RESEARCH

Open Access

An empirical comparison of the performances of single structure columnar in-memory and disk-resident data storage techniques using healthcare big data

R. F. Famutimi*, M. O. Oyelami, A. O. Ibitoye and O. M. Awoniran

*Correspondence:
ranti.famutimi@bowen.edu.ng

Bowen University, Iwo, Nigeria

Abstract

Healthcare data in images, texts and other unstructured formats have continued to grow exponentially while generating storage concerns. Even though there are other complexities, volume complexity is a major challenge for Disk-Resident technique in storage optimization. Hence, this research aimed to empirically compare the efficiency of Disk-Resident and In-Memory single structure database technique (as opposed to multiple structure In-Memory database), using descriptive and inferential big data analytical approaches. The essence was to discover a more cost-effective storage option for healthcare big data. Data from Nigerian Health Insurance Scheme (NHIS) alongside sample patients' history from Made-in-Nigeria Primary Healthcare Information System (MINPHIS) which included patients' investigation, patients' bio-data and patients' diagnoses were the primary data for this research. An implementation of both Disk-Resident and single structure In-Memory resident data storage was carried out on these big data sources. After storage, each quantity of data items stored for different data items in Disk-Resident was then compared with that of single structure In-Memory resident system using size of items as comparison criteria and different analyses made.

The results obtained showed that single structure In-Memory technique conserved up to 90.57% of memory spaces with respect to the traditional (common) Disk-Resident technique for text data items. This shows that with this In-Memory technique, an improved performance in terms of storage was obtained.

Keywords: Healthcare big data, Disk-resident database, Columnar in-memory-resident database, Big data analytics, Descriptive analysis, Big data volume complexity

Introduction

The amount of data produced worldwide in this digital era is said to double itself every three (3) years and with time, this will result in data overload [1]. The moment data is beyond the capacity of the existing tools to process it, it is gradually becoming big data. The characteristics of big data among others include the following five V's: Volume, Velocity, Variety, Veracity and Value. While Volume relates to huge amounts of data, Velocity applies to the high pace at which new data is generated, Variety is the level of complexity of the data, Veracity measures the genuineness of the data and Value evaluates the usefulness of the data [2, 3]. In order to ascertain the veracity of big data sources, a methodology that confirms the sources with the normal health data sources was proposed [4]. Big data has a very large size typically, to the extent that its manipulation and management present significant challenges such as accessing, storage, integration, privacy and security. Although there are common ways of addressing these challenges such as investing in more hardware infrastructure, through the use of distributed databases and the employment of divide and conquer approach, the implication of these approaches is that of continuous investment in hardware and personnel resources [5]. Since the emergence of Big Data Analytics in virtually every sphere of human activity including healthcare, it has continued to improve patient wellbeing, provide personalized healthcare delivery, improve provider relationships with patients and reduce medical spending [6] while increasing accuracy in healthcare decision support services. When big data is combined with high powered and efficient analytical tools, greater tasks such as the following can be accomplished: predicting the root cause of an illness from accumulated cases treated, a possible outbreak of an epidemic, across a geographical spread of a particular disease and the possible period of the year a disease is likely to occur. Also, analyzing treatment patterns by a physician can provide a better way of treating an ailment. In area of policy making, the vast amounts of data that are being generated by different sources creates an opportunity for public authorities and stakeholders to be able to create, analyse, evaluate and optimize policies [7]. This researcher worked on opportunities created by Big Data for public authorities in proper decision making. In manufacturing sector, industrial big data and artificial intelligence are now in the forefront of a new era of manufacturing known as smart manufacturing [8]. This author worked on the use of Big Data for smart manufacturing. Using Big Data, a fraudulent tradition can be detected before it affects an organization [9]. This author worked on the use of Big Data for fraudulent activities' detection. Consequently, based on National Health ICT Strategic Framework 2015–2020, developed for the achievement of Universal Health Coverage in Nigeria, two components expected from health-ICT are the ability to capture and exchange patient-level healthcare information and to be able to exchange and report aggregate healthcare information [10]. Big data in health ICT is the utmost approach for proper maintenance of aggregate healthcare information as well as comprehensive patient-level healthcare information in Nigeria.

Researchers [4, 7, 8] on Big Data did not focus on how to effectively manage the size in environments where resources are of paramount importance, they only worked on the usage of Big Data.

This research work deals with the Volume challenge in big data storage when using the Disk-Resident storage approach and hence implemented an improved In-Memory approach that stores data in the memory in coded form without the need to decode when extracting the stored data. In presenting this research work, "Resolving Big Data Voluminosity Challenge and Related Work" Section of this paper reviews literature on the voluminosity challenge of Big Data and some related work, "Descriptive and Inference Data Analytics" Section is on Descriptive and Inference Data Analytics on Big Data as a basis for justifying the result obtained in the study, "Methods" Section is on

Methods, data and environment used for the research work, "Results" Section is on the Result obtained, "Discussion—conclusions" Section on Discussion and Conclusions and "Summary" Section on the Summary of the work.

Resolving big data voluminosity challenge and related work

Recent technology has revealed that data can be managed by database systems using either In-disk-based techniques or In-Memory-based technique. While the In-disk-based techniques have been around for quite a long time, the In-Memory based technique is an evolving technique of data management, and it is still in its infancy [11, 12]. In an In-disk-based technique, data resides permanently in the hard disk and it is moved to and from the memory when needed depending on the algorithm being used for memory management. An In-Memory database system or main-memory database system is a breed of database management system that relies on main memory for storage instead of using secondary storage. For generic In-disk- based Database Management System (DBMS) architecture, it is becoming more and more challenging to process data and to produce analytical results in real time (Online Analytical Processing (OLAP)). For In-disk databases, disk I/O operations are the main bottlenecks, which are very slow operations and cannot be optimized beyond a limit, being mechanical in nature [13]. In spite of the fact that traditional In-disk DBMSs have introduced various measures to speed up their operations through different techniques, this has not brought the much-desired result. As a result of advancement in technological innovations, the cost of main memory is drastically reducing and this makes storage of large amount of data in main memory affordable. The moment data are stored in the main memory, the speed of accessing and manipulation improves drastically. If data resides in the main memory, there will be no need for caching, which on its own creates another problem. Therefore, with the noticeable advent of new applications and upcoming hardware improvements, tremendous changes are expected to take place in enterprise software. The next generation database technologies will clearly deviate from traditional databases of In-disk to In-Memory database technology. According to Plattner, it has been established that the use of In-Memory database technology enables performance improvements by factors of up to 100,000 [14]. The use of In-Memory database also resulted into a speedup of 1.35X to 11.28X [15]. Plattner's work was however based on multiple field structure databases. This work employs the use of single field structure database to reduce the overhead incurred by the multiple field structure databases.

Single and multiple field structure columnar in-memory databases

In a multiple structure Columnar In-Memory database, the identifier table contains the record identifier item, and for every available field item in the table, a field item identifier (that points to the dictionary) is created and maintained. For an example, if we are considering data items like surname, first name, phone number and city, the identifier table will contain the record identifier, surname, first name, phone number and city fields. All these fields will contain numbers that will be pointing to the dictionary. The number of fields being stored is proportional to the number of the attributes used.

