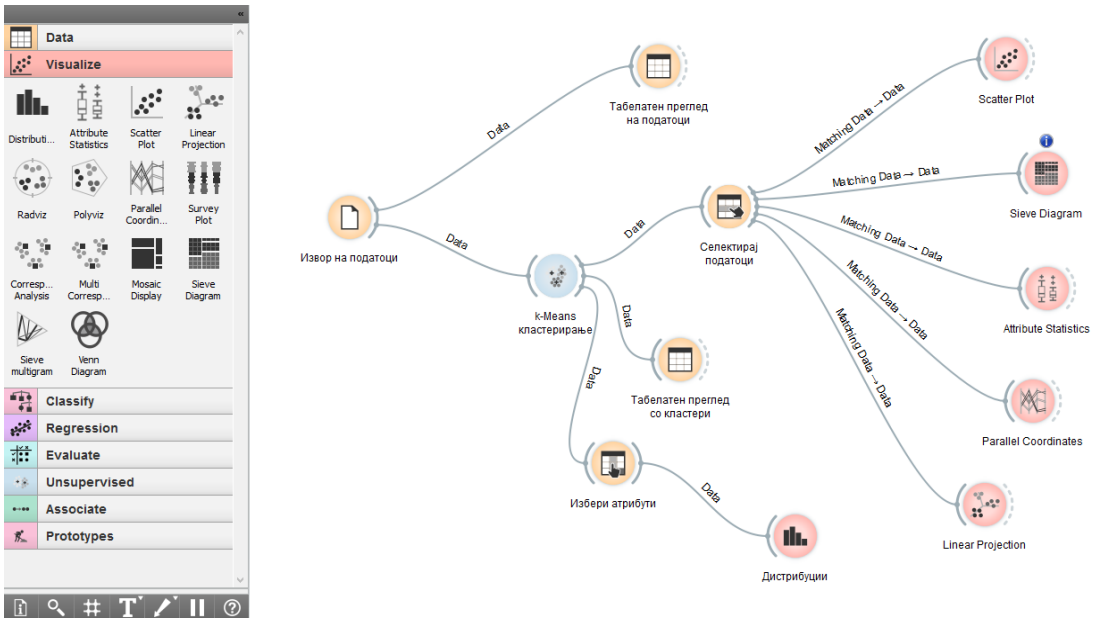
Prodazba Iznos	Column Labels						
	HORECA	LKA	NKA	OTHER	TT	WHOLESALER	Grand Total
Row Labels							
East	43,135,668	313,587,744	306,997,696	2,817,047	439,615,008	156,192,784	1,262,344,064
Кочани - Исток	8,748,143	50,865,564	90,036,816	1,521,518	141,177,424	38,992,040	331,341,568
Берово - Делчево	42,188	16,403,456	55,778,384	181,750	50,647,040	3,932,958	126,985,776
Виница		16,816,088	7,649,727	123,887	26,950,510		51,540,216
Кочани	8,705,954	17,646,026	26,608,688	1,215,880	63,579,940	35,059,076	152,815,568
Куманово	9,583,461	121,010,992	128,500,944	876,095	179,474,368	77,407,928	516,853,376
Кратово		4,111,350	394,044	88,547	9,429,247		14,023,188
Крива Паланка	522,274	7,311,033	20,506,154	92,801	25,488,752	12,295,462	66,216,476
Куманово	9,061,182	109,588,640	107,600,808	694,747	144,555,840	65,112,468	436,613,696
Штип - Св.Николе	24,804,032	141,710,896	88,459,704	419,434	118,962,504	39,792,696	414,149,088
Пробиштип		4,463,823	8,438,335	2,971	20,599,596		33,504,728
Свети Николе	250,250	30,365,584	1,324,232	48,250	18,842,308	4,757,998	55,588,620
Штип	24,553,772	106,881,432	78,697,200	368,213	79,520,440	35,034,704	325,055,744
Долни Балван					268,744		268,744
Карбинци					492,561		492,561
Крупиште					468,342		468,342
Таринци					25,150		25,150
Чифлик (Штипско)					174,279		174,279
САВАНА-Б ШТИПН - Lanci.Lanec; 50					174,279		174,279
Штип	24,553,784	106,881,464	78,697,200	368,213	78,091,384	35,034,700	323,626,752
North	226,567,136	598,143,808	1,761,016,704	36,333,520	1,046,012,800	377,987,168	4,046,056,960
Скопје	212,484,112	442,744,832	1,649,048,832	26,549,530	768,497,856	258,426,272	3,357,743,104
Скопје	212,484,320	442,744,128	1,649,045,376	26,549,536	768,493,376	258,426,272	3,357,743,104
Тетово - Гостивар	14,082,765	155,397,712	111,969,112	9,783,990	277,519,008	119,560,456	688,313,856
Гостивар		57,320,180	5,265,586	630,405	114,906,536	44,141,944	222,264,656
Тетово	14,082,764	98,077,376	106,703,504	9,153,585	162,613,408	75,418,544	466,049,216
Grand Total	269,702,688	911,729,472	2,068,011,776	39,150,576	1,485,626,112	534,179,296	5,308,400,128

Слика 50. Операција бушење надолу на податочна коцка
Figure 6.36 Drill down operation on data cube

6.4. КОРИСТЕЊЕ НА ТЕХНИКИ ЗА ПОДАТОЧНО РУДАРЕЊЕ

За разлика од претходите чекори кои се однесуваат на читање и анализирање на постоечките податоци и нивно интерпретирање во разбирлива форма, следен чекор е полуавтоматска анализа на податоците од податочниот склад и извлекување на претходно непознати и интересни шеми. За таа цел ќе употребиме кластерска анализа односно k-means кластерирање, метод на векторска квантификација.

