Grenze International Journal of Engineering and Technology, June Issue

Management of Memory in the Real Time Data Streams

Kavitha N[1], Kalpana Y[2] and Kumar V[3]
[1-2] Vels Institute of Science Technology and Advanced Studies, Department of Information Technology, Chennai, TN
Email: kavitanithyanandam@gmail.com, ykalpanaravi@gmail.com
[3]Agril. Engineering Department, Agricultural College and Research Institute, Madurai, Tamil Nadu
Email: vskumaran1955@gmail.com

***Abstract*—Real time data streams[11] or streaming of data has become popular in the field of Data science. Data size is growing fast and large whereas it has become challenge for the researchers and industries on managing the data. Real time analysis involves large volume of dynamic data with dynamic change in nature. Once the data is received, correctness of the data needs to be checked. Data mining[19] is the multi-disciplinary field that combines statistics, machine learning[2], Artificial intelligence[10] and database technology. The main characteristic of mining complex data streams is the huge volume of continuous incoming infinite data; the nature of the data is fast-changing and requires a fast real-time response. The data will be of multi-Dimensional. Data sets are complex in nature, main challenge lies on the unbound memory required to store the data and use the same for further analysis. This paper discusses about managing the real time data with available memory and the technologies used for handling it.**

***Index Terms*— Real time data, Data Streams, Data cleaning, Cron, NewSQL, VoltDB.**

I. INTRODUCTION

Real time data in the recent times has become very popular in the field of stock market analysis. Data from the national stock exchange of India has been taken for the analysis. Pre market data will be participating in the analysis. Once the data is collected, correctness of the data is checked. For the purpose of checking the data correctness, data is cleaned. Data cleaning steps has been discussed. Once the data is cleaned, data is ready to participate in the memory management algorithm. This algorithm is fed into the Cron scheduler and the scheduler is triggered for every 10 minutes. In the memory management algorithm, data is separated as trial data and test data. Trial data is sent to the database [1] and the test data is sent to the database [2]. Here the database used for the further analysis is VoltDB. Also we have discussed about the different databases, drawbacks and comparison for different databases also explained.

II. DATA STREAMS

Data Streaming [1] is the processing of continuous inflow of data fed into the stream processing software to arrive the desired insights. Key characteristics of the data streams are Time sensitive, Data is continuous, Heterogeneous, volatile and unrepeatable. On the other side is the real time data streams where the action on the data is taken at the time of generation. Processing is measured in the milli seconds rather than the minutes or hours. Real time data is often referred to the event data where each data point describes something that occurred at a given time. Some of the benefits of data stream processing are Reduce Infrastructure Cost, Reduce

Grenze ID: 01.GIJET.9.2.541_3

Preventable Losses, Increase Competitiveness and Customer Satisfaction.

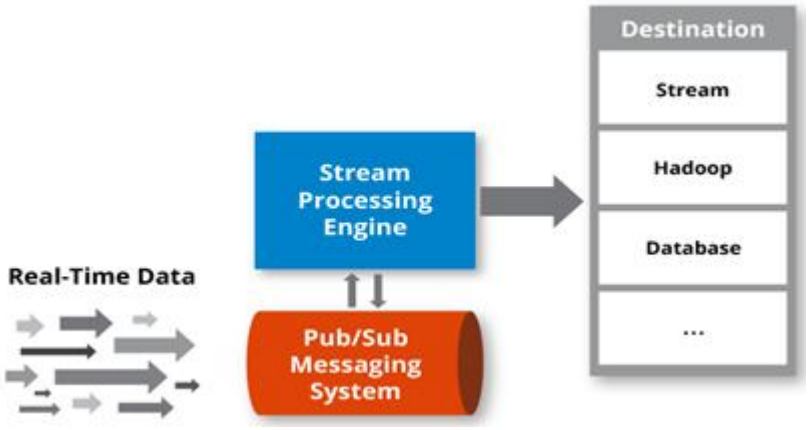

Figure 1:. Data Stream Architecture

III. Architecture of the Research

Real time is collected from the National Stock Exchange. Once the data is collected, data is incomplete, inconsistent, duplicated, improperly formatted. To overcome all this inconsistencies, data is cleaned using python. Once the data is cleaning is cleaned, it is ready to participate in the memory management algorithm. Once the algorithm is executed, data is pushed to different databases. Data in the primary database will be the trial data and the data in the secondary database is the test data.

