algorithms

MDPI

Article

An Introduction of NoSQL Databases Based on Their Categories and Application Industries †

Jeang-Kuo Chen * and Wei-Zhe Lee

Department of Information Management, Chaoyang University of Technology, Taichung 41349, Taiwan; s10514613@cyut.edu.tw

* Correspondence: jkchen@cyut.edu.tw

† This Paper is an Extended Version of the Conference Paper (ID: 1070) in Taichung, Taiwan, 6–8 December 2018, IS3C2018.

Received: 31 January 2019; Accepted: 9 May 2019; Published: 16 May 2019

Abstract: The popularization of big data makes the enterprise need to store more and more data. The data in the enterprise's database must be accessed as fast as possible, but the Relational Database (RDB) has the speed limitation due to the join operation. Many enterprises have changed to use a NoSQL database, which can meet the requirement of fast data access. However, there are more than hundreds of NoSQL databases. It is important to select a suitable NoSQL database for a certain enterprise because this decision will affect the performance of the enterprise operations. In this paper, fifteen categories of NoSQL databases will be introduced to find out the characteristics of every category. Some principles and examples are proposed to choose an appropriate NoSQL database for different industries.

Keywords: big data; NoSQL database; RDB; HBase; MongoDB; Neo4j

1. Introduction

The Relational Database (RDB) was developed from the 1970s to present. Through a powerful Relational Database Management System (RDBMS), RDB is easy to use and maintain, and becomes a widely used kind of database [1]. Due to the popularization of big data acquisition technologies and applications, enterprises need to store more data than ever before. The enterprise's database is desired to be accessed as fast as possible. To obtain complex information from multiple relations, RDB sometimes needs to perform SQL join operations to merge two or more relations at the same time, which can lead to performance bottlenecks. Besides, except the relational data storage format, other data storage formats have been proposed in many applications, such as key-value pairs, document-oriented, time series, etc. As a result, more and more enterprises have decided to use NoSQL databases to store big data [2–4].

However, there are more than 225 NoSQL databases [2]. How to choose an appropriate NoSQL database for a specific enterprise is very important because the change of database may affect the enterprise performance of the business operations. This paper introduces basic concepts, compares the data formats and features, and lists some actual products for every category of NoSQL databases. In addition, this paper also proposes principles and key points for different types of enterprises to choose an appropriate NoSQL database to solve the business problems and challenges.

2. Related Work

2.1. Relational Database Model (RDM)

Developed by E.F. Codd in the 1970s, the elements of RDM [1] contain data structure, integrity constraints, and so on. The details are described as follows. An RDB is a collection of relations, which

are the data structures of RDM. A relation is a two-dimensional, normalized table to organize data. Each relation consists of two parts, relation schema and relation instances, where relation schema includes the name of the relation, the names of the attributes in the relation and their domains; relation instances refer to the data records stored in the relation at a specific time. Table 1 shows an example of a relation as follows:

1. The name of this relation is Students;
2. The names of the attributes in this relation are SID, name, telephone, and birthday, respectively;
3. The domain of each attribute is a collection of acceptable data values for the attribute, for example, the acceptable data value of attribute birthday is date;
4. There are five data records in this relation.

Table 1. An example of a relation.

Table Name: Students			
SID: Char(5)	**Name: Char(10)**	**Telephone: Char(11)**	**Birthday: Date (dd/mm/yy)**
S0001	Alice	05-65976597	10/4/1994
S0002	Dora	06-45714571	15/5/1995
S0003	Ella	07-57865869	20/6/1996
S0004	Kevin	06-57995611	22/7/1997
S0005	Leo	05-12899821	26/8/1998

As part of an RDB design, integrity constraints are used to check the correctness of the data input into the database. Integrity constraints not only prevent authorized users from storing illegal data into the database but also avoid data inconsistency between relations. There are four common kinds of integrity constraints described as follows:

1. Key constraint: A relation must have a unique and minimal primary key;
2. Domain constraint: An attribute value of a relation must be an atomic one belonging to the corresponding domain of the attribute;
3. Entity integrity constraint: Some principles for a primary key;
4. Referential integrity constraint: Some principles for a foreign key.

2.2. Entity-Relationship Model (ER-Model)

As a tool for database analysis and design, the entity-relationship model (ER-model) [1] uses the geometries of entities, relationships, and attributes to show the blueprint of a conceptual database. An entity is an object recognized from the real world such as people, event, product, supplier, and so on, while a relationship refers to the relationship between two or more entities. An attribute is used to represent a feature of an entity or a relationship. A database design diagram drawn with the ER-Model is called the entity-relationship diagram (ERD). The common geometries of the ER-Model elements are shown in Table 2.

Table 2. The common geometries of the ER-model.

ER-Model Elements	Symbols
Entity	
Weak Entity	
Relationship-Entity (Bridge Entity)	
Relationship	
Identifying Relationship	
Attribute	
Key Attribute	
Composite Attribute	
Multivalued Attribute	

An ERD example is shown in Figure 1. This is an ERD of a simple school database with four entities: Students, selections, courses, and employees, where bridge-entity selections is converted from a many-to-many relationship. The relationships between entities describe as follows:

1. A student can select many courses and vice versa;
2. An employee can teach many courses, but a course can only be taught by an employee.