Како алатка ќе се користи Orange верзија 2.7.6, софтвер со отворен код кој служи за податочно рударење и визуелизација на добиените резултати. Целиот процес е поделен во три дела, односно утврдување на извор на податоци, користење на k-means алгоритам за креирање кластери и визуелно прикажување со интерпретација на добиени резултати прикажано на слика 6.37.



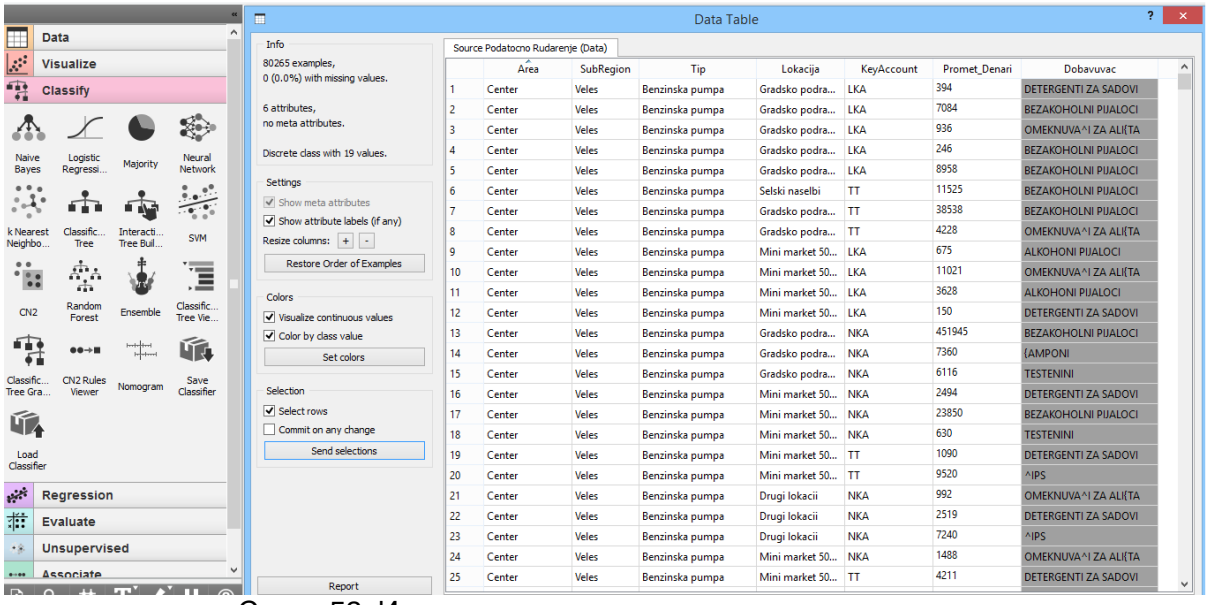
Слика 51. Дијаграм за k-means податочно рударење
Figure 51. K-means diagram for data minnig

6.4.1. ИЗВОР НА ПОДАТОЦИ ЗА ПОДАТОЧНО РУДАРЕЊЕ

Како извор на податоци, креиран е прашалник во податочниот склад со следниов SQL код:

```
SELECT dbo.tbl_NaseleniMesta.Area, dbo.tbl_NaseleniMesta.SubRegion,
dbo.tbl_ProdaznoMesto.Tip, dbo.tbl_ProdaznoMesto.Lokacija, dbo.tbl_Kupuvaci.KeyAccount,
SUM(ROUND(dbo.tbl_Prodazba.ProdazbaIznos, 0))
        AS Promet_Denari, dbo.tbl_Artikli.Dobavuvac
FROM    dbo.tbl_NaseleniMesta INNER JOIN
        dbo.tbl_ProdaznoMesto ON dbo.tbl_NaseleniMesta.NaselenoMestoID =
dbo.tbl_ProdaznoMesto.NaselenoMestoID INNER JOIN
        dbo.tbl_Artikli INNER JOIN
        dbo.tbl_Prodazba ON dbo.tbl_Artikli.Artikal_ID = dbo.tbl_Prodazba.ArtikalId INNER
JOIN
        dbo.tbl_Kupuvaci ON dbo.tbl_Prodazba.Kupuvac_ID = dbo.tbl_Kupuvaci.KupuvacId
ON dbo.tbl_ProdaznoMesto.ProdaznoMesto_Id = dbo.tbl_Prodazba.ProdaznoMestoID
GROUP BY dbo.tbl_NaseleniMesta.Area, dbo.tbl_NaseleniMesta.SubRegion,
dbo.tbl_ProdaznoMesto.Tip, dbo.tbl_ProdaznoMesto.Lokacija, dbo.tbl_Kupuvaci.KeyAccount,
dbo.tbl_Artikli.Dobavuvac
```

Како што може да се види од SQL кодот извршено е групирање, сумирање и селектирање на Area, SubRegion, Tip, Lokacija, KeyAccount, Promet_Denari и Dobavuvac. Исфрлени се некои податочни колони кои може да бидат опфатени во некои наредни анализи. Податоците се конвертирани во tab-delimited text фајл, формат кој е читлив од Orange Слика 52.