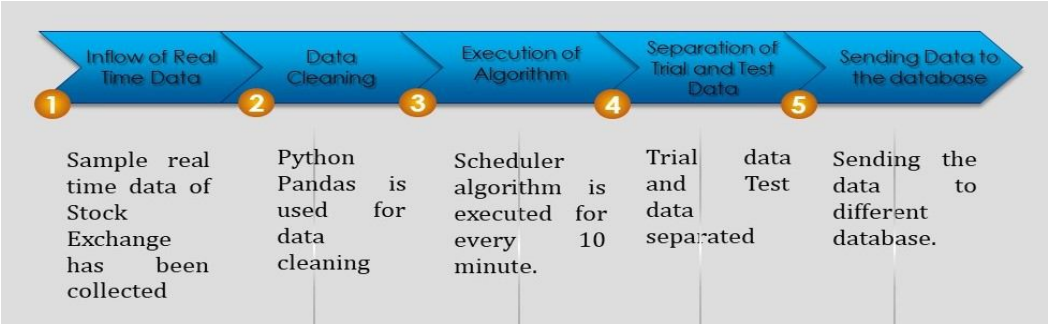

Figure 2. Research Architecture

IV. Data Cleaning

Data cleaning[31] is the process of correcting incorrect, incomplete, corrupted, duplicate, incorrectly formatted data from the dataset. While combining the data from the data source for different intervals, there are the high possibilities for the data to be duplicated or mislabeled. To overcome all these issue, a data set has to be cleaned. Data cleaning also called as data cleansing is the process of detecting and correcting the data set. It also identifies the incomplete, incorrect and corrupted data by replacing, modifying or deleting the dirty data. After the data is cleaned, data should be consistent with the other similar data set. Inconsistency in the data has to be identified and removed. NULL in the data set should be handled in the proper way where it should not be a problem while processing the data. Data should be cleaned at the time of entry level rather at the time of processing.
Different stage in the data cleaning cycle are shown in the below diagram.

A. Steps for Data Cleaning

Steps for cleaning the dataset are

- Step 1: Rename the column header using the common column name used in the database.
- Step 2: Looking for the statistical characteristic of the dataset.
- Step 3: Dropping the full NULL rows.

TABLE I. SAMPLE DATASET

SYMBOL	PREV. CLOSE	IEP	CHNG	%CHNG	FINAL	FINAL QUANTITY	VALUE	FFM CAP	NM 52W H	NM 52W L
POWERGRID	213.8	215.6	1.8	0.84	215.6	103655	2,23,48,018.00	7,30,76,23,92,938.37	248.35	186.35
SUNPHARMA	984.5	990.15	5.65	0.57	990.15	4430	43,86,364.50	10,62,96,53,75,084.25	1,072.15	789.9
BHARTIARTL	776.8	781.25	4.45	0.57	781.25	7488	58,50,000.00	19,45,09,88,53,400.52	860.55	628.75
AXISBANK	854.9	859.7	4.8	0.56	859.7	8876	76,30,697.20	23,13,21,78,32,937.09	970	618.25
KOTAKBANK	1,759.25	1,768.05	8.8	0.5	1,768.05	4719	83,43,427.95	25,50,28,38,07,207.08	1,997.55	1,631.00
EICHERMOT	3,285.00	3,300.00	15	0.46	3,300.00	417	13,76,100.00	4,58,17,14,00,807.00	3,889.65	2,159.55
TATAMOTORS	439.9	441.8	1.9	0.43	441.8	11823	52,23,401.40	7,88,95,92,09,925.76	511.5	366.2
HDFCLIFE	504.25	506.3	2.05	0.41	506.3	5636	28,53,506.80	4,98,51,18,88,284.78	620.6	473.7
ASIANPAINT	2,833.60	2,845.00	11.4	0.4	2,845.00	3409	96,98,605.00	12,77,45,19,43,139.68	3,582.90	2,560.00
ONGC	156.6	157.2	0.6	0.38	157.2	28948	45,50,625.60	6,10,72,22,34,334.48	194.95	119.85
BPCL	331.7	332.85	1.15	0.35	332.85	6594	21,94,812.90	3,23,79,35,10,833.16	398.8	288.05
ITC	383.4	384.7	1.3	0.34	384.7	23374	89,91,977.80	33,78,80,42,82,670.43	388.2	207
LT	2,226.35	2,233.70	7.35	0.33	2,233.70	2747	61,35,973.90	26,90,79,58,84,765.95	2,297.65	1,456.35
BAJAJFINSV	1,414.05	1,418.00	3.95	0.28	1,418.00	1998	28,33,164.00	7,65,78,90,38,412.42	1,844.00	1,072.72
HDFCBANK	1,655.90	1,660.00	4.1	0.25	1,660.00	12821	2,12,82,860.00	72,95,29,39,15,586.76	1,722.10	1,271.60
INDUSINDBK	1,114.70	1,117.25	2.55	0.23	1,117.25	2034	22,72,486.50	7,26,05,83,25,619.31	1,275.80	763.2

