A Project Report

on

ENHANCEMENT OF DATA ANALYTICS

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

In Partial Fulfillment of the Requirements for the Award of the Degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

Submitted By

S.M.RAZAELAHI - (15691A0579)

Under the Guidance of

Dr. P. Kuppusamy,Ph.D.,

Professor

Department of Computer Science & Engineering

MADANAPALLE INSTITUTE OF TECHNOLGY & SCIENCE

(UGC – AUTONOMOUS)

(Affiliated to JNTUA, Ananthapuramu)

Accredited by NBA, Approved by AICTE, New Delhi)

AN ISO 9001:2008 Certified Institution

P. B. No: 14, Angallu, Madanapalle – 517325

2015-2019

✆ : 08571 - 280255, 280590

MADANAPALLE INSTITUTE OF TECHNOLOGY & SCIENCE

AN ENGINEERING COLLEGE SPONSORED BY :
RATAKONDA RANGA REDDY EDUCATIONAL ACADEMY
Approved by AICTE, New Delhi and Affiliated to JNTUA, Anantapur
P.B. No. 14, Angallu, MADANAPALLE - 517 325. Chittoor Dist. A.P.
www.mits.edu

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that the project work entitled **“ENHANCEMENT OF DATA ANALYTICS”** is a bonafide work carried out by

S.M.RAZAELAHI - (15691A0579)

Submitted in partial fulfillment of the requirements for the award of degree **Bachelor of Technology** in the stream of **Computer Science & Engineering** in **Madanapalle Institute of Technology and Science, Madanapalle,** affiliated to **Jawaharlal Nehru Technological University Anantapur** during the academic year 2018-2019.

Guide	Head of the Department
Dr. P. Kuppusamy, M.E., Ph.D.,	**Dr. R. KALPANA, Ph.D.,**
Professor,	**Professor and HOD,**
Department of CSE	**Department of CSE**

Submitted for the University examination held on:

Internal Examiner	**External Examiner**
Date:	**Date:**

ACKNOWLEDGMENT

I sincerely thank the **MANAGEMENT** of **Madanapalle Institute of Technology and Science** for providing excellent infrastructure and lab facilities that helped me to complete this project.

I sincerely thank **Dr. C. Yuvaraj, M.E., Ph.D**., **Principal** for guiding and providing facilities for the successful completion of our project at **Madanapalle Institute of Technology and Science,** Madanapalle.

I express our deep sense of gratitude to **Dr. R. Kalpana, Ph.D., Professor and Head of the Department of CSE** for her valuable guidance and constant encouragement given to us during this work.

I express our deep gratitude to my guide **Dr. P. Kuppusamy, M.E., Ph.D.**, **Professor, Department of CSE** for his guidance and encouragement that helped us to complete this project.

I express our deep gratitude to my project Coordinator **Dr. V. Arun, Ph.D.**, **Associate Professor, Department of CSE** for his guidance and encouragement that helped us to complete this project.

I also wish to place on record my gratefulness to other **Faculty of CSE Department** and also to our friends and our parents for their help and cooperation during our project work.

DECLARATION

I hereby declare that the results embodied in this project **“Enhancement of Data Analytics”** by us under the guidance of **Dr. P. Kuppusamy, M.E., Ph.D., Professor, Dept. of CSE** in partial fulfillment of the award of **Bachelor of Technology** in **Computer Science & Engineering** from **Jawaharlal Nehru Technological University Anantapur** and I have not submitted the same to any other University/institute for award of any other degree.

Date:

Place:

PROJECT ASSOCIATE

S. M. Razaelahi - 15691A0579

I certify that above statement made by the students is correct to the best of my knowledge.

Date: **Guide**

INDEX

LIST OF FIGURES

ABSTRACT

The Microsoft SQL Server introduces a new data warehouse query acceleration feature based on a new index type called a column store index. The new index type combined with new query operators processing batches of rows greatly improves data warehouse query performance: in some cases, by hundreds of times and routinely a tenfold speedup for a broad range of decision support queries. Column store indexes are fully integrated with the rest of the system, including query processing and optimization. This project provides complete details on the design and implementation of column store indexes that includes enhancements to query processing and query optimization to take full advantage of the new indexes. The resulting performance improvements are illustrated by a number of example queries.

CHAPTER-1

INTRODUCTION

1. Introduction

Our project “Enhancement of Data Analytics” is an enhancement of the existing system to respond quickly to multiple Data Analytic inference queries. The existing system tries to respond to inference queries effectively but the response time is very high.

Database systems traditionally store data row-wise, that is, values from different columns of a record are stored together. This data organization works well for transaction processing where requests typically touch only a few records. However, it is not well suited for data warehousing where queries typically scan many records but touch only a few columns. In this case, a column-wise organization where values from the same column in different records are stored together performs much better. It reduces the data processed by a query because the query reads only the columns that it needs and, furthermore, column-wise data can be compressed efficiently. Systems using column-wise storage are usually referred to as column stores.

The Proposed system intends to improve performance on data warehousing queries using column-wise storage and efficient column-wise processing to the system. This capability is exposed as a new index type: a column store index. That is, an index can now be stored either row-wise in a B tree or column-wise in a column store index

To improve the performance of typical data analytic queries, all a user needs to do is build a column store index on the fact tables in the data warehouse. It may also be beneficial to build column store indexes on extremely large dimension tables (say more than 10 million rows). After that, queries can be submitted unchanged and the optimizer automatically decides whether or not to use a column store index exactly as it does for other indexes.

1.1 Domain Study

1.1.1 Data Mining:

Data Mining refers to extracting or mining information from large amounts of data. Data mining has attracted a great deal of attention in the information industry and in society as a whole in recent years, due to the wide availability of huge amounts of data and the imminent need for turning such data into useful information and knowledge.

Data mining, the extraction of hidden predictive information from large databases, is a powerful new technology with great potential to help companies focus on the most important information in their data warehouses. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions.

The automated, prospective analyses offered by data mining move beyond the analyses of past events provided by retrospective tools typical of decision support systems. Data mining tools can answer business questions that traditionally were too much time consuming to resolve. They scour databases for hidden patterns, finding predictive information that experts may miss because it lies outside their expectations.

Most companies already collect and refine massive quantities of data. Data mining techniques can be implemented rapidly on existing software and hardware platforms to enhance the value of existing information resources and can be integrated with new products and systems as they are brought on-line. When implemented on high-performance client/server or parallel processing computers, data mining tools can analyse massive databases to deliver answers to many questions.

The information and knowledge gained can be used for applications ranging from market analysis, fraud detection, and customer retention, to production control and science exploration. Data Mining plays an important role in online shopping for analyzing the subscribers' data and understanding their behaviors and making good decisions such that customer acquisition and customer retention are increased which gives high revenue.

1.2 Problem Statement

Database systems traditionally store data row-wise, that is, values from different columns of a record are stored together. This data organization works well for transaction processing where requests typically touch only a few records. However, it is not well suited for data warehousing where queries typically scan many records but touch only a few columns.

CHAPTER – 2

LITERATURE SURVEY

2.1 Introduction

In this chapter, the literature survey focuses on data analytic inference query processing and the existing methods that are already in use. Also, the Java Database Access Technology (JDBC) has been reviewed.

The following Fig (2.1) will show the query processing method.

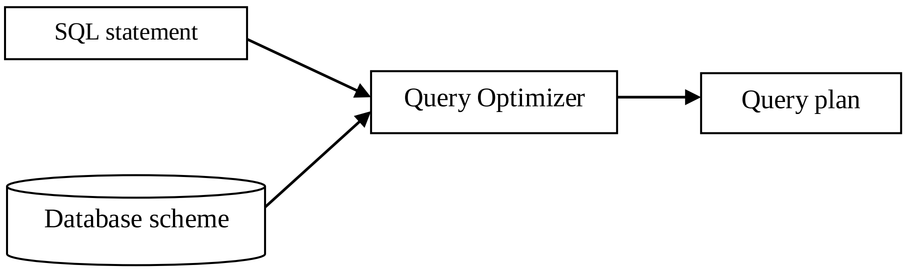


Fig 2.1 Query Optimization and processing

In the existing system, it concentrates on query optimization and query processing. It will focus on generating and processing the query in an optimized way. In the existing system, it will generate and process the query each time even though that query is already processed. It tries to increase the performance by making use of the query optimization concept. But it fails to give better results regarding the performance and time cost. It is not made use of the already processed query results. The time cost is too high because of repeatedly retrieving the same information each time from the database. The performance also decreased. It concentrates on whether the query is optimized one to fetch the information or not instead of to make use of already available information.

2.2 Database

A database is a collection of records, data or information that can be stored, modified or retrieved at any time. The essence of a database is to have an organized method of easy access to the required information which is kept in the database. Having a database will enable businesses to manage and update data details easily, effectively and with accuracy.

2.2.1 Query Optimization

The query processor applies rules to the internal data structures of the query to transform these structures into equivalent, but more efficient representations. The rules can be based upon mathematical models of the relational algebra expression and tree (heuristics), upon cost estimates of different algorithms applied to operations or upon the semantics within the query and the relations it involves. Selecting the proper rules to apply, when to apply them and how they are applied is the function of the query optimization engine.

2.3 Database Access Technology

2.3.1 ODBC

The Open Database Connectivity (ODBC) API is the industry standard for database-independent connectivity between the .NET programming language and a wide range of databases – SQL databases. The ODBC API provides a call-level API for SQL-based database access.

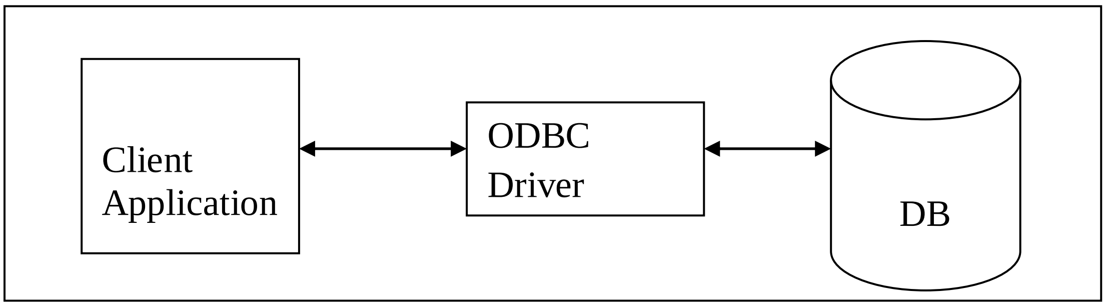


Fig 2.2 ODBC Connectivity diagram

2.3.2 JDBC

The Java Database Connectivity (JDBC) API is the industry standard for database-independent connectivity between the Java programming language and a wide range of databases – SQL databases. The JDBC API provides a call-level API for SQL-based database access.

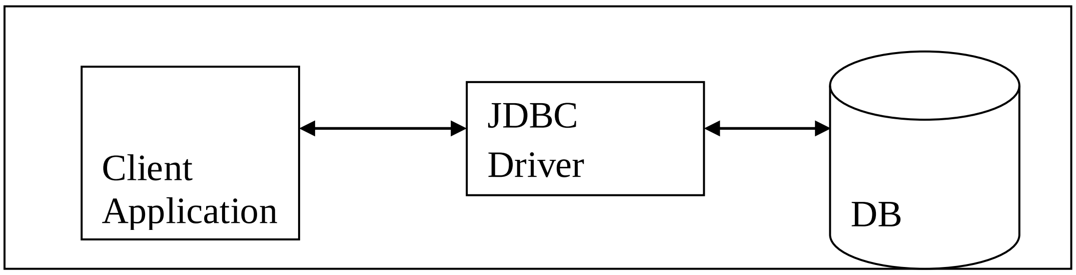


Fig 2.3JDBC Connectivity diagram

2.4 Related Methods and Technologies

Shao Xu song et al: A naive implementation of an inference query is to materialize a joint view by joining all the tables together, and then, perform GROUP BY (GB) operation to aggregate and eliminate the variables, not in the query. The joining tree for computing the materialized view of the joint distribution. Then, the GB operation aggregates the results to generate the marginal probability of the query.

In the database literature, techniques have been proposed to optimize the aggregation queries on joining trees. Push GROUP BY operation down into the joining tree. Since GROUP BY operation reduces the cardinality of a subquery result, early conduction of GROUP BY can potentially save the cost of subsequent joins. Interestingly, there is a similar strategy in the literature of probabilistic inference in Bayesian networks, called the variable elimination algorithm, which also studies the aggregation of variables one by one on the factors rather than the joint distribution. This is not surprising due to the similarity and correspondence between Bayesian networks and relational databases. Bravo and Ramakrishnan implement the VE Algorithm in relational databases and study a cost-based ordering heuristic for variable elimination.

The Bayesian networks can be naturally represented and stored in relational databases. Specifically, we transform each factor into a relational table. Other than variable attributes in a factor (relation), we introduce an extra attribute p to represent the probability value. For example, the corresponding relation of factor is tA (A, B, E,p), where p denotes the probability of P. According to properties of Bayesian networks, the joint distribution is specified by joining all the relations of factors, and can be represented by a relational database view. For example,

CREATE VIEW joint AS (SELECT B, E, A, J, M, tB.p * tE.p * tA.p * tJ.p * tM.p AS p FROM tB, tE, tA, tJ, tM WHERE tB.B=tA.B AND tE.E=tA.E AND tM.A=tA.A AND tJ.A=tA.A).

The time cost of query results by the minimum propagating join estimation (CTP min join) is the best in all these four groups of queries. The reason is that the JOIN-GROUP BY operator plays the most expensive role in the query evaluation. Thus, the queries with CTP min join estimation can achieve lower time cost. Note that the CTP min path does not have the minimum number of JOIN-GROUP BY steps in the results. The underlying reason is that the caching variables are updated incrementally, and the CTP min path estimation only greedily chooses the minimum propagating path for the current query. Consequently, after several queries with different pivots in these estimation approaches, the cached variables are various. Thus, the estimations for the next query are different as well due to the different historical caching data [1].

R. Chirkovaet al: Bayesian networks (BNs) have been widely used as a model for knowledge representation and probabilistic inferences. However, the single probability representation of conditional dependencies has been proven to be over constrained in realistic applications. Many efforts have proposed to represent the dependencies using probability intervals instead of single probabilities. In this paper, move one step further and adopt a probability distribution schema. This results in a higher order representation of uncertainties in a BN. Formulate probabilistic inferences in this context and then propose a mean/covariance propagation algorithm based on the well-known junction tree propagation for standard BNs. For algorithm validation, develop a two-layered Markov likelihood weighting approach that handles high-order uncertainties and provides “ground-truth” solutions to inferences, albeit very slowly. Experiments show that the mean/covariance propagation algorithm can efficiently produce high-quality solutions that compare favorably to results obtained through painstaking sampling.

To increase the expressive power of BNs, researchers have proposed interval and two-layered probabilities for realistic BN applications. Probabilistic interval representation and relevant inferences in BNs have been widely studied. The idea is to represent the sets of posterior probabilities as polytopes and represent a polytope using its vertices’ set. At each propagation step, the algorithm calculates the vertices representing the posteriors of a node. Its major drawback is that the vertices’ number grows very fast as the number of the parameters increases. A 2U algorithm, an exact interval propagation method for polytrees. But the 2U algorithm applies only to single-connected BNs with binary variables. Tessem proposed one of the earliest approximate algorithms to propagate interval probabilities. However, in this algorithm speed is achieved at the expense of accuracy and the interval bounds tend to diverge [0, 1] In addition to JT, a number of approaches exist for carrying out inferences using mathematics programming. The Multilinear Program (MLP) technique in creedal networks. It formulates an inference as an MLP and then solves the MLP. Its drawback is that the number of constraints grows explosively in the network size and that an MLP is still a difficult non-linear optimization problem [7].

S.K.M. Wong et al: Managing uncertain data using probabilistic frameworks has attracted much interest lately in the database literature, and a central computational challenge is a probabilistic inference. This paper presents a broad class of aggregate queries, called MPF queries, inspired by the literature on probabilistic inference in statistics and machine learning. An MPF (Marginalize a

Product Function) query is an aggregate query over a stylized join of several relations. In probabilistic inference, this join corresponds to taking the product of several probability distributions, while the aggregate operation corresponds to marginalization. Probabilistic inference can be expressed directly as MPF queries in a relational setting, and therefore, by optimizing evaluation of MPF queries, we provide scalable support for probabilistic inference in database systems. To optimize MPF queries, build on ideas from database query optimization as well as traditional algorithms such as Variable Elimination and Belief Propagation from the probabilistic inference literature. Although our main motivation for introducing MPF queries is to support easy expression and efficient evaluation of probabilistic inference in a DBMS, we observe that this class of queries is very useful for a range of decision support tasks. We present and optimize MPF queries in a general form where arbitrary functions (i.e., other than probability distributions) are handled and demonstrate their value for decision support applications.

