Data Modeling and Data Analytics: Big Data Perspective

Prof. Yogita S. Alone Prof. Ruchita A. Kale Prof. P. P. Deshmukh Prof. Gaurav J. Sawale

Abstract- **The volume of data we deal with has grown to terabytes and petabytes in internet. As the volume of data keeps growing, the types of data generated by applications become richer than before. Traditional relational databases are challenged to capture, store, search, and share, analyze, and visualize data. Information is now available in an over a abundance, hat distinguishing the noise from the signal has become very problematic. The collection and storage of information was the primary issue. Currently, there are massive amounts of data both structured and unstructured, that need to be analyzed in an iterative, In a time sensitive manner. In response to this need, data analytical tools and services have emerged as a means to solve this problem.**

Keywords*- *Big Data , Data Modeling, Data Analytics, Modeling Language

I. INTRODUCTION

An exponential growth of the volume of data produced and stored. This can be explained by the evolution of the technology that results in the proliferation of data with different formats from the most various domains (e.g. health care, banking, government or logistics) and sources (e.g. sensors, social networks or mobile devices). We have assisted a paradigm shift from simple books to sophisticated databases that keep being populated every second at an immensely fast rate. Internet and social media also highly contribute to the worsening of this situation [1]. Facebook, for example, has an average of 4.75 billion pieces of content shared among friends every day [2]. Traditional Relational Database Management Systems (RDBMSs) and Data Warehouses (DWs) are designed to handle a certain amount of data, typically structured, which is completely different from the reality that we are facing nowadays. Business is generating enormous quantities of data that are too big to be processed and analyzed by the traditional RDBMSs and DWs technologies, which are struggling to meet the performance and scalability requirements.

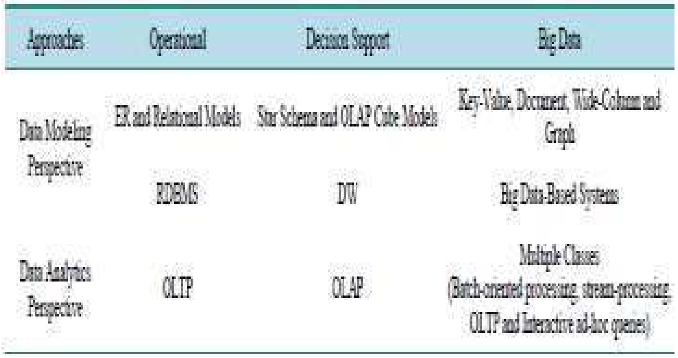
Therefore, in the recent years, a new approach that aims to mitigate these limitations has emerged. Companies like Facebook, Google, Yahoo and Amazon are the pioneers in creating solutions to deal with these "Big Data" scenarios, namely recurring to technologies like Hadoop [3] [4] and MapReduce [5]. Big data is a generic term used to refer to massive and complex datasets, which are made of a variety of data structures (structured, semi structured and unstructured data) from a multitude of sources [6]. Big Data can be categorized by three Vs: volume (amount of data), velocity (speed of data in and out) and variety (kinds of data types and sources) [7]. Still, there are added some other Vs for variability, veracity and value [8]. Implementing Big Data-based technologies not only moderates the problems existing above, but also opens new perspectives that allow extracting value from Big Data. Big Data-based technologies are being applied with success in multiple scenarios [1] [9] [10] like in:

(1) e-commerce and marketing, where count the clicks that the crowds do on the network permit identifying trends that improve campaigns, evaluate personal profiles of a user, so that the content shown is the one he will most likely enjoy;

(2) Government and public health, allowing the detection and tracking of disease outbreaks via social media or detect frauds; (3) transportation, industry and surveillance, with real-time improved estimated times of arrival and smart use of resources.

II. DATA MODELING

This segment gives an in detail look of the most popular data models used to define and support Operational Databases, Data Warehouses and Big Data technologies.