Figure 3. Data cleaning cycle

- Step 4: Replacing the null, N/A and Nan values with 0.
- Step 5: As per the table structure, casting the column in the dataset (i.e.) data transformation.
- Step 6: Checking for the statistical analysis of the data.

V. CRON

Cron is scheduling the task repeatedly at the specific interval. cron is the command line utility like scheduler on the unix like operating system. This cron job will run on every minute of every hour of every day of every

month, each day of the week. This cron scheduler also referred as job scheduler. Cron is most suitable for scheduling repetitive tasks. Scheduling one-time tasks can be accomplished using the associated at utility.

A. Cron job

Task scheduling in the cron is called as cron job. User can determine the task to be automated and when it should be executed. cron is the daemon which means a background process executing non-interactive jobs. It is the simple text file and the default cron configuration file is /etc/crontab. Using cron job user can automate system maintenance, disk space monitoring and backup schedules. It is the greatest utility for the computers which runs on 24*7 basis. cron uses the simple shell script. There are two basic cron job operations. one is system crontab, where jobs will be scheduled system wide with root privilege and the other is user crontab, where user can create and edit the cron job and apply at the user level. Common syntax for the cron scheduler is shown in the diagram below.

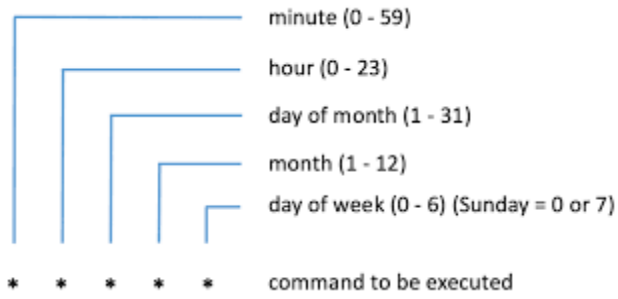

Figure 4. Syntax for Cron

Nonstandard predefined scheduling definitions:
Some cron implementations support the following non-standard macros.

TABLE II. SCHEDULING DEFINITIONS

Entry	Description	Equivalent to
@yearly (or @annually)	Run once a year at midnight of 1 January	0 0 1 1 *
@monthly	Run once a month at midnight of the first day of the month	0 0 1 * *
@weekly	Run once a week at midnight on Sunday morning	0 0 * * 0
@daily (or @midnight)	Run once a day at midnight	0 0 * * *
@hourly	Run once an hour at the beginning of the hour	0 * * * *
@reboot	Run at startup	—

B. Cron permissions

These two files play an important role:

- /etc/cron.allow - If this file exists, it must contain the user's name for that user to be allowed to use cron jobs.
- /etc/cron.deny - If the cron.allow file does not exist but the /etc/cron.deny file does exist then, to use cron jobs, users must not be listed in the /etc/cron.deny file.

