ENTITY AWARE ASSOCIATION RULE BASED BIG DATA MIGRATION

[1]V. Rathika , and [2]Dr. L. Arockiam

[1] Research Scholar, [2] Associate Professor
[1] Department of Computer Science, [2] Department of Computer Science,
[1] Mother Teresa Women's University, Kodaikanal, Tamil Nadu, India.
[2] St. Joseph's College, Trichy, Tamil Nadu, India.

ABSTRACT : In the recent development of technologies, the existing structured databases are immature to handle the unstructured data. In view of this issue, most of the software application-based industries are upgrading or changing their databases. Most of the businesses are running lively and it is hard to shut down the database for many hours for data migration. It is noted that migrating big data from relational database to non-relational database is taking approximately ten to fifteen hours in real-time environment. This will be a tedious task for the enterprises. To address these issues, this research work is proposed to handle live data migration. The entity aware association technique is proposed to reduce the database downtime during the big data migration.

Keywords: **Big Data Migration, NoSQL Database, RDBMS, Map Reduce, Association Rule Mining.**

I. INTRODUCTION

Planning a migration with reserved downtime is hard. However in specific applications; a major part of downtime is unsatisfactory. If so, at that point it's a basic process to include application code that tracks which records has been moved and which haven't. This enables researchers and industrialists to inquire and incrementally redesign parts of the new database architecture while existing together with existing data and existing application code.

With the enhancement and progression of web advancements, individuals utilize the web increasingly in everyday life. In this manner, tremendous measure of information is created with different arrangements. The approaches to store heterogeneous and homogeneous information to find helpful data end up with some critical issues. Distributed computing innovations are created for these issues. For example, Google introduces the following technologies to handle the database storage and process issues. Table 1 addresses the above said statement.

Table 1. Technologies Introduced by Google for Big Data Handling

S. No.	Technology	Description
1.	Google File System (GFS)	To store heterogeneous data separately
2.	MapReduce	To run the distributed data in parallel environment
3.	Big Table	To store unstructured data (NoSQL Database)

The expression "Distributed Computing" isn't an innovation, yet an idea of setting up a server group in Cloud by virtualization advancements and preparing the vast measure of information put away in Cloud through the web [2, 17]. Google projected the Map Reduce system to part information into little pieces and execute the related occupations on hubs [20]. The outcomes were gathered from hubs, incorporated and afterward return back to clients. Along these lines, Map Reduce changes a solitary hub preparing occupation to a parallel handling employment to enhance the execution productivity.

In spite of the fact that GFS, MapReduce and BigTable [18] were distributed, the source code was not discharged. After the network created Hadoop [4], ventures and software engineers have a stage like GFS and MapReduce system to create MapReduce related innovations [3].

Most undertakings still utilize social database (RDB) for business. In any case, as an ever-increasing number of information created, RDB comes up short on the capacity to deal with such size of information. Utilizing Cloud database is a conceivable arrangement and the undertakings require a device to relocate information from RDB to Cloud database for execution as far as database usage.

In the time of data blast, anything can be digitalized, e.g. books and figures. As time passes by, the extent of computerized information develops and is difficult to realize how substantial it really is. IDC says the aggregate information size of Digital universe is 0.18 ZB in 2006. RDB is clearly not ready to preparing such sort of information regarding size. Be that as it may, Cloud Computing is intended for the Big Data including information putting away, preparing and breaking down.

Apache Sqoop was created for information transfer from RDB to NoSQL database. Sqoop turns into the best level venture in 2012 which empower clients to move extensive size of information to Cloud advancements. Not quite the same as the conventional means, records are maintained in various Virtual Machines (VMs). In any case, there will be an issue if a JOIN activity is performed and its information of tables is circulated and put away on various VMs.

Sqoop parts each table into four sections as a matter of course and utilize the Mapper of MapReduce structure to store information in the group by means of JDBC driver amidst information relocation. Information of tables is then put away in the VMs where Hadoop executes the Mapper haphazardly. The information in this manner is conveyed in the VM bunch. Due to execution of Mapper randomly in VMs, the terrible area of information in the group may expand the question time.

Hadoop, an Apache top level open source venture, gives a disseminated framework design and is mostly comprised of Hadoop Distributed File System (HDFS) and MapReduce. HDFS was created dependent on the idea of GFS to interface hubs together and frame an expansive size of disseminated document framework. HDFS was intended to process huge measure of information and give safe stockpiling engineering to keep away from equipment disappointment. It was likewise intended for composing once-read-many records get to show, which gives basic consistency component and increments the throughput of information change. HDFS copies the information on various information hubs to ensure clients can even now get to their information if any information hub crashes.

This research work addresses above issues and upgrades information region by investigating the record of inquiries. RDB logs everything including the questions which are submitted to get to database. By breaking down the log, tables which are often utilized can be found and put away as nearer as conceivable in the VM group to keep away from conceivable information change and enhance the general execution.

II. LITERATURE REVIEW

The advancements in technologies, ventures need to stay aware of information development blast. RDB comes up short on the capacity to deal with such measure of information for ongoing framework, e.g., telecom charging frameworks. An exploration [6] analyzed MapReduce and SQL on substantial scale information handling. The outcome demonstrates RDB to execute with little information; however MapReduce performed better while size of information expanded. LoadAtomizer [1] introduced calculations to illuminate the planning and load adjusting issues.

NoSQL databases are intended to process unstructured information [9, 10], yet ventures need to use the capacity of NoSQL to process structure information. Clydesdale [5] proposed a system on Hadoop for above issue without adjusting the muddled engineering of Hadoop yet displaying a few strategies to accelerate the task of handling organized information.