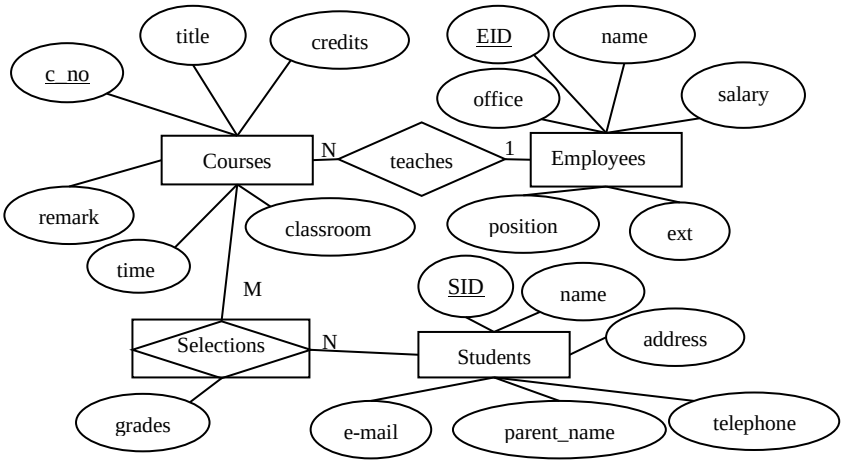

Figure 1. An entity-relationship diagram (ERD) example.

2.3. Big Data

What is big data? Different ages have different answers. Today, big data refers to materials that are difficult to store in RDBs and cannot be processed by stand-alone data analysis and statistical tools.

This data needs to be stored in a large parallel system with tens or hundreds of machines, while the NoSQL database system just has these features, suitable for storing big data and can quickly access data for various application processing. Big data can apply to all areas of daily life (such as social networking, e-commerce, etc.) and scientific research (such as astronomical meteorology, clinical medicine, etc.), and the continued growth of data has forced people to reconsider the storage and management of data [3,4].

The features of big data (i.e., 4V) are described as follows [3].

1. Volume: It refers to the large-scale growth of data volume faced by enterprises.
2. Variety: It refers to the type of data including a variety of texts, videos, pictures, geographic locations, and information generated by sensors.
3. Value: It refers to the commercial value of the data after analysis. In the case of video, for example, one-hour of video, in continuous monitoring, the information that may be useful is only one or two seconds. Therefore, how to refine the value of data more quickly through powerful machine learning algorithms is an important issue of big data.
4. Velocity: It refers to that enterprises not only need to know how to quickly collect data, but also must know how to process, analyze, and pass back the results to users to meet their immediate needs.

2.4. NoSQL Databases

The definition of NoSQL can be found on the official website [2] as follows. NoSQL databases are next generation databases mostly addressing some of the points: Being non-relational, distributed, open-source, and horizontally scalable [2]. The original intention of NoSQL development is to become modern web-scale databases. The development began early 2009 and is growing rapidly. Often more characteristics apply to NoSQL databases such as: Schema-free, easy replication support, simple API, eventually consistent/BASE (basically available, soft-state, eventual consistency [3]), a huge amount of data and more. In addition, the misleading term "NoSQL" can also be explained as "Not Only SQL" that means if RDB is suitable to use then use it while if RDB is unsuitable to use then use alternatives [3].

The features of NoSQL databases are described as follows [1–5].

1. Non-relational: NoSQL databases do not use relational database model, neither does support SQL join operations. In addition, unlike RDBs to obtain advanced data through join operations, NoSQL databases do not support join operations, the related data needs to be stored together to improve the speed of data access.
2. Distributed: Data in NoSQL databases is usually stored in different servers and the locations of the stored data are managed by metadata.
3. Open-source: Unlike most RDBs that require a fee to purchase, most NoSQL databases are open source and free to download.
4. Horizontally scalable: Increase or decrease multiple normal servers to meet the data processing capacity of NoSQL database.
5. Schema-free: Unlike RDBs need to define database schema before inserting data, NoSQL databases do not need to do this. Therefore, NoSQL databases can flexibly add data.
6. Easy replication support: NoSQL databases mostly support master-slave replication or peer-to-peer replication, making it easier for NoSQL databases to ensure high availability.
7. Simple API: The NoSQL database provides APIs for network delivery, data collection, etc. for programmers to use, so that programmers do not need to design additional programs to make writing programs easier.
8. BASE is an abbreviation for "basically available, soft-state, and eventual consistency," and the meanings are described as follows.