C. Limitations of cron job

Though the utility is highly appreciative for the scheduling the task without any intervention, it has its own limitations. They are
Shortest interval between the job is 60 seconds. User cannot able to repeat the job before 59 seconds.
Cron jobs cannot be distributed to multiple computers on network. It is centralised to one computer. If the system crashes schedule task will not be executed and the same has be executed manually.
Cron is designed to execute for the specific time. If the task fails it has to wait for the next schedule time. There is no auto retry mechanism. This makes cron unsuitable for incremental tasks.

VI. DATABASE AND DBMS

A database is the organized collection of information or data stored in the electronic form. Database is controlled by Database Management System. It is a system software for creating and managing database. DBMS make the

end user to create, insert, update, delete and protect the database. DBMS serves as an interface between database and application program. It also ensures that the data can be easily accessible by the end user. DBMS supports typical database administrative tasks like change management, tuning, performance monitoring, security, backup and recovery. Some of the components of the DBMS are storage engine, Metadata catalog, database query language, query processor, data utilities and log manager etc.

Types of databases:

- Relational DBMS
- NoSQL Databases
- New SQL Databases

A. RDBMS - SQL

SQL or the structured query language is the standard query language used for interacting with the relational database management system. It stores the data in the form of rows and columns using the fixed schema. Each row has same columns with the defined datatype. Features of the SQL are ACID property used for maintaining the reliability of the transactions. Normalization, for designing the efficient database and the vertical scalability. Some of the disadvantages of the SQL are unable to handle the large data needs, rise of the cloud-based solutions, dynamic change in the schema and handling of Big data.

How SQL works?

After executing the SQL command for any RDBMS, system determines the best way to carry out the request and SQL engine figures out the work to be carried out. Various components of the SQL are Query Dispatcher, Optimization Engines, Classic Query Engine and SQL Query Engine.

TABLE III. COMMANDS IN SQL

Data Definition Language (DDL)	
CREATE	Creates a new table, a view of a table, or other abject in the
ALTER	Modifies an existing database object, such as a table
DROP	Deletes an entire table, a view of a table or other objects in the database
Data Manipulation Language (DML)	
SELECT	Retrieves certain records from ane or more tables
INSERT	creates a record
UPDATE	Modifies records
DELETE	Deletes records
Data Control Language (DCL)	
GRANT	Gives a privilege to user
REVOKE	Takes back privileges granted from user

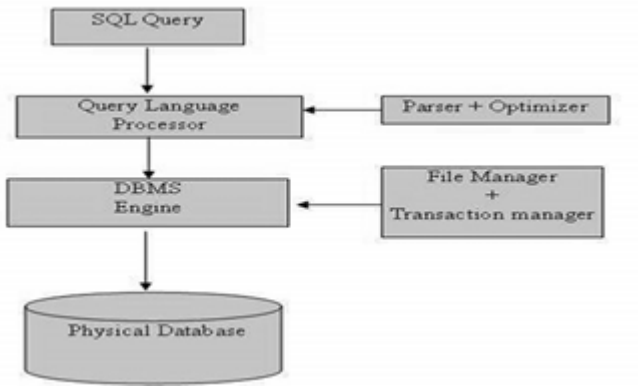

Figure 4.SQL basic commands

B. NoSQL

NoSQL stands for Not only SQL or Not SQL. NoSQL originally built for scale, unstructured data and did not use relational table-based schema. Stores data in the database as a key- value pair with unique key and associated values. Wide column database stores large number of columns with flexible schema. Stores data in the graph form to highlight the connection between the different data elements. Some of the features of the NoSQL are BASE principle – Basically available soft – state, lack of schema, eventually consistent and horizontally

scalable. Some of the limitations are limited query capabilities, does not support the ACID properties. As the data is stored as key-value pairs it is difficult to maintain as the data grows.
NoSQL database features:

- Horizontal scaling
- Fast queries due to the data model
- Ease of use for developers

Types of NoSQL database:

- Document databases - Stores data in Database like JSON objects. Each document contains pair of fields and values.
- Key-value databases - Type of database where each item contains keys and values.
- Wide-column stores - Stores data in tables, rows and dynamic columns
- Graph databases - Stores the data in nodes and edges. Nodes stores information whereas as edges stores relationship between the nodes.