Hive creates MapReduce employments for inquiries one by one. In this manner, the connection between questions isn't considered. YSmart [11] tended to this issue and endeavored to blend activities as indicated by the connection between inquiries. YSmart effectively maintained a strategic distance from superfluous JOIN activity, enhanced the question time and was coordinated to hive in 2012.

JOIN is an every now and again utilized database activity. Numerous look into had proposed answers to enhance the execution of JOIN [7, 8]. Hive [12] gives SQL-like question dialect, HiveQL, to get to information in NoSQL stockpiling, e.g., HBase. Be that as it may, the MapReduce employments of JOIN tasks created by Hive are not productive.

III. OBJECTIVES

From the above background study, the most of the related works have been carried out using Sqoop technique for the database migration. Sqoop technique is not good enough for the live database migration. It also discussed the issues of data migration where the database is lively used by the users. Most of the data migration phases are processed after shutting down the database. This has become an important challenge for the enterprise. In order to address these issues, the following objectives are solved for an effective live big data migration. The objectives are as follows:

- To find the frequently accessed entity and attribute
- To rank the frequently associated entity and attribute
- To build automated query translator for live migration

IV. AUTOMATED QUERY MANAGEMENT FRAMEWORK

The overall process of the proposed work is represented in the figure 1. The figure makes easy to understand the overall process involved to migrate the big data from structured database to unstructured database without shutdown the database. This proposed methodology allows the users to retrieve and write the data into the source and new databases simultaneously. The proposed automated query management technique transfers the big data parallel in the live environment. The proposed live big data migration technique uses the replicas of the source database. Recent image of the archive files of is used as the source database. This avoids the data loss and maintains the integrity of the database.

4.1 Process of Automated Query Management

This section explains the overall process of the AQM in detail. There are six process involved in the AQM based live big data migration. The processes are described as follows:

Process 1:

Every relational database maintains the SQL query log files. The query log files have the Meta data of the queries executed and the status of the database. These log files play a major role in the live data migration. This could avoid the downtime of database or shutdown of the database. The information collected from the SQL log files are preprocessed to find the maximum-minimum queries which hit the database. The Meta data stores the details of the database schema and data types of the entities.

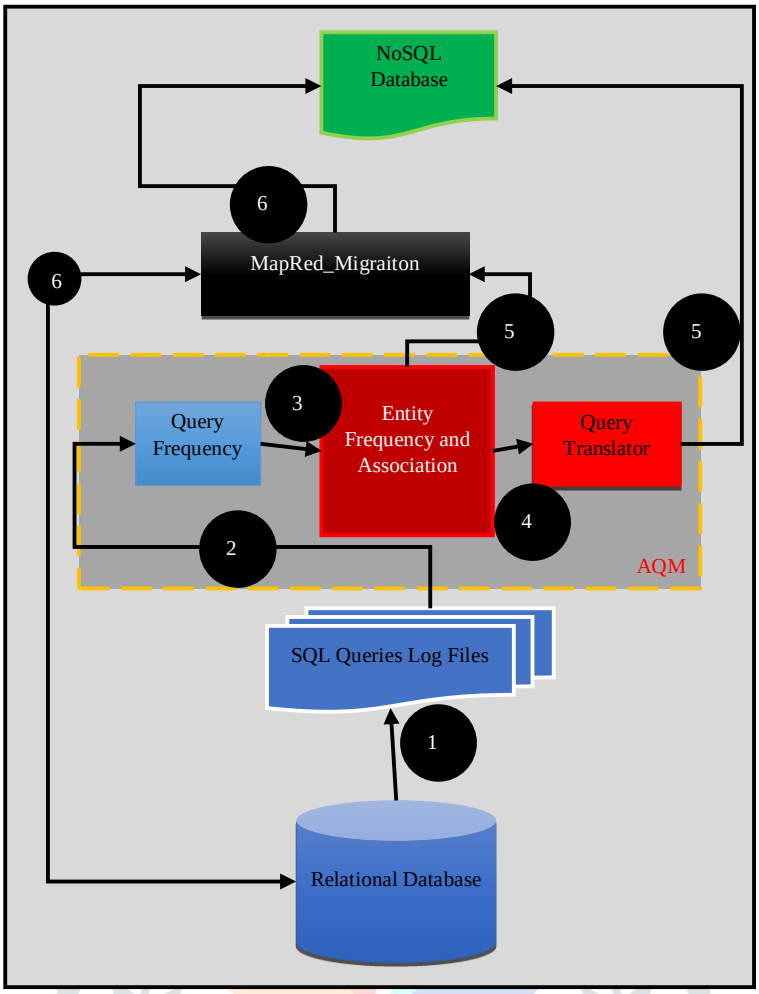

Figure 1. Automated Query Management Flow Diagram

Process 2:

The "query frequency" identifier is used to identify the degrees/orders of the queries issued on the database by the users/admin. The basic ranking method is applied for identifying the query frequency. The detailed algorithm is provided in the successive sub section.

Process 3:

For migrating the entire database without shutting down or downtime, it is a must to migrate the most active entities and attributes of the current database to the source database without any interruption. This process plays a major role to associate the higher frequency of the correlated entities. A novel method is proposed to find the most correlated entity and columns.

Process 4:

Avoiding the database shutdown while data migration, it is to ensure the queries to be used for the target database. "Query Translator" is the process translates the most frequently used query of the source database to the query of the target database. The accuracy of the query translator is manually calculated and evaluated with the existing techniques.

Process 5:

There are two processes running simultaneously at this stage. The first process is to create a token to the appropriate collections and fields created in the NoSQL database based on the ranking of the query associator of the process 3. The second process of this stage is to migrate the data from the SQL database to NoSQL database based on the tokenized collections and fields. The process is explained in detail in the section 5.5.

Process 6:

The final process of this AQM architecture is migrating the big data from SQL to NoSQL using the MapReduce migration process. This function collects data from SQL based on the tokenized column and migrated into the appropriate collections and fields of the NoSQL database.