Aggregate queries overviews, there are two options for evaluating an MPF query: The relation defined by view V is materialized, and queries are evaluated directly on the materialized view; or, each query is rewritten using V's definition and then evaluated, so that constructing the relation defined by V is an intermediate step. The first approach requires that the materialized view is updated as base relations change. In the latter, the problem of view maintenance is avoided, but this approach is prohibitive if computing V's relation is too expensive. The rewriting option is likely to be appropriate for answering individual queries, and variations of the former might be appropriate if we have knowledge of the anticipated query workload. In this paper, we study the query rewrite approach. Chaudhuri and Shim define an algorithm for optimizing aggregate query evaluation based on pushing aggregate operations inside join trees. Evaluate the Variable Elimination (VE) technique from the literature on optimizing probabilistic inference and show similar gains over existing systems. Additionally, we present extensions to VE based on ideas in the Chaudhuri and Shim algorithm that yield better plans than traditional VE [4],[5], [2].

R. Chirk et al: Clique tree propagation has a compilation step that transforms a BN into a secondary structure called a clique tree or junction tree. The secondary structure allows CTP to compute the answers to all queries with one query variable and a fixed set of observations in twice the time needed to answer one such query in the clique tree. For many applications, this is a desirable property since a user might want to compare the posterior probabilities of different variables. CTP takes work to build the secondary structure before any observations have been received. When the Bayesian network is reused, the cost of building the secondary structure can be amortized over many cases. Each observation entails a propagation through the network. Given all of the observations, VE processes one query at a time. If a user wants the posterior probabilities of several variables, or for a sequence of observations, she needs to run VE for each of the variables and observation sets.

The cost, in terms of the number of summations and multiplications, of answering a single query with no observations using VE is of the same order of magnitude as using CTP. A particular clique tree and propagation sequence encode an elimination ordering; using VE on that elimination ordering results in approximately the same summations and multiplications of factors as in the CTP (there is some discrepancy, as VE does not actually form the marginal on the cliques, but works with conditional probabilities directly). Observations make VE simpler (the observed variables are eliminated at the start of the algorithm), but each observation in CTP requires the propagation of evidence. Because VE is query oriented, we can prune nodes that are irrelevant to specific queries.In CTP, on the other hand, the clique tree structure is kept static at run time and hence does not allow pruning of irrelevant nodes.

Bayesian networks place no restrictions on how a node depends on its parents. Unfortunately, this means that in the most general case we need to specify an exponential (in the number of parents) number of conditional probabilities for each node. There are many cases where there is structure in the probability tables that can be exploited for both acquisitions and for inference. One such case that we investigate in this paper is known as ‘causal independence'[7].

2.5 EXISTING SYSTEM

2.5.1. Description

The existing systems store data row-wise, that is, values from different columns of a record are stored together. This data organization works well for transaction processing where requests typically touch only a few records. However, it is not well suited for data warehousing where queries typically scan many records but touch only a few columns. So the performance of the system will decrease because of processing the query row-wise and also the time is taken to respond to the user will be high.

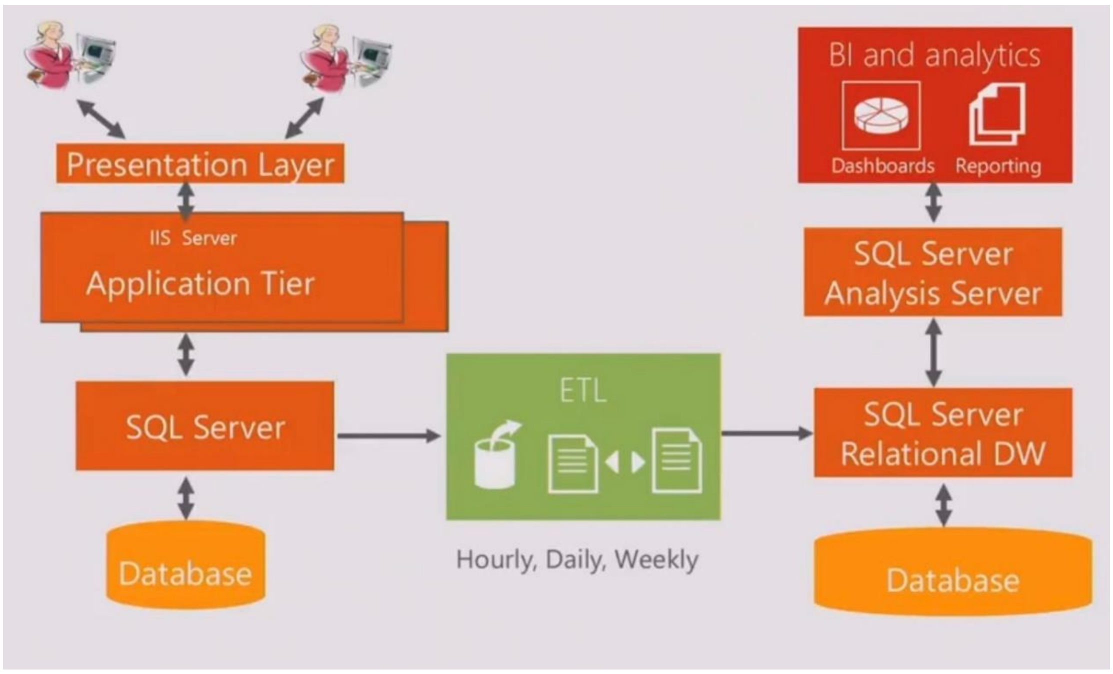


Fig 2.4 Existing System

2.5.2. Drawbacks

- Complex Implementation
- Requires Two servers (CapEx and OpEx)
- Data Latency in Analytics
- More Business Demand/require real-time Analytics

2.6 PROPOSED SYSTEM

2.6.1 Data Analytics

Data analytics uses an updateable Column store index on a row store table. The Column store index maintains a copy of the data, so the OLTP and analytics workloads run against separate copies of the data. This minimizes the performance impact of both workloads running at the same time. SQL Server automatically maintains index changes so OLTP changes are always up-to-date for analytics. With this design, it is possible and practical to run analytics in real-time on up-to-date data.

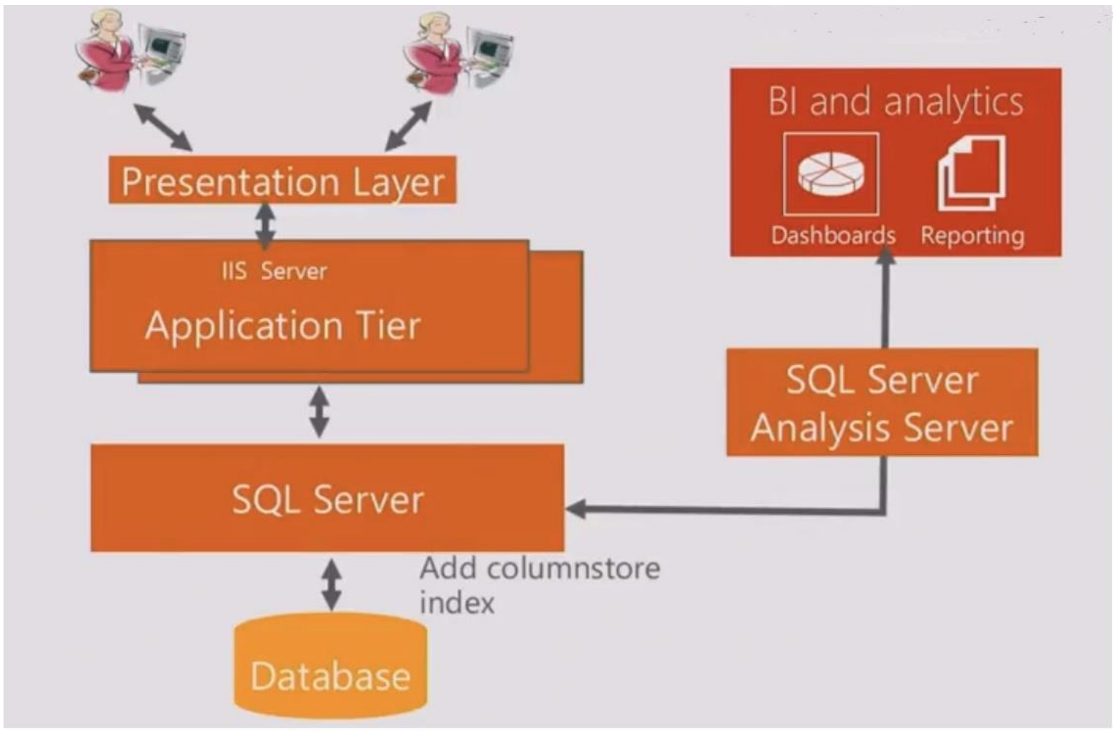


Fig 2.5 Proposed System

The general agreement on the previous version of Microsoft SQL Server was that it delivered Exceptional data protection, access control, and compliance. SQL Server 2008 R2 came equipped with various capabilities for organizations to leverage as well as, however not restricted to, clear Data Encryption (TDE) to guard information at rest, protractible Key Management (EKM) to meet information and key separation, Kerberos Authentication to attain the strongest authentication, and SQL Server Audit, Policy-Based Management, and alter information capture for fulfilling compliance necessities. SQL Server 2008 R2 was unquestionably strong from a security perspective and a fanatical leader in the database-platform industry, with the least amount of vulnerabilities.

2.6.2: Security enhancements in SQL server:

The following list shows some of the enhancements that may attractiveness to organizations wanting to realize maximum security and control of their database platform:

■ Security manageability improvements
• Default schema for groups
• User-defined server roles
■ Audit enhancements
• Audit supported on all SKUs
• Improved resilience

• User-defined audit event

• Record filtering

• T-SQL stack information

■ Database authentication enhancement: contained databases authentication

58 PART 1 Database Administration

■ Crypto changes

• Hashing algorithms

• Certificate key length

• Service master key and database master key encryption changes from 3DES to AES

■ Miscellaneous security enhancements

• Tight Integration with SharePoint and Active Directory

• Provisioning enhancements

• New permissions

This chapter describes every of those new security enhancements introduced in SQL Server 2012, starting with security manageability improvements.

2.6.3 Column Store index

The new Column store storage model significantly improves data warehouse query speeds for many reasons. First, data organized in a column shares many more similar characteristics than data organized across rows. As a result, a much higher level of compression can be achieved compared to data organized across rows. Moreover, the Columnstore index within SQL Server uses the Vert iPAQ compression algorithm technology, which in SQL Server 2008 R2 was found only in Analysis Server for PowerPivot. Vert iPAQ compression is far superior to the traditional row and page compression used in the Database Engine, with compression rates of up to 15 to 1 having been achieved with VertiPag. When data is compressed, queries require less IO because the amount of data transferred from disk to memory is significantly reduced. Reducing IO when processing queries equates to faster performance-response times. The advantages of Columnstore do not come to an end here. With less data transferred to memory, less space is required in memory to hold the working set affiliated with the query. Second, when a user runs a query using the Columnstore index, SQL Server fetches data only for the columns that are required for the query.

2.6.4 Security Enhancements

It has been approximately 10 years since Microsoft initiated its trustworthy computing initiative. Since then, SQL Server has had the best track record with the least amount of vulnerabilities and exposures among the major database players in the industry. The product continues to expand on this solid foundation to deliver enhanced security and compliance within the database platform. For detailed information of all the security enhancements associated with the Database Engine. SQL Server 2012 Editions and Engine Enhancements 11 "Security Enhancements." For now, here is a snapshot of some of the new enterprise-ready security capabilities and controls that enable organizations to meet strict compliance policies and regulations:

- ■ User-defined server roles for easier separation of duties
- ■ Audit enhancements to improve compliance and resiliency
- ■ Simplified security management, with a default schema for groups
- ■ Contained Database Authentication, which provides database authentication that uses self-contained access information without the need for server logins
- ■ SharePoint and Active Directory security models for higher data security in end-user reports.

2.6.5 Programmability Enhancements

There has also been a tremendous investment in SQL Server 2012 regarding programmability. Specifically, there is support for "beyond relational" elements such as XML, Spatial, Documents, Digital Media, Scientific Records, factoids, and other unstructured data types. Why such investments? Organizations have demanded they be given a way to reduce the costs associated with managing both structured and nanostructured data. They wanted to simplify the development of applications over all data, and they wanted the management and search capabilities for all data improved. Take a minute to review some of the SQL Server 2012 investments that positively impact programmability. For more information associated with programmability and beyond relational elements, please review Chapter 5, "Programmability and Beyond-Relational Enhancements."

■ **File Table:** Applications typically store data within a relational database engine; however, a myriad of applications also maintains the data in unstructured formats, such as documents, media files, and XML. Unstructured data usually resides on a file server and not directly in a relational database such as SQL Server. As you can imagine, it becomes challenging for organizations to not only manage their structured and unstructured data across these disparate systems, but to also keep them in sync. File Table, a new capability in SQL Server 2012, addresses these challenges. It builds on FILESTREAM technology that was first introduced with SQL Server 2008. File Table offers organizations Windows file namespace support and application compatibility with the file data stored in SQL Server. As an added bonus, when applications are allowed to integrate storage and data management within SQL Server, full text and semantic search is achievable over unstructured and structured data

■ **Statistical Semantic Search:** By introducing new semantic search functionality, SQL Server 2012 allows organizations to achieve deeper insight into unstructured data stored within the Database Engine. Three new Transact-SQL row set functions were introduced to query not only the words in a document, but also the meaning of the document.

■ **Full-Text Search Enhancements:** Full-text search in SQL Server 2012 offers better query performance and scale. It also introduces property-scoped searching functionality, which allows organizations the ability to search properties such as Author and Title without the need 12 PART 1 Database Administration for developers to maintain file properties in a separate database. Developers can now also benefit by customizing proximity search by using the new NEAR operator that allows them to specify the maximum number of non-search terms that separate the first and last search terms in a match.

■ **Extended Events Enhancements**: This new user interface was introduced to help simplify the management associated with extended events. New extended events for functional and performance troubleshooting were also introduced in SQL Server 2012.

2.6.6 Data Compression

Column store indexes come through up to 10x larger knowledge compression than row store indexes. This greatly reduces the I/O needed to execute analytics queries and so improves question performance. Column store indexes scan compressed knowledge from disk, which means fewer bytes of data need to be read into memory. Column store indexes store knowledge in compressed type in memory that reduces I/O by reducing the amount of times an equivalent knowledge is scan into memory.
For example, with 10x compression, column store indexes can keep 10x more data in memory compared to storing the data in uncompressed form. With more data in memory, it is more likely that the column store index will find the data it needs in memory with incurring

additional reads from disk. Column store indexes compress knowledge by columns rather than by rows that achieves high compression rates and reduces the scale of the info hold on on disk. Each column is compressed and stored independently. Data within a column always has the same data type and tends to have similar values. Data compression techniques are very good at achieving higher compression rates when values are similar. For example, if a fact table stores customer addresses and has a column for country, the total number of possible values is fewer than 200. Some of those values will be repeated many times. If the actual fact table has a hundred million rows, the country column can compress simply and need little or no storage. Row-by-row compression is not able to capitalize on the similarity of column values in this way and will use more bytes to compress the values in the country column [4], [3].

2.6.7 Column Elimination

- Columnstore indexes skip reading in columns that are not required for the query result. This ability, called column elimination, further reduces I/O for query execution and therefore improves query performance.
- Column elimination is possible because the data is organized and compressed column by column. In contrast, when data is stored row-by-row, the column values in each row are physically stored together and cannot be easily separated. The query processor needs to read in an entire row to retrieve specific column values, which increases I/O because extra data is unnecessarily read into memory.
- For example, if a table has 50 columns and the query only uses 5 of those columns, the column store index only fetches the 5 columns from disk. It skips reading in the other 45 columns. This reduces I/O by another 90% assuming all columns are of similar size. If the same data are stored in a row store, the query processor needs to read the additional 45 columns.

2.7 Functional Specification

This system will be used in two which are the Administrator and user. As all of these have different requirements the modules are designed to meet their needs and avoid any type of confusion. The uses of two user modules have been described below.

1) The administrator can do the following function
   - Manage user information.
   - Manage the Column Store Index
   - Add new Column Store Index
   - Keep Index Fragmentation levels in check
   - Monitor the Response Time of Queries
2) User can do the following function
   - Users can Access any reports with Minimum Response time
   - User can run multiple Analytical Inference Queries

2.8 Hardware and Software Requirements

2.8.1 Hardware Requirements:

• Processor	: Intel Pentium IV and higher
• Processor Speed	: 2.5 GHz and higher
• Memory Size	: 16GB and higher
• Hard Disk Drive	: 100GB and higher
• Pointing Device	: Scroll Mouse
• Keyboard	: 101Standard Keyboard

2.8.2 Software Requirements:

• Operating System	: Windows Server 2012 R2
• Core Language	: JAVA
• Front End	: JDK 1.6, Tableau
• Database	: SQL Server 2016

2.9Tools Survey

2.9.1Tableau

Tableau is one of the most fast-growing data visualization tools which is currently being used in the BI industry. It is the best way to change or transform the raw set of data into an easily understandable format with zero technical skills and coding knowledge.

Data visualization is important because we understand things that are visually well descriptive and interesting. So, working with data visualization tools like Tableau will help anyone understand data better, as it gives us access to the amount of data in easily digestible visuals. Also, well-designed data graphics is usually the simplest and the most powerful. Tableau is greatly used because data can be analyzed very quickly with Tableau. Also, visualizations are generated as dashboards and worksheets. Tableau allows us to create dashboards that provide actionable insights and drives the business forward. The products of Tableau always operate in virtualized environments when they are configured with the proper underlying operating system and hardware. Tableau is used to explore data with limitless visual analytics.