(1) Basically available: The DB system can execute and always provide services. Some parts of the DB system may have partial failures and the rest of the DB system can continue to operate. Some NoSQL DBs typically keep several copies of specific data on different servers, which allows the DB system to respond to all queries even if few of the servers fail.
(2) Soft-state: The DB system does not require a state of strong consistency. Strong consistency means that no matter which replication of a certain data is updated, all later reading operations of the data must be able to obtain the latest information.
(3) Eventual consistency: The DB system needs to meet the consistency requirement after a certain time. Sometimes the DB may be in an inconsistent state. For example, some NoSQL DBs keep multiple copies of certain data on multiple servers. However, these copies may be inconsistent in a short time, which may happen when a copy of the data is updated while the other copies continue to have data from the old version. Eventually, the replication mechanism in the NoSQL DB system will update all replicas to be consistent.

According to the statistics of the NoSQL database official website [2], the current number of NoSQL databases has more than 225. Moreover, some NoSQL databases are widely used in many famous enterprises such as Google, Yahoo, Facebook, Twitter, Taobao, Amazon, and so on [3].

2.5. The Survey Papers of NoSQL Databases

There are several survey papers discussing NoSQL databases related technologies, features and examples, as well as several factors that affect the applicability of NoSQL databases. The focuses of these papers are described as follows.

(1) Hecht and Jablonski [6] evaluated the relevant technologies of some of the four common NoSQL database categories (i.e., key value store, document Store, wide column store, and graph databases) to assist users in selecting an appropriate NoSQL database. Related technologies include data models, queries, concurrency controls, partitions, and replication.
(2) Lourenço et al. [7] compared several quality attributes for several NoSQL databases. The evaluated NoSQL databases contain Aerospike, Cassandra, Couchbase, CouchDB, HBase, MongoDB, and Voldemort, while the quality attributes include availability, consistency, durability, maintainability, read and write performance, recovery time, reliability, robustness, scalability, and stabilization time.
(3) Corbellini et al. [8] reviewed the basic concepts of four common categories of NoSQL databases and compared some databases for each category. In addition, this paper also discussed how to select an appropriate NoSQL database from existing databases. The decision-making factors include data analysis, hardware scalability (horizontally scalable and BASE [3,4]), flexibility schema, fast deployment of servers (replication and sharding configuration), distributed technology, etc.
(4) Khazaei et al. [9] illustrated the basic concepts of four popular NoSQL database models and evaluated some databases for each model. In this paper, the authors discussed several factors to be considered in order to select an appropriate NoSQL database, such as data model, access patterns, queries, non-functional requirements (including data access performance, replication, partition, horizontally scalable, BASE [3,4], software development and maintenance, etc.).
(5) Gessert et al. [10] linked functional requirements, non-functional requirements in the NoSQL database to the used technologies, and provided decision trees to assist users in selecting the appropriate NoSQL database, where:

    (a) Functional requirements include sorting, full-text search, and so on;
    (b) Non-functional requirements include data scalability, elasticity, and so on;
    (c) Used technologies include sharding, replication, storage management, and query processing;

(d) Evaluated NoSQL databases contain MongoDB, Redis, HBase, Riak, and Cassandra.

(6) Davoudian et al. [11] clarified four factors for deciding a suitable NoSQL database, such as data model, consistency model, data partitioning, and CAP theorem, and further explained the available strategies, features, advantages, and disadvantages for them. This is helpful for selecting an appropriate NoSQL database.

3. The Categories of NoSQL Databases

According to the classification of the NoSQL database official website [2], there are 15 categories of NoSQL databases such as wide column store, document store, key value store, graph databases, and so on, which are based on different data models. This section will explain the basic concepts of each category of the NoSQL database and analyze the characteristics of the data that each category of the NoSQL database is suitable for processing.

3.1. Wide Column Store

This category of NoSQL databases has a complex table schema described as follows [5,12–14].

1. A row key is an identification that has a unique value used to identify a specific record, similar to the primary key of a relation in RDB.
2. A timestamp (abbreviated as ts) is an integer used to identify a specific version of a data value.
3. At least one column family that has the format of "Family: Qualifier = Value," where "Family" is the name of a column family, "Qualifier" is the name of a column qualifier, and "Value" is a real value of a column qualifier stored in text.
4. The name of a column family need to be defined when the table is created, but the name of a column qualifier does not.
5. Users can find the actual data value through the value of a specific row key, the name of a specific column family, the name of a specific column qualifier, and the value of a specific timestamp.

An example is illustrated as follows. An inventory table of 3C products in a wide column store database is shown in Table 3, where:

1. Products_Inventory is the name of the inventory table, which contains two column families, products, and inventory, and has three records with the product codes P001, P002, and P003 as the values of three row keys, respectively;
2. An increasing integer t_i ($i = 1, 2, \ldots, 18$) is the value of timestamp for each column qualifier when a data value of a column qualifier is inserted into the table;
3. Column family products includes four column qualifiers: Classes, title, descriptions, price, and their data values, for example, are "TV", "SONY 55 inch 4K OLED Smart Networked TV", "TBD", and "24999", respectively;
4. Column family inventory includes two column qualifiers: Quantity, place, and their data values, for example, are "10" and "1A", respectively.

Table 3. An example of a data table in wide column store.