The above six processes help to perform the Live Big Data Migration without any interruption. This reduces the downtime and avoids the database shutdown.

The proposed technique is evaluated with the e-learning portal database with more than 1 million rows. This web portal is a beta version which allows the public to practice themselves for competitive exams. The datasets used for live big data migration are described in the table 2.

Table 2. Dataset Description

S. No	Database	No. of Tables	No. of Columns
1.	e-Learning Portal Database	84	549

V. AUTOMATED QUERY MANAGEMENT TECHNIQUE

Every database typically has a log document to store activities including setup, change and inquiry. The log document is typically used to screen the database. What's more, the logged tasks can be treated as the inquiries getting to the database. Consequently, when the log record is

examined, it is able to easily identify the entities which are as often as possible use by SQL queries. The above section discussed the overall process of finding the ranking of most frequent entities and attributes. This section provides the techniques and mathematical concepts involved in the automated query management for an efficient live big data migration.

The pseudo code for the overall process is given below in the algorithm 1. There are four major components involved in the process of the Live Big Data Migration. They are described as follows.

- Query Frequency – It helps to identify the most frequent query hit in relational database. The function is explained in the algorithm 2.
- Tables/Entities Frequency – This function supports to measure the degree of frequently used entity and to find the associated entity based on the proposed tf-idf methodology. Algorithm 3 and 4 depicts the process of entity frequency with mathematical notations.
- Query Translator–Query translator plays a significant role in translating the most frequently hit queries of the source database into the query for the target database. This helps to provide the end users to use the database without any interruption. The automated query builder translates the query based on the pre-defined functions stored in a newly created database dictionary. The detailed process of this function is presented in algorithm 5.
- MapReduce based Data Migration –Here the data are migrated from source database into the target database parallelly using the map reduce functions. Algorithm 5 describes the overall process of this function.

Algorithm 1: Automated Query Management for Live Big Data Migration

Input: Relational Database

Output: Non-Relational Database

```
1:  Start
2:  Connect SQL Database
3:  Read the Schema of SQL Database
4:  Create Schema in NoSQL Database
5:  Read SQL Database Query log files
6:  Query_Freq()                    // Algorithm 2
7:  Tables_Freq()                   // Algorithm 3
8:  Query_Translator()              //Algorithm 4
9:  MapRed_Data()          //Algorithm 5
10: Stop
```

5.1 Proposed Method for Query Ranking

The query log files are stored in a single file or in separate files under a specific location. Algorithm 2 explains the procedure to find the maximum query hit in the database. The maximum hit query is ranked based on the frequency of the query used. It is calculated by the MAX( ) function. The retrieved data are stored with the ranking of the queries. This result will impact the execution of the entity associator.

Algorithm 2: Query Frequency Analyzer (QFA)

Input: Query Log Files

Output: Ranking the Query from Higher to Lower degrees

```
1:  Start
2:  d = number of log files
3:  q = number of queries in log files
4:  for i in range(d):
5:      for j in range(q):
6:          count q_j                    // counting frequency of similar queries
7:  MAX(q_j)              // finding the maximum degrees of queries
8:      end for
9:  end for
10: Sort(q_j)                    // sorting the query with higher to lower degrees
11: Stop
```

Mathematical Model for QFA:

Let us assume, d is log files andq is queries, then the following equation (1) & (2) resembles the log files and queries maintaining in the SQL database.

$d \in D, \forall D = \{d_1, d_2, d_3 \ldots d_m\}$ …… (1)

$q \in Q, \forall Q = \{q_1, q_2, q_3 \ldots. q_n\}$ ……... (2)

Queries are stored and updated in log files, therefore q is presented in d. The log files are maintained separately or collectively based on the database administrators' interests. i.e., query will be present in any of the log files. To find that, it is mandatory to union all the log files, which is given in the equation 3.

$q \in \cup D$, where $D = \{d_1, d_2, d_3 \ldots d_m\}$ …… (3)

MAX(q) a function is implied in the equation 3 to find the maximum query used in the SQL database, the derived equation 4 is as follows:

$$MAX(q(\cup D) \quad \ldots\ldots (4)$$

5.2 Entity Associate Aware Technique

The most frequently hit query of the database clearly shows the frequently accessed data by the client side. The query contains the entity and attributes name. There are following possible cases to understand from the query logs, they are as follows:

Case 1: Single entity and single attribute read/write from the database.

Case 2: Multiple entities and multiple attributes read/write from the database.

Case 3: Single entity and multiple attributes read/write from the database.

Case 4: Multiple entities and single attribute read/write from the database.

The above discussed cases produced the complex part to find the most used entity and columns. It is also a challengeable portion to identify the most associated term. This is the major part to build the collections and fields on demand. The new method "entity association aware technique" is proposed to rectify the above issues. This technique finds the most important entity and attribute to migrate at first. It prioritizes the terms based on the Tf-idf along with association rules. The method is explained in the mathematical model of EAAT followed by the algorithm given below. The overall process of EAAT is explained in the algorithm 3.

Algorithm 3: Entity Associate Aware Technique (EAAT)

Input: Sorted query from the query log files

Output: Frequently Accessed Entities (FAE)

1: Start
2: Read Sort(q_j)
3: Tokenize entities E_i and attributes A_j *//i =1,2,3…n and j = 1,2,3…m*
4: Find the frequent of E
5: Find the frequent of A
6: Apply associator_rule for E and A
7: Apply the Associated Attribute-Entity Ranker
8: Stop

The most frequently occurred queries are identified and filtered by the query frequency analyzer which is explained in the above section. These queries are passed in to the EAAT technique. This EAAT process is divided into two stages as depicted in Fig 2. Association rule generator is the first stage, and entity-attribute ranking is the second stage. The core awareness is to remove an arrangement of non-repetitive terms, interrelating to inquiry terms connections in a relevant way. It utilizes the most intriguing and quality connections to process another positioning measure. Later, it calculates maximum support and confidence of the extracted term. Most commonly used tf-idf method is applied to ranking the associated terms in the selected queries.