Advantages of Using Tableau

- **Fantastic Visualizations:** You can now work with a lot of data that doesn't have any order to it and create a range of visualizations. Well, thanks to the in-built features of Tableau which help you create visualizations that surely stand out of the crowd. You also have the option of switching between different visualizations to bring about a greater context, ways of drilling down data and exploring the data at a minute level.

- **In-depth Insights:** Tableau can help enterprises futuristically to analyze data without any specific goals in mind. You can explore visualizations and have a look at the same data from different angles. You can frame 'what if' queries and work with data by hypothetically visualizing it in a different manner and dynamically adding components for comparison and analysis. When you are working with real-time data, then these capabilities are highlighted in a huge manner.

- **User-friendly Approach:** This is the greatest strength of Tableau. It is built from the ground level for people who don't have any technical skills or coding experience. So, everything can be done with this tool by anybody without any prior set of skills. Since most of the features are in a drag-and-drop format, each visualization is so intuitive and self-depicting.
- **Working with Disparate Data Sources:** Tableau has a powerful reason to be included by various organizations in today's data-driven world where data can come from any point and any disparate sources. This is where Tableau has an edge over other Business Intelligence and Analytics tools. Tableau lets you work by connecting to various data sources, data warehouses, files that exist in the cloud, big data, data that exists in spreadsheets, and non-relational data, among other types of data. Tableau effortlessly blends all different types of data to help organizations come up with compelling visualizations.

- **Adding Data Sets**: Be it a database or an Excel workbook, with Tableau one can easily add new data sets that get automatically blended with Tableau using common fields.

- **Switching Between Visualizations:** You also have the option of switching between different visualizations to bring about a greater context, ways of drilling down data and exploring the data at a minute level.

Common uses of Tableau

- Tableau is a powerful and fastest growing data visualization tool used in the Business Intelligence Industry
- The Tableau Product Suite consists of
- Tableau Desktop
- Tableau Public
- Tableau Online
- Tableau Server and Tableau Reader
- Tableau Desktop has a rich feature set and allows you to code and customize reports
- In Tableau public, workbooks created cannot be saved locally, in turn, it should be saved to the Tableau's public cloud which can be viewed and accessed by anyone

- Tableau server is specifically used to share the workbooks, visualizations that are created in the Tableau Desktop application across the organization
- Tableau online has all the similar functionalities of the Tableau Server, but the data is stored on servers hosted in the cloud which are maintained by the Tableau group.
- Tableau Reader is a free tool that allows you to view the workbooks and visualizations created using Tableau Desktop or Tableau Public.
- Tableau connects and extracts the data stored in various places. It can pull data from any platform imaginable.
- The spreadsheet application used for manipulating the data while Tableau is a perfect visualization tool used for analysis

Characteristics Tableau:

- Real-time analysis
- Data Blending
- Collaboration of data

2.9.2Sql Server:

SQL Server is Microsoft's relational database management system (RDBMS). It is a full-featured database primarily designed to compete against competitors Oracle Database (DB) and MySQL.

Like all major RBDMS, SQL Server supports ANSI SQL, the standard SQL language. However, SQL Server also contains T-SQL, its own SQL implementation. SQL Server Management Studio (SSMS) (previously known as Enterprise Manager) is SQL Server's main interface tool, and it supports 32-bit and 64-bit environments.

SQL Server is sometimes referred to as MSSQL and Microsoft SQL Server.

Originally released in 1989 as version 1.0 by Microsoft, in conjunction with Sybase, SQL Server and its early versions were very similar to Sybase. However, the Microsoft-Sybase partnership dissolved in the early 1990s, and Microsoft retained the rights to the SQL Server trade name. Since then, Microsoft has released 2000, 2005 and 2008 versions, which feature more advanced options and better security.

Examples of some features include XML data type support, dynamic management views (DMVs), full-text search capability and database mirroring.

SQL Server is offered in several editions with the different feature set and pricing options to meet a variety of user needs, including the following:

- Enterprise: Designed for large enterprises with complex data requirements, data warehousing, and Web-enabled databases. It has all the features of SQL Server, and its license pricing is the most expensive.
- Standard: Targeted toward small and medium organizations. It also supports e-commerce and data warehousing.
- Workgroup: For small organizations. No size or user limits and may be used as the backend database for small Web servers or branch offices.
- Express: Free for distribution. Has the fewest number of features and limits database size and users. May be used as a replacement for an Access database.

SQL Architecture:

The protocol layer implements the external interface to SQL Server. All operations that can be invoked on SQL Server are communicated to it via a Microsoft-defined format, called Tabular Data Stream (TDS). TDS is an application layer protocol, used to transfer data between a database server and a client. Initially designed and developed by Sybase Inc. for their Sybase SQL Server. The relational database engine in 1984, and later by Microsoft in Microsoft SQL Server, TDS packets can be encased in other physical transport dependent protocols, TCP/IP, named pipes, and Shared Memory. Consequently, access to SQL Server is available over these protocols [3].

2.9.3JAVA:

Java is a programming language developed by James Gosling with other team members named Mike Sheridan and Patrick Naughton also called as Green Team in the year 1995. There are lots of applications and websites that will not work unless you have Java installed, and more are created every day. Java is fast, secure, and reliable.

History of Java:

The internet and the World Wide Web were starting to emerge in 1996 and Java was not originally designed with the internet in mind. Instead, Sun Microsystems engineers envisioned small, appliance-sized, interconnected devices that could communicate with each other.

As a result, the Java programming language paid more attention to the task of network programming than other competing languages. Through the java.net APIs, the Java programming language took large strides in simplifying the traditionally difficult task of programming across a network.

The first full increment of Java occurred on Jan. 23, 1996. The well-known JavaBeans interface was introduced in Java 1.1 in February 1997.

Later versions of Java releases have received nicknames, such as JDK 1.2 being referred to as Java 2. Java 2 saw considerable improvements to API collections, while Java 5 included significant changes to Java syntax through a new feature called Generics.

In October 2009, Google released the Android software developer's kit (SDK), a standard development kit that made it possible for mobile device developers to write applications for Android-based devices using Java APIs.

Oracle took over the Java platform when it acquired Sun Microsystems in January 2010. The acquisition delayed the release of Java 7, and Oracle scaled back some of the more ambitious plans for it.

Java 8 was released in March 2014. It included Lambda expressions, which are common features in many competing languages but had been absent in Java. With Lambda expressions, developers can write applications using a functional approach, as opposed to an object-oriented one.

March of 2018 saw the release of Java 10 followed by Java 11 in September 2018. Java 12 was released in March of 2019.

Java platforms:

There are three key platforms upon which programmers develop Java applications:

1. Java SE- Simple, stand-alone applications are developed using Java Standard Edition. Formerly known as J2SE, Java SE provides all of the API's needed to develop traditional desktop applications.
2. Java EE- the Java Enterprise Edition, formerly known as J2EE, provides the ability to create server-side components that can respond to a web-based request-response cycle. This arrangement allows the creation of Java programs that can interact with Internet-based clients, including web browsers, CORBA-based clients and even REST- and SOAP-based web services.
3. Java ME- Java also provides a lightweight platform for mobile development known as Java Micro Edition, formerly known as J2ME. Java ME has proved a prevalent platform for embedded device development, but it struggled to gain traction in the smartphone development arena.

Main Uses of Java:

It is easy for developers to write programs that employ popular software design patterns and best practices using the various components found in Java EE. For example, frameworks such as Struts and Java Server Faces all use a Java servlet to implement the front controller design pattern for centralizing requests.

A big part of the Java ecosystem is the large variety of open source and community built projects, software platforms, and APIs. For example, the Apache Foundation hosts a variety of projects written using Java, including simple logging frameworks for Java (SLF4J), both Yarn and Hadoop processing frameworks, Micro services development platforms and integration platforms.

Java EE environments can be used in the cloud as well. Developers can build, deploy, debug and monitor Java applications on Google Cloud at a scalable level.

In terms of mobile development, Java is commonly used as the programming language for Android applications. Java tends to be preferred by Android developers because of Java's security, object-oriented paradigms, regularly updated and maintained feature sets, use of JVM and frameworks for networking, IO and threading.

Although Java is widely used, it still has fair criticisms. Java syntax is often criticized for being too verbose. In response, several peripheral languages have emerged to address these issues, including Groovy. Due to the way Java references objects internally, complex and concurrent list-based operations slow the JVM. The Scala language addresses many of the shortcomings of the Java language that reduces its ability to scale.

2.9.4NET

The .NET is 4the technology from Microsoft, on which all other Microsoft technologies will be depending on in the future. It is a major technology change, introduced by Microsoft, to catch the market from the SUN's Java. A few years back, Microsoft had only VC++ and VB to compete with Java, but Java was catching the market very fast. With the world depending more and more on the Internet/ Web and java related tools becoming the best choice for web applications, Microsoft seemed to be losing the battle. Thousands of programmers moved to java from VC++ and VB. To recover the market. Microsoft announced .NET.

But Microsoft has a wonderful history of starting late but catching up quickly. This is true in the case of .NET too. Microsoft put their best men at work for a secret project called Next Generation Windows Services (NGWS). Under the direct supervision of Mr. Bill Gates.

The outcome of the project is what we now know as .NET. Even though .NET has borrowed most of its ideas from Sun's J2EE, it has really outperformed its competitors.

Microsoft's VC++ was a powerful tool. But it was too complex. It has too many data types, and developers had to learn many libraries including Windows SDK, MFC, ATL, COM, etc. There were many data type compatibility issues while exchanging data between different layers. Visual Basic was too easy, and many serious programmers hated it just for that reason. Even though Visual basic was very easy to use, it was not very flexible to develop serious applications. SUN's Java became a very good choice for these reasons. It had the flexibility and power of C++ and at the same time easy enough to catch the attention of VB programmers.

Microsoft recognized these factors and they introduced the .NET considering all these factors. All unwanted complexities are eliminated and a pure object-oriented programming model was introduced. This makes the programmer's life very easy.

.NET is said to be the Microsoft development model in which software becomes platform and device independent and data becomes available over the internet. Due to this vision Microsoft .NET is also called Microsoft strategy for connecting systems, information and devices through web services so people can collaborate and communicates effectively.

The .NET PLATFORM

The .NET platform is a set of technologies. Microsoft .NET platform simplifies software development (Windows or WEB) by building applications of XML Web services.

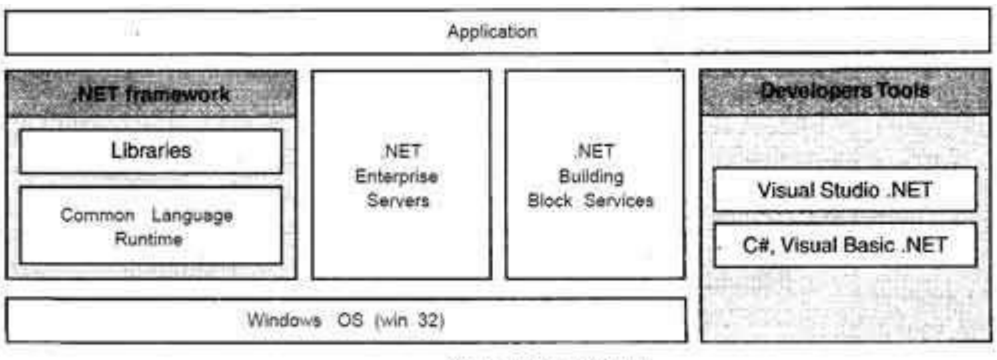
The the.net platform consists of the following core technologies:

• The .NET Framework

• The .NET Enterprise Servers

• Building block services

• Visual Studio .NET



The .NET platform

Fig 2.6 .NET Platform

CHAPTER – 3

DESIGN SPECIFICATION

3.1 Modular Function Diagram

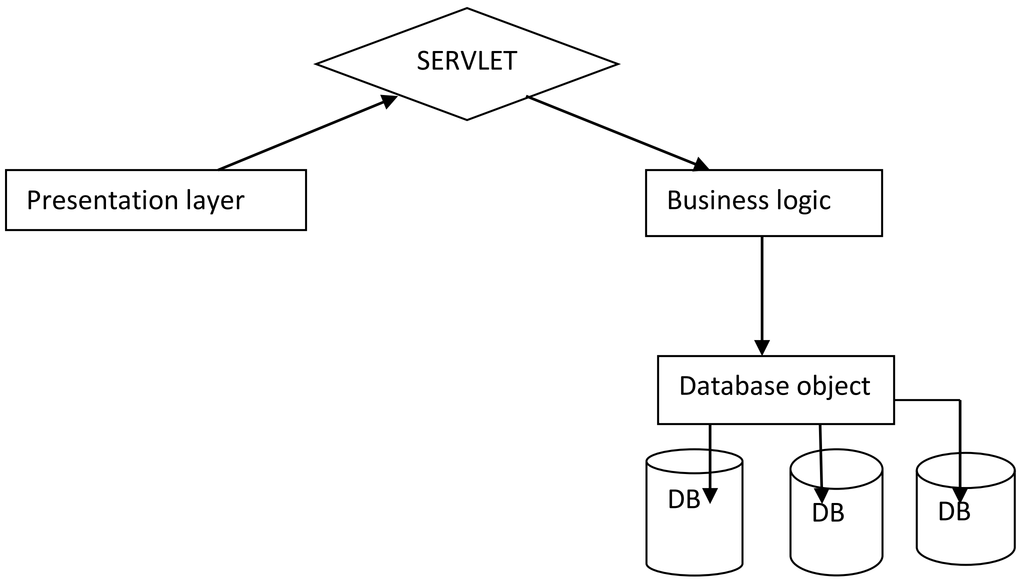


Fig 3.1 Modular Function Diagram

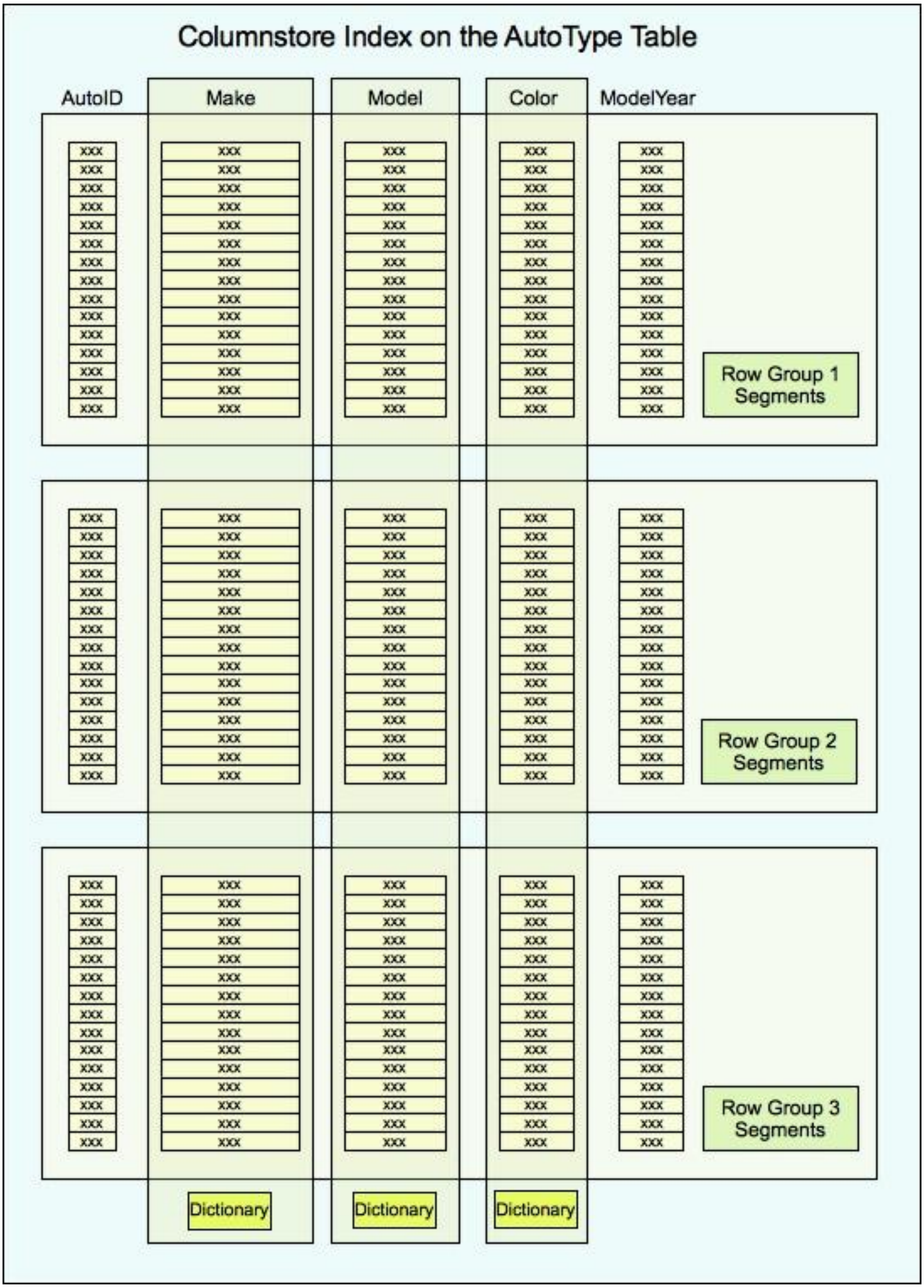


Fig 3.2 Modular Function Table

3.3 Specification of Function Modules

Module Design:

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement has been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

NUMBER OF MODULES

The system after a careful analysis has been identified to be presented with the functions of the following module:

1. **Admin**
2. **User**
3. **Database Objects**
4. **Columnstore Indexes**
5. **Data Analysis Reports**

3.4 System Design

3.4.1 Data/Control Flow Diagrams

A data flow diagram is a graphical tool used to describe and analyze the movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations.