Table Name: Products_Inventory			
Row Key	**ts**	**Column Family Products**	**Column Family Inventory**
P001	t1	Products: Classes = "TV"	
	t2	Products: Title = "SONY 55 inch 4K OLED Smart Networked TV"	
	t3	Products: Descriptions = "TBD"	
	t4	Products: Price = "24999"	
	t5		Inventory: Quantity = "10"
	t6		Inventory: Place = "1A"
P002	t7	Products: Classes = "Laptop"	
	t8	Products: Title = "ACER SF514 14-inch laptop"	
	t9	Products: Descriptions = "TBD"	
	t10	Products: Price = "31000"	
	t11		Inventory: Quantity = "20"
	t12		Inventory: Place = "2A"
P003	t13	Products: Classes = "Mobile phone"	
	t14	Products: Title = "ZenFone 5Z"	
	t15	Products: Descriptions = "TBD"	
	t16	Products: Price = "5000"	
	t17		Inventory: Quantity = "8"
	t18		Inventory: Place = "2B"

According to the statistics of the DB-Engines Ranking website [15], Apache Cassandra and Apache HBase are the more widely discussed ones of the wide column store databases.

3.2. Document Store

The terms related to the database model of the document store are described below [5].

- A collection is a group of documents. The documents within a collection are usually related to the same subject, such as employees, products, and so on.
- A document is a set of ordered key-value pairs, where key is a string used to reference a particular value, and value can be either a string or a document.
- JSON (JavaScript Object Notation), BSON (Binary JSON), and XML (eXtensible Markup Language) are formats commonly used to define documents.
- Embedded documents are documents within documents. An embedded document enables users to store related data in a single document to improve database performance.
- Document store databases do not require users to formally specify the structure of documents prior to adding documents to a collection. Therefore, document databases are called schemaless ones. Application programs should verify rules about the structure of a document.

An example of a collection in a document store database is shown in Figure 2. As a JSON file format, this document stores school curriculum data. There are three courses, "Accounting", "Economics", and "Computer Science", in this file. Each course contains four fields, c_no, title, credits, and instructor.

```
{
  {
    "c_no": "C001",
    "title": "Accounting",
    "credits": 3,
    "instructor": "Zoe"
  },
  {
    "c_no": "C002",
    "title": "Economics",
    "credits": 3,
    "instructor": "Wendy"
  },
  {
    "c_no": "C003",
    "title": "Computer Science",
    "credits": 3,
    "instructor": "Cathy"
  }
}
```

Figure 2. An example of a collection in a document store database.

According to the statistics of the DB-Engines Ranking Website [15], the MongoDB and Couchbase Server are the more widely discussed ones of the document store databases.

3.3. Key Value Store

The data in this category of NoSQL databases is stored with the format of "Key → Value" [5], where

1. Key is a string used to identify a unique value;
2. Value is an object whose value can be a simple string, numeric value, or a complex BLOB (binary large object), JSON object, image, audio, and so on;
3. In key value store databases, operations on values are derived from keys. Users can retrieve, set, and delete a value by a key;
4. A namespace is a logical data structure that can contain any number of key-value pairs.

Suppose that an online shopping website uses a key value store database to store data as shown in Figure 3. This database includes several namespaces, such as "products" and "customers" [5], where

1. The key in the namespace "Products" is the ID of products, and the value is the details about products;
2. The key in the namespace "Customers" is the ID of customers, and the value is the details about customers.

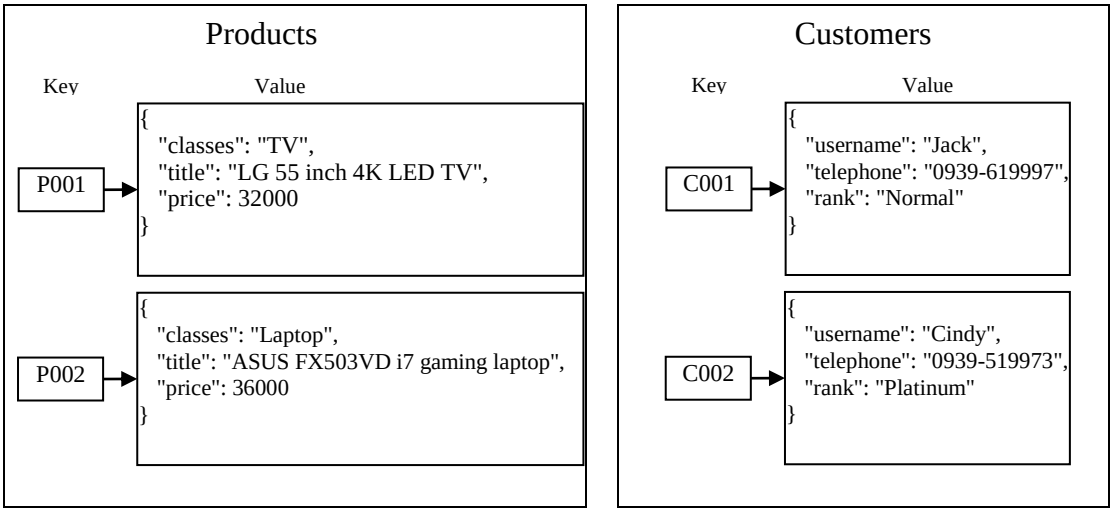

Figure 3. An example of two namespaces in a key value store database.