Approaches	Operational	Decision Support	Big Data
Data Modeling Perspective	ER and Relational Models	Star Schema and OLAP Cube Models	Key-Value, Document, Wide-Column and Graph
	RDBMS	DW	Big Data-Based Systems
Data Analytics Perspective	OLTP	OLAP	Multiple Classes (Batch-oriented processing, stream-processing, OLTP and Interactive ad-hoc queries)

Table 1: Approaches and perspectives for Big Data

Databases are broadly used either for individual or enterprise use, namely due to their strong ACID guarantees (atomicity, consistency, isolation and durability) guarantees and the maturity level of Database Management Systems (DBMSs) that support them [15]. The data modeling process may involve the definition of three data models (or schemas) defined at different abstraction levels, namely Conceptual, Logical and Physical data models [15] [16]. Figure 1 shows amount of the three data models for the AMS case study. All these models define three entities (Person, Student and Professor) and their main relationships (teach and supervise associations). Conceptual Data Model: A conceptual data model is used to define, at a very high and platform-independent level of abstraction, the entities or concepts, which represent the data of the problem domain, and their relationships. It leaves further details about the entities (such as their attributes, types or primary keys) for the next steps. This model is typically used to explore domain concepts with the stakeholders and can be omitted or used instead of the logical data model.

Despite being independent of any DBMS, this model can easily be mapped on to a physical data model thanks to the details it provides.

Summarizing, the complexity and detail increase from a conceptual to a physical data model. First, it is important to perceive at a higher level of abstraction, the data entities and their relationships using a Conceptual Data Model. Then, the spotlight is on detailing those entities without perturbing about implementation details using a Logical Data Model. Finally, a Physical Data Model allows representing how data is supported by a given DBMS.

2.1 Operational Databases

Databases had a great boost with the popularity of the Relational Model [17] proposed by E. F. Codd in 1970. The Relational Model incapacitated the problems of ancestors data models (namely the Hierarchical Model and the Navigational Model [18]). The Relational Model caused the emergence of Relational Database Management Systems (RDBMSs), which are the most used and popular DBMSs, as well as the definition of the Structured Query Language (SQL) [19] as the standard language for defining and manipulating data in RDBMSs. RDBMSs are widely used for maintaining data of daily operations. Considering the data modeling of operational databases there are two main models: the Relational and the Entity-Relationship (ER) models. Relational Model. The Relational Model is based on the mathematical concept of relation. A relation is defined as a set (in mathematics terminology) and is represented as a table, which is a matrix of columns and rows, holding information about the domain entities and the relationships among them. Each column of the table corresponds to an entity attribute and specifies the attribute's name and its type (known as domain).

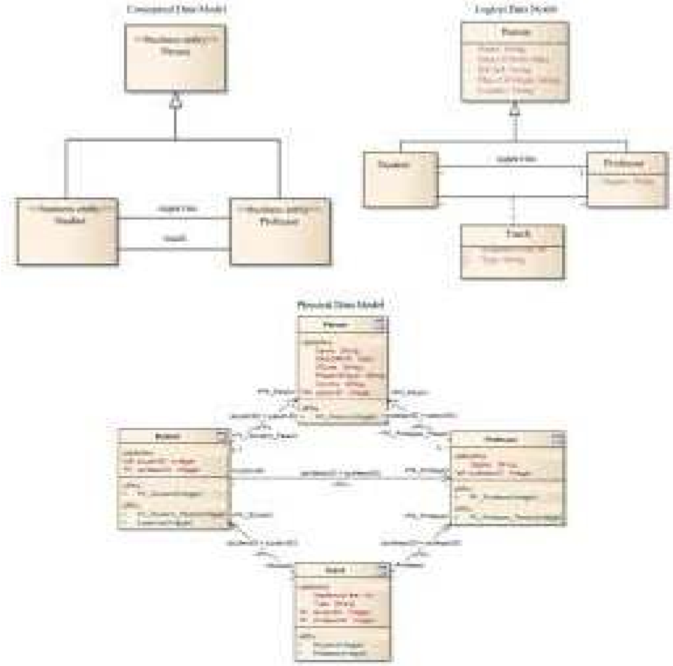


Fig 1: Three Data Model Example

The Fig 1 shows example of Academic Management System, the table (known as tuple) corresponds to a single element of the represented domain entity. In the Relational Model each row is unique and therefore a table has an attribute or set of attributes known as primary key, used to univocally identify those rows. Tables are related with each other by sharing one or more common attributes. These attributes correspond to a primary key in the referenced (parent) table and are known as foreign keys in the referencing (child) table. In one-to-many relationships, the referenced table corresponds to the entity of the "one" side of the relationship and the referencing table corresponds to the entity of the "many" side. In many to-many relationships, it is used an additional association table that associates the entities involved through their respective primary keys. The Relational Model also features the concept of View, which is like tables whose rows are not explicitly stored in the database, but are computed as needed from a view definition. Instead, a view is defined as a query on one or more base tables or other views [17].