Types of Data Flow Diagrams

1. Current Physical
2. Current Logical
3. New Logical
4. New Physical

Context

1-Level DFD:

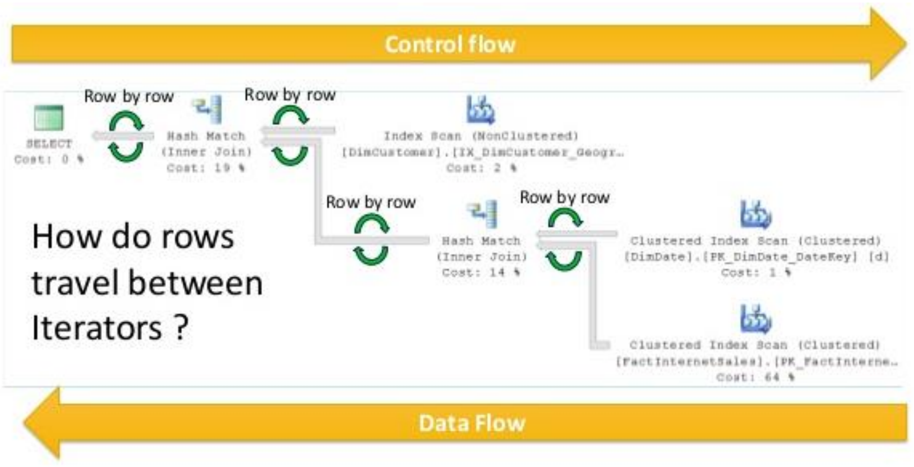

Control flow
Row by row
Row by row
SELECT
Cost: 0 %
Hash Match
(Inner Join)
Cost: 19 %
Index Scan (NonClustered)
[DimCustomer].[IX_DimCustomer_Geogr…
Cost: 2 %
Row by row
Row by row
How do rows travel between Iterators ?
Hash Match
(Inner Join)
Cost: 14 %
Clustered Index Scan (Clustered)
[DimDate].[PK_DimDate_DateKey] [d]
Cost: 1 %
Clustered Index Scan (Clustered)
[FactInternetSales].[PK_FactInterne…
Cost: 64 %
Data Flow

Fig 3.3 Data Flow Diagram

3.4.2 Use Case Diagram

Use Case diagrams identify the functionality provided by the system (use cases), the users who interact with the system (actors), and the association between the users and the functionality. Use Cases are used in the Analysis phase of software development to articulate the high-level requirements of the system. A single use case diagram captures a particular functionality of a system. So to model the entire system numbers of use case diagrams are used. So in brief, the purposes of use case diagrams can be as follows:

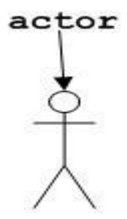
- Used to gather the requirements of a system.
- Used to get an outside view of a system.
- Identify external and internal factors influencing the system.
- Show the interacting among the requirements are actors.

Graphical Notation

The basic components of Use Case diagrams are the Actor, the Use Case, and the Association.

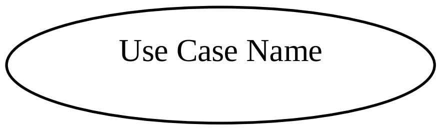
Actor

An Actor, as mentioned, is a user of the system, and is depicted using a stick figure. The role of the user is written beneath the icon. Actors are not limited to humans. If a system communicates with another application, and expects input or delivers output, then that application can also be considered an actor.



Use Case

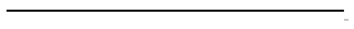
A Use Case is a functionality provided by the system, typically described as verb + object (e.g. Register Car, Delete User). Use Cases are depicted with an ellipse. The name of the use case is written within the ellipse



Use Case

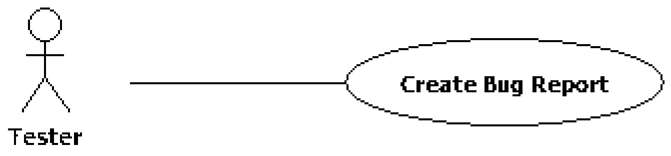
Association

Associations are used to link Actors with Use Cases and indicate that an Actor participates in the Use Case in some form. Associations are depicted by a line connecting the Actor and the Use Case.



Association

The following image shows how these three basic elements work together to form a use case diagram.



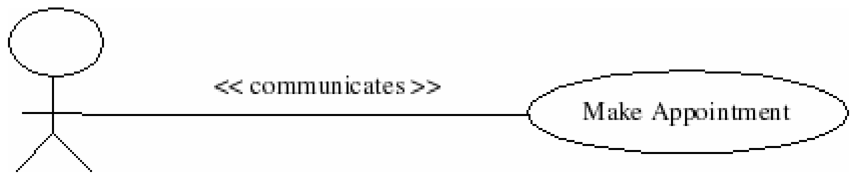
Use Case example

Use case diagrams describe what a system does from the standpoint of an external observer. The emphasis is on *what* a system does rather than *how.*

Relationships in Use Case

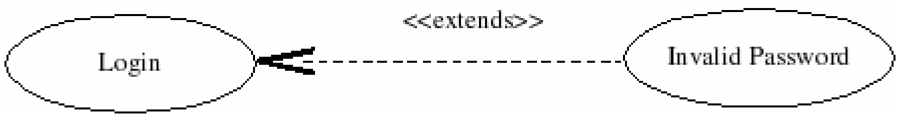
Use cases share different kinds of relationships. A relationship between two use cases is basically a dependency between the two use cases. Defining the relationship between two use cases is the decision of the modeller of the use case diagram. This use of an existing use case using different types of relationships reduces the overall effort required in defining use cases in a system. Use case relationships can be one of the following:

- **Communicates:** The participation of an actor in a use case is shown by connecting the actor symbol to use case symbol by a solid path. The actor is said to 'communicate' with the use case. This is the only relation between an actor and a use case. See figure 2.20.



Communicates relationship

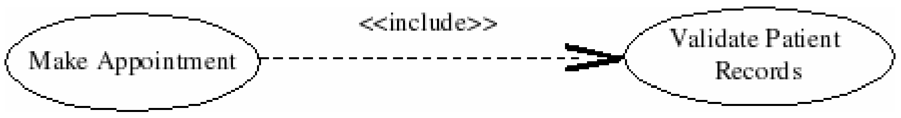
- **Extends** In an extend*e*d relationship between two use cases, the child use case adds to the existing functionality and characteristics of the parent use case. An extend relationship is depicted with a directed arrow having a dotted shaft, similar to the include relationship. The tip of the arrowhead points to the parent use case and the child use case is connected at the base of the arrow. The stereotype “<extends>” identifies as an extend relationship as shown in figure 2.21.



An example of an extend relationship

For example, validating the user for a system. An invalid password is an extension of validating password use case as shown in figure 3.5.

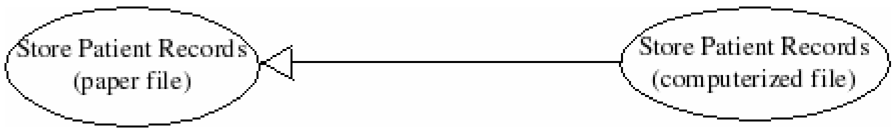
Include or uses: When a use case is depicted as using the functionality of another use case, this relationship between the use cases is named as include or uses relationship. In other words, in an include relationship, a use case includes the functionality described in the use case as a part of its business process flow. A uses relationship from use case A to use case B indicates that an instance of the use case will also include the behaviour as specified by B. An include relationship is depicted with a directed arrow having a dotted shaft. The tip of arrowhead points to the child use case and the parent use case connected at the base of the arrow. The stereotype "<include>" identifies the relationship as an include relationship.



An example of an include relationship

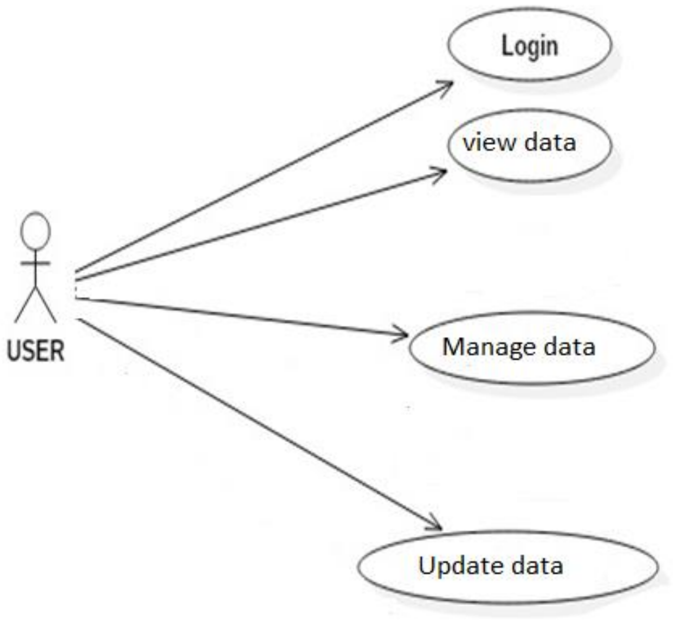
The system boundary is potentially the entire system as defined in the requirements document. For large and complex systems, each module may be the system boundary. For example, for an ERP system for an organization, each of the modules such as personal, payroll, accounting, etc. can form a system boundary for use cases specific to each of these business functions. The entire system can span all of these modules depicting the overall system boundary.

Generalization: A generalization relationship is also a parent-child relationship between use cases. The child use case in the generalization relationship has the underlying business process meaning but is an enhancement of the parent use case. In a use case diagram, generalization is shown as a directed arrow with a triangle arrowhead. The child use case is connected at the base of the arrow. The tip of the arrow is connected to the parent use case.


Store Patient Records (paper file)
Store Patient Records (computerized file)

An example of a generalization relationship

The Use Case Diagram



3.4.3 Class Diagram

The class diagram is a static diagram. It represents the static view of an application. The class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing an executable code of the software application.

The class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object-oriented systems because they are the only UML diagrams that can be mapped directly with object-oriented languages.

The class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

Graphical Notation

UML *class* is represented by the diagram shown below. The diagram is divided into four parts.

- The top section is used to name the class.
- The second one is used to show the attributes of the class.
- The third section is used to describe the operations performed by the class.
- The fourth section is optional to show any additional components.

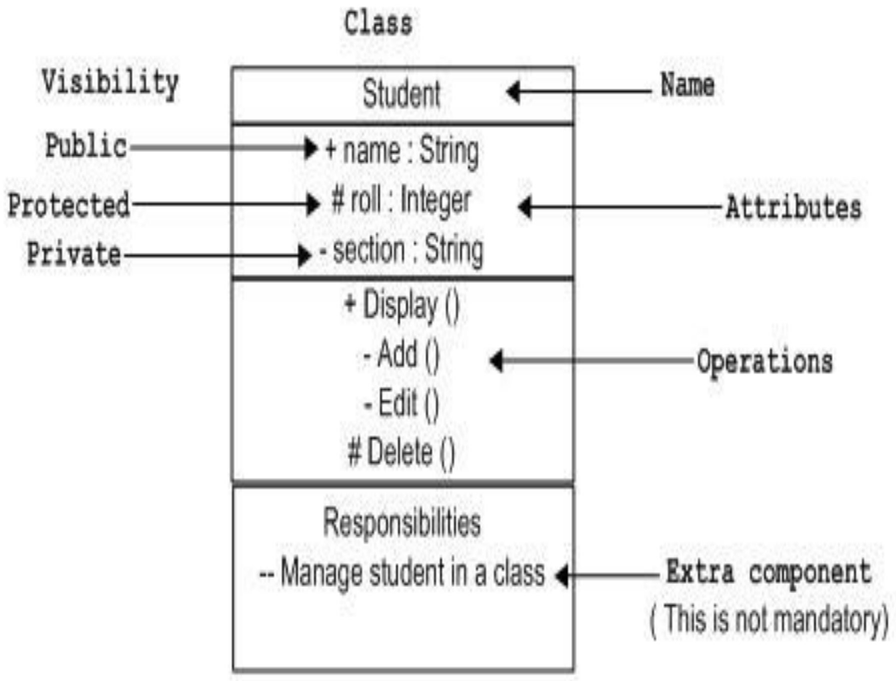


Fig No. 3.4 Class diagram representation

Classes are used to represent objects. Objects can be anything having properties and responsibilities.

Relationships in Class Diagram

The Class diagram has three kinds of relationships.

- **Association** -- a relationship between instances of the two classes. There is an association between two classes if an instance of one class must know about the other in order to perform its work. In a diagram, an association is a link connecting two classes.
- **Aggregation** -- an association in which one class belongs to a collection. An aggregation has a diamond end pointing to the part containing the whole.
- **Generalization** -- an inheritance link indicating one class is a superclass of the other. A generalization has a triangle pointing to the superclass.

An association has two ends. An end may have a role name to clarify the nature of the association. For example, an Order Detail is a line item of each Order. A navigability arrow on an association shows which direction the association can be traversed or queried. Associations with no navigability arrows are bi-directional. The multiplicity of an association end is the number of possible instances of the class associated with a single instance of the other end. Multiplicities are single numbers or ranges of numbers. This table gives the most common multiplicities.

Multiplicities	**Meaning**
0..1	Zero or one instance. The notation ***n... M*** indicates ***n*** to***m*** instances.
0..* *or* *****	No limit on the number of instances (including none).
1	exactly one instance
1..*	at least one instance

Class diagram multiplicity

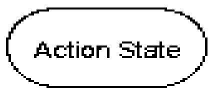
Every class diagram has classes, associations, and multiplicities. Navigability and roles are optional items placed in a diagram to provide clarity.

3.4.4 Activity Diagram

Activity diagrams are used to document workflows in a system, from the business level down to the operational level. When looking at an Activity diagram, you'll notice elements from State diagrams. In fact, the Activity diagram is a variation of the state diagram where the "states" represent operations, and the transitions represent the activities that happen when the operation is complete. The general purpose of Activity diagrams is to focus on flows driven by internal processing vs. external events.

ACTIVITY STATES

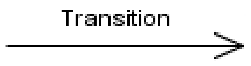
Activity states mark an action by an object. The notations for these states are rounded rectangles, the same notation as found in State chart diagrams.



Notation of activity state

TRANSITION

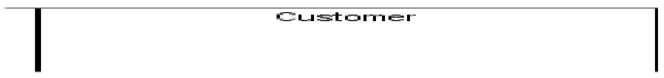
When an Activity State is completed, processing moves to another Activity State. Transitions are used to mark this movement. Transitions are modeled using arrows.



Notation of transition

SWIM LANE

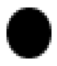
Swim lanes divide activities according to objects by arranging objects in column format and placing activities by that object within that column. Objects are listed at the top of the column, and vertical bars separate the columns to form the swim lanes.



Notation of swim lane

INITIAL STATE

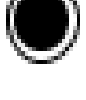
The Initial State marks the entry point and the initial Activity State. The notation for the Initial State is the same as in State chart diagrams, a solid circle. There can only be one Initial State on a diagram.



Notation of initial state

FINAL STATE

The final State mark the end of the modeled workflow. There can be the multiple Final States on a diagram, and these states are modeled using a solid circle surrounded by another circle.



Notation of final state

SYNCHRONIZATION BAR

Activities often can be done in parallel. To split processing ("fork"), or to resume processing when multiple activities have been completed ("join"), Synchronization Bars are used. These are modeled as solid rectangles, with multiple transitions going in and/or out.