According to the statistics of the DB-Engines Ranking Website [15], both Redis and DynamoDB are the more widely discussed ones of the key value store databases.

3.4. Graph Databases

The graph database model (GDM) is composed of vertices and edges [5], where

1. A vertex is an entity instance, which is equivalent to a tuple in RDM;
2. An edge is used to define the relationship between vertices;
3. Each vertex and edge contains any number of attributes that store the actual data value.

An Oceania airline is illustrated as an example. The airline needs to store flight hours among some cities. The data can be stored in a graph database as shown in Figure 4. In this graph database, each vertex contains some data such as nation, city, and A2C_time (time from an airport to a city center), and each edge represents the flight duration between two cities [5].

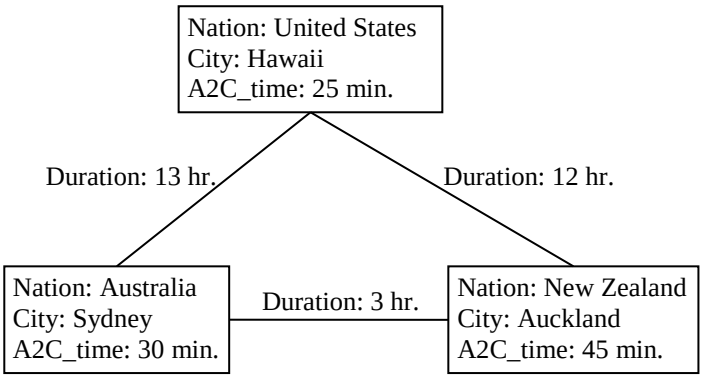

Figure 4. An example of data stored in a graph database.

According to the statistics of the DB-Engines Ranking website [15], Neo4J and FlockDB are the more widely discussed ones of the graph databases.

3.5. Multimodel Databases

The data format of this category of NoSQL databases contains more than two data formats of the other categories of NoSQL databases [16]. According to the statistics of the DB-Engines Ranking

website [15], OrientDB and ArangoDB are more widely discussed ones of multimodel databases. OrientDB contains the data formats of object database, document store, graph database, and key value store; while ArangoDB contains the data formats of document store, graph database, and key value store [2].

3.6. Object Databases

This category of NoSQL databases combines the functions of object-oriented programming languages and traditional databases [1]. A web-based application system, which provides users to order lunch boxes, is illustrated as an example. The data in the object databases are described in the form of a class diagram as shown in Figure 5 [17]. In Figure 5, each rectangle is an object that includes both data items and data processing functions. For example, the object Customers has four data items (account, password, telephone, and e-mail) and two data processing functions (readData() and writeData()). According to the statistics of the DB-Engines Ranking website [15], db4o and Versant are the more widely discussed ones of the object databases.

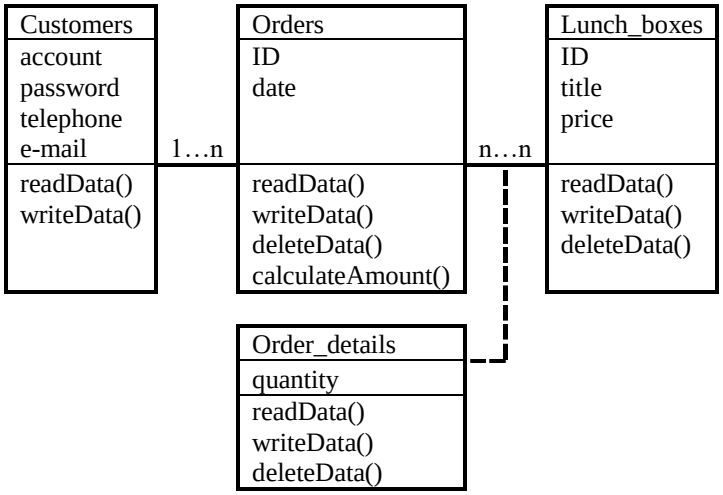

Figure 5. An example of a class diagram in an object database.

3.7. Grid and Cloud Database Solutions

This category of NoSQL databases stores recent access data in random access memory (RAM) and uses grid computing to speed up the time of access data from a database [2]. According to the statistics of the DB-Engines Ranking website [15], Hazelcast and Oracle Coherence are more widely discussed ones of grid and cloud database solutions.