For the single structure Columnar In-Memory database used in this work, the identifier table contains only the record identifier field and the items identifier field. The total

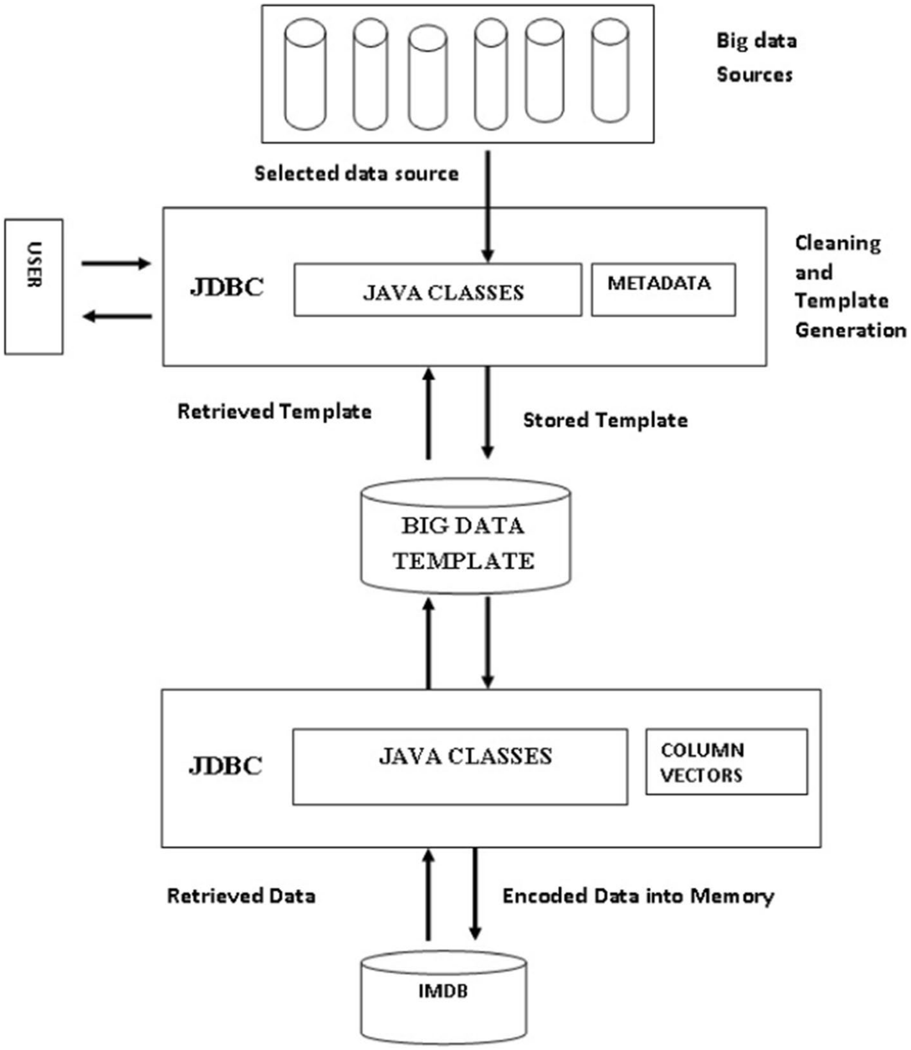


Fig. 1 Metadata Extraction Process

number of fields used in this case was always two irrespective of the number of attributes (in the record) to be managed. All the items' fields are concatenated into a single field which is split into individual fields while retrieving records from the dictionary. "Descriptive and inference data analytics" Section describes how data analysis could be carried out on Big Data using either descriptive or inferential statistics or both so as to justify the comparison criteria used in this study.

Descriptive and inference data analytics

Descriptive data analytics explains the characteristics of a given dataset [16]. It gives a description of the properties of the dataset and their possible relationships. The three important measures in descriptive analytics are the distribution, measures of central tendency and measures of dispersion [17]. The distribution gives the frequency of the different outcomes of the properties of a dataset. Value counts are the most important aspect of a distribution. Measures of central tendencies summarize datasets by giving the central values in the dataset. Common measures under this include mean, median and mode. Measures of dispersion (or variability) describe the spread of values in the dataset. The identification of dispersion within a dataset relies heavily on the understanding of its central tendencies. Common measures of dispersion are standard deviation, variance, range, kurtosis and skewness. Results of descriptive analytics are usually concisely represented using graphs, tables and charts [18]. Inferential data analytics makes inference es on a larger population of data by using a sample from the data [19, 20]. The main aim of this process is to draw conclusions from the sample data and extend the conclusions to a larger dataset from which the sample data was drawn. Probability theory is used to determine the characteristics of the sample that will best represent the overall population of interest. Common methodologies under inferential data analytics are hypothesis tests and analysis of variance [21].

Methods

Two data sources were used for the implementation. The first is Nigeria Health Insurance Scheme (NHIS) registration data, as of December 16, 2015. It was obtained from Wise Health Services Limited; an NHIS accredited Health Maintenance Organization (HMO) with ethic and legal compliance and approval. The second data set was Patient history obtained (with the approval of ethics and legal committee as well) from "Made in Nigeria Primary Healthcare Information System (MINPHIS)", of which the lead author of this paper is a member of the research team that implemented the system.

In order to carry out the comparative study on volume complexity using healthcare big data, an implementation of both Disk-Resident (conventional data storage on disk) and an In-Memory resident (textual data residing in memory) big data was carried out on specific healthcare big data sources. The big data sources were clinical data which included patients' investigation and imaging data, patients' bio-data, patients' diagnoses, medications, billings, epidemiology and behavioral data, national health insurance scheme data and genetics data. These sources served as raw textual formats which were encoded using dictionary encoding format before storage in the memory (In-Memory coded format). Figure 1 illustrates, the process of clinical metadata extraction.

The metadata extraction stage consists of the following sub-stages: big data sources input, user interface interaction, Java Database connectivity and the Java classes modules used to manipulate the metadata. The first stage generates the big data template. This is then operated upon by another set of JDBC and Java classes modules together with the individual data field known as column vectors to produce the In-Memory database. In order to ensure the security of the In-Memory Database, Network security mechanism was deployed.

MySQL Database Management System (DBMS) was then used for the Disk-Resident implementation, for storing the data and the volume occupied by individual attributes were recorded accordingly. Any DBMS could be used, however, MySQL was used because of its open source. The same approach was used for the In-Memory technique for managing big data complexities [22]. Huffman algorithm's prefix coding methodology was adopted for the implementation of the dictionary encoding system