Notation of synchronization bar

Activity diagrams are mainly used as a flow chart consists of activities performed by the system. But the activity diagram is not exactly a flow chart as they have some additional capabilities. These additional capabilities include branching, parallel flow, swim lane, etc. Before drawing an activity diagram we must have a clear understanding of the elements used in the activity diagram. The main element of an activity diagram is the activity itself. An activity is a function performed by the system. After identifying the activities, we need to understand how they are associated with constraints and conditions. So before drawing an activity diagram, we should identify the following elements:

- Activities
- Association
- Conditions
- Constraints

The Activity Diagram for Login

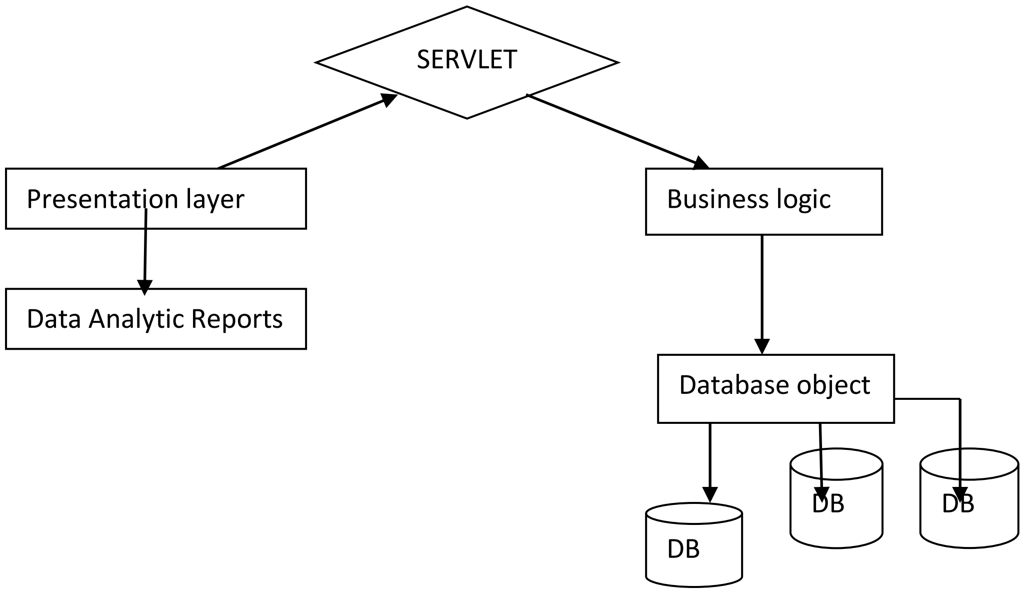


Fig No. 3.5 Activity Diagram for Online Login

3.4.5 Activity Diagram for User

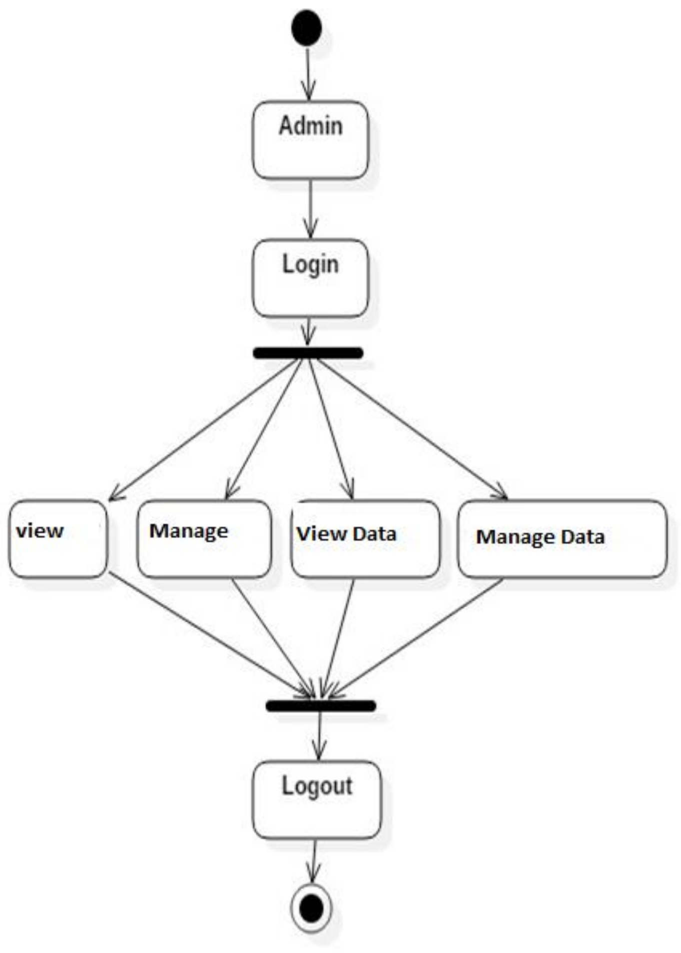


Fig No. 3.6 Activity Diagram for Online User

3.4.6Sequence Diagram

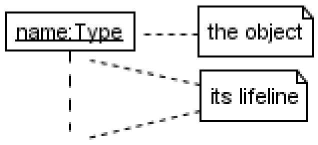
A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart.

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines (*lifelines*), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

Objects, as well as classes, can be targets on a sequence diagram, which means that messages can be sent to them. A target is displayed as a rectangle with some text in it. Below the target, its lifeline extends for as long as the target exists. The lifeline is displayed as a vertical dashed line.

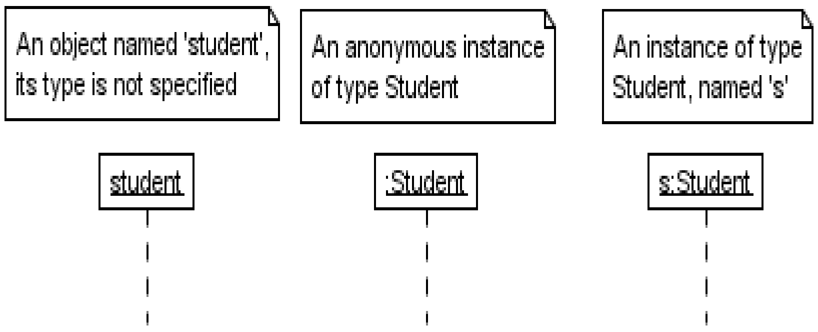
Object



Basic notation for an object

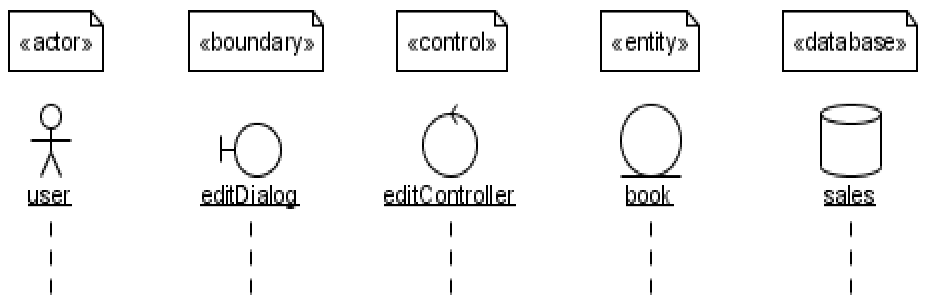
Where 'name' is the name of the object in the context of the diagram and 'Type' indicates the type of which the object is an instance. Note that the object doesn't have to be a *direct* instance of Type, a type of which it is an *indirect* instance is possible too. So 'Type' can be an abstract type as well.

Both name and type are optional, but at least one of them should be present. Some **example**:


An object named 'student', its type is not specified
An anonymous instance of type Student
An instance of type Student, named 's'
student
:Student
s:Student

Example 1

As with any UML-element, you can add a stereotype to a target. Some often used stereotypes for objects are «actor», «boundary», «control», «entity» and «database». They can be displayed with icons as well:



Example 2

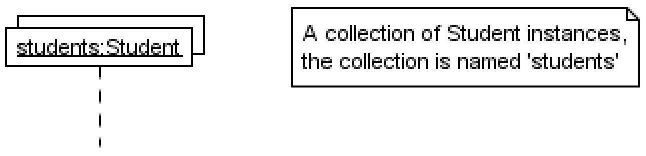
An object should be named only if at least one of the following applies

- You want to refer to it during the interaction as a message parameter or return value
- You don't mention its type
- There are other anonymous objects of the same type and giving them names is the only way to differentiate them

Try to avoid long but non-descriptive names when you're also specifying the type of the object (e.g. don't use 'aStudent' for an instance of type Student). A shorter name carries the same amount of information and doesn't clutter the diagram (e.g. use’s’ instead).

Multi-Object

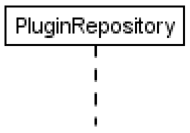
When you want to show how a client interacts with the elements of a collection, you can use a multi-object.


students:Student
A collection of Student instances, the collection is named 'students'

Basic notation of multi-object

Again, a name and/or type can be specified. Note however that the 'Type' part designates the type of the elements and not the type of the collection itself.

Class



Basic notation for a class

Only class messages (e.g. shared or static methods in some programming languages) can be sent to a class. Note that the text of a class is not underlined, which is how you can distinguish it from an object.

3.4.7 Sequence Diagram For Admin

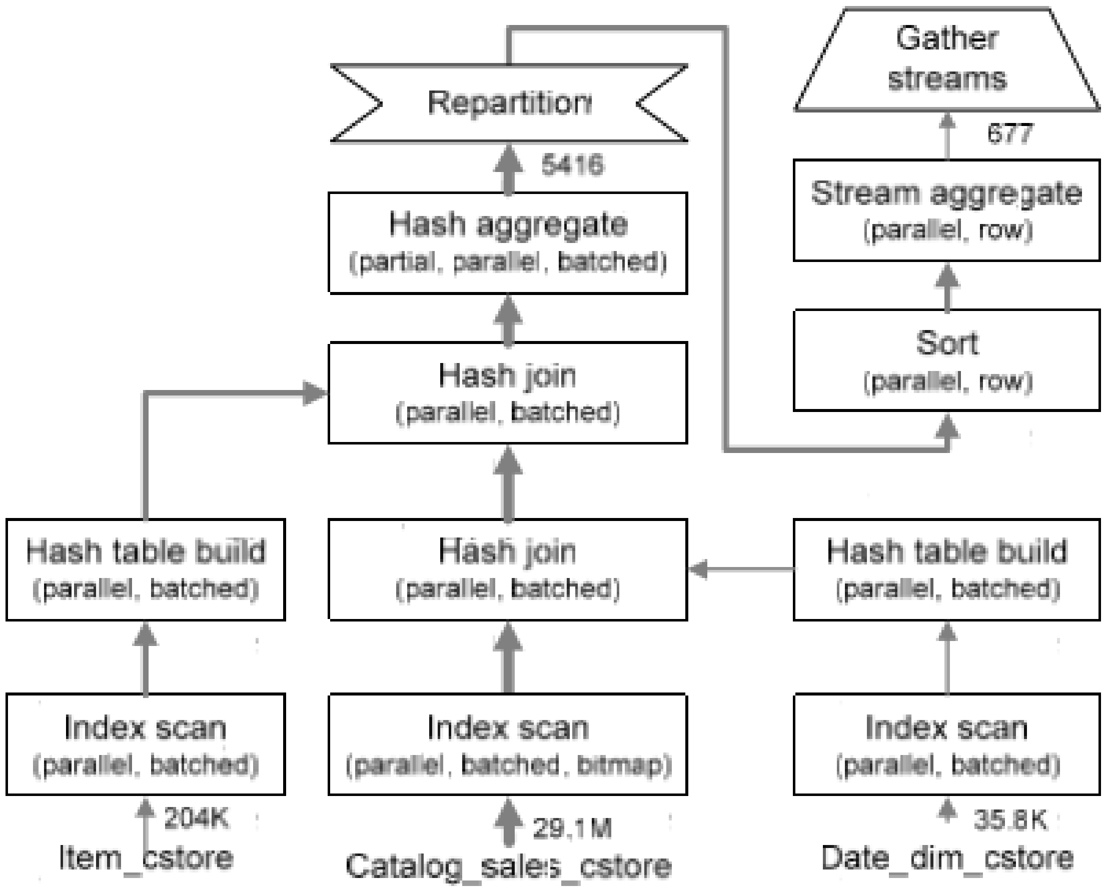

Gather streams
677
Stream aggregate (parallel, row)
Sort (parallel, row)
Repartition
5416
Hash aggregate (partial, parallel, batched)
Hash join (parallel, batched)
Hash table build (parallel, batched)
Hash join (parallel, batched)
Hash table build (parallel, batched)
Index scan (parallel, batched)
Index scan (parallel, batched, bitmap)
Index scan (parallel, batched)
204K
Item_cstore
29.1M
Catalog_sales_cstore
35.8K
Date_dim_cstore

Fig 3.7Sequence Diagram for Admin

CHAPTER-4

IMPLEMENTATION DETAILS

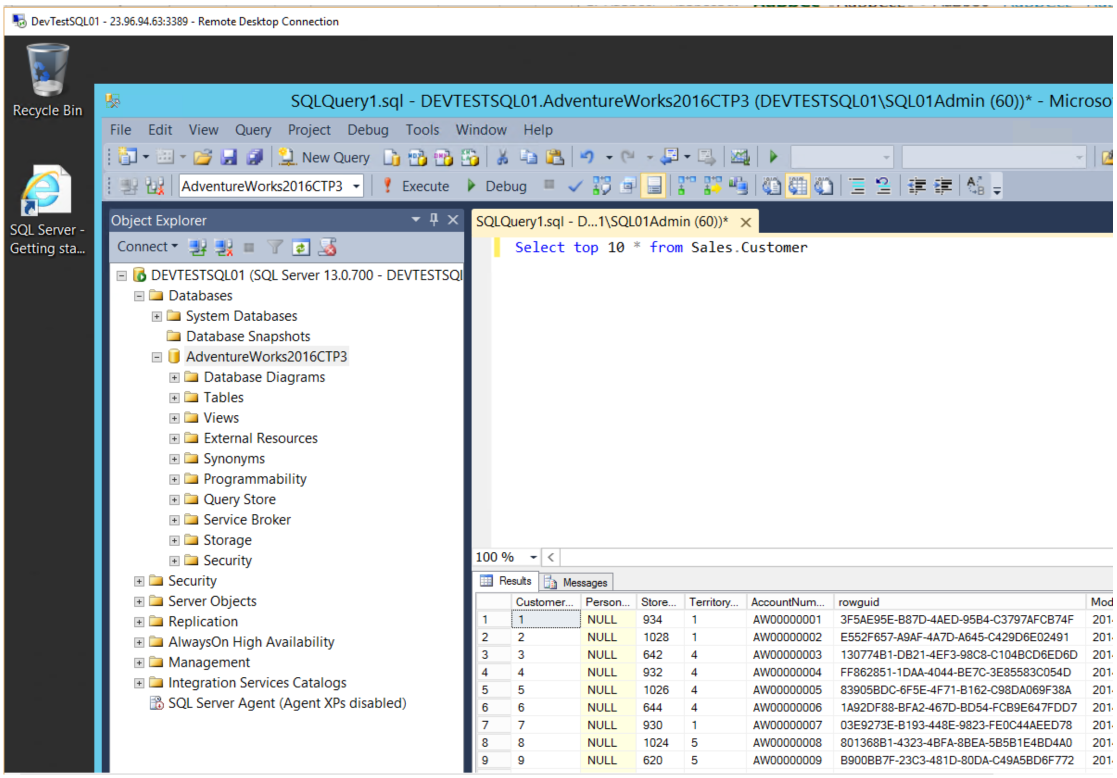
4.1Sample Code

```
--Creating disk-based Orders Table
create table orders
(
AccountKeyint not null,
customernamenvarchar (50),
OrderNumberbigint,
PurchasePrice decimal (9,2),
OrderStatussmallint not NULL,
OrderStatusDescnvarchar (50)
)
```

Create a Clustered index on 'OrderStatus'

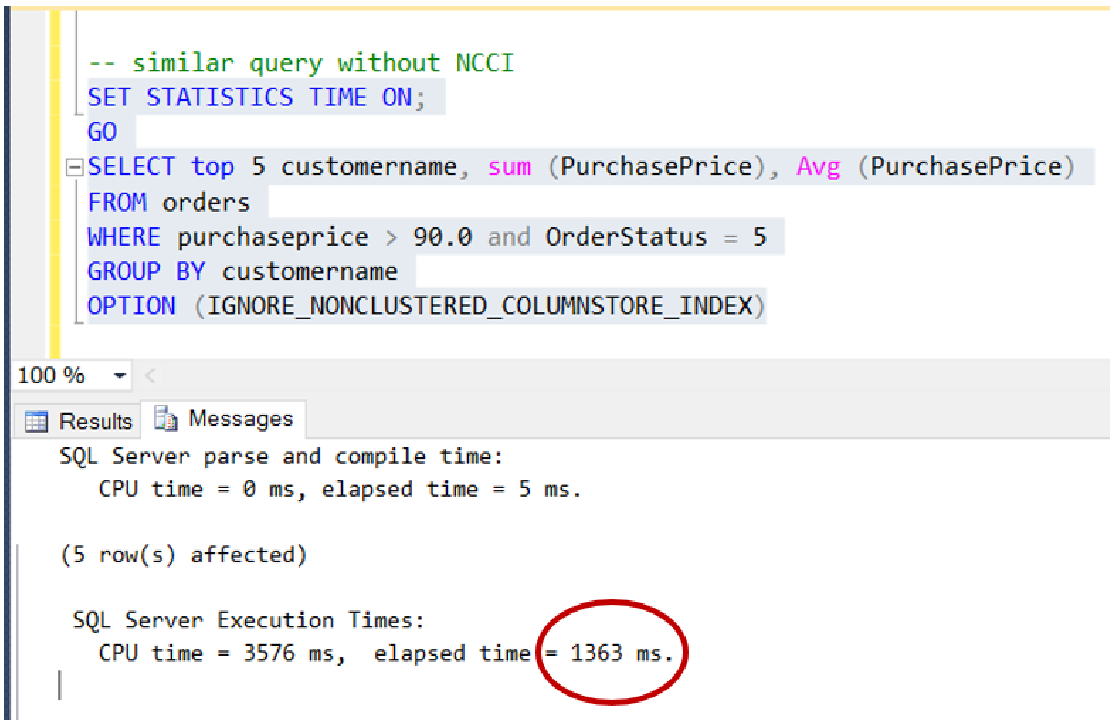
```
--Creating Clustered Index on OrderStatus
create
clustered
indexorders_ci on orders(OrderStatus)
declare @outerloopint = 0
declare @iint = 0
declare @purchaseprice decimal (9,2)
declare @customernamenvarchar (50)
declare @accountkeyint
declare @orderstatussmallint
declare @orderstatusdescnvarchar(50)
declare @ordernumberbigint

while (@outerloop< 3000000)

begin

Select @i = 0

begintran

while (@i< 2000)

begin

set @ordernumber = @outerloop + @i

set @purchaseprice = rand() * 1000.0

set @accountkey = convert (int, RAND ()*1000)

set @orderstatus = convert (smallint, RAND()*100)

if (@orderstatus>= 5)

set @orderstatus = 5

set @orderstatusdesc =

case @orderstatus

WHEN 0 THEN 'Order Started'

WHEN 1 THEN 'Order Closed'

WHEN 2 THEN 'Order Paid'

WHEN 3 THEN 'Order Fullfillment'

WHEN 4 THEN 'Order Shipped'

WHEN 5 THEN 'Order Received'

END
```

```
insert orders values (@accountkey, (convert (varchar (6), @accountkey) + 'firstname'),@ordernumber, @purchaseprice,@orderstatus, @orderstatusdesc)

set @i += 1;

end

commit

set @outerloop = @outerloop + 2000

set @i = 0

end

go
```



```
Select top 10 * from Sales.Customer
```