A. Association Rule Generator

In this work, entity associate aware technique utilizes a strategy which is dependent on strong association among the entity's relationship. Later, it passes to rank affiliation rules as indicated by a blend of an arrangement of measures. This strategy enables to locate the most significant views among the large database. After the completion of the QFA strategy, the queries are passed to the EAAT technique. The queries are presented with the mixed terms. It contains SQL clause statements, entity, attribute, condition statement and connection statement.

Preprocessing is the initial phase in entity-attribute mining process. The queries are tokenized entities-attribute are utilized, illicit characters and stop words (and, or, between, time, where, etc.,) are eliminated using the new stop words dictionary.

In the second phase, the Association Rule Generator (ARG) algorithm is used to find the association among the most frequent entity and attribute relationships. Algorithm 4 describes the pseudo code of the association rule generation. The affiliation among the associated entities and attributes are discovered by using the ARG algorithm. It executes to discover the recurrence of entities-attributes and generates all possible affiliation rules. The following common measures are utilized to evaluate the association rules, they are as follows:

- Confidence and
- Support.

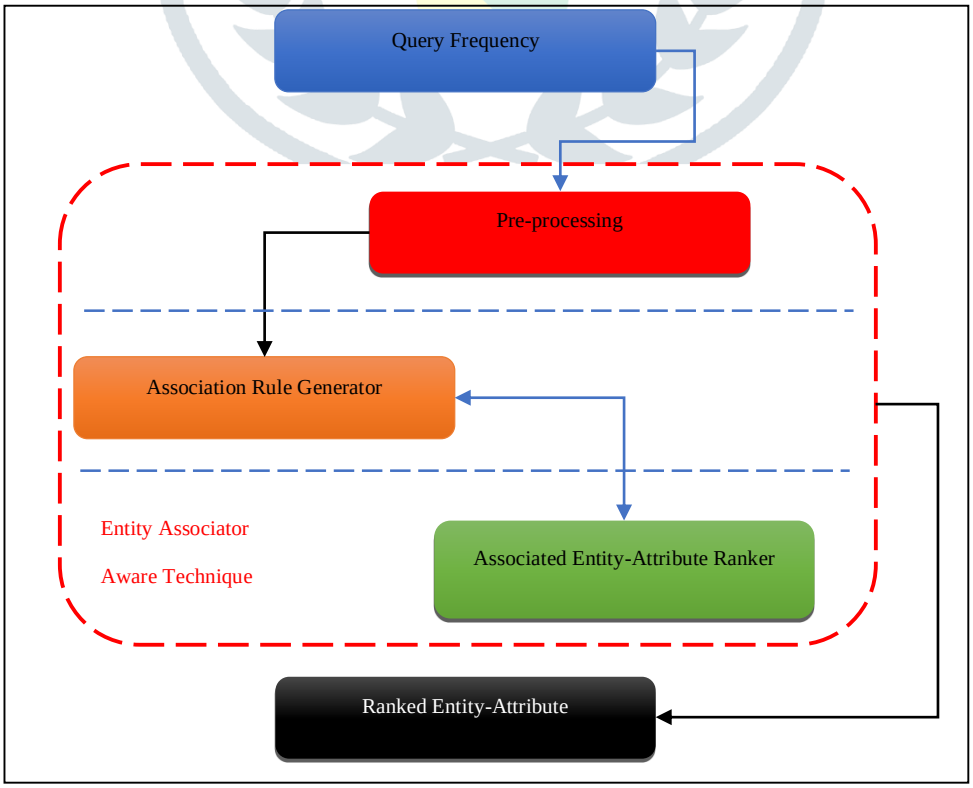

Figure 2. The Process of Entity Associate Aware Technique

The frequently used entity and attribute appeared in the query of transaction is indicated by the S*upport* measure. The support of entity and attribute with respect to Q are defined as the amount of transactions t in the query logs which contain the item set q. The equation 1 and 2 resembles the formula for calculating the support of entity and attributes.

$$Supp(E) = \frac{|\{q \in Q; e \subseteq q\}|}{|Q|} \ldots\ldots\ldots\ldots\ldots\ldots (1)$$

$$Supp(A) = \frac{|\{q \in Q; a \subseteq q\}|}{|Q|} \ldots\ldots\ldots\ldots\ldots\ldots (2)$$

Where E and A are entity and attribute respectively.

How frequent the rule has been occurred in calculated by the term Confidence and it is defined as follows:

Let Q_1 be the query one and be the query two.

The confidence value of an associated rule generator$(Q_1 \Longrightarrow Q_2)$ with respect to a set of transactions T, is the amount of the transactions that holdsQ_1 which also contains Q_2, but it performs different operations. The equation 3 describes the confidence calculation.

$$\text{Conf}(Q_1 \Longrightarrow Q_2) = \frac{Supp(Q_1 \cup Q_2)}{Supp(Q_1)} \ldots\ldots\ldots\ldots (3)$$

Where Q_1 and Q_2 contains the term e and a i.e., entity and attribute respectively.