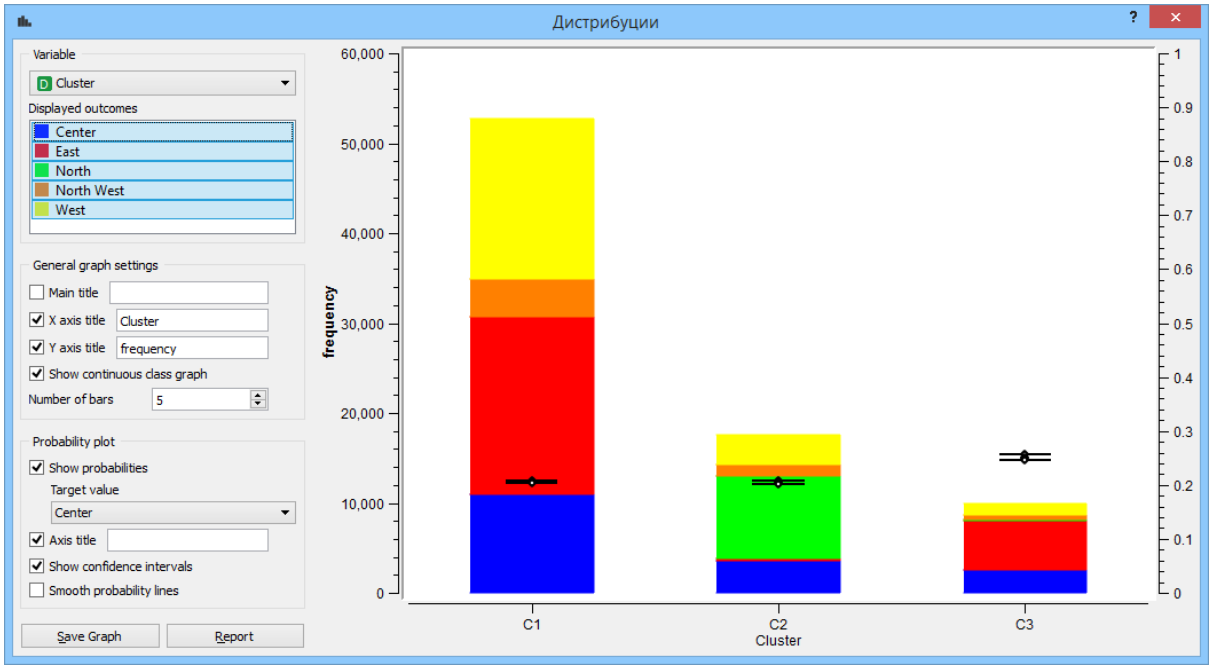
Слика 52. Извор на податоци за податочно рударење
Figure 52. Data source for data mining

6.4.2. ВИЗУЕЛИЗАЦИЈА И ИНТЕРПРЕТАЦИЈА НА РЕЗУЛТАТИ

Целта на трудот е да покаже како се имплементира систем за податочно рударење и од таа гледна точка ќе се направи објаснување на процесот и детална интерпретација на добиените резултати.

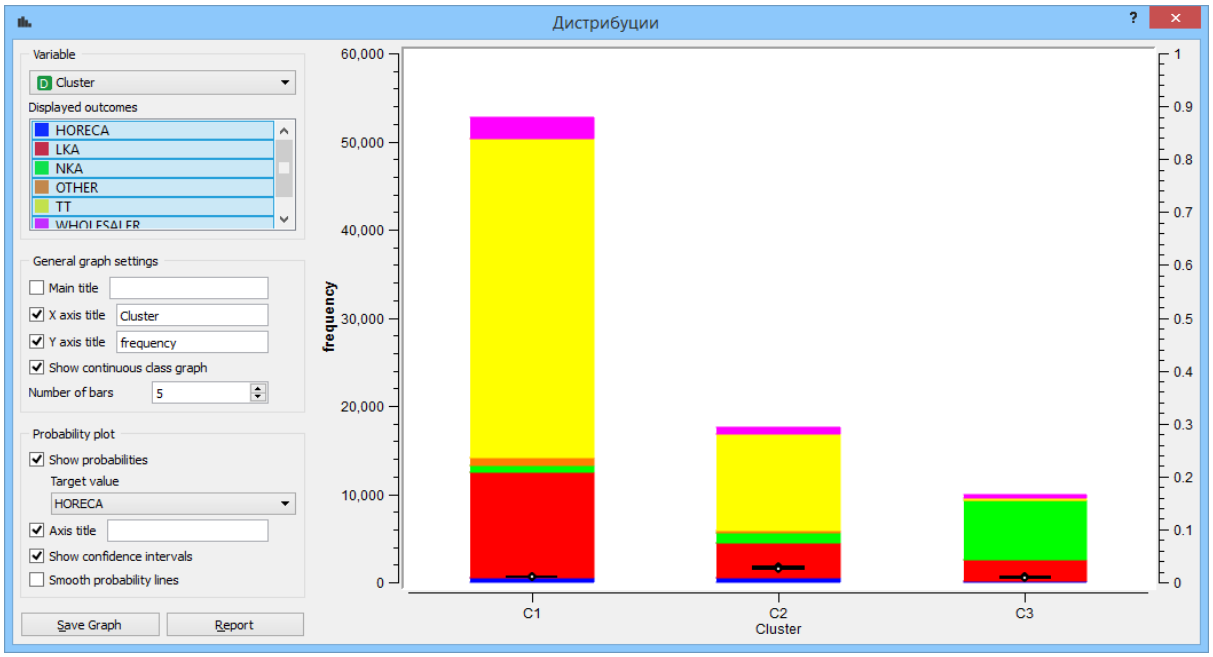
Со извршување на алгоритамот, креирани се три кластери и сега неопходно е да се види дистрибуцијата на податоци поединечно за секој атрибут. Вкупниот број на податоци или фреквенција кој се анализира е 80265.