Entity-Relationship (ER) Model:The Entity Relationship (ER) Model [20], proposed by Chen in 1976, appeared as an alternative to the Relational Model in order to provide more expressiveness and semantics into the database design from the user's point of view. The ER model is a semantic data model, goals to signify the meaning of the data involved on some detailed domain. This model was originally defined by three main concepts: entities, relationships and attributes. The Enhanced ER Model [21] provided additional concepts to represent more complex requirements, such as generalization, specialization, aggregation and composition. Other popular variants of ER diagram notations are Crow's foot, Bachman, Barker's, IDEF1X and UML Profile for Data Modeling [22].

2.2 Decision Support Databases

The growth of relational databases to verdict provision databases, here in afterward indistinctly referred as "Data Warehouses" (DWs), occurred with the need of storing operational but also historical data, and the need of analyzing that data in complex dashboards and reports. Even though a DW looks to be a relational database, it is dissimilar in the intellect that DWs are more suitable for supporting query and analysis operations (fast reads) instead of transaction processing (fast reads and writes) operations. DWs contain historical data that come from transactional data, but they also might include other data sources [23]. DWs are mainly used for OLAP (online analytical processing) operations. OLAP is the approach to provide report data from DW through multi-dimensional queries and it is required to create a multi-dimensional database [24]. Usually, DWs include a framework that allows extracting data from multiple data sources and transform it before loading to the repository, which is known as ETL (Extract Transform Load) framework [23]. Data modeling in DW consists in defining fact tables with several dimension tables, suggesting star or snowflake schema data models [23]. A star schema has a central fact table linked with dimension tables. Usually, a fact table has a large number of attributes (in many cases in a denormalized way), with many foreign keys that are the primary keys to the dimension tables. The dimension tables represent characteristics that describe the fact table. When star schemas developed too multifaceted to be queried proficiently they are malformed into multi-dimensional arrays of data called OLAP cubes (for more information on how this transformation is performed the reader can consult the following references [24] [25]). A star schema is converted to a cube by placing the datum table on the front face that we are facing and the dimensions on the other faces of the cube [24]. For this reason, cubes can be equivalent to star schemas in content, but they are accessed with more platform-specific languages than SQL that have more analytic capabilities (e.g. MDX or XMLA). A cube with three proportions is theoretically easier to visualize and understand, but the OLAP cube model provisions more than three measurements, and is called a hypercube. Figure 2 shows two examples of star schemas regarding the case study AMS. The star schema on the leftward denotes the data model for the Student's fact, while the data model on the right represents the On the other hand, Figure 3 shows a cube model with three dimensions for the Student. These dimensions are represented by sides of the cube (Student, Country and Date). This cube is valuable to perform queries such as: the students by country enrolled for the first time in a given year. A challenge that DWs face is the growth of data, since it affects the number of dimensions and levels in either the star schema or the cube orders. The increasing number of dimensions over time makes the management of such systems often impracticable; this problem becomes even more serious when dealing with Big Data scenarios, where data is continuously being generated [23].