3.8. XML Databases

The files stored in this category of NoSQL databases are based on the XML format [18]. An example of a school curriculum file stored in an XML database is shown in Figure 6. In this XML file, there are three courses, internet of things, artificial neural network, and big data, which have course numbers (c_no), C001, C002, and C003, credits, 3, 4, and 2, and instructors, Amy, Zoe, and Mary, respectively. According to the statistics of the DB-Engines Ranking website [15], Oracle Berkeley DB and BaseX are the more widely discussed ones of the XML databases.

```
<?xml version="1.0" encoding="UTF-8"?>
<Courses>
    <Course c_no="C001">
        <title>Internet of Things</title>
        <credits>3</credits>
        <instructor>Amy</instructor>
    </Course>
    <Course c_no="C002">
        <title>Artificial Neural Network</title>
        <credits>4</credits>
        <instructor>Zoe</instructor>
    </Course>
    <Course c_no="C003">
        <title>Big Data</title>
        <credits>2</credits>
        <instructor>Mary</instructor>
    </Course>
    <!-- ... -->
</Courses>
```

Figure 6. An example of a data file in an XML database.

3.9. Multidimensional Databases

The data in this category of NoSQL databases is stored in a multidimensional array in order to analyze the value of each array element. Suppose a printing company stores data in a multidimensional database as shown in Figure 7 [19]. The printing company needs to analyze the total sales amount of printed products based on three dimensions: Products, branches, and customer rank. For example, the company has two branches, Taipei and Tainan, three products, copy paper, photo paper, and poster, and two customer ranks, platinum member and normal member. The boss of the printing company wants the total sales amount of each branch, each product, and each customer rank. According to the statistics of the DB-Engines Ranking website [15], intersystems cache and GT.M are the more widely discussed ones of the multidimensional databases.

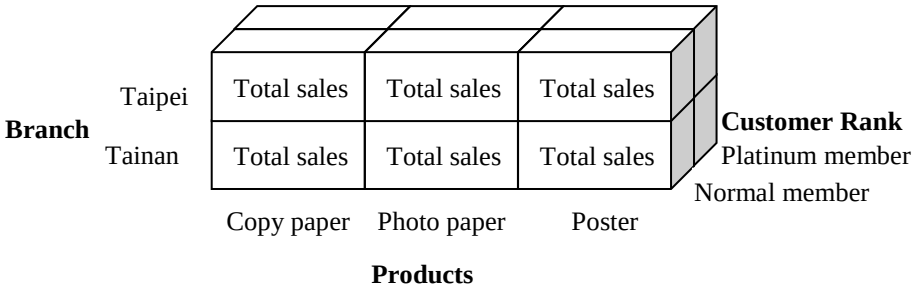

Figure 7. An example of a three-dimensional array in a multidimensional database.

3.10. Multivalue Databases

This category of NoSQL databases is suitable for storing data of multivalued attributes or composite attributes [20]. An example of student data is illustrated in a table of multivalue databases as shown in Table 4. The schema of the table is students (SID, name, and society), where name is a composite attribute composed of the two attributes, First_name and Last_name, society is a multivalued attribute. There are six records in this data table, the name of each student is divided into two parts to save into the attributes, First_name and Last_name, respectively, and the attending societies of each student can

have more than one value. According to the statistics of the DB-Engines Ranking website [15], jBASE and Model 204 Database are the more widely discussed ones of the multivalue databases.

Table 4. An example of a data table in a multivalue database.

Table Name: Students			
SID	**Name**		**Society**
	First_Name	**Last_Name**	
S001	Frank	Lee	{Pop music, Choir}
S002	George	Lu	{Choir, Poetry}
S003	Hank	Wu	{Computer, Guitar}
S004	Ivy	Lin	{Computer, Pop music}
S005	Jack	Chen	{Pop music, Guitar}
S006	Kevin	Yang	{Computer, Poetry}

3.11. Event Sourcing

This category of NoSQL databases is suitable for storing events that occurred in the past in order to track the status of a specific event. An example about a lecture registration system to store the data in an event sourcing database is shown in Table 5. In this table, the first two fields, time and person, can be considered as an event, and the last field current enrolment number is used to track the number of people currently enrolled in the lecture [21]. According to the statistics of the DB-Engines Ranking website [15], event store is the most widely discussed one of the event sourcing databases.

Table 5. An example of a data table in an event sourcing database.

Time (dd/mm/yy)	**Person**	**Current Enrolment Number**
22/12/2018 12:30	Amy	1
25/12/2018 10:40	Ruby	2
28/12/2018 13:20	Cindy	3
29/12/2018 14:10	John	4
30/12/2018 15:00	Mary	5
31/12/2018 16:50	Zoe	6

3.12. Time Series Databases (TSDBs)

This category of NoSQL databases is designed to handle time series data [22,23]. An example of air quality data is illustrated as follows. Assume that an observing station measures the air quality index (AQI) and the density of PM2.5 once an hour and transmits the measurement result to a time series database (TSDB), and the results in 2018 are shown in Table 6 [24]. According to the statistics of the DB-Engines Ranking website [15], Informix Time Series Solution and influxdata are the more widely discussed ones of the TSDBs.

Table 6. An example of a data table in a time series database (TSDB).

Measurement Time (dd/mm/yy)	Air Quality Index (AQI)	The Density of PM2.5
01/01/2018 00:00	156	45
01/01/2018 01:00	101	29
01/01/2018 02:00	97	19
...	...	...
31/12/2018 21:00	133	34
31/12/2018 22:00	135	36
31/12/2018 23:00	141	43

3.13. Scientific and Specialized DBs

This category of NoSQL databases is designed to solve scientific and professional issues. For example, BayesDB allows users who have not been statistically trained to solve basic science problems, and GPUdb is a database suitable for distributed computing [2].