На Слика 53 може да се види графички приказ на дистрибуцијата на атрибутот Area. На x оската се прикажани трите кластери, додека на y оската е прикажана фреквенцијата на податоци. Секој од регионите е означен со боја, при што може да се види дека кластер C1 ги претставува сите региони со исклучок на North, C2 го претставува North, делумно West и Center додека кластерот C3 го претставува главно East и делумно West и Center. Врз основа на оваа дистрибуција добиваме сознанија за географската застапеност по дадените кластери, а кое понатаму ќе ни помогне при анализата и процесот на донесување на одлуки кои се однесуваат за дадените региони.



Слика 53. Дистрибуција на аргументот Area
Figure 53. Area argument distribution

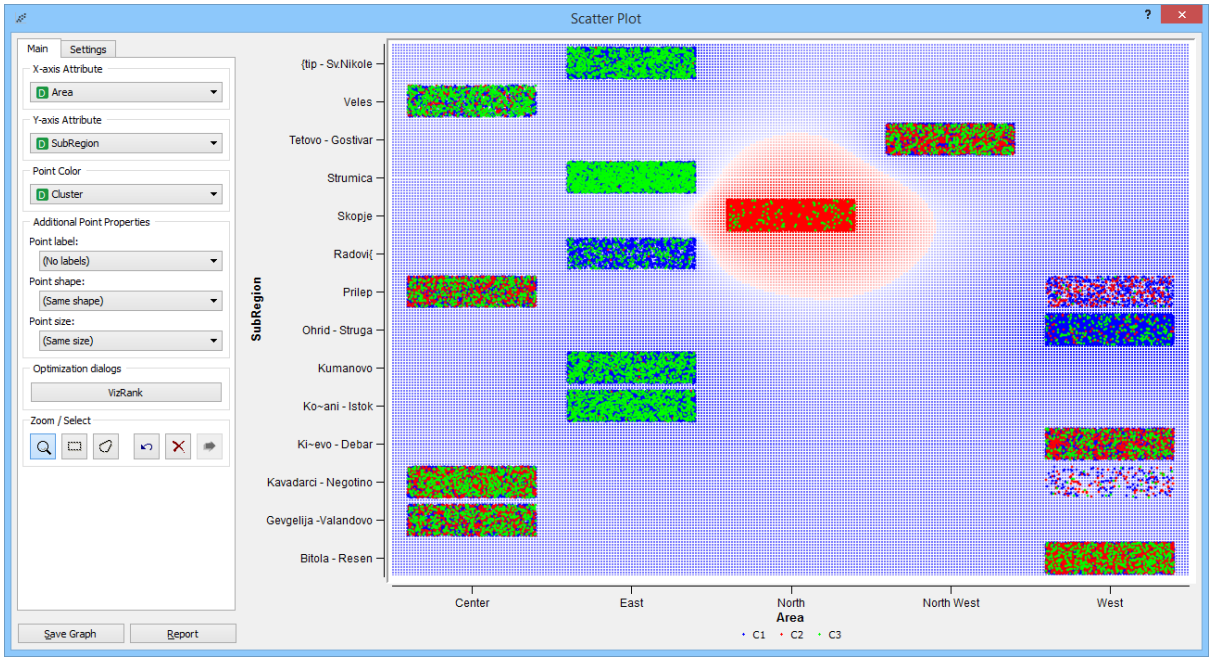
Слично како и претходната дистрибуција, во Слика 54 е прикажана дистрибуцијата на атрибутот KeyAccount по кластери, односно дистрибуција на малопродажните ланци според генерална класификација. На x оската се прикажани трите кластери, додека на y оската е прикажана фреквенцијата на податоци. Како што може да се види од графиконот, кластерот C1 опфаќа над 52000 податоци и ги опфаќа каналите на продажба Local Key Account, Traditional Trade, Wholesalers. Кластерот C2 ги опфаќа Local Key Account, Traditional Trade, додека кластерот C3 ги претставува главно National Key Account и Local Key Account. Значи кластерите C1 и C2 ги претставуваат малите малопродажни објекти како канал на продажба како додека C3 ги претставува каналите на продажба во големи купувачи. Врз основа на оваа дистрибуција добиваме сознанија за застапеност на типови малопродажни ланци по дадените кластери, а кое понатаму ќе ни помогне при анализата и процесот на донесување на одлуки кои се однесуваат за дадените типови малопродажни објекти.