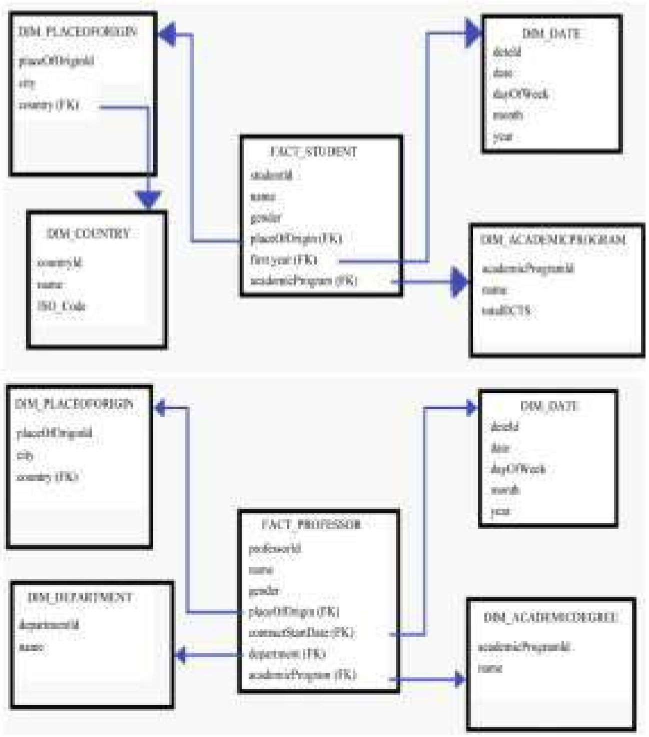


Fig 2: Two Star Schema Model Example

2.2 Big Data Technologies

The volume of data has been exponentially increasing over the last years, namely due to the simultaneous growth. of the number of sources (e.g. users, systems or sensors) that are continuously producing data. These data sources produce huge amounts of data with variable representations that make their management by the traditional RDBMSs and DWs often impracticable.

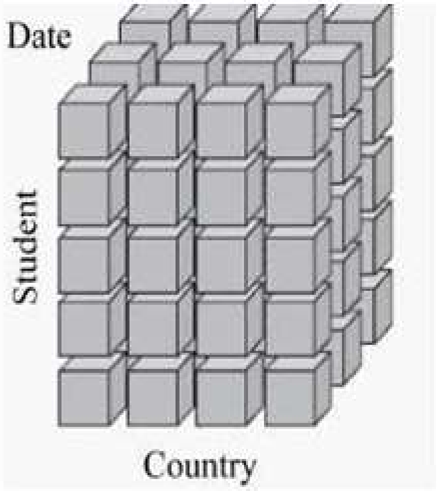
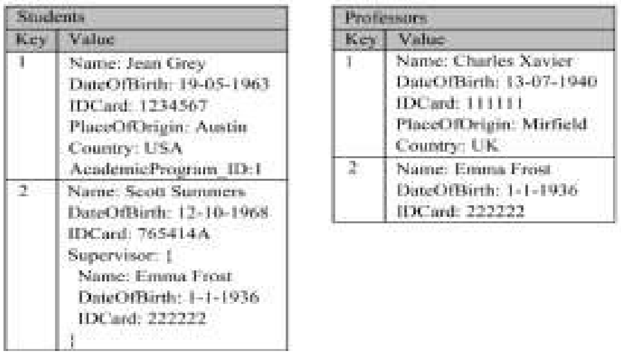


Fig 3 : Cube Model Example

Therefore, there is a necessity to plan new data models and technologies that can handle such Big Data. NoSQL (Not Only SQL) [26] is one of the most popular approaches to deal with this problem. It involves in a group of non-relational DBMSs that accordingly do not signify databases using tables and usually do not use SQL for data manipulation. NoSQL systems allow managing and storing large- scale denormalized datasets, and are designed to scale horizontally. They achieve that by compromising consistency in favor of availability and partition-tolerance, according to Brewer's CAP theorem [27]. Therefore, NoSQL systems are "eventually consistent", i.e. assume that writes on the data are eventually propagated over time, but there are limited guarantees that different users will read the same value at the same time. NoSQL provides BASE guarantees (Basically Available, Soft state and Eventually consistent) instead of the traditional ACID guarantees, in order to greatly improve performance and scalability [28]. NoSQL databases can be classified in four categories [29]: Key-value stores, (2) Document-oriented databases, (3) Wide-column stores, and (4) Graph databases.