Key Features of NoSQL

- Dynamic schema
- Horizontal scalability
- Document-based
- Key-value-based
- Column-based
- *Flexibility*
- Performance

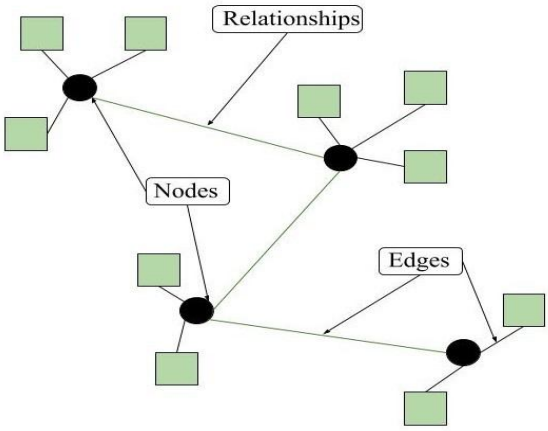

Figure 5. NoSQL Data Architecture Patterns

C. NewSQL

A class of modern relational Database Management System provides scalable performance of NoSQL databases. Supports the ACID properties. It is the SQL database with distributed and fault-tolerant architecture. NewSQL database offers in-memory capabilities and clustered database services with ability to deploy in the cloud. It can also handle large amount of data while maintaining the transactional consistency. Some of the key features of the NewSQL databases are Main Memory storage of OLTP databases which enables in-memory computations of databases, vertical and horizontal scalability, supports ACID properties, enhanced concurrency control system, presence of secondary index which allows for the faster query processing, strong data durability, minimized downtime, fault tolerant and crash recovery mechanism.

NewSQL Database Features:

- In-memory storage and data processing supply fast query results.
- Partitioning scales the database into units. Queries execute on many shards and combine into a single result.
- ACID properties preserve the features of RDBMS.
- Secondary indexing results in faster query processing and information retrieval.

- High availability due to the database replication mechanism.
- A built-in crash recovery mechanism delivers fault tolerance and minimizes downtime.

TABLE IV. COMPARISON OF DATABASES

Feature	SQL	NoSQL	NewSQL
Relational Property	Yes	No	Yes
ACID	Yes	No	Yes
SQL	Yes	No	Yes
OLTP	Inefficient for OLTP databses	Supports, not best suited	Fully supported
Scaling	Vertical Scaling	Horizontal Scaling	vertical + Horizontal Scaling
Distributed Database	No	Yes	Yes

VII. VOLTDB

VoltDB[26] is an in-memory database and an ACID-compliant RDBMS that uses a share-nothing architecture. It supports SQL access form within the pre-compiled java stored procedure. VoltDB database is optimized for a specific application by partitioning the database tables and the stored procedures that access those tables across multiple "sites" or partitions on one or more host machines to create the distributed database. Parallelly multiple queries can be executed. VoltDB balances maximum performance requirements with the flexibility to accommodate less intense but equally important queries that cross partitions. Using this database, data is pushed in different database in the pre-defined table structure.

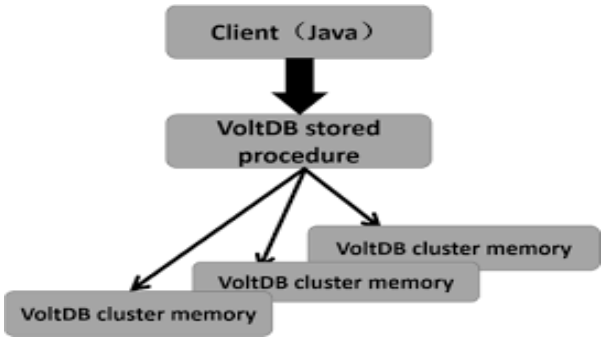

Figure 6. VoltDB Architecture

Memory Management [34] Algorithm

- Step 1: Start
- Step 2: Declare variables for Prev close, IEP price and Quantity
- Step 3: Read values for Prev close, IEP price and Quantity
- Step 4: Formula for Change value = IEP price - Prev close
- Step 5: Formula for % Change value = (Change value / Prev close) * 100
- Step 6: Final price = IEP price
- Step 7: Value = IEP price * Quantity
- Step 8: Separation of Trial Data and the Test data
- Step 9: Data is pushed to multiple databases.
- Step 10: End