Слика 54. Дистрибуција на аргументот KeyAccount
Figure 54. KeyAccount argument distribution

Врз основа на направена анализа на дистрибуцијата на аргументите Area и KeyAccount добивме информации за географската дистрибуција и

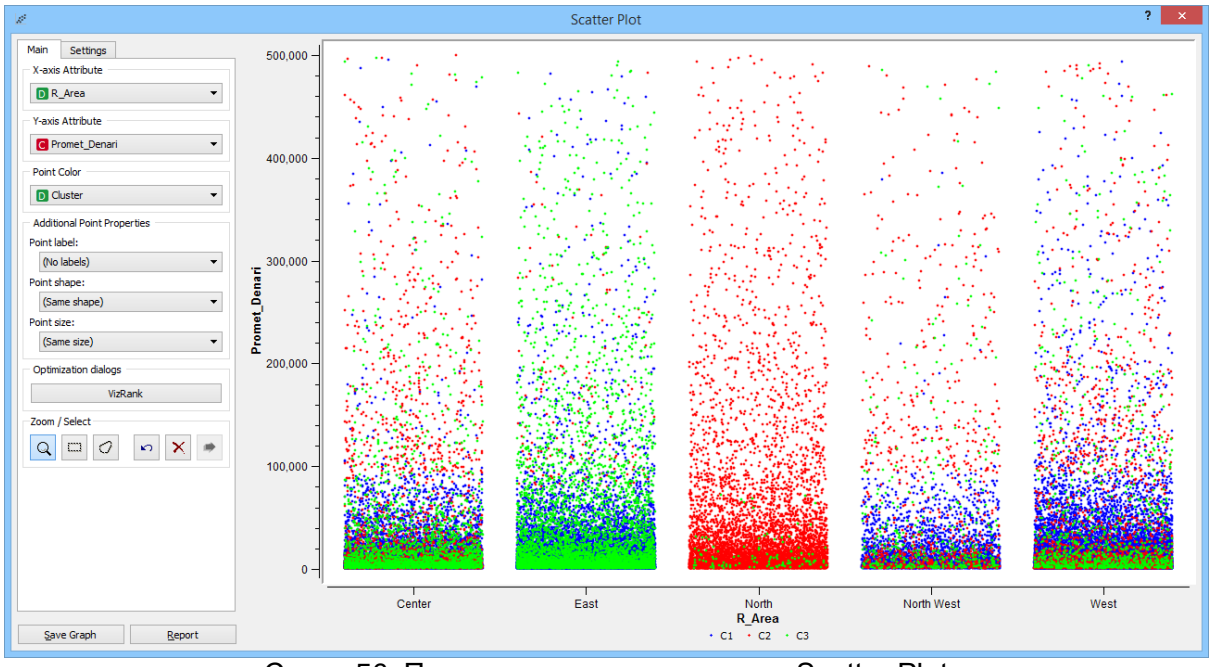
дистрибуцијата според големина на купувачи. Како што е прикажано на Слика 54 на x оската се прикажани регионите, на y оската градовите додека со сина боја е прикажан кластер C1, црвена кластер C2 и зелена кластер C3. По стекнатото знаење за дистрибуција на аргументи по кластер се преминува на анализа на кластерите, како што е прикажано на Слика 55. Посебно интересна е шемата или знаењето што го добиваме од овој приказ. Голема е веројатноста да се зголемува значењето на кластерот C2 во регион North (град Скопје) што е прикажано со облакот во црвена боја. Ова означува дека значењето на каналите на продажба Traditional Trade и Local Key Account, односно малите купувачи во регионот на град Скопје нема да се намали. Ова наметнува деловна одлука за задржување и проширување на дистрибутивна мрежа за овие канали на продажба. За разлика од град Скопје во град Струмица веројатноста за зголемување на бројот на National Key Account е голема, што наметнува потреба за размислување за ангажирање на комерцијалист специјализиран за продажба по големи купувачи. Со ваква анализа се доаѓа до сознанија за сите останати градови и региони поодделно.



Слика 55. Прикажување на кластер со Scatter Plot
Figure 55. Scatter Plot cluster visualization

На Слика 56 го имаме прикажано учеството во промет на одредени кластери по регион. Така на x оската се прикажани регионите, у оската промет

во денари, додека со сина боја е прикажан кластер C1, црвена кластер C2 и зелена кластер C3. Во регионите Center, East и донекаде West ќе се зголемува учеството на кластер C3, односно учеството на големите купувачи, додека во регионот North учеството во промет на Traditional Trade и Local Key Account сè уште ќе има значителен удел. Регионот North West ја задржува класичната дистрибуција на промет. Врз основа на овие сознанија се наметнува потреба од размислување за промена на дистрибутивната мрежа во регионите Center, East и West. Размислувањето се темели на согледаната динамика на пазарот.



Слика 56. Прикажување на кластер со Scatter Plot
Figure 56. Scatter Plot cluster visualization

7. ЗАКЛУЧОК

Тргнувајќи од основната цел за создавање методологија за креирање на податочен склад и систем за податочно рударење и можноста да се извлече скриено знаење од историските бази на податоци кои значајно би помогнале во донесување стратешки одлуки во една компанија, истражувањето наметнува повеќе насоки и заклучоци.