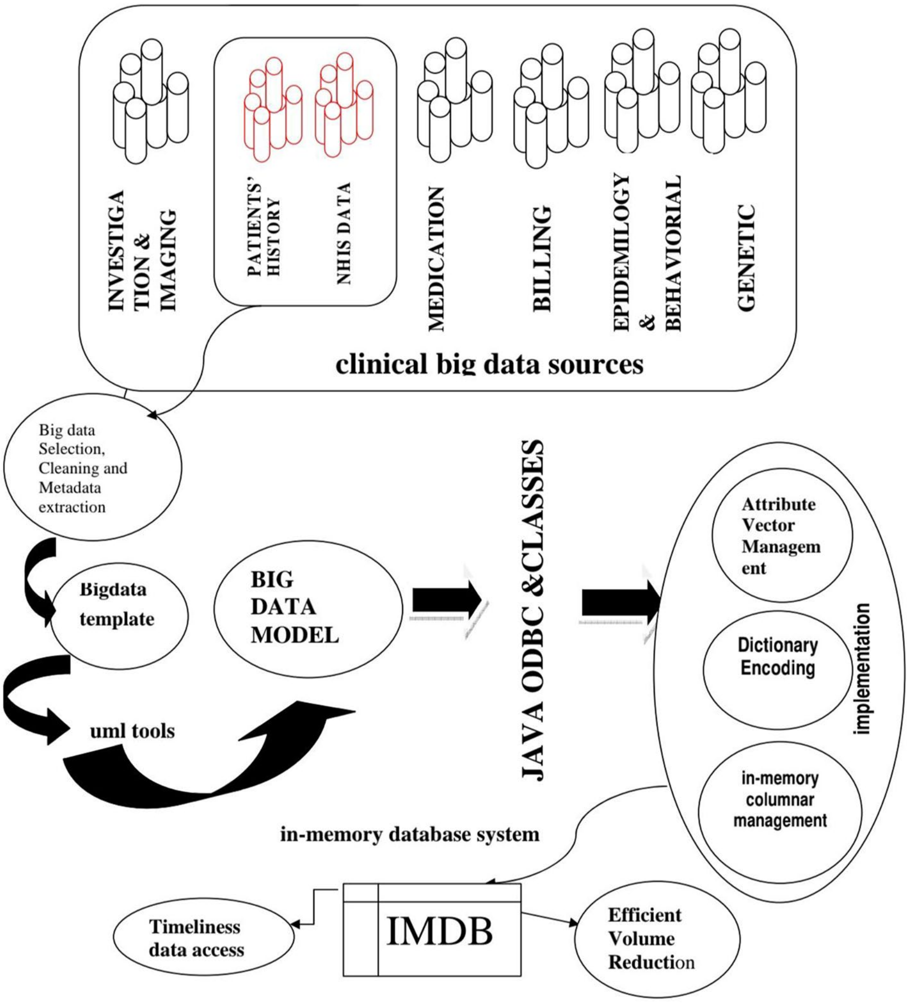

INVESTIGA TION & IMAGING
PATIENTS' HISTORY
NHIS DATA
MEDICATION
BILLING
EPIDEMILOGY & BEHAVIORIAL
GENETIC
clinical big data sources
Big data Selection, Cleaning and Metadata extraction
Bigdata template
BIG DATA MODEL
uml tools
JAVA ODBC &CLASSES
Attribute Vector Managem ent
Dictionary Encoding
in-memory columnar management
implementation
in-memory database system
IMDB
Timeliness data access
Efficient Volume Reduction

Fig. 2 Framework of the Research Model [7]

for the In-Memory database. This is denoted by Eq. 1. 'S' stands for message (data) and 'W' stands for code word.

$$C = \left\{\left(s^1, w^1\right), \left(s^2, w^2\right), \ldots\ldots\ldots\ldots..\left(s^n, w^n\right); s^k <> s^m \text{due to encoding}\right\} \quad (1)$$

The code word for message s^1in S is w^1,the code word for message s^2in S is w^2 and the code.

word for message s^nin S is w^n. The total size (entries) of theIn-Memorydatabase is denoted by Eq. 2.

$$In - Memory\, database = \sum\nolimits_{k=0}^{n} \left(S^k + W^k\right) \quad (2)$$

(where S and W are entries).

The phases consist of the transformation system and the In-Memory database construction. In the transformation system, appropriate health related data was selected and

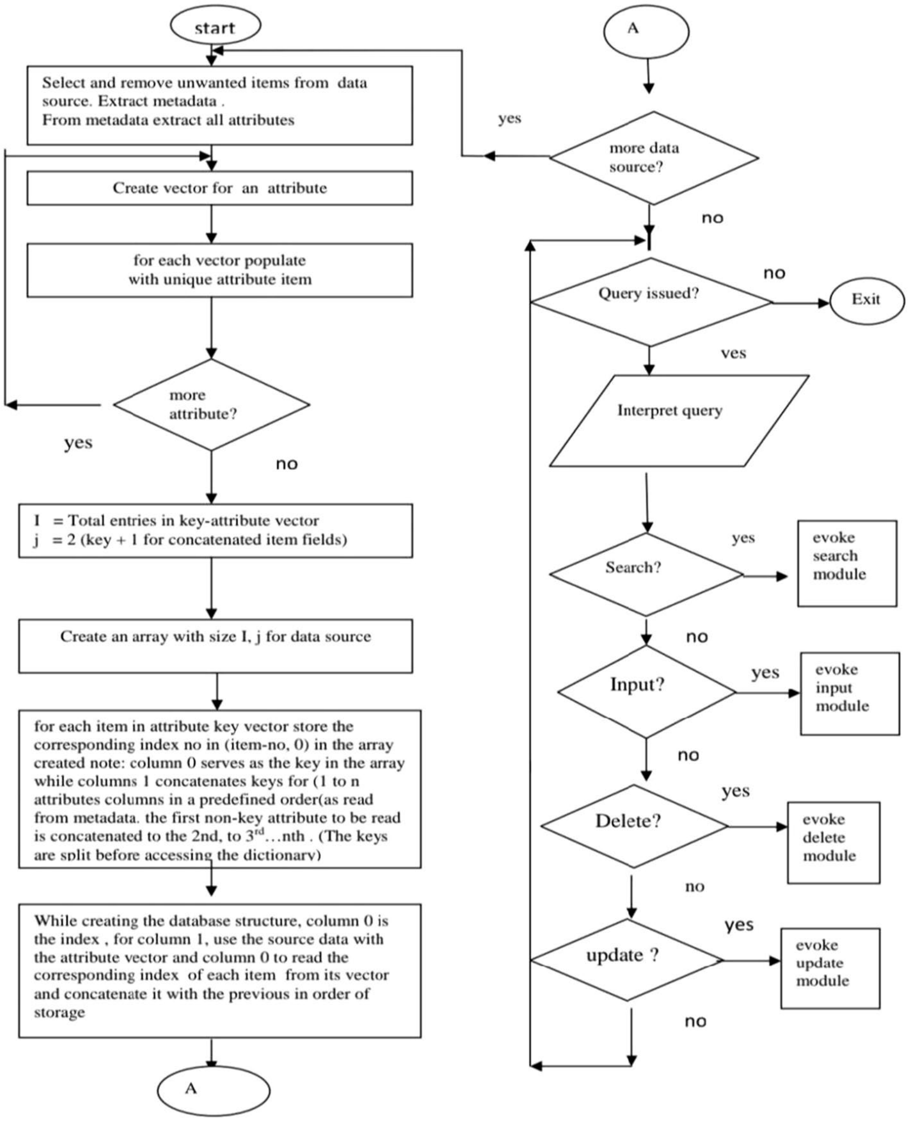

start
Select and remove unwanted items from data source. Extract metadata .
From metadata extract all attributes
Create vector for an attribute
for each vector populate with unique attribute item
more attribute?
yes
no
I = Total entries in key-attribute vector
j = 2 (key + 1 for concatenated item fields)
Create an array with size I, j for data source
for each item in attribute key vector store the corresponding index no in (item-no, 0) in the array created note: column 0 serves as the key in the array while columns 1 concatenates keys for (1 to n attributes columns in a predefined order(as read from metadata. the first non-key attribute to be read is concatenated to the 2nd, to 3rd…nth . (The keys are split before accessing the dictionary)
While creating the database structure, column 0 is the index , for column 1, use the source data with the attribute vector and column 0 to read the corresponding index of each item from its vector and concatenate it with the previous in order of storage
A
A
yes
more data source?
no
no
Query issued?
Exit
yes
Interpret query
yes
Search?
evoke search module
no
Input?
yes
evoke input module
no
yes
Delete?
evoke delete module
no
yes
update ?
evoke update module
no

Fig. 3 Operational Flowchart

cleaned, after which metadata was extracted from this source. Then integrating, mapping and reduction of the data source were carried out so as to produce a transformed big data template.