	Customer...	Person...	Store...	Territory...	AccountNum...	rowguid	Mod
1	1	NULL	934	1	AW00000001	3F5AE95E-B87D-4AED-95B4-C3797AFCB74F	201
2	2	NULL	1028	1	AW00000002	E552F657-A9AF-4A7D-A645-C429D6E02491	201
3	3	NULL	642	4	AW00000003	130774B1-DB21-4EF3-98C8-C104BCD6ED6D	201
4	4	NULL	932	4	AW00000004	FF862851-1DAA-4044-BE7C-3E85583C054D	201
5	5	NULL	1026	4	AW00000005	83905BDC-6F5E-4F71-B162-C98DA069F38A	201
6	6	NULL	644	4	AW00000006	1A92DF88-BFA2-467D-BD54-FCB9E647FDD7	201
7	7	NULL	930	1	AW00000007	03E9273E-B193-448E-9823-FE0C44AEED78	201
8	8	NULL	1024	5	AW00000008	801368B1-4323-4BFA-8BEA-5B5B1E4BD4A0	201
9	9	NULL	620	5	AW00000009	B900BB7F-23C3-481D-80DA-C49A5BD6F772	201



```
-- similar query without NCCI
SET STATISTICS TIME ON;
GO
SELECT top 5 customername, sum (PurchasePrice), Avg (PurchasePrice)
FROM orders
WHERE purchaseprice > 90.0 and OrderStatus = 5
GROUP BY customername
OPTION (IGNORE_NONCLUSTERED_COLUMNSTORE_INDEX)
```

```
SQL Server parse and compile time:
   CPU time = 0 ms, elapsed time = 5 ms.

(5 row(s) affected)

 SQL Server Execution Times:
   CPU time = 3576 ms,  elapsed time = 1363 ms.
```

CREATE NONCLUSTERED COLUMNSTORE INDEX orders_ncci ON orders (accountkey, customername, purchaseprice, orderstatus, orderstatusdesc)

declare @outerloopint = 3000000

declare @iint = 0

declare @purchaseprice decimal (9,2)

declare @customernamenvarchar (50)

declare @accountkeyint

declare @orderstatussmallint

declare @orderstatusdescnvarchar(50)

declare @ordernumberbigint

while (@outerloop< 3200000)

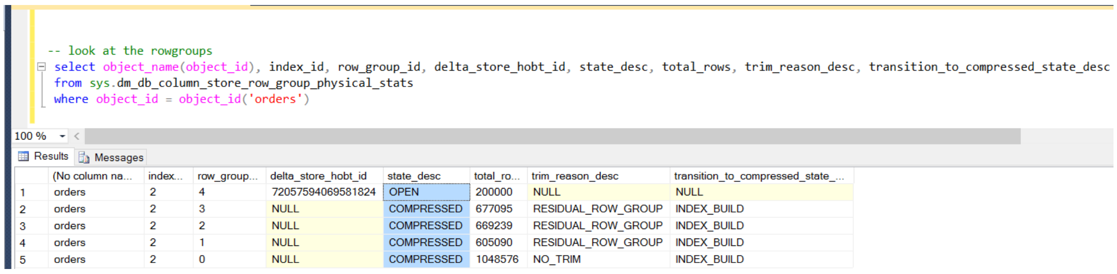
begin

Select @i = 0

begintran

```
while (@i< 2000)

begin

set @ordernumber = @outerloop + @i

set @purchaseprice = rand() * 1000.0

set @accountkey = convert (int, RAND ()*1000)

set @orderstatus = convert (smallint, RAND()*5)

set @orderstatusdesc =

case @orderstatus

WHEN 0 THEN 'Order Started'

WHEN 1 THEN 'Order Closed'

WHEN 2 THEN 'Order Paid'

WHEN 3 THEN 'Order Fulfillment'

WHEN 4 THEN 'Order Shipped'

WHEN 5 THEN 'Order Received'

END

insert orders values (@accountkey, (convert(varchar(6), @accountkey) + 'firstname'),
@ordernumber, @purchaseprice, @orderstatus, @orderstatusdesc)

set @i += 1;

end

commit

set @outerloop = @outerloop + 2000

set @i = 0

end
```

go

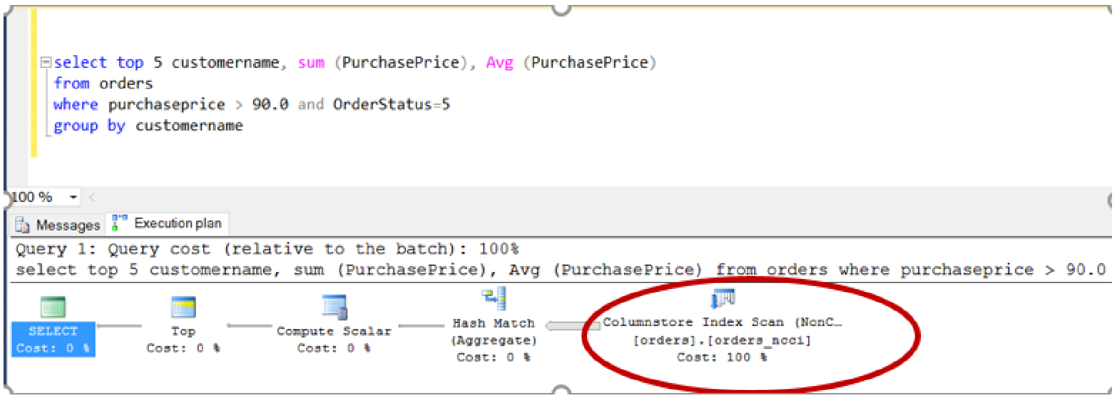


```sql
-- look at the rowgroups
select object_name(object_id), index_id, row_group_id, delta_store_hobt_id, state_desc, total_rows, trim_reason_desc, transition_to_compressed_state_desc
from sys.dm_db_column_store_row_group_physical_stats
where object_id = object_id('orders')
```

100 %

Results | Messages

	(No column na...	index...	row_group...	delta_store_hobt_id	state_desc	total_ro...	trim_reason_desc	transition_to_compressed_state_...
1	orders	2	4	72057594069581824	OPEN	200000	NULL	NULL
2	orders	2	3	NULL	COMPRESSED	677095	RESIDUAL_ROW_GROUP	INDEX_BUILD
3	orders	2	2	NULL	COMPRESSED	669239	RESIDUAL_ROW_GROUP	INDEX_BUILD
4	orders	2	1	NULL	COMPRESSED	605090	RESIDUAL_ROW_GROUP	INDEX_BUILD
5	orders	2	0	NULL	COMPRESSED	1048576	NO_TRIM	INDEX_BUILD

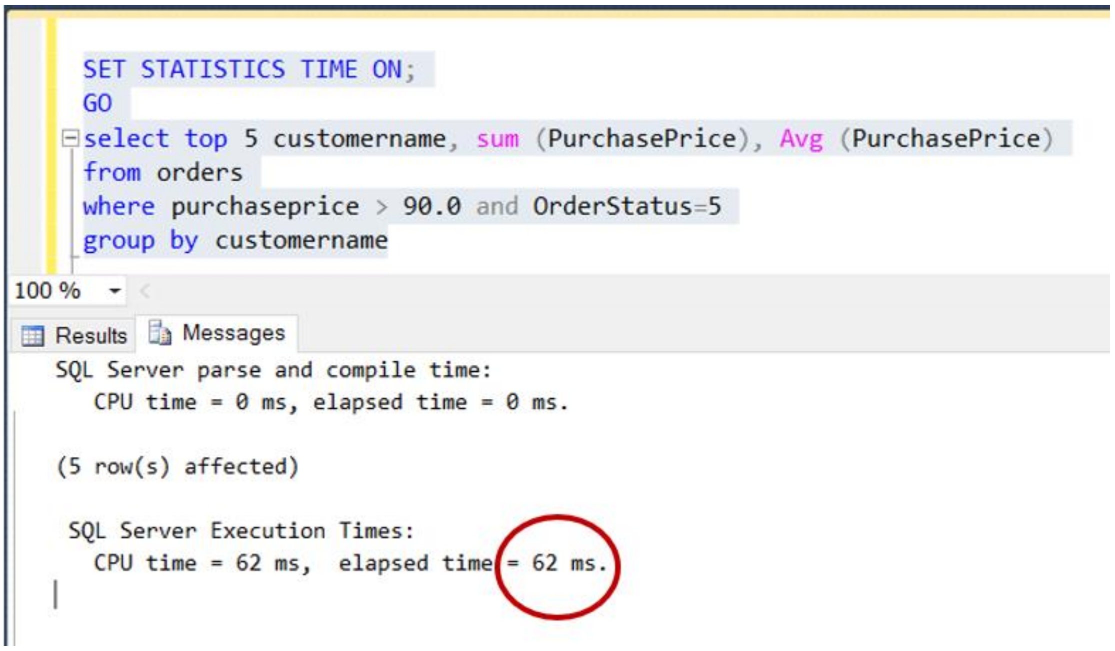


```sql
select top 5 customername, sum (PurchasePrice), Avg (PurchasePrice)
from orders
where purchaseprice > 90.0 and OrderStatus=5
group by customername
```

100 %

Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
select top 5 customername, sum (PurchasePrice), Avg (PurchasePrice) from orders where purchaseprice > 90.0
```



```sql
SET STATISTICS TIME ON;
GO
select top 5 customername, sum (PurchasePrice), Avg (PurchasePrice)
from orders
where purchaseprice > 90.0 and OrderStatus=5
group by customername
```

100 %

Results | Messages

```
SQL Server parse and compile time:
   CPU time = 0 ms, elapsed time = 0 ms.

(5 row(s) affected)

 SQL Server Execution Times:
   CPU time = 62 ms,  elapsed time = 62 ms.
```

--Using a filtered condition to separate hot data in a row store table

-- From "warm" data in a Columnstore index.

CREATE TABLE

orders_filteredNCCI (

```
AccountKeyint not null,

Customernamenvarchar (50),

OrderNumberbigint,

PurchasePrice        decimal (9,2),

OrderStatussmallint not null,

OrderStatusDescnvarchar (50))

-- OrderStatusDesc

-- 0 => 'Order Started'

-- 1 => 'Order Closed'

-- 2 => 'Order Paid'

-- 3 => 'Order Fulfillment Wait'

-- 4 => 'Order Shipped'

-- 5 => 'Order Received'

--Create the Columnstore index with a filtered condition

CREATE NONCLUSTERED COLUMNSTORE INDEX orders_ncci_filteredindex ON
orders_filteredNCCI (accountkey, customername, purchaseprice, orderstatus)

whereorderstatus = 5

--Create Orders Tables

CREATE TABLE orders_filteredNCCI (

AccountKeyint not null,

Customernamenvarchar (50),

OrderNumberbigint,

PurchasePrice        decimal (9,2),
OrderStatussmallint not null,

OrderStatusDescnvarchar (50))

-- OrderStatusDesc

-- 0 => 'Order Started'

-- 1 => 'Order Closed'

-- 2 => 'Order Paid'

-- 3 => 'Order Fulfillment Wait'

-- 4 => 'Order Shipped'

-- 5 => 'Order Received'

--Creating Btree Indexes

CREATE CLUSTERED INDEX orders_ci_filteredindex ON orders_filteredNCCI(OrderStatus)

-- Loading 3 million rows with the data pattern that 95% of the orders have already been received by
the customer.

declare @outerloopint = 0

declare @iint = 0

declare @purchaseprice decimal (9,2)

declare @customernamenvarchar (50)

declare @accountkeyint

declare @orderstatussmallint

declare @orderstatusdescnvarchar(50)

declare @ordernumberbigint

while (@outerloop< 3000000)

begin
```

```
Select @i = 0

begin

tran

while (@i< 2000)

begin

set @ordernumber = @outerloop + @i

set @purchaseprice = rand() * 1000.0

set @accountkey = convert (int, RAND ()*1000)

set @orderstatus = convert (smallint, RAND()*100)

if (@orderstatus>= 5)

set @orderstatus = 5

set @orderstatusdesc =

case @orderstatus

WHEN 0 THEN 'Order Started'

WHEN 1 THEN 'Order Closed'

WHEN 2 THEN 'Order Paid'

WHEN 3 THEN 'Order Fulfillment'

WHEN 4 THEN 'Order Shipped'

WHEN 5 THEN 'Order Received'

END

insert orders values (@accountkey, (convert (varchar (6), @accountkey) +
'firstname'),@ordernumber, @purchaseprice,@orderstatus, @orderstatusdesc)

set @i += 1;
```

end

commit

--Create the columnstore index with a filtered condition

CREATE NONCLUSTERED COLUMNSTORE INDEX orders_ncci_filteredindex ON orders_filteredNCCI (accountkey, customername, purchaseprice, orderstatus)

whereorderstatus = 5

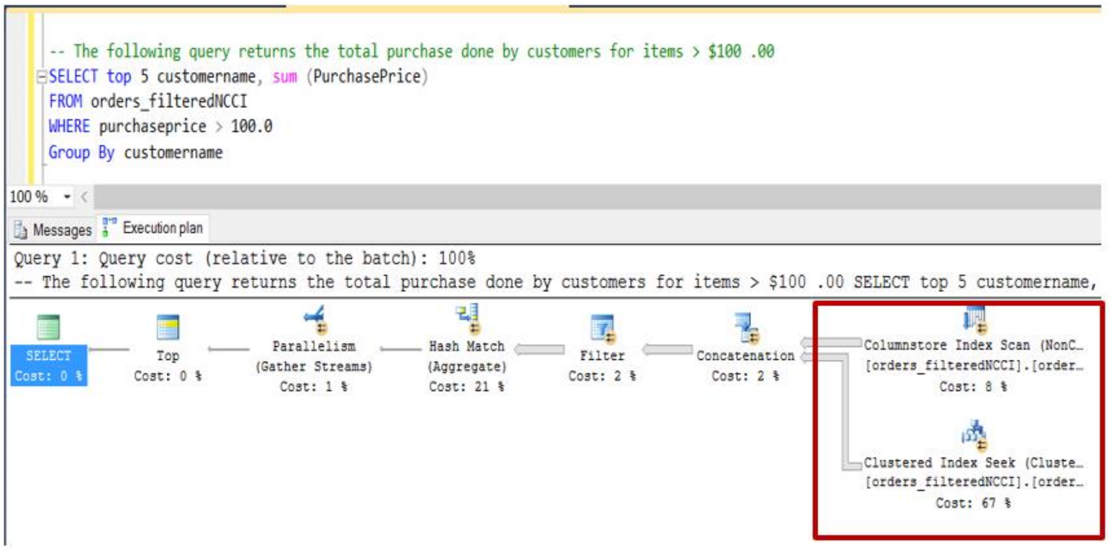
-- The following query returns the total purchase done by customers for items > $100 .00

SELECT top 5 customername, sum (PurchasePrice)

FROM orders_filteredNCCI

WHERE purchaseprice> 100.0

Group By customername

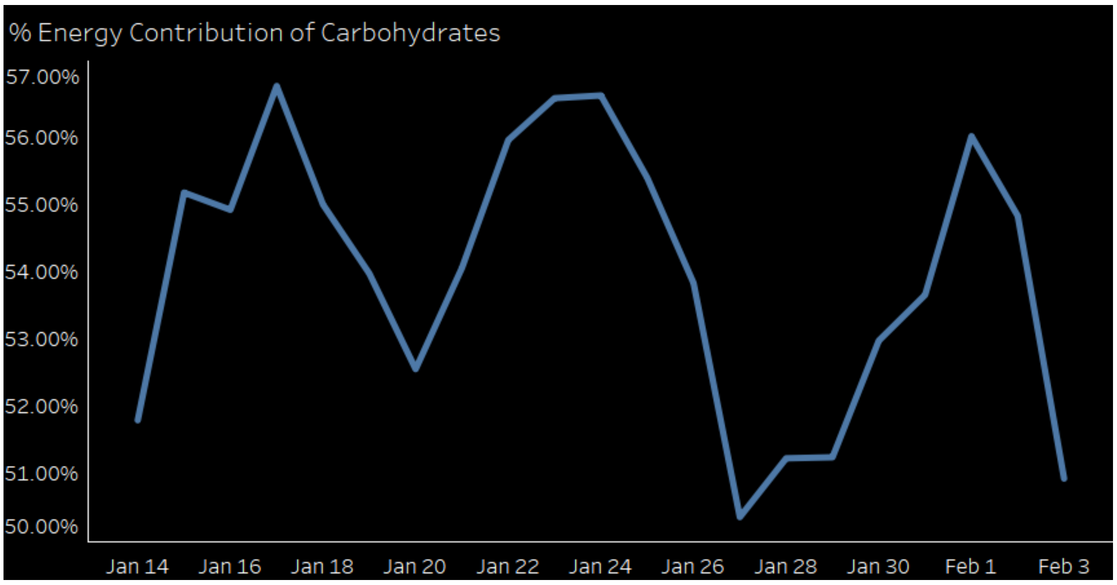