1. Историските бази на податоци кои не се користеа во работењето на компанијата се интегрираа во податочниот склад и се овозможи нивно тековно искористување. Тековната трансакциона база од 2015 година, исто така, се интегрира во податочниот склад со што се овозможи интегрално анализирање на сите податоци. Податочниот склад и процесот на ETL овозможија една нова платформа за понатамошен развој на системот, што беше и појдовна база за имплементација за замисленият систем за податочно рударење.
2. Креираниот податочен склад, податочната коцка и имплементираното податочно рударење даде резултати кои се во согласност со поставената цел. Добиените извештаи дадоа една нова димензија во размислувањето на менаџерските структури. Отворените можности придонесоа за активно вклучување на менаџерските структури во имплементацијата како и во барањата за креирање извештаи за откривање на скриени знаења, односно можност за донесување одлуки врз основа на податоци а не само врз искуство.
3. Имплементацијата на систем за податочно рударење се наметна како приоритет во работењето на компанијата. Креираната инфраструктура, како и обуката на одговорното лице за ИТ во компанијата овозможи една добра основа за понатамошен развој.
4. Магистерската теза ќе придонесе за зголемување на продуктивноста на компанијата преку донесување на одлуки поврзани со организација на работењето, а исто така донесените стратешки одлуки во делот на организацијата на продажната сила прелиминарно покажуваат намалување на трошоците и истовремено зголемување на продажбата.

5. Серверите се користат само за OLTP, притоа имаме 10% искористување од нивниот потенцијал. Се наметнува заклучок дека хардверот не претставува проблем за развој на систем за податочно рударење.
6. Оперативен систем и системот за релациони бази на податоци се современи и можат да одговорат на современите предизвици. Можностите на сегашниот софтвер се далеку над искористувањата и имплементацијата на податочниот склад е возможна без дополнителни софтвери.
7. Во поглавјето 6.4.3 е претставен пример за практично користење на креираниот систем преку визуелизација и интерпретација на резултати. Добиените резултати имаат практична примена и се во согласност на поставените цели на трудот.

КОРИСТЕНА ЛИТЕРАТУРА

1. **Bonnet, Dennis Shasha and Philippe.** *Database Tuning: Principles, Experiments, and Troubleshooting Techniques.* б.м. : Morgan Kaufmann, 2002.

2. **Berry, G. S. L. Michael J.A.** *Data Mining Techniques For Marketing, Sales, and Customer Relationship Management.* б.м. : Wiley Publishing, Inc., Indianapolis, Indiana, 2004.

3. **Larson, Brian.** *Delivering Business Intelligence with Microsoft SQL Server 2008.* б.м. : The McGraw-Hill Companies, 2009.

4. **Foster Provost, Tom Fawcett.** *Data Science for Business.* б.м. : O'Reilly, 2013.

5. **Little, Roderick J.A. и Rubin, Donald B.** *Statistical Analysis with Missing Data. Wiley Series in Probability and Mathematical Statistics.* New York : John Wiley & Sons, 1987.

6. **Kamber, Jiawei Han and Micheline.** *Data Mining: Concepts and Techniques, Second Edition.* б.м. : Elsevier Inc, 2006.

7. **Galit Shmueli, Nitin R. Patel, Peter C. Bruce.** *Data Mining for Business Intelligence: Concepts, Techniques, and application in Microsoft Office Excel with XLMiner.* б.м. : John Wiley & Sons, 2012.

8. **Javier Torrenteras, Carlos Martinez.** SSAS Cube Exploration:Digging Through the Details with Drillthrough. [В Интернете] 14 02 2013 г. http://blogs.solidq.com/en/businessanalytics/ssas-cube-exploration-digging-details-drillthrough/.

9. **David L. Olson, Dursun Delen.** *Advanced Data Mining Techniques.* б.м. : Springer-Verlag Berlin Heidelbergh, 2008.

10. **Sayad, Dr. Saed.** An Introduction to Data Mining. [В Интернете] 16 03 2012 г. http://chem-eng.utoronto.ca/~datamining/dmc/data_mining_map.htm.

11. **Association analysis: Basic concepts and algorithms. [В Интернете] 25 10 2014 г. http://www-users.cs.umn.edu/~kumar/dmbook/ch6.pdf.**

12. Hahsler, Michael. A Probabilistic Comparison of Commonly Used Interest Measures for Association Rules. [В Интернете] 2015 г. http://michael.hahsler.net/research/association_rules/measures.html.

13. *Mining Association Rules between Sets of Items in Large Databases.* Swami, Rakesh Agrawal Tomasz Imielinski Arun. IBM Almaden Research Center.

14. King, Ronald S. *Cluster Analysis and Data Mining.* б.м. : Mercury Learning & Information, 2014.

15. SAS. SAS/STAT 9.2 Users Guide. SAS Institute. *Hierarchical clustering.* [В Интернете] 19.10.2014 г. http://support.sas.com/documentation/cdl/en/statug/63033/HTML/default/viewer.htm#statug_distance_sect016.htm.

16. *Standardization and Its Effects on K-Means Clustering Algorithm.* Usman, Ismail Bin Mohamad and Dauda. 2013 г., Research Journal of Applied Sciences, Engineering and Technology.

17. *RedR: A dataflow programming interface for R.* [В Интернете] 19.08.2013 г. https://code.google.com/p/r-orange/source/browse/branches/linux/orngClustering.py?r=318.

18. University of Ljubljana. Data table (Table). [В Интернете] 26.12.2014 г. http://docs.orange.biolab.si/reference/rst/Orange.data.table.html.

19. —. Hierarchical clustering (hierarchical). [В Интернете] 26.12.2014 г. http://docs.orange.biolab.si/reference/rst/Orange. clustering.hierarchical.html#cluster-analysis.