The In-Memory database construction employed three subsections to work on the transformed big data template before producing the In-Memory database. These subsections are: attribute vector management, dictionary encoding and In-Memory columnar management. The attribute vector constructs and manages the attribute of the new In-Memory database, the dictionary encoding is used for data compression and decompression purposes on the data while the In-Memory columnar management manages the generated column wise record. The management of In-Memory database consists

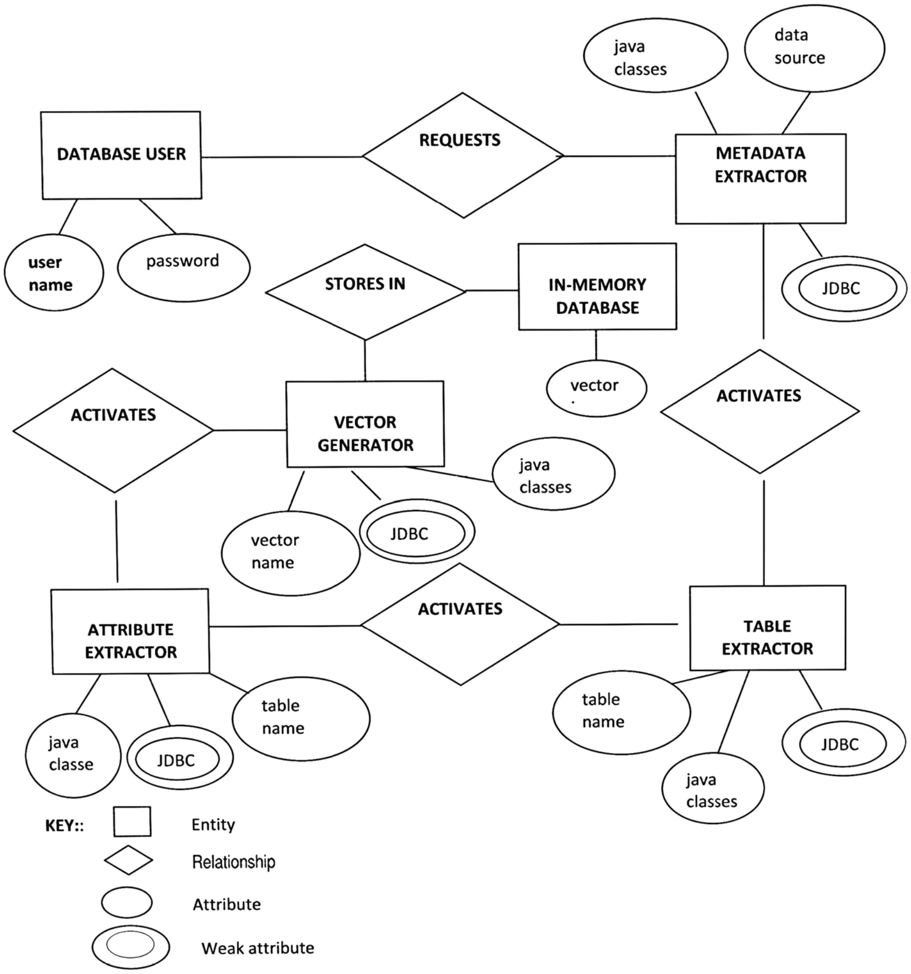


Fig. 4 Conceptual Model

of: inserting new records, updating the existing ones and deleting unwanted records. The framework of the research model is shown in Fig. 2. The various health related data sources are shown. Among these are: investigation and imaging, patients' history, NHIS data, medication, billing, epidemiology, behavioural and genetic. From the big data sources, those that had repeated items most were selected cleaned and metadata extracted to obtain a big data template. From the big data template, unified modeling language (UML) tools were applied to generate the big data model. The modeled big data was then implemented as In-Memory big data management using Java Open Database Connectivity (JDBC) tools and Java Classes.

The resultant implementation was then tested for volume reduction. Figure 3 showed the operational flowchart of the system. It illustrates the flow of activities from the selection of big data source, how memory vectors were created and populated with the contents of the selected data source, the creation of the dictionary encoding using the index numbers of the vectors and also the various sub-modules that were used for the maintenance such as the search, input, delete and update. Some of the UML diagrams are:

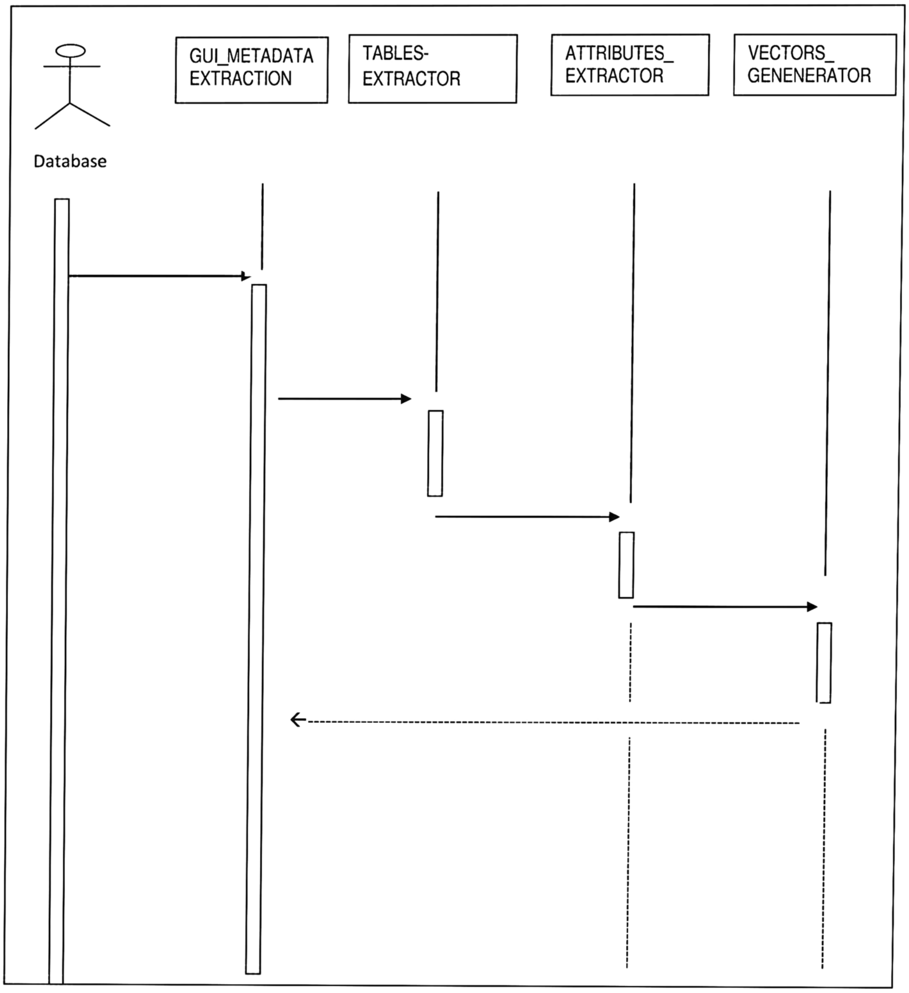


Fig. 5 Sequence diagram of vector generation

Fig. 4 the conceptual model, Fig. 5 sequence diagram of In-Memory vector generation, Fig. 6 class diagram and Fig. 7 the activity diagram of In-Memory vector management.