Key-value Stores: A Key-Value store represents data as a collection (known as dictionary or map) of key value pairs. Every key consists in a unique alphanumeric identifier that works like an index, which is used to access a corresponding value. Values can be simple text strings or more complex structures like arrays. The Key-value model can be extended to an ordered model whose keys are stored in lexicographical order. The fact of being a simple data model makes Key-value stores ideally suited to retrieve information in a very fast, available and scalable way. For instance, Amazon makes extensive use of a Key-value store system, named Dynamo, to manage the products in its shopping cart [30]. Amazon's Dynamo and Voldemort, which is used by Linkedin, are two examples of systems that apply this data model with success. In case of a key-value store for both students and professors of the Academic Managements organization is shown in Figure 4.

Document-oriented Databases :Document-oriented databases (or document stores) were originally created to store traditional documents, like a notepad text file or Microsoft Word document. However, their concept of document goes beyond that, and a document can be any kind of domain object [26]. Documents contain encoded data in a standard format like XML, YAML, JSON or BSON (Binary JSON) and are univocally identified in the database by a unique key. Documents encompass semi-structured data signified as name-value pairs, which can vary according to the row and can nest other documents. Unlike key-value stores, these systems support secondary indexes and allow fully searching either by keys or values. Document databases are well suited for storing and managing huge collections of textual documents (e.g. text files or email messages), as well as semi- structured or denormalized data that would require an extensive use of "nulls" in an RDBMS [30]. MongoDB and CouchDB are two of the most standard Document-oriented database systems. Figure 5 illustrates two collections of documents for both students and professors of the Academic Management System.



Students	
Key	Value
1	Name: Jean Grey DateOfBirth: 19-05-1963 IDCard: 1234567 PlaceOfOrigin: Austin Country: USA AcademicProgram_ID:1

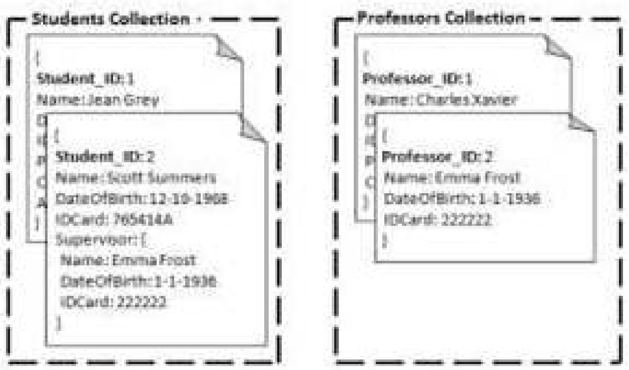
2	Name: Scott Summers
	DateOfBirth: 12-10-1968
	IDCard: 765414A
	Supervisor: {
	Name: Emma Frost
	DateOfBirth: 1-1-1936
	IDCard: 222222
	}

Professors	
Key	Value
1	Name: Charles Xavier DateOfBirth: 13-07-1940 IDCard: 111111 PlaceOfOrigin: Mirfield Country: UK
2	Name: Emma Frost DateOfBirth: 1-1-1936 IDCard: 222222

Fig 4 : Key-Value Store Example

Wide-column Stores.Wide-column stores (also identified as column-family stores, extensible record stores or column-oriented databases) signify and manage data as sectors of columns rather than rows (like in RDBMS).

Each sector is collected of key-value pairs, where the keys are rows and the values are sets of columns, known as column families. Each row is identified by a primary key and can have column families different of the other rows. Each column family also acts as a primary key of the set of columns it contains. In turn each column of column family consists in a name-value pair. Column families can level be grouped in great column families [29]. This data model was highly inspired by Google's Big Table [31]. Wide-column stores are suited for scenarios like: (1) Distributed data storage; (2) Large-scale and batch-oriented data processing, using the famous MapReduce method for tasks like sorting, parsing, querying or conversion and; (3) Exploratory and predictive analytics. Cassandra and HadoopHBase are two popular frameworks of such data management systems [29].