VIII. CONCLUSION

Data is cleaned, and values are calculated using the algorithm. Once the values are calculated, data is pushed to multiple databases. Here the primary data will be at database1, and the rest of the values will be pushed to a different database with the common column symbol. By achieving this, memory will be managed, and the unused columns will be at different servers. Unused data can be analysed in the scheduled intervals, and the same can be discarded once the analysis is done. By discarding the analysed values, memory will be free, and it optimizes the overall system performance; the load on the machine will be reduced. The future scope of the research will be on model prediction and evaluating the overall model performance using machine learning.

ACKNOWLEDGMENTS

I would like to extent my sincere thanks to Dr.Y.Kalpana, Professor VISTAS for guiding me in my research with innovative ideas and advanced techniques. Also providing support as and when required. I would like to thank my professor Dr. Kumar V who seeded the idea of research in my career. I like to thank Dr.G.Suseendran (Late) for providing me the topic of research and giving me the opportunity to work with VISTAS.

REFERENCES

[1] Bifet, R. Kirkby, Data Stream Mining -A Practical Approach.
[2] S. K. Sen, and B. K. Ratha, "A Comprehensive Study on Distributed Data Mining and Learning Algorithms," Xi, J. B. Ni, "Deploying Mobile Agents in Distributed Data Mining," PAKDD 2007 Workshops, pp. 322–331, 2007.
[3] S. Bailey, R. Grossman, H. Sivakumar, and A. Turinsky, "Papyrus: A System for Data Mining over Local and Wide Area Clusters and Super-Clusters".
[4] V. Sawant and K. Shah, "A review of Distributed Data Mining using agents", International Journal of Advanced Technology & Engineering Research (IJATER), vol. 3, no. 5, pp. 27-33, 2013.
[5] S. Kumar, P. N. Santosh Kumar, and C. Venugopal, "An Apriori Algorithm in Distributed Data Mining System", Global Journal of Computer Science and Technology Software & Data Engineering, vol 13, No. 12, 2013.
[6] K. Das, K. Bhaduri, and H. Kargupta, "A local asynchronous distributed privacy preserving feature selection algorithm for large peer-to peer networks", J. Knowledge and Information Systems, vol. 24(3), pp. 341-367, Sept. 2010.
[7] R. Vilalta, C. Giraud-Carrier, P. Brazdil, and C. Soares, "Using Meta-Learning to Support Data Mining," International Journal of Computer Science & Applications, vol. 1, No. 1, pp. 31-45, 2004.
[8] S. C. Frank, Y-H. Tseng, and Y-H, Min, "Toward boosting distributed association rule mining by data de-clustering," Journal of Information Sciences, vol. 180, Issue 22, pp. 4263-4289, Nov. 2010.
[9] G. S. Bhamra, A. K. Verma, and R. B. Patel, "Agent Enriched Distributed Association Rules Mining," ADMI 2011, pp. 30–45, 2012.
[10] J. Costa da Silva, and M. Klusch, "Inferences in Distributed Data Mining", Engineering Applications of Artificial Intelligence, vol. 19, pp. 363 -369, 2006.
[11] M. A. Naeem, "A robust join operator to process streaming data in real time data warehousing," in Eighth International Conference on Digital Information Management (ICDIM 2013), pp. 119–124, IEEE, 2013.
[12] H. Isah, T. Abughofa, S. Mahfuz, D. Ajerla, F. Zulkernine, and S. Khan, "A survey of distributed data stream processing frameworks," IEEE Access, vol. 7, pp. 154300–154316, 2019.
[13] M. A. Naeem, G. Dobbie, and G. Weber, "Efficient processing of streaming updates with archived master data in near-real-time data warehousing," Knowledge and information systems, vol. 40, no. 3, pp. 615–637, 2014.
[14] M. Babar and F. Arif, "Real-time data processing scheme using big data analytics in internet of things based smart transportation environment," J. Ambient Intelligence and Humanized Computing, vol. 10, no. 10, pp. 4167–4177, 2019.
[15] N. Biswas, A. Sarkar, and K. C. Mondal, "Efficient incremental loading in etl processing for real-time data integration," Innovations in Systems and Software Engineering, pp. 1–9, 2019.
[16] M. A. Naeem, G. Dobbie, I. S. Bajwa, and G. Weber, "Resource optimization for processing of stream data in data warehouse environment," in Proceedings of the International Conference on Advances in Computing, Communications and Informatics, pp. 62–68, ACM, 2012.
[17] R. Mukherjee and P. Kar, "A comparative review of data warehousing etl tools with new trends and industry insight," in 2017 IEEE 7th International Advance Computing Conference (IACC), pp. 943–948, IEEE, 2017.
[18] H. Bouali, J. Akaichi, and A. Gaaloul, "Real-time data warehouse loading methodology and architecture: a healthcare use case," Int. J. Data Analysis Techniques and Strategies, vol. 11, no. 4, pp. 310–327, 2019.
[19] R. Duan, R. Prodan, and T. Fahringer, "Short Paper: Data Mining-based Fault Prediction and Detection on the Grid," Proceedings of 15th IEEE International Symposium on High Performance Distributed Computing, DOI: 10.1109/HPDC.2006.1652162, pp.305-308, 2006
[20] N. Khayat, Semantic Instrumentation and Measurement of Data Mining Algorithms, Technical Report on R&D 2, Hochschule Bonn-Rhein-Sieg, 2009.
[21] S. Datta, K. Bhaduri, C. Giannella, R. Wolff, and H. Kargupta. Distributed data mining in peer-to-peer networks. Internet Computing, IEEE, vol. 10, No. 4, pp. 18–26, 2006.
[22] N. Khan, I. Yaqoob, I. A. Hashem, Z. Inayat, W. K. Ali, M. Alam, et al, "Big data: survey, technologies, opportunities, and challenges," Scientific World J, Article ID 712826, 2014
[23] M. Steen, G. Pierre, and S. Voulgaris, "Challenges in very large distributed systems," J Internet Serv Appl, vol. 23(1), pp. 59–66.
[24] G. Tsoumakas, and I. Vlahavas, Distributed data mining. In: Encyclopedia of Data Warehousing and Mining, IGI Global, Hershey, PA, USA, 2009, 709–715.
[25] S. Cong, J. Han, J. Hoeflinger, and D. Padua, A sampling based framework for parallel data mining. In: Proc. of the ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, Chicago, Illinois, USA, 2005, 255–265.