Results

Some sample screenshots of the implementation are shown as follows: Encoded Dictionary Vector shown in Fig. 8, In-Memory Database Management window in Fig. 9, In-Memory Database result for patient id attribute in Fig. 10, In-Memory Database result for patient surname attribute in Fig. 11 and In-Memory Database result for patient sex attribute in Fig. 12. When the National Health Insurance Scheme was the chosen data source, Table 1 shows the descriptive and inference statistics of storage space usage for both Disk-Resident and single structure Columnar In-Memory resident databases in terms of the number of entries (items). For identifier attribute, the Disk-Resident approach stored 1,056,132 entries while the In-Memory approach stored 9 entries; this gave a compression factor of 117,348 and savings of 99.9%. For first-name attribute, the Disk-Resident approach stored 1,056,132 entries while

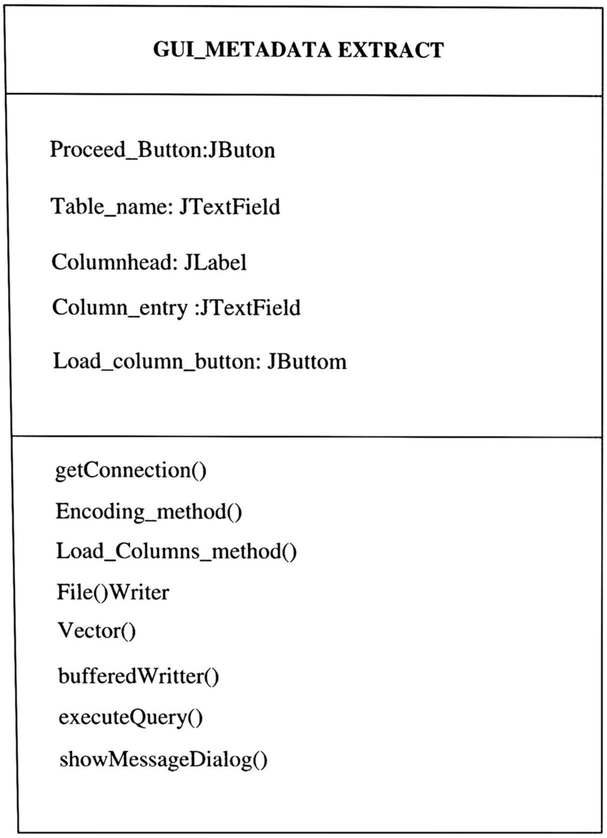


Fig. 6 Class diagram of In-Memory vector generation

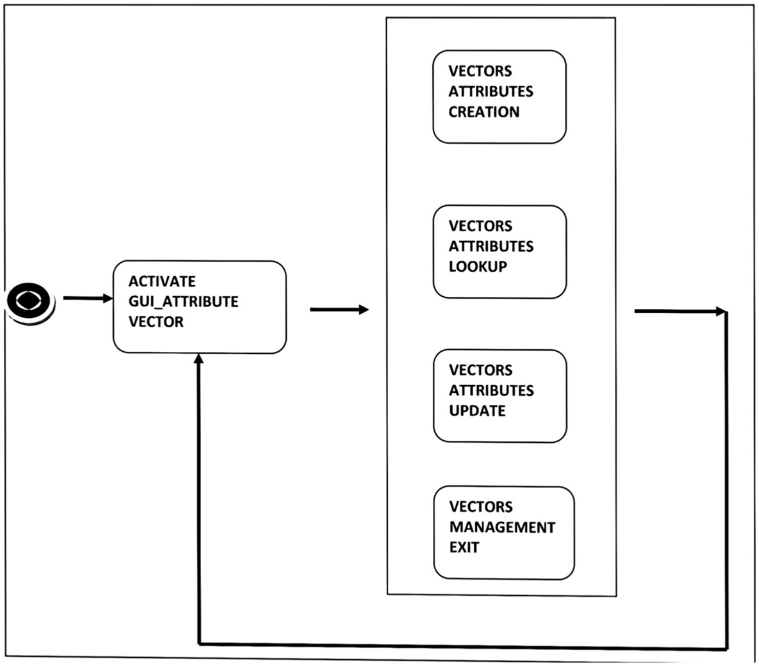
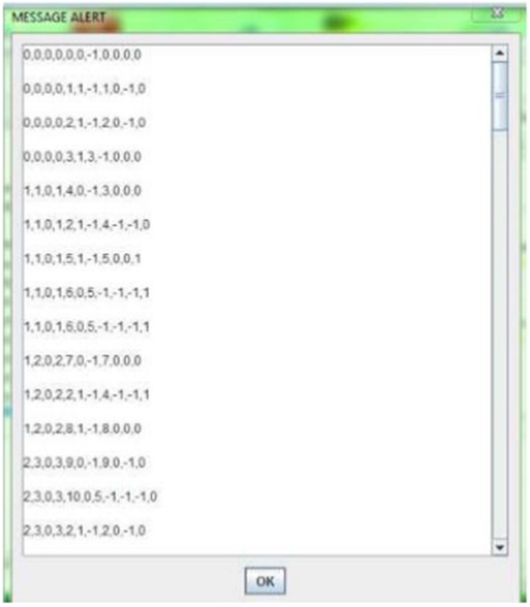
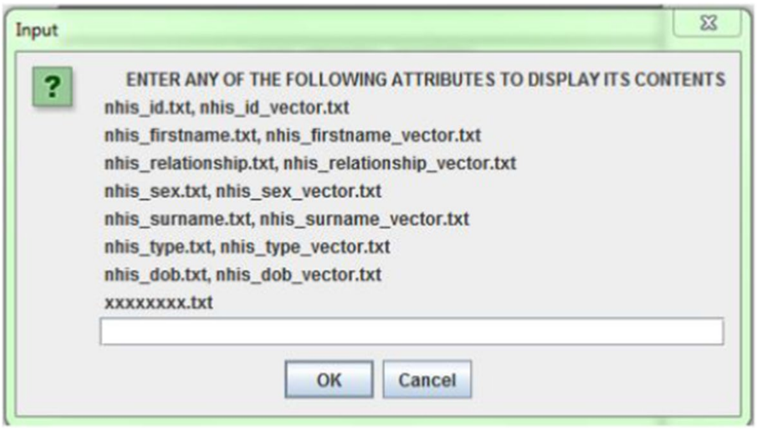

VECTORS ATTRIBUTES CREATION
VECTORS ATTRIBUTES LOOKUP
ACTIVATE GUI_ATTRIBUTE VECTOR
VECTORS ATTRIBUTES UPDATE
VECTORS MANAGEMENT EXIT

Fig. 7 Activity diagram of vector attribute management



```
0,0,0,0,0,0,-1,0,0,0,0
0,0,0,0,1,1,-1,1,0,-1,0
0,0,0,0,2,1,-1,2,0,-1,0
0,0,0,0,3,1,3,-1,0,0,0
1,1,0,1,4,0,-1,3,0,0,0
1,1,0,1,2,1,-1,4,-1,-1,0
1,1,0,1,5,1,-1,5,0,0,1
1,1,0,1,6,0,5,-1,-1,-1,1
1,1,0,1,6,0,5,-1,-1,-1,1
1,2,0,2,7,0,-1,7,0,0,0
1,2,0,2,2,1,-1,4,-1,-1,1
1,2,0,2,8,1,-1,8,0,0,0
2,3,0,3,9,0,-1,9,0,-1,0
2,3,0,3,10,0,5,-1,-1,-1,0
2,3,0,3,2,1,-1,2,0,-1,0
```