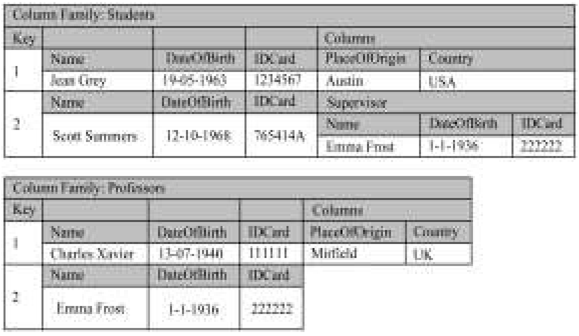
{
Student_ID:1
Name:Jean Grey
}

{
Student_ID: 2
Name: Scott Summers
DateOfBirth: 12-10-1968
IDCard: 765414A
Supervisor: {
Name: Emma Frost
DateOfBirth: 1-1-1936
IDCard: 222222
}

{
Professor_ID:1
Name: Charles Xavier
}

{
Professor_ID: 2
Name: Emma Frost
DateOfBirth: 1-1-1936
IDCard: 222222
}

Fig 5 : document oriented database example



Column Family: Students

Key				Columns		
1	Name	DateOfBirth	IDCard	PlaceOfOrigin	Country	
	Jean Grey	19-05-1963	1234567	Austin	USA	
2	Name	DateOfBirth	IDCard	Supervisor		
				Name	DateOfBirth	IDCard
	Scott Summers	12-10-1968	765414A	Emma Frost	1-1-1936	222222

Column Family: Professors

Key				Columns	
1	Name	DateOfBirth	IDCard	PlaceOfOrigin	Country
	Charles Xavier	13-07-1940	111111	Mirfield	UK
2	Name	DateOfBirth	IDCard		
	Emma Frost	1-1-1936	222222		

Fig 6 : Example of Wide- Column Stores

III. DATA ANALYTICS

This section presents and discusses the types of operations that can be performed over the data models described in the previous section and also establishes comparisons between them.

3.1 Operational Databases Systems

Operational databases are intended to handle a high number of transactions that usually implement changes to the operational data, i.e. the data an organization needs to assure its everyday normal operation. These systems are called Online Transaction Processing (OLTP) systems and they are the reason why RDBMSs are so essential nowadays. RDBMSs have increasingly been optimized to perform well in OLTP systems, namely providing reliable and efficient data processing [16]. The set of operations supported by RDBMSs is derived from the relational algebra and calculus underlying the Relational Model [15]. As mentioned before, SQL is the standard language to perform these operations. SQL can be divided in two parts involving different types of operations: Data Definition Language (SQL-DDL) and Data Manipulation Language (SQL-DML).

SQL-DDL allows performing the creation (CREATE), update (UPDATE) and deletion (DROP).

```
CREATE TABLE Student (Student ID NOT NULL
IDENTITY,Name   VARCHAR(255)NOT NULL, Date of
Birth DATE NOT NULL, ID Card VARCHAR(255)   NOT
NULL,    Place of   Origin VARCHAR(255), Country
VARCHAR(255), PRIMARY KEY(Student ID))
```

3.2 Decision Support Databases

The most common data model used in DW is the OLAP cube, which offers a set of operations to analyze the cube model [23]. Since data is conceptualized as a cube with hierarchical dimensions, its operations have familiar names when manipulating a cube, such as slice, dice, drill and pivot. Figure 7 depicts these operations considering the Student's facts of the AMS case study (see Figure 2). The slice operation begins by selecting one of the dimensions (or faces) of the cube. This dimension is the one we want to consult and it is followed by "slicing" the cube to a specific depth of interest. The slice operation leaves us with a more restricted selection of the cube, namely the dimension we wanted (front face) and the layer of that dimension (the sliced section). In the example of Figure 7 (top-left), the cube was sliced to consider only data of the year 2004. Dice is the operation that allows restricting the front face of the cube by reducing its size to a smaller targeted domain. This means that the user produces a smaller "front face" than the one he had at the start. Figure 7 (top-right) shows that the set of students has decreased after the dice operation. Drill is the operation that allows to navigate by specifying different levels of the dimensions, ranging from the most detailed ones (drill down) to the most summarized ones (drill up). Figure 7 (bottom-left) shows the drill down so the user can see the cities from where the students of the country Portugal come from. The pivot operation allows changing the dimension that is being faced (change the current front face) to one that is adjacent to it by rotating the cube. [23] [24]. Figure 7 (bottom-right) shows a pivot operation where years are arranged vertically and countries horizontally. The usual operations issued over the OLAP cube are about just querying historical events stored in it.