```
-- The following query returns the total purchase done by customers for items > $100 .00
SELECT top 5 customername, sum (PurchasePrice)
FROM orders_filteredNCCI
WHERE purchaseprice > 100.0
Group By customername
```

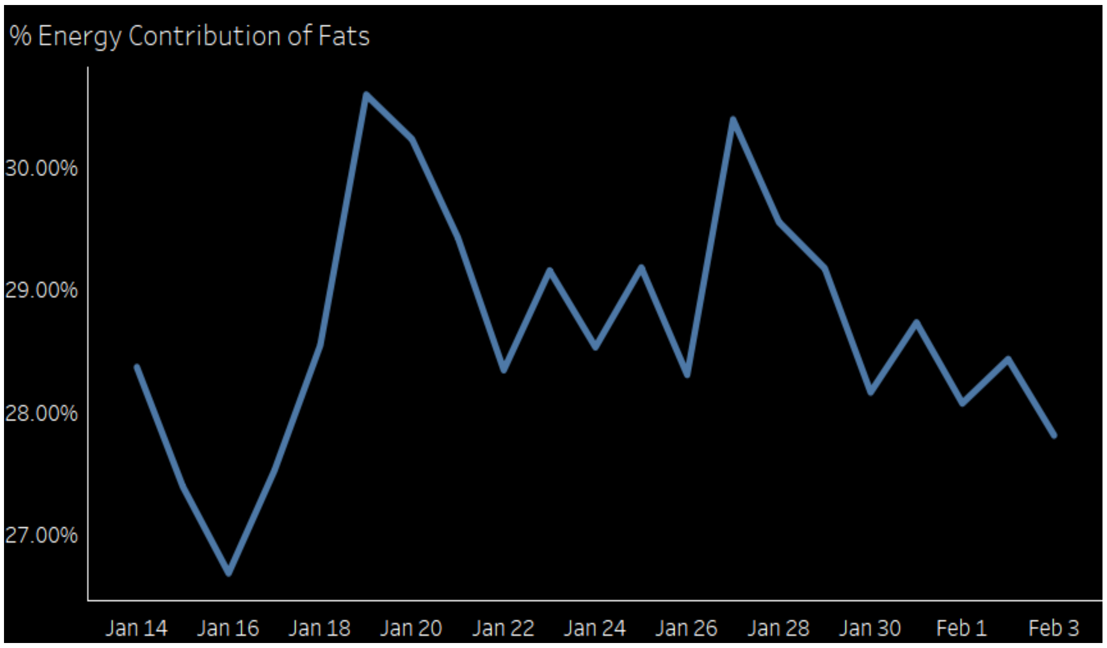
```
Query 1: Query cost (relative to the batch): 100%
-- The following query returns the total purchase done by customers for items > $100 .00 SELECT top 5 customername,
```

4.2 Screenshots

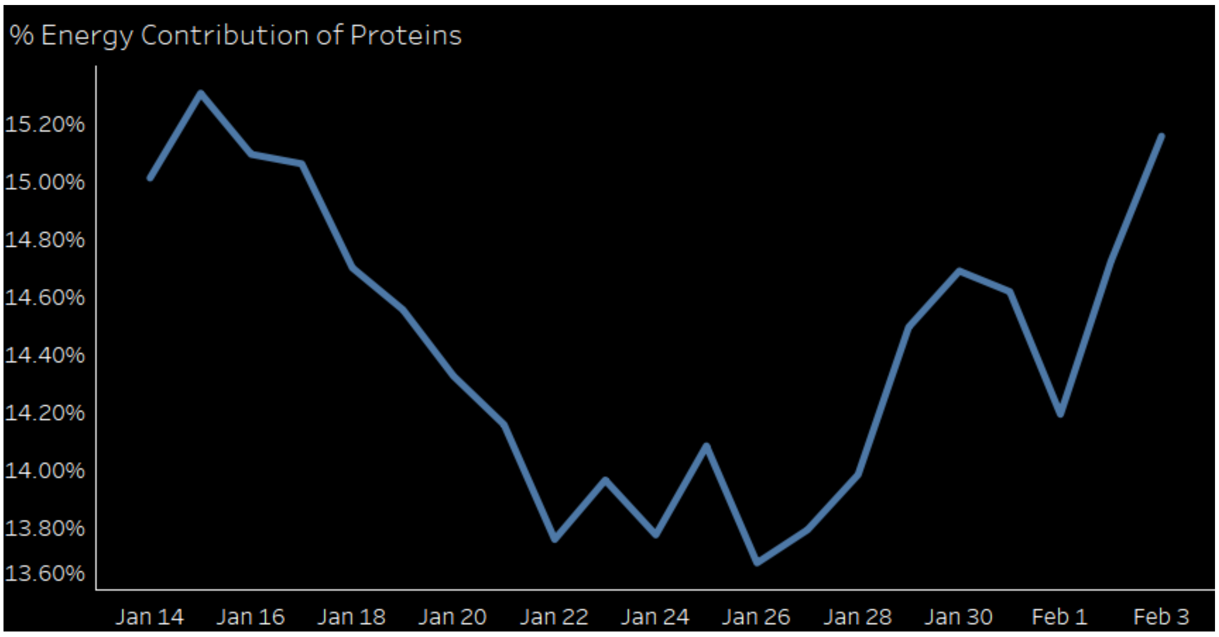
Analytical Reports:



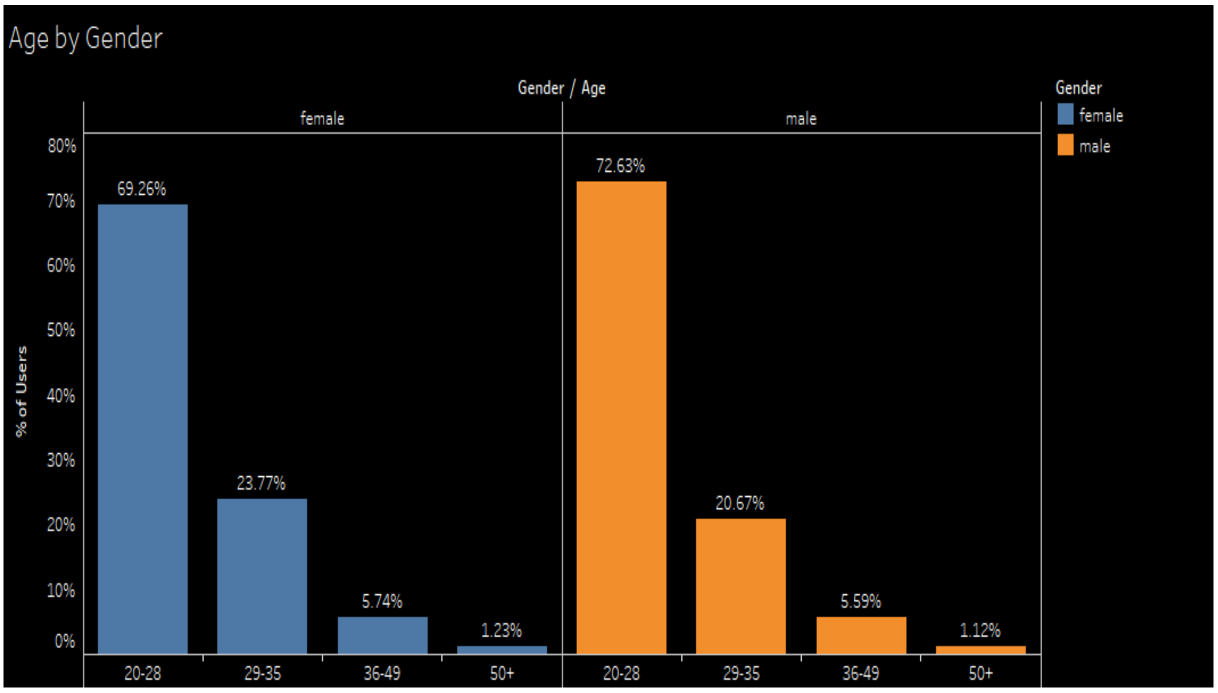
Carbs Breakfast



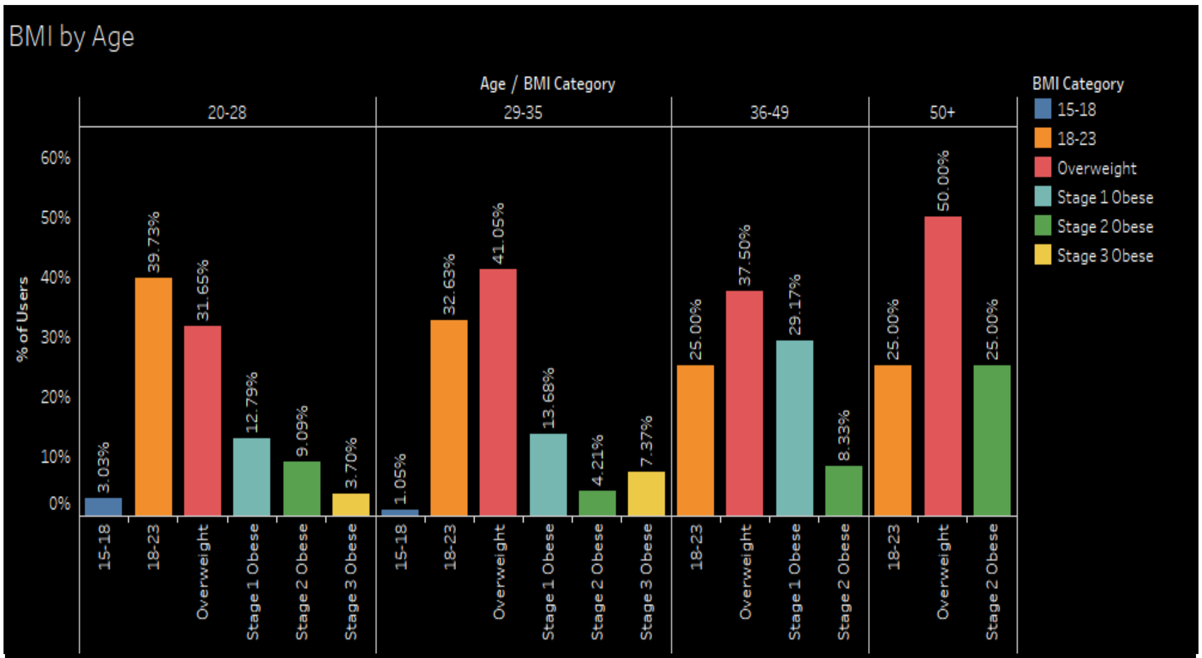
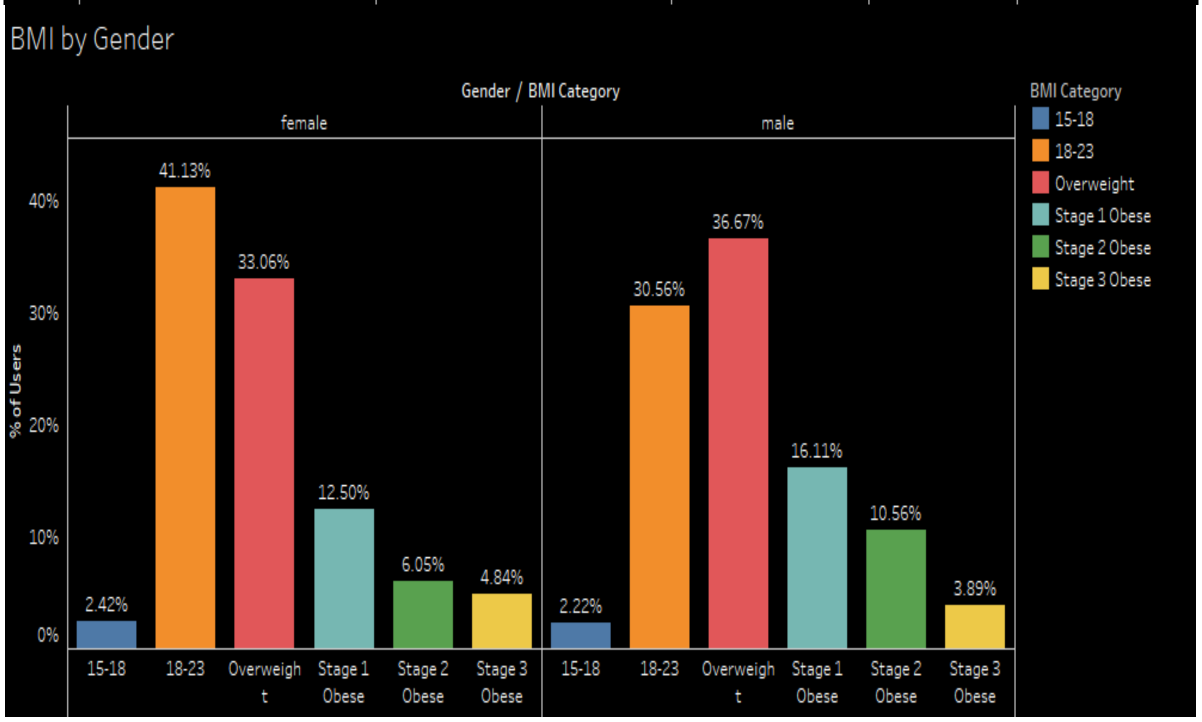
Fats Lunch

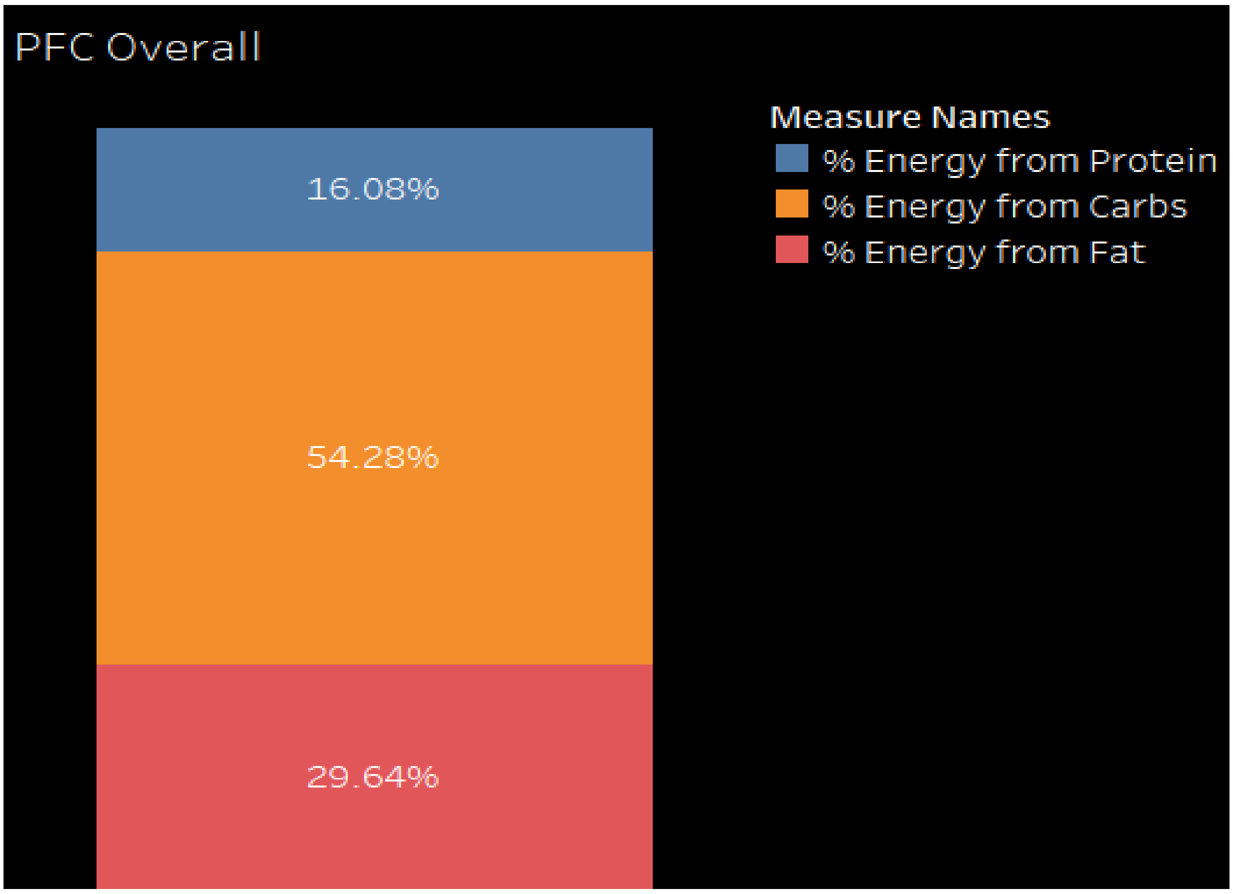


Protein Dinner

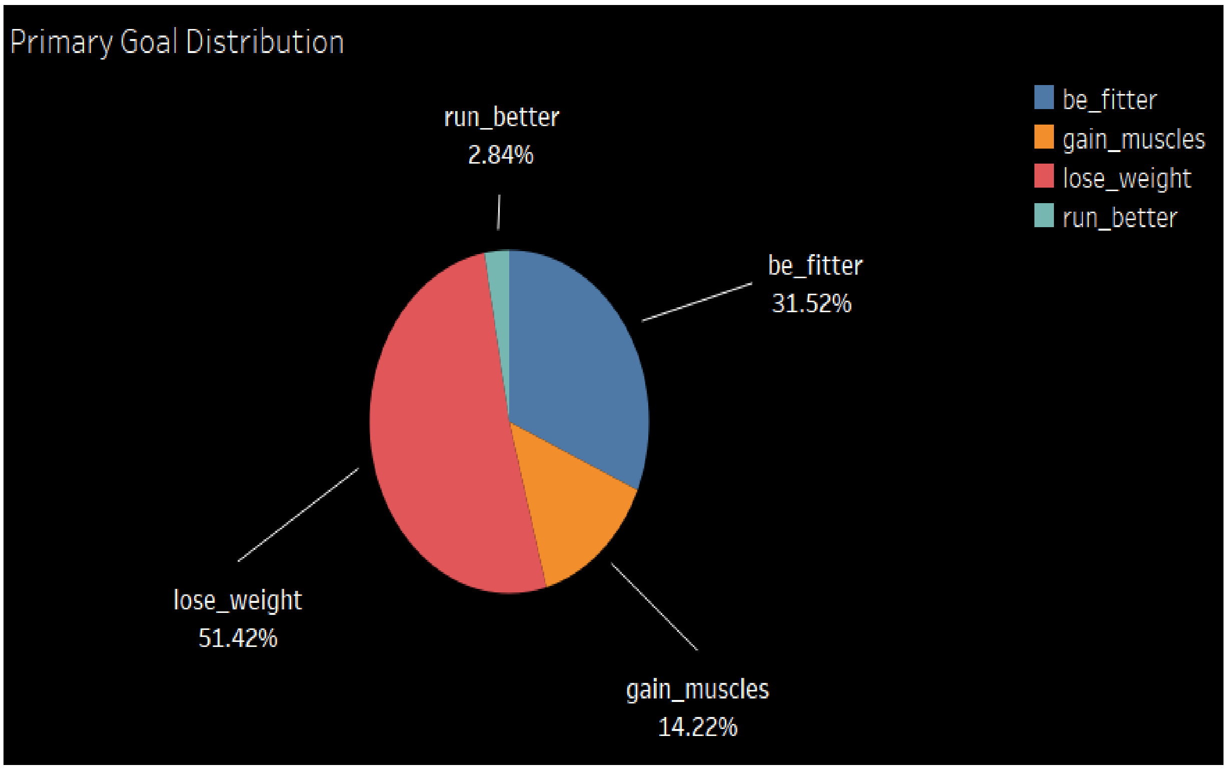


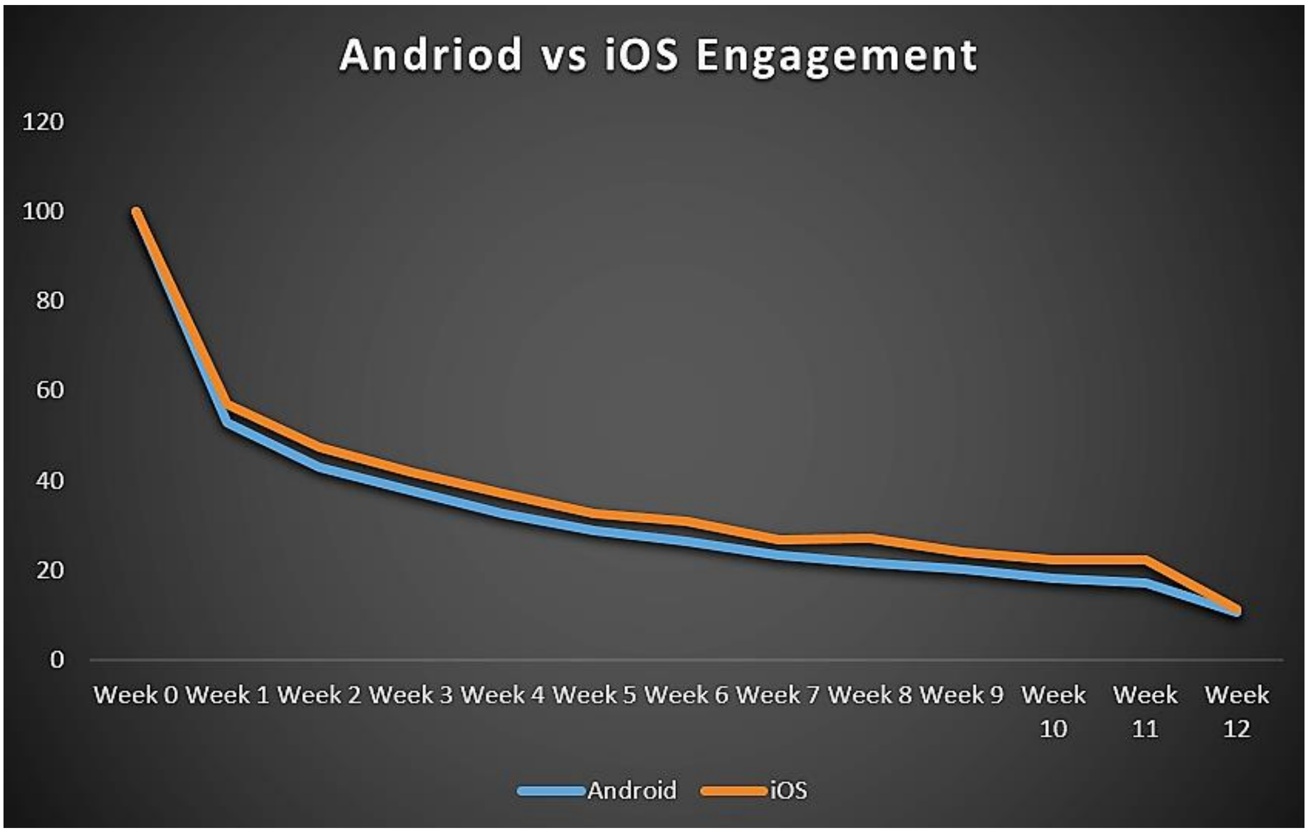
Age By Gender Distribution


BMI by Age
Age / BMI Category
20-28
29-35
36-49
50+
% of Users
60%
50%
40%
30%
20%
10%
0%
3.03%
39.73%
31.65%
12.79%
9.09%
3.70%
1.05%
32.63%
41.05%
13.68%
4.21%
7.37%
25.00%
37.50%
29.17%
8.33%
25.00%
50.00%
25.00%
15-18
18-23
Overweight
Stage 1 Obese
Stage 2 Obese
Stage 3 Obese
BMI Category
15-18
18-23
Overweight
Stage 1 Obese
Stage 2 Obese
Stage 3 Obese

BMI by Gender
Gender / BMI Category
female
male
% of Users
40%
30%
20%
10%
0%
2.42%
41.13%
33.06%
12.50%
6.05%
4.84%
2.22%
30.56%
36.67%
16.11%
10.56%
3.89%
15-18
18-23
Overweight
Stage 1 Obese
Stage 2 Obese
Stage 3 Obese
BMI Category
15-18
18-23
Overweight
Stage 1 Obese
Stage 2 Obese
Stage 3 Obese

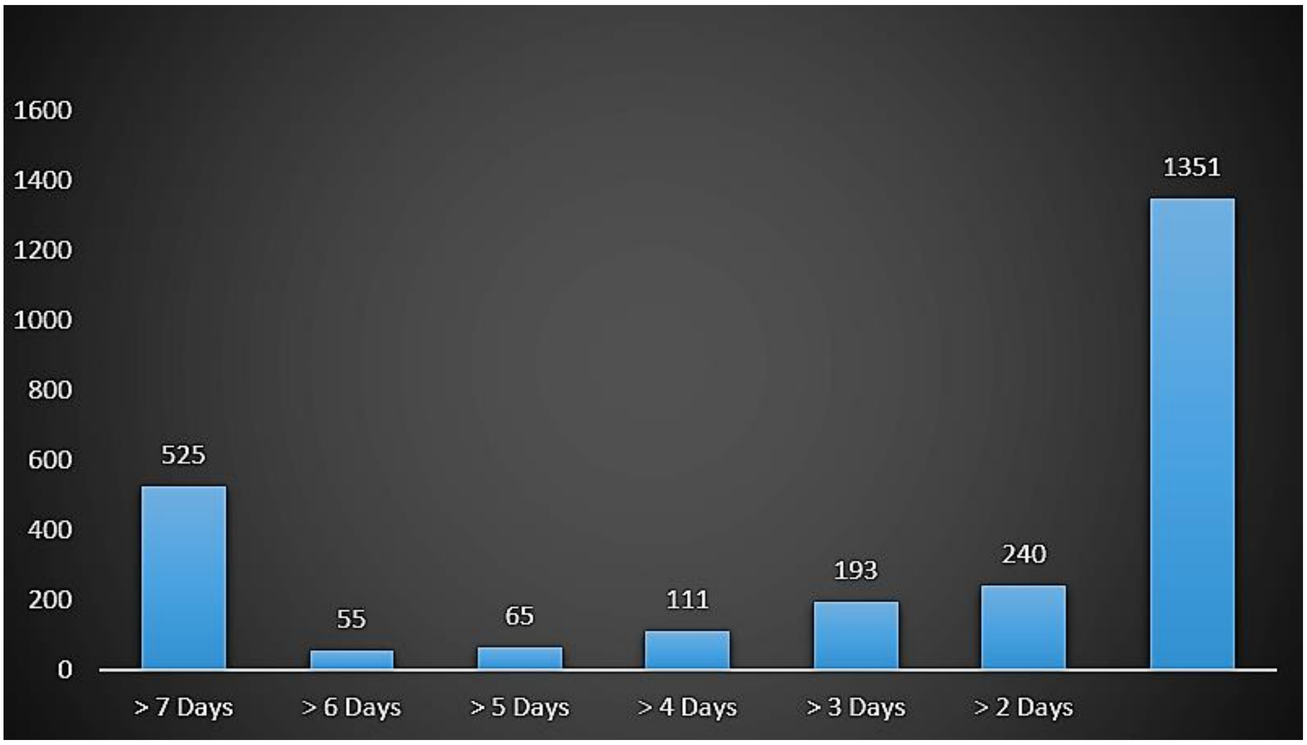


PFC BreakDownOverall

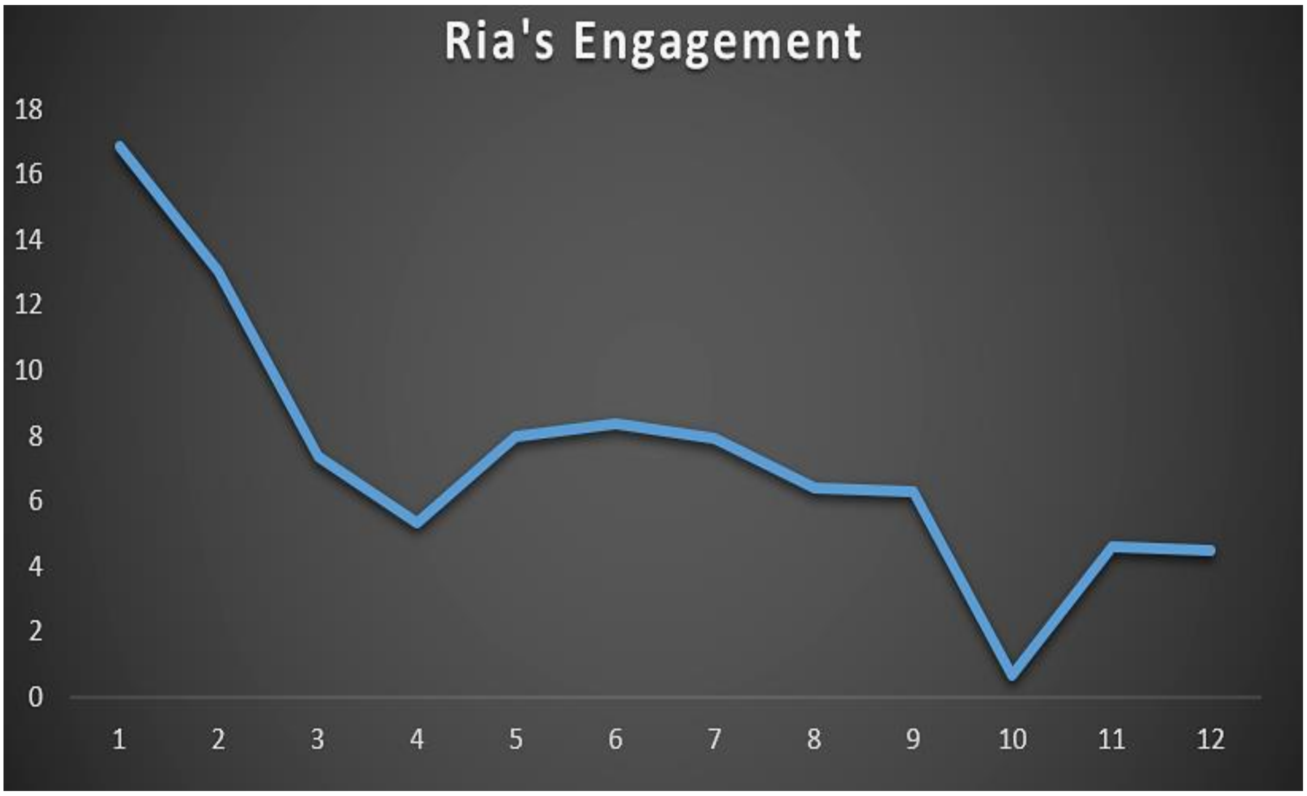




Android vs IOS Engagement



Day Wise Engagement



Graphical Data Representation

CHAPTER-5

CONCLUSION

CONCLUSION

A column store index also has a number of limitations and restrictions (too many to list here), such as which data types can be indexed, the number of columns that can be included, the type of objects that support column store indexing, and numerous other considerations. For more details about what you can and cannot do with column store indexes, see the topic "Columnstore Indexes" in SQL Server Books Online or at MSDN. The topic also includes details about how to update data in a column store index. Essentially, you can't do it, but you can take steps such as dropping the index or partitioning the table.