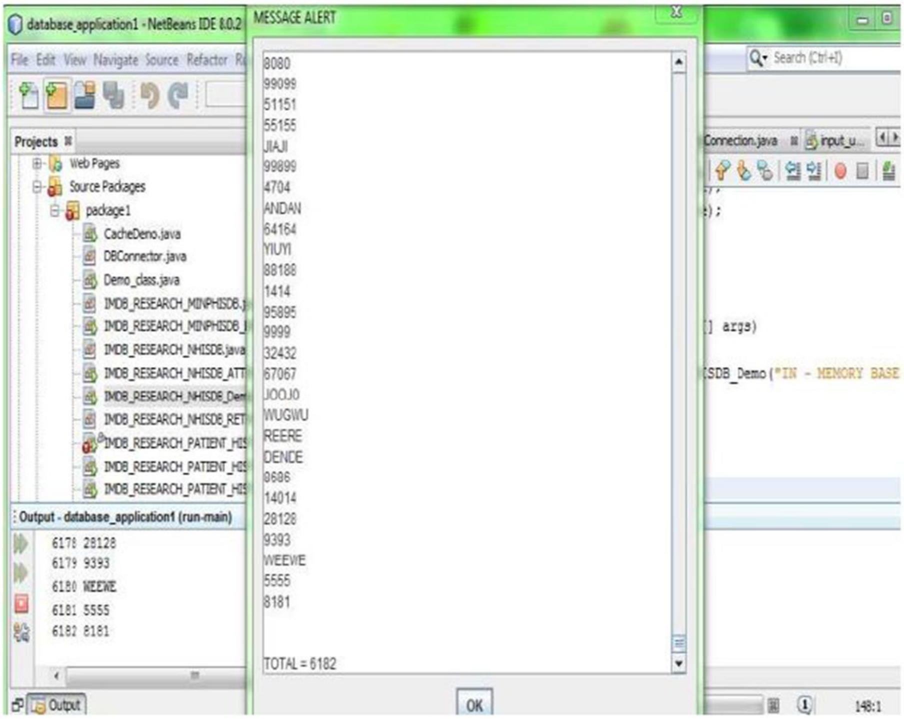
Fig. 8 Encoded Dictionary Vector snapshot



ENTER ANY OF THE FOLLOWING ATTRIBUTES TO DISPLAY ITS CONTENTS
nhis_id.txt, nhis_id_vector.txt
nhis_firstname.txt, nhis_firstname_vector.txt
nhis_relationship.txt, nhis_relationship_vector.txt
nhis_sex.txt, nhis_sex_vector.txt
nhis_surname.txt, nhis_surname_vector.txt
nhis_type.txt, nhis_type_vector.txt
nhis_dob.txt, nhis_dob_vector.txt
xxxxxxxx.txt

Fig. 9 In-Memory Database Management window

the In-Memory approach stored 123,753 entries; this produced a compression factor of 8.5342 and savings of 88.28%. For relationship attribute, the Disk-Resident approach stored 1,056,132 entries while the In-Memory approach stored 6 entries; this produced a compression factor of 176,022 and savings of 99.9%. For sex attribute, the Disk-Resident approach stored 1,056,132 entries while the In-Memory approach stored 2 entries; this produced a compression factor of 528,066 and savings of 99.9%For surname attribute, the Disk-Resident approach stored 1,056,132 entries while the In-Memory approach stored 89,592 entries; this produced a


database_application1 - NetBeans IDE 8.0.2
MESSAGE ALERT
File Edit View Navigate Source Refactor
Search (Ctrl+I)
Projects
Web Pages
Source Packages
package1
CacheDeno.java
DBConnector.java
Demo_class.java
IMDB_RESEARCH_MINPHISDB.j
IMDB_RESEARCH_MINPHISDB_
IMDB_RESEARCH_NHISDB.java
IMDB_RESEARCH_NHISDB_ATT
IMDB_RESEARCH_NHISDB_Dem
IMDB_RESEARCH_NHISDB_RET
IMDB_RESEARCH_PATIENT_HIS
IMDB_RESEARCH_PATIENT_HIS
IMDB_RESEARCH_PATIENT_HIS
Output - database_application1 (run-main)
Connection.java
input_u...
Output
OK
148:1

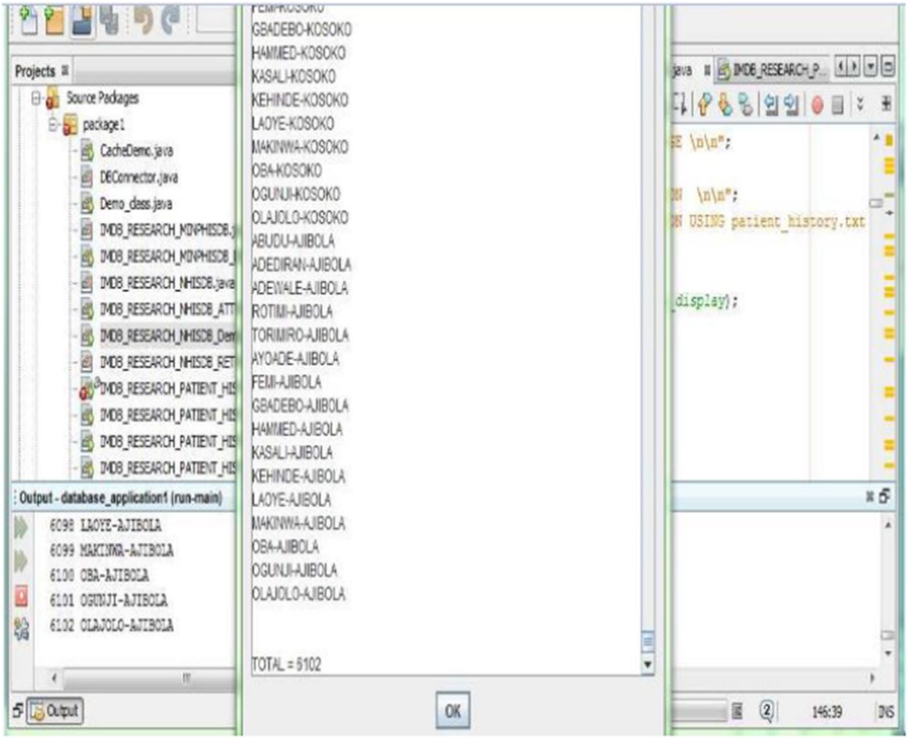
```
8080
99099
51151
55155
JIAJI
99899
4704
ANDAN
64164
YIUYI
88188
1414
95895
9999
32432
67067
JOOJO
WUGWU
REERE
DENDE
8686
14014
28128
9393
WEEWE
5555
8181


TOTAL = 6182
```

```
6178 28128
6179 9393
6180 WEEWE
6181 5555
6182 8181
```

```
] args)
SDB_Demo("IN - MEMORY BASE
```

Fig. 10 In-Memory Database result for patient id attribute window



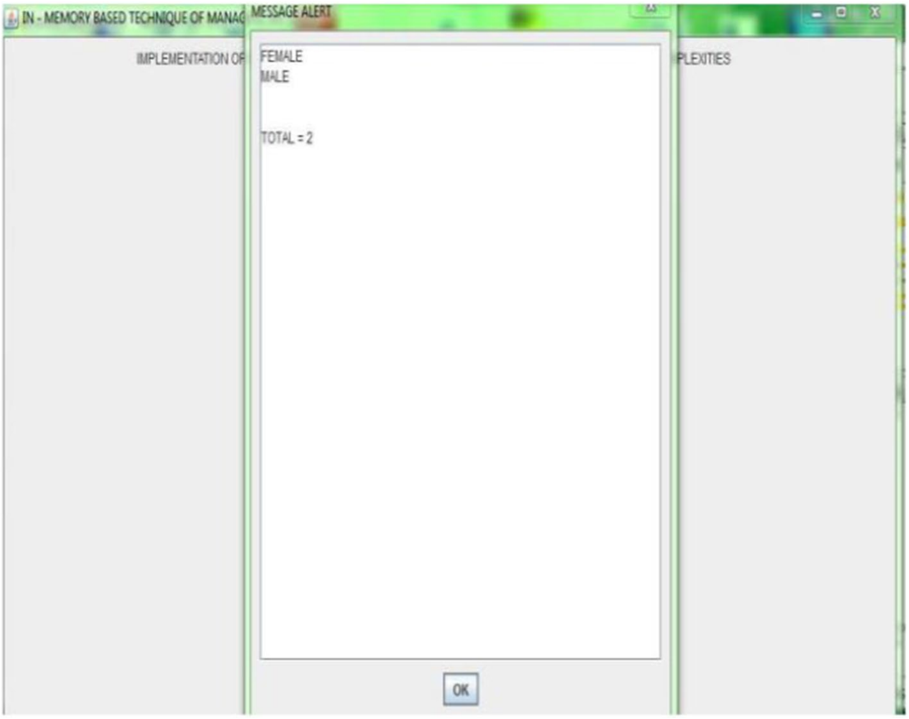
```
FEMI-KOSOKO
GBADEBO-KOSOKO
HAMMED-KOSOKO
KASALI-KOSOKO
KEHINDE-KOSOKO
LAOYE-KOSOKO
MAKINWA-KOSOKO
OBA-KOSOKO
OGUNJI-KOSOKO
OLAJOLO-KOSOKO
ABUDU-AJIBOLA
ADEDIRAN-AJIBOLA
ADEWALE-AJIBOLA
ROTIMI-AJIBOLA
TORIMIRO-AJIBOLA
AYOADE-AJIBOLA
FEMI-AJIBOLA
GBADEBO-AJIBOLA
HAMMED-AJIBOLA
KASALI-AJIBOLA
KEHINDE-AJIBOLA
LAOYE-AJIBOLA
MAKINWA-AJIBOLA
OBA-AJIBOLA
OGUNJI-AJIBOLA
OLAJOLO-AJIBOLA


TOTAL = 6102
```

```
6098 LAOYE-AJIBOLA
6099 MAKINWA-AJIBOLA
6100 OBA-AJIBOLA
6101 OGUNJI-AJIBOLA
6102 OLAJOLO-AJIBOLA
```

```
E \n\n";
N  \n\n";
N USING patient_history.txt
display);
```