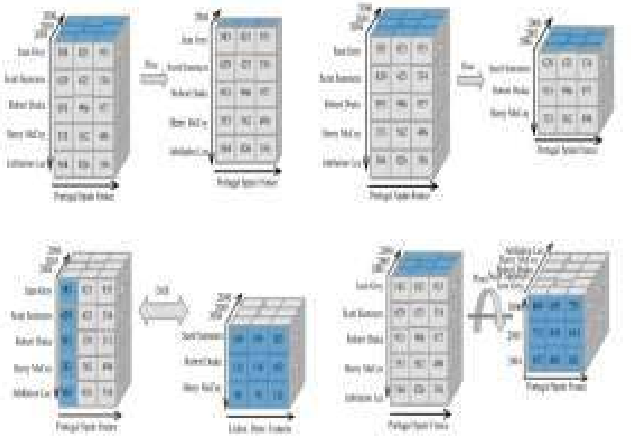


Fig 7 : Representation of Cube Operation

The SELECT clause sets the query axes as the name and the gender of the Student dimension and the year 2015 of the Date dimension. The FROM clause indicates the data source, here being the Students cube, and the WHERE clause defines the slicer axis as the “Computer Science” value of the Academic Program dimension. This query returns the students (by names and gender) that have enrolled in Computer Science in the year 2015.

SELECT{ [Student].[Name],[Student].[Gender]} **ON COLUMNS**{ [Date].[Academic Year] &[2015] } **ON ROWSFROM** [Students Cube]**WHERE** ([Academic Program].[Name] &[Computer Science])

3.3 Big Data Technologies

Big Data Analytics consists in the process of discovering and extracting potentially useful information hidden in huge amounts of data (e.g. discover unknown patterns and correlations). Big Data Analytics can be separated in the following categories: (1) Batch-oriented processing; (2) Stream processing; (3) OLTP and; (4) Interactive ad-hoc queries and analysis.

Batch-oriented processing: is a paradigm where a large volume of data is firstly stored and only then analyzed, as opposed to Stream processing. This paradigm is very common to perform large-scale recurring tasks in parallel like parsing, sorting or counting. The most popular batch-oriented processing model is Map Reduce [5], and more specifically its open-source implementation in Hadoop1.

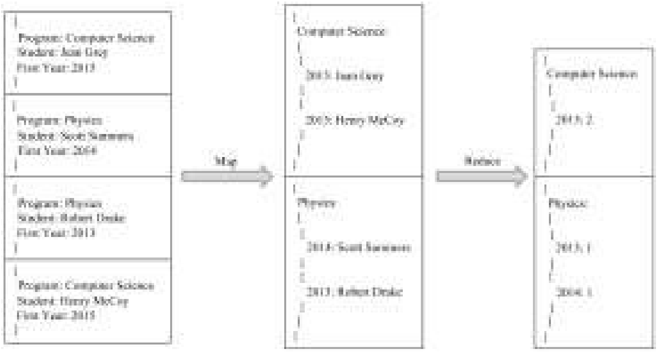


Fig 8: Example

IV. Conclusion

In recent years, the term Big Data has appeared to classify the huge datasets that are continuously being produced from various sources and that are represented in a variety of structures. Handling this kind of data represents new challenges, because the traditional RDBMSs and DWs reveal serious limitations in terms of performance and scalability when dealing with such a volume and variety of data. Therefore, it is needed to reinvent the ways in which data is represented and analyzed, in order to be able to extract value from it. This paper presents a survey focused on both these two perspectives: data modeling and data analytics, which are reviewed in terms of the three representative approaches nowadays: operational databases, decision support databases and Big Data technologies. First, concerning data modeling, this paper discusses the most common data models, namely: relational model and ER model for operational databases; star schema model and OLAP cube model for decision support databases; and key-value store, document-oriented database, wide-column store and graph database for Big Data-based technologies. Second, regarding data analytics, this paper discusses the common operations used for each approach. Namely, it observes that operational databases are more suitable for OLTP applications, decision support databases are more suited for OLAP applications, and Big Data technologies are more appropriate for scenarios like batch-oriented processing, stream processing, OLTP and interactive ad-hoc queries and analysis. Third, it compares these approaches in terms of the two perspectives and based on some features of analysis. From the data modeling perspective, there are considered features like the data model, its abstraction level, its concepts, the concrete languages used to described, as well as the modeling and database tools that support it. On the other hand, from the data analytics perspective, there are taken into account features like the class of application domains, the most common operations and the concrete languages used to specify those operations. From this analysis, it is possible to verify that there are several data models for Big Data, but none of them is represented by any modeling language, neither supported by a respective modeling tool.