Algorithm 4: Association Rule Generator

1. Start
2. Function ARG (Q, T, S_{min}, C_{min}, e_{max}, a_{max})
3. q ∈ Q, where q = {q_1, q_2, q_3…..q_m)
4. Let e and a = non-empty set
5. $C_{e|a}$:= $S_{i \in I}${i};
6. While $F_{e|a} \neq \emptyset$ and e and a are maximum
7. Check the C_i of e and a where i = {1,2, 3…. n}
8. MAX(C_i) = MAX(C_{i+1}) */* Check up to n */*
9. Find the S_{max}
10. If S and C are minimum, remove it from the T
11. Generate R using tf-idf */* explained in AER, Algorithm 5 */*
12. Find most entity, attribute and associated entity-attribute
13. Stop

Notations used in the algorithms are described as follows:

Q → Query
C → Confidence
S → Support
Max → Maximum
Min → Minimum
T → Transaction
e → entity
a→ attribute

B. Associated Entity-Attribute Ranker:

This stage comprises in ranking the terms of entity and attribute with the end goal to assure that the retrieved records are important. This is the final phase of EAAT process which is represented in the figure 2. It is built using the vector space model for ranking the terms. The support and confidence of the associator rule generator is calculated based on the rank produced by AEAR. The new ranking measure is proposed to calculate for the associated entity-attribute ranker.

Vector Space Model (VSM)

In the vector space model, [13]an entity in a group of documents is conceptually denoted as vector. Each element of the vector reflects the entity that is extracted from the document with weights representing the position of entity in the document. Normally, the occurrence frequency of an entity in the document is calculated by a weight function [14]. The weight function is calculated by the presence of term in the whole document. If a term or entity is a part of SQL query then it satisfies a non-zero (not empty) value in the document-vector along the dimension equivalent to the term [15, 16]. The VSM can be divided in to the following process.

- Index the entity or attribute in the SQL queries obtained from the QFA.
- Calculate the weight of the indexed entity or attribute.
- Ranking the indexed entity or attribute using the similarity measures.

$w_{e|a}$ is a weighting factor that would be computed for each entity e or attribute a in query q_i to indicate the entity and attribute importance. To end, each query q_i, it would be denoted by a vector of weighted entity or attribute.

$$\begin{cases} w_e > 0 & if\ entity\ e\ occurs\ in\ query\ q \\ w_a > 0 & if\ attribute\ a\ occurs\ in\ query\ q \\ w_e = 0;\ w_a = 0 & otherwise \end{cases} \ldots\ldots\ldots\ldots\ldots\ldots (1)$$

The term frequency tf is the number of times a term appears in a document.

$$tf_{iq} = \frac{f_{iq}}{n_q} \ldots\ldots\ldots\ldots (2)$$

Where f_{iq} is the frequency of entity or attribute i in query q, and n_q is the total number of entities and attributes in the query logs q.

The inverse document frequency is calculated by using the following equation (3)

$$idf_q = \log\left(\frac{n}{n_i}\right) + 1 \quad n_q > 0 \ldots\ldots\ldots\ldots (3)$$

Where n is the number of queries and n_i is the number of entity or attribute in which queryq occurs. The concept of term frequency and inverse document frequency are combined, toproduce a composite weight for each term in each document [17].

$$tf\text{-}idf = tf * idf \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4)$$

The proposed Associated Entity-Attribute Ranker (AEAR)measure is defined as follows in the equation 5. AEAR is obtained by combining the equations (1), (2), (3) and (4).

$$Rank(Q_{e,a}) = \begin{cases} \alpha * tf\text{-}idf(E,A,Q) + (1-\alpha) * conf_{max}(E,A,Qw_{e|a}) * nb \\ if\ conf_{max}(E,A,Q_i) \neq 0 \\ \alpha * tf\text{-}idf(E,A,Q), otherwise \end{cases} \quad \ldots\ldots\ldots (5)$$

Where:

- *tf-idf (E,A, Q)* is the vector space model ranking measure of E and A inQuery *Q*according to query *Q* defined in (2).
- $e \in E, a \in A$where e = {e_1, e_2, e_3 …… e_n} and a = {a_1, a_2, a_3, …... a_m}
- $conf_{max}(E, A, Q_{wi})$ is the maximum of confidences of association rules concerning queryQ, that contains a query keywords $Q_{wi}$where i=1,2,…n
- *nb*is the number of entity and attribute keywords contained in the rule *R*.
- $\alpha = 0.5$,is a weighting parameter.

5.3 Process of SQL to NoSQL Query Translator

Another important process in live big data migration is to plan the process of server and user interaction. To produce uninterrupted service in post migration phase, the user must be able to request and get response from the server database. Hence, it is important build the NoSQL queries according to the SQL queries. Here the query translator plays a vital role to translate the SQL queries to NoSQL queries. This translator build NoSQL query based on the output received by the query frequency analyzer. The most frequently occurred queries are translated to the NoSQL query. This ensures the uninterrupted service and avoids the necessity of database shutdown during the data migration process. The process flow of the query translator is represented in the figure 3. The "sql-to-mongo-db-query-converter[1]" tool is used to translate the MySQL query in to the respective MongoDB query.

The query log file contains the activities of the database and other information. The queries that are used to read and write the data into the database is also stored in the query log files. The QFA analyzes and finds the most frequently accessed queries. As a result, it will return the most accessed queries with its ranking. The top most occurred query is then translated into the MongoDB query by using the above-mentioned tool.