Fig. 11 In-Memory Database result for patient surname attribute window



```
FEMALE
MALE


TOTAL = 2
```

Fig. 12 In-Memory Database result for patient sex attribute window

Table 1 Descriptive and inference statistics of storage space usage (NHIS)

(A) Attribute	(B) (Items) Size in Disk DB	(C) (Items) Size in In-Memory	(D) Compression Factor (B/C)	(E) (% Space Saving) (1—(C/B))* 100
Identifier	1,056,132	9	117,348	99.9991
Firstname	1,056,132	123,753	8.5342	88.2824
Relationship	1,056,132	6	176,022	99.9994
Sex	1,056,132	2	528,066	99.9998
Surname	1,056,132	89,592	11.7882	91.517
Type	1,056,132	6	176,022	99.9994
Date of birth	1,056,132	22,037	47.9254	97.9134
Dictionary				
Encoding	–	1,056,132		
TOTAL:	7,392,924	1,291,537		
MEAN = 1,056.132 161,442.125				
STD DEV = 0364,639.4471				
MEAN SPACE SPACING % $= (1—(\text{TOTAL}_C/\text{TOTAL}_B))* 100 = (1—(1291537/7392924))* 100 = 82.53\%$				

compression factor of 11.788 and savings of 91.517%. For type attribute, the Disk-Resident approach stored 1,056,132 entries while the In-Memory approach stored 6 entries; this produced a compression factor of 176,022 and percentage of 99.9%.For date-of-birth attribute, the Disk-Resident approach stored 1,056,132 entries while

Table 2 Descriptive and inference statistics of storage space usage (Patient History)

(A) Attribute	(B) (Items) Size in Disk DB	(C) (Items) Size in In-Memory	(D) Compression Factor (B/C)	(E) (% Space Saving) (1—(C/B)) * 100
Pat-No	1,003,164	6,182	162.3	99.3837
Surname	1,003,164	6121	163.9	99.3898
Other-name	1,003,164	112,279	8.9	88.8075
Sex	1,003,164	2	501,582.0	99.9998
Age	1,003,164	112	8,956.8	99.9888
Doa	1,003,164	3701	271.1	99.6311
Source	1,003,164	2	501,582.0	99.9998
Provider	1,003,164	239	4,197.3	99.9762
Dod	1,003,164	3290	304.9	99.6720
Sod	1,003,164	2	501,582.0	99.9998
Condition	1,003,164	2	501,582.0	99.9998
Diagnosis	1,003,164	8	125,395.5	99.9992
Dictionary				
Encoding	–	1,003,164		
TOTAL:	12,037,968.00	1,135,104.00		
MEAN = 1,003,164.00 87,315.69				
STD DEV = 0 319,224.11				
MEAN SPACE SPACING % = (1—(TOTAL$_C$/TOTAL$_B$))* 100= (1—(1,135,104/12,037,968))* 100 = 90.57%				

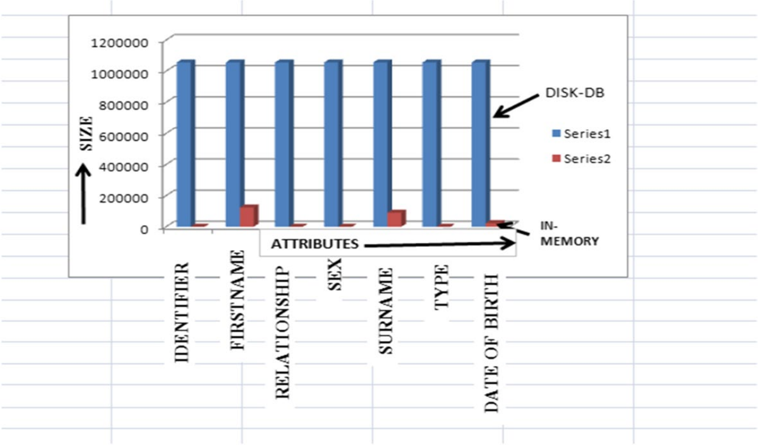


Fig. 13 Volume Representation in Disk-based and In-MemoryDatabase (NHIS)

the In-Memory approach stored 22,037 entries; this produced a compression factor of 47.92 and savings of 97.9%. For dictionary-encoding, the Disk-Resident approach stored no entries while the In-Memory approach stored 1,056,132 entries; this produced a compression factor of 1 (this means there is no compression with respect to Disk Database since Disk Database did not use this item) and hence there is no savings. Table 2 shows the data for patient history. The graph of volume representation in Disk-Resident and In-Memory Database using the NHIS table is depicted in Fig. 13 and that of Patient History shown in Fig. 14.

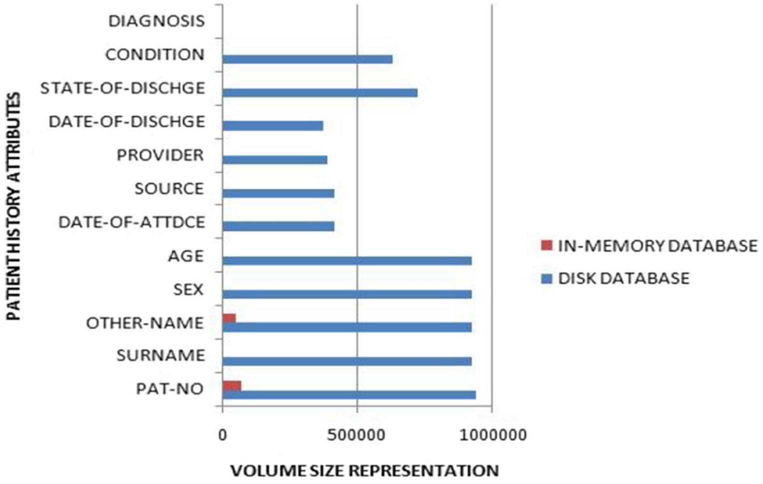


Fig. 14 Volume Representation in Disk-based and In-MemoryDatabase (Patient History)

Discussion—conclusions

This study has compared an In-Memory and Disk-based techniques for managing volume complexity associated with big data using textual big health data sources. In the study, vast amount of data was stored in compressed form using dictionary encoding that ensured no need to uncompress them when accessing the data since most compression techniques require that data be uncompressed before they can be used. In the study, repeated items irrespective of the number of occurrence are stored once, and digits which require less storage space used as the means of retrieving the items. Most stored items that accumulate into big data in healthcare domain are often of repeated nature. When the Disk-Resident database and the In-Memory database were compared, the In-Memory database was found to have addressed significantly,the volume challenge of healthcare big data (up to 90.57% space saving) in comparison with the disk-based database system when various big data sizes were considered. With this technique, administrators need not embark on intensive acquisition of storage resources when managing big data in critical applications such as in healthcare domain. However, the study used text data only in its implementation; hence the effects of other data formats on this technique are yet to be ascertained. This is intended to be carried out in future work.