REFERENCES

1. Mayer-Schönberger, V. and Cukier, K. (2014) Big Data: A Revolution That Will Transform How We Live, Work, and Think. Houghton Mifflin Harcourt, New York.
2. Noyes, D. (2015) The Top 20 Valuable Facebook Statistics. https://zephoria.com/top-15-valuable-facebook-statistics
3. Shvachko, K., HairongKuang, K., Radia, S. and Chansler, R. (2010) TheHadoop Distributed File System. 26th Symposium on Mass Storage Systems and Technologies (MSST), Incline Village, 3-7 May 2010,1-10. http://dx.doi.org/10.1109/msst.2010.5496972
4. White, T. (2012) Hadoop: The Definitive Guide. 3rd Edition, O'Reilly Media, Inc., Sebastopol.
5. Dean, J. and Ghemawat, S. (2008) MapReduce: Simplified Data Processing on Large Clusters. Communications, 51, 107- 113. http://dx.doi.org/10.1145/1327452.1327492
6. Hurwitz, J., Nugent, A., Halper, F. and Kaufman, M. (2013) Big Data for Dummies. John Wiley & Sons, Hoboken.
7. Beyer, M.A. and Laney, D. (2012) The Importance of "Big Data": A Definition. Gartner. https://www.gartner.com/doc/2057415
8. Duncan, A.D. (2014) Focus on the "Three Vs" of Big Data Analytics: Variability, Veracity and Value. Gartner. https://www.gartner.com/doc/2921417/focus-vs-big-data-analytics
9. Agrawal, D., Das, S. and El Abbadi, A. (2011) Big Data and Cloud Computing: Current State and Future Opportunities. Proceedings of the 14th International Conference on Extending Database Technology, Uppsala, 21-24 March, 530-533. http://dx.doi.org/10.1145/1951365.1951432
10. McAfee, A. and Brynjolfsson, E. (2012) Big Data: The Management Revolution. Harvard Business Review.
11. DataStorm Project Website. http://dmir.inesc- id.pt/project/DataStorm.
12. Stahl, T., Voelter, M. and Czarnecki, K. (2006) Model- Driven Software Development: Technology, Engineering, Management. John Wiley & Sons, Inc., New York.
13. Schmidt, D.C. (2006) Guest Editor's Introduction: Model- Driven Engineering. IEEE Computer, 39, 25-31. http://dx.doi.org/10.1109/MC.2006.58
14. Silva, A.R. (2015) Model-Driven Engineering: A Survey Supported by the Unified Conceptual Model. Computer Languages, Systems & Structures, 43, 139-155.
15. Ramakrishnan, R. and Gehrke, J. (2012) Database Management Systems. 3rd Edition, McGraw-Hill, Inc., New York.
16. Connolly, T.M. and Begg, C.E. (2005) Database Systems: A Practical Approach to Design, Implementation, and Management. 4th Edition, Pearson Education, Harlow.
17. Codd, E.F. (1970) A Relational Model of Data for Large Shared Data Banks. Communications of the ACM, 13, 377- 387. http://dx.doi.org/10.1145/362384.362685
18. Bachman, C.W. (1969) Data Structure Diagrams. ACM SIGMIS Database, 1,4-10. http://dx.doi.org/10.1145/1017466.1017467
19. Chamberlin, D.D. and Boyce, R.F. (1974) SEQUEL: A Structured English Query Language. In: Proceedings of the 1974 ACM SIGFIDET (Now SIGMOD) Workshop on Data Description, Access and Control (SIGFIDET' 74), ACM Press, Ann Harbor, 249-264.
20. Chen, P.P.S. (1976) The Entity-Relationship Model— Toward a Unified View of Data. ACM Transactions on Database Systems, 1,9-36.http://dx.doi.org/10.1145/320434.320440