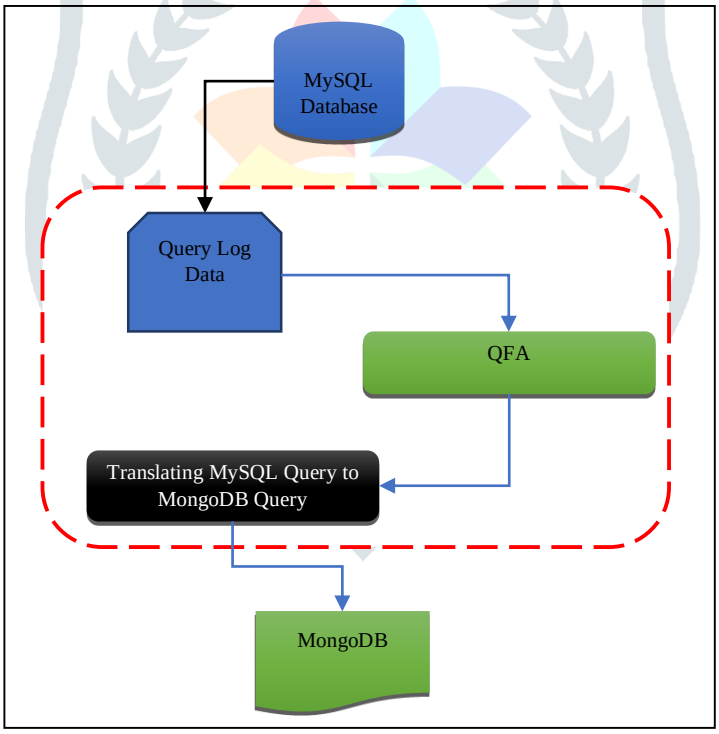

Figure 3. Process Flow of Query Translator

5.4 Process of Migrating Big Data from SQL to NoSQL using Map-Reduce Model

This is the final process of the proposed AQM technique for big data migration from SQL database to NoSQL database. Map-Reduce based algorithm is used for the huge data migration [19]. The data migration is carried out by the ranking method. The most frequently used entity and attribute are built in the target database by using the above process. Data are being migrated based on the high-relevance order.

Initially, it transfers the data of the most occurred entities and attributes of the source database to target database. Later it will move the rest data. Each entity and attribute of the source database are tokenized and converted to the target database with specific ID. Data that are stored in the rarely used entity and attribute are migrated at the end of the entire process. In erratic cases, this step may conflict in the data retrieval. The process of Map-Reduce based data migration from SQL to NoSQL database is given in the algorithm 5.

[1]https://github.com/vincentrussell/sql-to-mongo-db-query-converter

The overall process of the Map-Reduce based data migration for this proposed technique is represented in the figure 4. The process of EAAT method is described in the figure 4. The processed data from the EAAT is passed to the map-reduce phase. The most occurred entity and attribute is considered as key and the instances of the entity and attribute is considered as values. Based on this it generated the key-value pairs. The number of mappers is limited to four, due to lack of the huge data. Four mappers are sufficient to process the collected data.

After the EAAT process, the map-reduce process is initiated. The pseudo code of this process is given in the algorithm 5.

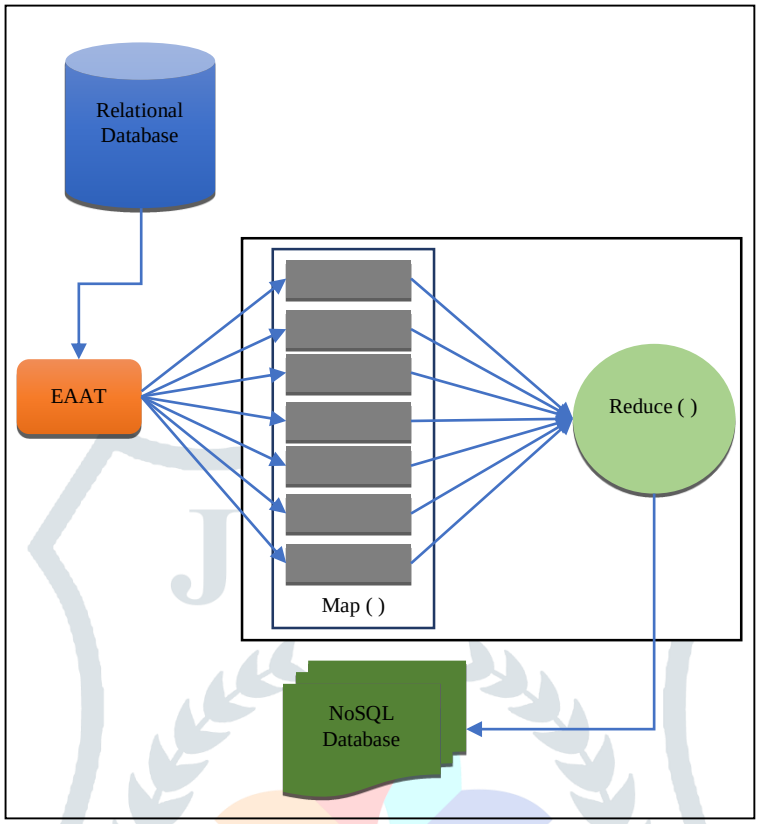

Figure 4. Processes of Map-Reduce Big Data Migration

Algorithm 5: Map Reduce based Data Migration from SQL to NoSQL Database

Input: MySQL database

Output: MongoDB Database

1: Start
2: Mapper function: *// input data*
3: Set the number of mappers
4: Set number of nodes
5: Assigns the entity and attribute as key
6: Assigns the instance as the value of the keys
7: If keys ≠ 0; go to step 8
8: else go to step 15
9: Sort function: *// data arrangements*
10: Obtained the rank of entity and attributes from EAAT
11: Sorted based on the high relevancy to low
12: Reduce Function: *// output data (migrated)*
13: Create appropriate collections and fields in target database
14: Migrate data based on the key-value pairs
15: Stop

The Map phase works with the following steps:

1. The number of mappers is set to 4, since the data is not very huge. Based on the mappers across the compute nodes, it chunks the input data which are stored in relational database.
2. Each mapper splits the data into individual pieces for each chunk.
3. The mapper function receives the number chunks split by the splits.
4. The chunks are the internal part of the storage disk.
5. Based on the entity and attribute produced by the EAAT, the Key is assigned as the entity and attribute and value is the instances.