Summary

This study implemented an In-Memory Technique that manages the Volume complexity associated with big data. It employed the single structure column wise (columnar) storage technique as well as dictionary encoding for storage in the main memory. The study used the Transaction Processing Control (TPC) Performance standard to evaluate the volume reduction. The technique was evaluated with a MySQL disk resident database management system. When evaluated, the technique was found to have addressed the volume challenge of big data considerably in comparison with the disk-based database system when various big data attribute sizes were considered. With this technique, big data administrators need not embark on resource intensive approach of managing big data, also real-time processing in critical applications such as in healthcare domain is enhanced.

The study was also able to store vast amount of data in compressed form without the need to uncompress them when accessing the compressed data. Most compression techniques require that data be uncompressed before they can be used. Repeated items irrespective of the number of occurrence are stored once, and numbers which require less storage space are used as the means of retrieving data items. Most stored items that accumulate into big data in healthcare domain are often of repeated nature.

Acknowledgements
All members of Obafemi Awolowo University Health Information Systems Research Group (Ife Nigeria) are acknowledged for shaping the research and offering all constructive criticisms.

Author contributions
RF was the principal author and the main researcher, MO reframed the title to be in accordance with the paper content, reviewed all sections of the paper and offered suggestions to be done for improving the quality of the work, was a major contributor to the paper AO wrote on the Big Data Analytics aspect while OM wrote the Descriptive and Inference Data Analytics section. All authors read and approved the final manuscript.

Funding
Not applicable.

Availability of data and materials
The datasets used and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Ethics approval and consent to participate
Two data sources were used for the implementation. The first dehumanized data set was The Nigeria Health Insurance Scheme (NHIS) registration data obtained from Wise Health Services Limited; an NHIS accredited Health Maintenance Organisation (HMO) with approval. The second dehumanized data set was Patient history that was obtained from "Made in Nigeria Primary Healthcare Information System, (MINPHIS)" designed by the Obafemi Awolowo University Health Information Systems Research Group, for which the corresponding author is a member and also approval obtained from ethics committee.

Consent for publication
Not applicable.

Competing interests
Not applicable.

Received: 20 April 2022 Accepted: 21 January 2023
Published online: 19 February 2023

References

1. Gartner. Pattern-Based Strategy: Getting Value from big dat. Gartner Group press release. July 2011. https://www.gartner.com/en/documents/1727419/pattern-based-strategy-getting-value-from-big-data. Accessed 14 April 2022
2. Demchenko Y, Ngo C, Membrey P, "Architecture Framework and Components for the Big Data Ecosystem. System and Network Engineering (SNE) publication. Amsterdam: Universiteit van Amsterdam; 2013.
3. Famutimi RF, Soriyan HA, IbitoyeA O, Famutimi TI. A case for the adoption of an in-memory based technique for healthcare big data management. Int J Comput. 2017;27(1):141–5.
4. Mavrogiorgou A, Kiourtis A, Perakis K, Miltiadou D, Pitsios S, Kyriazis D. Analyzing data and data sources towards a unified approach for ensuring end-to-end data and data sources quality in healthcare 40. Comput Method Progr Biomed. 2019;181:104967.
5. Press G. 12 big data Definitions-the little black book of Billionaire secrets .*Forbes Publication*.2014 (5):pp.1-20.
6. LidongWang andCheryl Ann Alexander. *Big Data Analytics in Healthcare Systems*. 2019. https://www.researchgate.net/publication/330073687_Big_Data_Analytics_in_Healthcare_Systems. Accessed 14 Jun 2021.
7. Kyriazis D. et al. "Policycloud: analytics as a service facilitating efficient data-driven public policy management. In: Maglogiannis I, Iliadis L, Pimenidis E, editors. "IFIP international conference on artificial intelligence applications and innovations. Cham: Springer; 2020.
8. Escobar CA, McGovern ME, Morales-Menendez R. Quality 4.0: a review of big data challenges in manufacturing. J Int Manufactur. 2021;32(8):2319–34.

9. Burghard C. Big Data and Analytics, Key to Accountable Care Success. *Idiopathic.Dilated Cardiomyopathy Health Insights Publication*. 2012. https://www.coursehero.com/file/14345607/Big-Data-and-Analytics-Key-to/. Accessed 14 Jun 2021.
10. Famutimi R.F. 2018 An In-Memory Technique of managing big data complexities. Unpublished PhD Thesis submitted to the Department of Computer Science and Engineering, Obafemi Awolowo University, Ile-Ife, Nigeria.
11. Gordon K. Principles of data management—facilitating information sharing. Swindon, UK: The British Computer Society Publishing and Information Product; 2007. p. 1–320.
12. Zhang H, Chen G, Ooi BC, Tan K, Zhang M. In-memory big data management and processing: a survey. Inst Electr Electr Eng Trans Knowl Data Eng. 2015;27(7):201–10.
13. Gupta MK, Verma V, Verma MS. In-memory database systems—a paradigmshift. *International Journal of Engineering Trends and Technology*. 2013. http://www.ijettjournal.org. Accessed 14 April 2022.
14. Plattner H. The Inner Mechanics of In-Memory Databases. Hasso Plattner Institue of IT Systems Engineering, Universitat Potsdam Technical report. https://open.hpi.de/courses/imdb2015. Accessed 10 Feb 2022.
15. Yesdaulet Izenov, Asoke Datta, Florin Rusu, Jun Hyung Shin. COMPASS: Online Sketch-based Query Optimization for In-Memory Databases. SIGMOD '21, June 20–25, 2021, Virtual Event, China.
16. Guetterman TC. Basics of statistics for primary care research. Family Med Commun Health. 2019;7(2):e000067. https://doi.org/10.1136/fmch-2018-000067.
17. Reid N, Cox DR. On some principles of statistical inference. Int Stat Rev. 2014;83(2):293–308. https://doi.org/10.1111/insr.12067.
18. Rahlf T. Statistical Inference. In: Diebolt C, Haupert M, editors. Handbook of Cliometrics (Springer Reference Series). Berlin/Heidelberg: Springer; 2014.
19. Byrne G. A Statistical primer: understanding descriptive and inferential statistics. Evidence Based Library Inform Practice. 2007;2(1):32–47. https://doi.org/10.18438/B8FW2H.
20. Amrhein V, Trafimow D, Greenland S. Inferential statistics as descriptive statistics: there is no replication crisis if we don't expect replication. Am Stat. 2019;73(sup1):262–70. https://doi.org/10.1080/00031305.2018.1543137.
21. Weihs C, Ickstadt K. Data science: the impact of statistics. Int J Data Sci Anal. 2018;6:189–94. https://doi.org/10.1007/s41060-018-0102-5.
22. Famutimi RF, Ibitoye AO, Ikono RN, Famutimi TI. Comparative study of disk resident and column oriented memory resident technique for healthcare big data management using retrieval time. Inte J Comput. 2018;31(1):92–9.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.