20. The Transition of Data into Wisdom. [В Интернете] 20.03.2012 г. http://www.information-management.com/news/2784-1.html.

21. *Oracle: Big Data for the Enterprise.* Dijcks, Jean Pierre. 2012 г., An Oracle White Paper.

22. www.datawarehouse4u.info, 2008-2009. OLTP vs. OLAP. [В Интернете] 20.03.2012 г. http://datawarehouse4u.info/OLTP-vs-OLAP.html.

23. *Data Mining, knowledge discovery With Neural Network Support.* Aman Kumar, Tarandeep Singh. 2012 г., International Journal of Engineering Research and Applications (IJERA) , стр. 93.

24. *Shared Memory Parallelization of Data Mining.* Ruoming Jin, Ge Yang, and Gagan Agrawal. 2004 г., IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, стр. 19.

25. Ian H. Witten, Eibe Frank, Mark A. Hall. *Data Mining – practical Machine Learning Tools and Techniques.* б.м. : Elsevier Inc, 2011.

26. Andrew R. Webb, Keith D. Copsey. *Statistical Pattern Recognition.* б.м. : John Wiley & Sons, Ltd, 2011.

27. Jamie MacLennan, ZhaoHui Tang, Bogdan Crivat. *Data mining with Microsoft SQL Server 2008.* б.м. : Wiley Publishing, Inc., Indianapolis, Indiana., 2009.

28. Siegel, Eric. *Predictive analytics : the power to predict who will click, buy, lie, or die.* б.м. : John Wiley & Sons, Inc., Hoboken, New Jersey, 2013.

29. Harrngton, Peter. *Machine Learning in Action.* б.м. : Manning Publications Co, 2012.

30. Oracle. Oracle Advanced Analytics SQL API Data Mining Algorithms. [В Интернете] 23.07.2013 г. http://www.oracle.com/technetwork/database/enterprise-edition/odm-techniques-algorithms-097163.html.

31. Mohammed J. Zaki, Wagner Meira, Jr. *Data Mining and Analysis: Fundamental Concepts and Algorithms.* б.м. : Cambridge University press, 2014.

32. Rokach, Lior. *Data Mining with Decision Trees: Theory and Applications.* б.м. : World Scientific Publishing Co, 2008.

33. Pang-Ning Tan, Michael Steinbach, Vipin Kumar. *Introduction to Data Mining.* б.м. : Addison-Wesley, 2013.

34. Microsoft. Data Mining Algorithms (Analysis Services - Data Mining). [В Интернете] 20.12.2014 г. https://msdn.microsoft.com/en-us/library/ms175595.aspx.

35. University of Ljubljana. Selection (selection). [В Интернете] 23.12.2014 г. http://docs.orange.biolab.si/reference/rst/Orange.feature.selection.html.

36. MacKay, David. Information Theory, Inference and Learning Algorithms. б.м. : Cambridge University Press, 2013.

37. Wu, C. F. Jeff. On the Convergence Properties of the EM Algorithm. *The Annals of Statistics .* б.м. : Institute of Mathematical Statistics, 1983.

38. Burns, S. Q. M. Mark Whitehorn and M. Keith. Microsoft Corporation, 07.2008. [В Интернете] 23 7 2012 г. http://technet.microsoft.com/en-us/library/cc719165(v=sql.100).aspx.

39. Nullege. Nullege Python source code. [В Интернете] 18.04.2014 г. http://nullege.com/codes/show/src@o@r@Orange-2.7.2@Orange@clustering@consensus.py/68/Orange.clustering.kmeans.

ПРИЛОГ

(K-MEANS АЛГОРИТАМ)

Класата Clustering го имплементира k-means алгоритамот. Извршувањето на алгоритамот, притоа опишувајќи го целиот процес е наведен во продолжение.

```
class Clustering:

    """Имплементира k-means кластеринг алгоритам:

    #. Избор на број на кластери, k = 3;
    # Се мери евклидово растојание;
    # Бројот на рестарти е 1;
    # Избор на множество од k иницијални центроиди;
    # Доделување на секоја инстанца од податочното множество на најблискиот центроид;
    #. За секој кластер, пресметка на нов центроид како центар на кластерираните податочни
       инстанци;
    #. Повторување на претходните два чекори, се додека некој конвергентен критериум не
    се исполни (на пример доделувањето на кластер не е променето). Главна предност на овој
    алгоритам е едноставноста и малите барања за работна меморија. Главен недостаток е
    зависност на резултатите од селекцијата на иницијалното множество од центроиди.

    .. attribute:: k

        Број на кластери.

    .. attribute:: data

        Инстанци во кластерот.

    .. attribute:: centroids

        Тековно множество од центроиди.

    .. attribute:: scoring

        Тековен кластеринг резултат.

    .. attribute:: iteration

        Тековно кластеринг повторување.

    .. attribute:: clusters

        Листа на кластер индекси. I-от елемент обезбедува индекс на центроидот поврзан со
    i-та податочна инстанца од влезното податочно множество.
    """

    def __init__(self, data=None, centroids=3, maxiters=None, minscorechange=None,
                 stopchanges=0, nstart=1, initialization=init_random,
                 distance=Orange.distance.Euclidean,
                 scoring=score_distance_to_centroids, inner_callback=None,
                 outer_callback=None):
```