3.14. Other NoSQL Related Databases

The NoSQL databases in this category seem to be able to be categorized into several other categories mentioned earlier, but the official website of NoSQL database [2] categorizes them into this special category without giving any explanation for the characteristics of this category of NoSQL databases. Therefore, we have no way to know why this category is needed and the reasons why these NoSQL databases are assigned to this category. According to the statistics of the DB-Engines Ranking website [15], eXtremeDB is the most widely discussed one of other NoSQL related databases.

3.15. Unresolved and Uncategorized

Any NoSQL database will be assigned to this category of NoSQL databases if it cannot be classified into any of the previously mentioned categories of NoSQL databases. According to the statistics of the DB-Engines Ranking website [15], Adabas and CodernityDB are the more widely discussed ones of the unresolved and uncategorized databases. By the way, the characteristics of the two categories of NoSQL databases, unresolved and uncategorized and other NoSQL related databases, are similar. This is based on the classification of the NoSQL database official website [2]. We do not know the basis of the classification.

3.16. Summary

The basic concepts of each category of NoSQL databases have been described. Then, all the categories of NoSQL databases are analyzed to get the results that each NoSQL database is suitable for processing certain features of data. The results are summarized in Table 7.

Table 7. Summary of suitable data features for NoSQL databases.

Categories of NoSQL Databases	Suitable Data Features
Wide Column Store	Three-dimensional data. Applications that often search for specific field data.
Document Store	Semi-structured files, such as XML, JSON, and so on.
Key Value Store	One-dimensional data, which is stored in key-value pairs.
Graph Databases	Data stored in a graphic structure. Suitable for data of social network relations, recommendation systems, and so on.
Multimodel Databases	Determine data features suitable processing based on the data format of a specific database.
Object Databases	The object-oriented concepts are used to describe the data itself and the relationship among the data. Suitable for computer aided design (CAD) and office automation.
Grid and Cloud Database Solutions	Applications that need to search recent access data frequently.
XML Databases	Data stored in XML files.
Multidimensional Databases	Applications that often analyze data in multiple dimensions.
Multivalue Databases	Data with multivalued attributes or composite attributes.
Event Sourcing	Data with events that occurred in the past for tracking the status of something.
Time Series Databases	Data related to time series.
Other NoSQL Related Databases	Unable to know.
Scientific and Specialized DBs	Data suitable for scientific research or computing.
Unresolved and Uncategorized	Data based on the data format of a specific database.

4. Choose an Appropriate Database

4.1. The Principles of Database Selection

If an enterprise prepares to choose a NoSQL database, it must understand the following questions according to the cultures and characteristics of the enterprise.

1. Understand the current problems, goals, and challenges of the corporate operation database.
2. The engineers of the IT center or database administrators (DBAs) must decide to continue using the current RDB or change using a NoSQL database based on the needs of enterprise and their expertise.
3. If changing to use a NoSQL database, the IT engineers or DBA first select a suitable category of NoSQL databases based on the features and formats of the enterprise's operating data.
4. When deciding which NoSQL database to choose, the IT engineers or DBA can make a decision according to the needs of the enterprise, the characteristics of each database, as well as the reputation and popularity of each database on websites (for example, DB-Engines Ranking website [15], vschart [25]). The more websites we query for this information, the more accurate the reputation and popularity of each database, and the more we can find the right NoSQL database.

4.2. Database Selection Case 1

Suppose that a 3C shopping website uses an RDB to store data for a long period of time, and this RDB generates 300,000 records per day. Users reflect that the website is slower, and hope the data processing speed to be as fast as possible, so the business owner asks the information department staff to solve this problem.

The head of the information department traced the reasons according to the boss instructions and found that the reasons for the slower access speed of the data are not only a large amount of data generated every day but also the need for many users to merge several tables of RDB with a

large amount of data. Therefore, the supervisor recommends using the NoSQL database as a solution because NoSQL databases can merge some tables of RDB in advance so that when querying a NoSQL database, the desired data can be read quickly without waiting much time to do join operations.

After the business owner agrees, the head of the information department will then decide which NoSQL database to use. The decision process is as follows.

1. The most suitable category of NoSQL database is the wide column store because access to the database often requires searching for data in a specific field.
2. According to the DB-Engines Ranking website [15], the wide column store databases that are more commonly discussed on the internet are Apache Cassandra and Apache HBase.
3. According to the experimental results of Chen et al. [26], the time of Apache HBase to read data is less than that of Apache Cassandra. Therefore, Apache HBase is recommended as the NoSQL database used by the enterprise.

4.3. Database Selection Case 2

In order to understand the news reporting strategy of a peer, a famous newspaper must collect online news from various newspapers or media, so about tens of thousands of online news are stored for analysis every day. The information staff found that RDB could not provide quick access to a large amount of data in time; therefore, it is recommended to use the NoSQL database to solve the problem of too slow access rate.