Despite the limitations of the column store index, they can provide performance improvements that far exceed what you can get with the row store structures of clustered and no clustered indexes. When used under the right circumstances, column store indexes can significantly reduce disk I/O and well as utilize memory more efficiently. These indexes are made for the type of analytical processing your BI workloads often require, an important consideration in this age of ever-increasing data.

5.1 Advantages

The proposed system is an online system: so any persons can browse the site and find out the best institute.

- Less time-consuming.
- Highly secure in data storing.
- It is more users friendly: the sections such as, registration, login and posting ads are available.
- Legal help is there for the user.

5.2 Limitations

- Only one Columnstore index can be created per table.

5.3 Future Enhancement

In the future, we will be able to add more columns and store data in them thereby optimization can be further improved.

CHAPTER-6

REFERENCES

REFERENCES

[1] S. Song, L. Chen and J. X. Yu, "Answering Frequent Probabilistic Inference Queries in Databases," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 4, pp. 512-526, April 2011.

[2] K. M. Wong, S & Butz, Cory & Xiang, Yuanfei. (2002). A Method for Implementing a Probabilistic Model as a Relational Database.

[3] Per-Åke Larson , Adrian Birka , Eric N. Hanson , Weiyun Huang , Michal Nowakiewicz , VassilisPapadimos, Real-time analytical processing with SQL server, Proceedings of the VLDB Endowment, v.8 n.12, August 2015

[4] S.K.M. Wong, C.J. Butz, and Y. Xiang, “A Method for Implementing a probabilistic Model as a Relational Database,” Proc. Conf.Uncertainty in Artificial Intelligence (UAI), pp. 556-564, 1995.

[5] S.K.M. Wong, D. Wu, and C.J. Butz, “Probabilistic Reasoning in
Bayesian Networks: A Relational Database Approach,” Proc. Conf.Artificial Intelligence (AI), pp. 583-590, 2003.

[6] J. Goldstein and P.A. Larson, “Optimizing Queries UsingMaterialized Views: A Practical, Scalable Solution,” Proc. ACMSIGMOD, pp. 331-342, 2001.

[7] R. Chirkova and C. Li, “Materializing Views with Minimal Size toAnswer Queries,” Proc. Symp. Principles of Database Systems (PODS), pp. 38-48, 2003.