"""
:param data: **Податочни инстанци да се кластерираат. Ако не е None, кластерирањето ќе се изврши веднаш освен ако** ``initialize_only=True``.
:type data: :class:`~Orange.data.Table` **или ништо.**
:param centroids: **или да се определи бројот на кластери или да се обезбеди листа на примери кои ќе служат како кластерирање центроиди.**
:type centroids: :obj:`int` **или** :obj:`list` од :class:`~Orange.data.Instance`
:param nstart: **Ако е повеќе од еден, nstart работи и кластеринг алгоритамот ќе се изврши, враќајќи кластеринг со најдобар (најмал) резултат.**
:type nstart: int
:param distance: **пример на** distance **конструктор, кој го мери растојанието помеѓу две инстанци.**
:type distance: :class:`~Orange.distance.DistanceConstructor`
:param initialization: **функција за селектирање дадени податочни инстанци на центроид, k и пример за distance функција. Овој модул имплементира различни пристапи (:obj:`init_random`, :obj:`init_diversity`, :obj:`init_hclustering`).**
:param scoring**: функција која го зема кластеринг објектот и враќа кластеринг резултат. Може да се користи за инстанци во процедури кои повторуваат nstart - пати, враќајќи кластеринг со најнизок резултат.**
:param inner_callback: **повикува после секое кластеринг повторување.**
:param outer_callback: **повикува после секое кластеринг рестартирање** (if nstart is greater than 1). (17)

Критериуми за стопирање:

:param maxiters**: максимален број на кластеринг повторувања;**
:type maxiters: integer;
:param minscorechange: **минимално поправување на резултат од претходна генерација(ако е помал, кластерирањето ќе застане). Ако е None, резултатот ќе се пресмета помеѓу повторувањата,**
:type minscorechange: float **или** None;
:param stopchanges: **ако бројот на повторувањата кои го менуваат кластерот е помал или еднаков од stopchanges, стопирај го кластерирањено;**
:type stopchanges: integer
"""

```
self.data = data
self.k = centroids if type(centroids)==int else len(centroids)
self.centroids = centroids if type(centroids) == orange.ExampleTable else None
self.maxiters = maxiters
self.minscorechange = minscorechange
self.stopchanges = stopchanges
self.nstart = nstart
self.initialization = initialization
self.distance_constructor = distance
self.distance = self.distance_constructor(self.data) if self.data else None
self.scoring = scoring
self.minimize_score = True if hasattr(scoring, 'minimize') else False
self.inner_callback = inner_callback
self.outer_callback = outer_callback
if self.data:
    self.run()

def __call__(self, data = None):
    """Извршува k-means кластеринг алгоритам, со clustering algorithm, with опционални нови податоци."""
    if data:
```

```
            self.data = data
            self.distance = self.distance_constructor(self.data)
        self.run()

    def init_centroids(self):
        """Инцијализација на кластер центроиди"""
        if self.centroids and not self.nstart > 1: # центроидите се утврдени
            return
        self.centroids = self.initialization(self.data, self.k, self.distance)

    def compute_centeroid(self, data):
        """Враќа центроид од податочното множество."""
        return data_center(data)

    def compute_cluster(self):
        """пресметува членство во кластерите"""
        return [minindex([self.distance(s, d) for s in self.centroids]) for d in self.data]

    def runone(self):
        """Извршува едно кластеринг повторување, почнувајќи со повторна пресметка на
центроидите, следено со пресметка на членството на податоците (поврзувајќи
податочни инстанци со нивните најблиски центроиди)."""
        self.centroids = [self.compute_centeroid(self.data.getitems(
            [i for i, c in enumerate(self.clusters) if c == cl])) for cl in range(self.k)]
        self.clusters = self.compute_cluster() (18)

    def run(self):
        """
        Извршува сè додека не се исполнат конвергентните улови. Ако nstart е поголем од
еден, nstart работи и кластеринг алгоритамот ќе биде извршен, враќајќи кластеринг со
најдобар (најмал) резултат.
        """
        self.winner = None
        for startindx in range(self.nstart):
            self.init_centroids()
            self.clusters = old_cluster = self.compute_cluster()
            if self.minscorechange != None:
                self.score = old_score = self.scoring(self)
            self.nchanges = len(self.data)
            self.iteration = 0
            stopcondition = False
            if self.inner_callback:
                self.inner_callback(self)
            while not stopcondition:
                self.iteration += 1
                self.runone()
                self.nchanges = sum(map(lambda x,y: x!=y, old_cluster, self.clusters))
                old_cluster = self.clusters
                if self.minscorechange != None:
                    self.score = self.scoring(self)
                    scorechange = (self.score - old_score) / old_score if old_score > 0 else
self.minscorechange
                    if self.minimize_score:
                        scorechange = -scorechange
                    old_score = self.score
                stopcondition = (self.nchanges <= self.stopchanges or
                                 self.iteration == self.maxiters or
```

```
                    (self.minscorechange != None and
                     scorechange <= self.minscorechange))
            if self.inner_callback:
                self.inner_callback(self)
        if self.scoring and self.minscorechange == None:
            self.score = self.scoring(self)
        if self.nstart > 1:
            if not self.winner or (self.score < self.winner[0] if
                    self.minimize_score else self.score > self.winner[0]):
                self.winner = (self.score, self.clusters, self.centroids)
            if self.outer_callback:
                self.outer_callback(self)

    if self.nstart > 1:
        self.score, self.clusters, self.centroids = self.winner (19)
```