[26] Ramu Vookanti, E. Aravindan, M. Varaprasad - Study on Conversion of Relational Databases to Big Data (VoltDB) - International Journal For Research & Development In Technology, vol.15, No. 3 (Mar-21). pp. 127-130
[27] M. Last, “Online classification of nonstationary data streams,” Intelligent Data Analysis, vol. 6, No. 2, pp. 129-147, 2002.
[28] Shearer, “The CRISP-DM model: the new blueprint for data mining,” J. Data Warehousing, vol. 5, No. 4, pp. 4-15, 2000.
[29] C. Aggrawal, Data Streams: Models and Algorithms, Springer, 2000
[30] L. O'Callaghan, N. Mishra, A. Meyerson, S. Guha, and R. Motwani, “Streaming-data algorithms for high-quality clustering,” in Proc. 2003 IEEE International Conference on Data Engineering.
[31] Fakhitah Ridzuan, Wan Mohd Nazmee Wan Zainonv- A Review on Data Cleansing Methods for Big Data, Procedia Computer Science 161 (2019) 731–738
[32] S. Muthukrishnan, Data streams: algorithms and applications, Proceedings of the fourteenth annual ACM-SIAM symposium on discrete algorithms, 2003
[33] Zafar Ali Khan: Memory Management Strategies for Different Real Time Operating Systems.
[34] Shah Vatsalkumar Hasmukhbhai- Memory Management in Real-Time Operating System, 2018
[35] Ahmed Faraz: A review of memory allocation and management in computer systems, An International Journal (CSEIJ), Vol.6, No.4, August 2016.