From the above process it is clearly understood that the map phase's job is to transform the raw data into a sequence of key-value pairs. These key-value pairs are sorted by the sorting function. The sorting function is an internal process of the map-reduce process. The sort function ordered the input data based on the requirement. It is performed based on the rank produced by the EAAT technique. The data are wrapped from all the mappers and discharged all of their key-value pairs. Later, those values are arranged by their keys and sent to the reducer phase. In this research work, it has considered four mappers and one reducer.

The Reduce phase works with the following steps:

Reducer gets the key-value pairs so that all key-value pairs having a similar key dependably go to the reducer. The collections and fields of target databases are mapped with the entity and attribute of the source database. The key of the source database is mapped with the key of the target database. The values are migrated if the keys are similar since, the values are dependent of the keys. Due to the data size the reducer is limited to one. Reducer also performs some kind of sorting to store the migrated data.

VI. RESULTS AND DISCUSSIONS

The proposed AQM technique is evaluated with two different datasets. Dataset obtained from the www.examey.comis a real-time dataset. Another dataset is W3School database. The performance of the proposed association rule generation is compared with the existing Jaccard and baseline tf-idf methods. The results are interpreted in different ways.

6.1 Comparison Matrix of Associated Entities

Due to the large space requirement, top 8 entities out of 84 entities in the examey database are considered for the comparison matrix evaluation. The top entities are filtered by considering the degree above 25. Table 2 represents the comparison matrix of the associated entity. The values are referred as degrees, in simple term; the total number of access among entity is given as degree. The relationship is calculated based on one-to-one and one-to-many relationship. In the table 2, T represents the table/entities in the database. From the table 2, it is evident that entity 1 (T1) associates with entity 76 (T76) is more frequently accessed.

Table 3. Matrix Comparison of Associated Entities

	T1	T2	T6	T8	T11	T15	T29	T76
T1	56	28	45	25	34	31	35	76
T2		36	25	37	29	31	29	25
T6			35	33	27	27	25	29
T8				31	25	26	25	28
T11					25	27	28	29
T15						25	27	26
T29							34	27
T76								58

Figure 5 represents the chart of the matrix comparison distribution of the associated entities. X-axis denotes the table/entities and Y-axis denotes the degree of the table association.

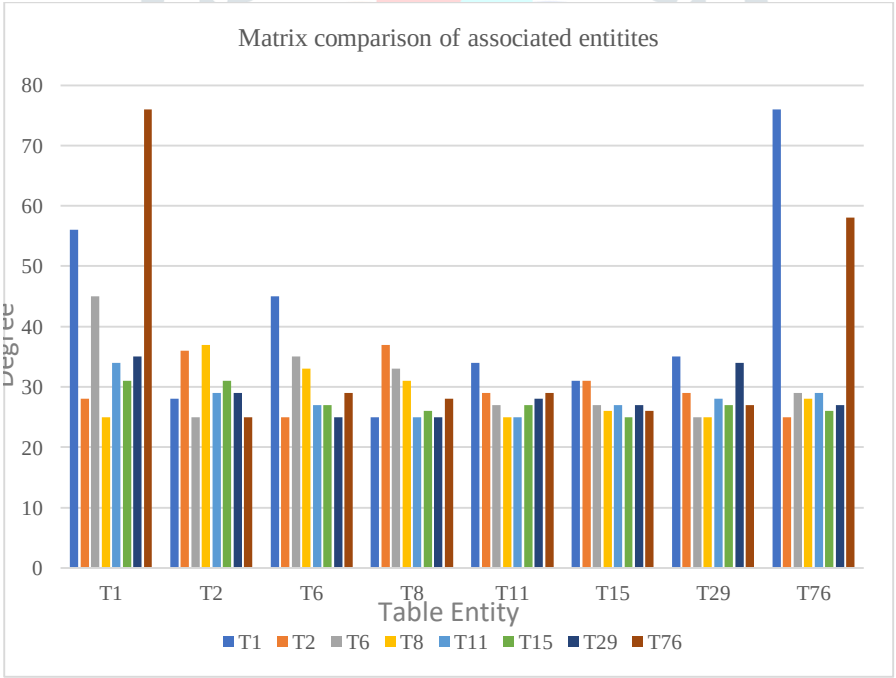

Figure 5. Entity Association Comparison Matrix Chart

The proposed EAAT model is compared with the existing Jaccard and Tf-Idf ranking methods. Vector-Space model based ranking method produced better results than the existing models. The results are represented in the table 4 given below.

Table 4. Accuracy of Entity-Attribute Correlation

Method	Precision	Recall	F-Score	Accuracy
Jaccard	91.7	89.4	90.5	92.1
Tf-Idf (baseline)	92.3	90.2	91.3	93.9
EAAT	94.3	93.7	94	95

The graphical comparison of the proposed EAAT model with the existing ranking models is shown in the figure 6. Accuracy, precision, recall and F-score of the proposed EAAT model is higher than the existing models. The proposed tf-idf along with the vector space model produced better results than the existing tf-idf baseline model.

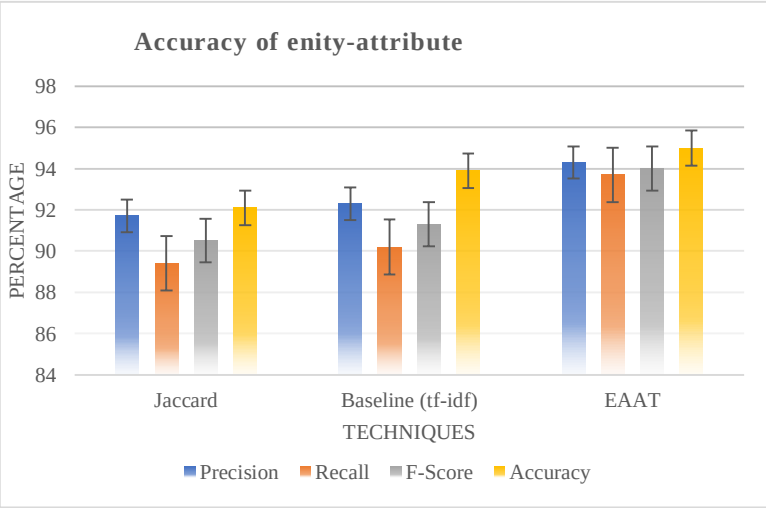

Figure 6. Accuracy of entity - attribute

VII. CONCLUSIONS

The proposed technique is well-suited for the live big data migration. The purpose of database server shutdown is dominantly avoided and provides live data migration. In the existing methodology, most of the database migration practices were carried out using the database shutdown downtime. This affects the real-time user. The proposed methodology overcomes the above issue. The two processes have played a vital role in supporting the live data migration. The proposed technique has several features such as query frequency analyzer, associator rule generator, associated entity-attribute ranker, query translator and map-reduce based data migration.

A novel feature Entity-Attribute Associate Aware Technique plays a major role in finding the associated entity and attribute. New formula has been proposed to find the rank of the associated terms. Vector space model along with the proposed associate rule generator is used to find the rank of the terms. This produced better results than the existing similarity measures. It is evident that there is a 2 percentage of increase in the precision, recall, F-score and accuracy of the proposed work when compared with the existing work. It also found that the total migration time taken by this technique is costlier than the previous research contributions. The migration time of the frequently accessed entity and attribute is very minimum than the previous works.

References:

[1] Masato Asahara, Shinji Nakadai and Takuya Araki, “LoadAtomizer: A Locality and I/O Load aware Task Scheduler for MapReduce”, in 4th IEEE International Conference on Cloud Computing Technology and Science (CloudCom), pp. 317-324, 2012.
[2] Kasim SelcukCandan, Jong Wook Kim, ParthNagarkar, Mithila Nagendra and Ren-wei Yu, “Scalable Multimedia Data Processing in Server Clusters,” IEEE MultiMedia, pp. 3-5, 2010.
[3] C. Jin and R. Buyya, “Mapreduce programming model for net-based cloud computing, ”in Proceedings of the 15th International Euro-Par Conference on Parallel Processing, Euro-Par (Berlin, Heidelberg), pp. 417–428, 2009.
[4] MateiZaharia, Andy Konwinski, Anthony D. Joseph, Randy Katz and Ion Stoica, “Improving MapReduce Performance in Heterogeneous Environments,” 8th Symposium on Operating Systems Design and Implementation, pp. 29–42, 2008. Dec.
[5] Andrey Balmin, Tim Kaldewey, Sandeep Tata”, Clydesdale: Structured Data Processing on Hadoop, “ACM SIGMOD International Conference on Management of Data, pp. 705-708,2012.
[6] Jenq-Shiou Leu, Yun-Sun Yee, Wa-Lin Chen, “Comparison of Map-Reduce and SQL on Large-scale Data Processing,” International Symposium on Parallel and Distributed Processing with Applications, pp. 244-248, 2010.
[7] Steven Lynden, Yusuke Tanimura, Isao Kojima and AkiyoshiMatono,” Dynamic Data Redistribution for MapReduce Joins,” IEEE International Conference on Coud Computing Technology and Science, pp. 717-723, 2011.
[8] Dawei Jiang, Anthony K. H. Tung, and Gang Chen,” MAP-JOIN-REDUCE: Toward Scalable and Efficient Data Analysis on Large Clusters,” IEEE Transactions on knowledge and Data Engineering, vol. 23, no. 9, pp. 1299-1311, 2011.
[9] Hung-Ping Lin, “Structured Data Processing on MapReduce in NoSQL Database”, Master Thesis in National Chiao Tung University, 2010.
[10] Meng-Ju Hsieh, Chao-Rui Chang, Jan-Jan Wu, Pangfeng Liu and Li-Yung Ho, ”SQLMR : A Scalable Database Management System for Cloud Computing, ”International Conference on Parallel Processing (ICPP), pp. 315-324, 2011.
[11] Rubao Lee, Tian Luo, Yin Huai, Fusheng Wang, Yongqiang He, and Xiaodong Zhang, ”YSmart: Yet Another SQL-to-MapReduce Translator, ” International Conference on Distributed Computing Systems, pp. 25-36, 2011.
[12] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, and R. Murthy, “Hive - a warehousing solution over a Map-Reduce framework, ”PVLDB, vol. 2, no. 2, pp. 1626–1629, 2009.
[13] Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. Communications of the ACM, 18(11), 613-620.

[14] Dumais, S. T. (1991). improving the retrieval of information from external sources. Behavior Research Methods, 23(2), 229-236.
[15] Siham JABRI, Azzeddine DAHBI, Abdelhak BASSIR, Taoufiq GADI "Ranking of Text Documents using TF-IDF Weighting and Association Rules mining", IEEE, 2018.
[16] Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. Information processing & management, 24(5), 513-523.
[17] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt and Andrew Warfield, "Xen and the Art of Virtualization", SOSP '03 Proceedings of the nineteenth ACM symposium on Operating systems principles, vol. 37, Issue 5, pp. 164-177, December 2003.
[18] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C., Hsieh Deborah A., Wallach Mike Burrws, Tushar Chandra, Andrew Fikes, and Robert E.Gruber, "Bigtable: A Distributed Storage System for Structured Data," 7th UENIX Symposium on Operating Systems Design and Implementation, pp. 205-218, 2006.
[19] Jeffrey Dean and Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Communications of the ACM, vol. 51, no. 1, pp. 107–113, 2008.
[20] Sven Groot, "Jumbo: Beyond MapReduce for Workload Balancing," Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference on Cloud Computing Technology and Science, vol. 4, pp. 2675-2678, 2011. July.