In response to this question, the head of the information department must decide which NoSQL database to use for the newspaper company. The decision-making process is as follows.

1. Since the newspaper needs to collect files generated by a large number of instant messages such as tens of thousands of online news and related readers' messages every day, it is necessary to replace the RDB with a NoSQL database.
2. There are fifteen categories of NoSQL databases available, and the category found to be suitable for storing news multimedia materials is the document store.
3. According to the DB-Engines Ranking website [15], the document store database that is often discussed on the internet has two NoSQL databases, MongoDB and Couchbase Server. Since the former has a higher market share than the latter, it is recommended to use MongoDB as the NoSQL database for the company.

4.4. Database Selection Case 3

An American multinational retail enterprise has the following challenges of its website [27].

1. This corporation intends to promote "online recommendation system optimization" to recommend suitable products for each customer who visits the website of this corporation.
2. To achieve the goal, the database must join a large number of customer and product data quickly to analyze in time the needs and trends of customers to products.
3. However, the RDB this company used cannot meet the above requirement.

For the above reasons, the head of the information department in this corporation decides to use a NoSQL database to replace RDB. The decision process is described as follows.

1. The most suitable category of NoSQL database for the enterprise is graph databases because graph databases is the most suitable for the recommendation system as described in Table 7.
2. According to the statistics of DB-Engines Ranking website [15], the most discussed NoSQL database in graph databases are Neo4j and FlockDB.
3. Since Neo4j has the best market share among all graph databases [27]; thereby, Neo4j is recommended as the NoSQL database used by this enterprise.

5. Conclusions

The main contents of this paper are as follows. First of all, we introduce the basic characteristics of the fifteen categories of NoSQL database (such as the wide column store, document store, key value store, and graph databases, etc.) in the NoSQL database official website [2]. Then we analyze the characteristics of the data that each category of NoSQL database is suitable for processing. Next, we propose some principles and key points for reference to help enterprises to find an appropriate NoSQL database from more than 225 ones when enterprises intend to abandon the use of RDB to use NoSQL database. Finally, we illustrate three cases, 3C shopping website, newspapers, and the US retail industry, to demonstrate how a particular company can choose a suitable NoSQL database to improve its competitiveness and customer services.

In summary, if a company abandons RDB and switches to NoSQL DB, it needs to consider the characteristics of the company's data in order to find the right DB. The transaction data of the e-commerce industry often needs to be related, the suitable NoSQL DB category is the wide column store, and Apache HBase is a good choice. The news materials of the news industry have semi-structured features. The suitable NoSQL DB category is the document store, and the better choice is MongoDB. The retailer data needs to be used by the recommendation system, so the suitable NoSQL DB category is the graph databases, and the best choice is Neo4j. We hope that these principles and examples will help decision makers to change databases correctly.

Author Contributions: Conceptualization, J.-K.C.; methodology, J.-K.C. and W.-Z.L.; writing—original draft preparation, W.-Z.L.; writing—review and editing, J.-K.C.; supervision, J.-K.C.; project administration, J.-K.C.

Funding: This research received no external funding.

Acknowledgments: Thanks to the reviewers for providing a lot of valuable comments to make this paper more complete.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chen, H.A. *Database System: Concept, Design, and Implementation*, 3rd ed.; XBOOK MARKETING Co., Ltd.: Taipei, Taiwan, 2013. (In Chinese)
2. NoSQL Databases. Available online: http://nosql-database.org/ (accessed on 20 January 2019).
3. Pi, S.J. *Establish the Cornerstone of Big Data: NoSQL Database Technique*, 2nd ed.; TopTeam Information Co., Ltd.: Taipei, Taiwan, 2016. (In Chinese)
4. Lu, J.H. *Challenge Big Data, How to Process Big Data in Facebook, Google, Amazon? Use NoSQL to Get 10 Billion Annual Hard Disk Data*, 2nd ed.; TopTeam Information Co., Ltd.: Taipei, Taiwan, 2015. (In Chinese)
5. Sullivan, D. *NoSQL for Mere Mortals*, 1st ed.; Pearson P T R: London, UK, 2015.
6. Hecht, R.; Jablonski, S. NoSQL Evaluation: A Use Case Oriented Survey. In Proceedings of the 2011 International Conference on Cloud and Service Computing, Hong Kong, China, 12–14 December 2011.
7. Lourenço, J.R.; Cabral, B.; Carreiro, P.; Vieira, M.; Bernardino, J. Choosing the right NoSQL database for the job: A quality attribute evaluation. *J. Big Data* **2015**, *2*, 18:1–18:26. [CrossRef]
8. Corbellini, A.; Mateos, C.; Zunino, A.; Godoy, D.; Schiaffino, S. Persisting big-data: The NoSQL landscape. *Inf. Syst.* **2016**, *63*, 1–23. [CrossRef]
9. Khazaei, H.; Fokaefs, M.; Zareian, S.; Beigi-Mohammadi, N.; Ramprasad, B.; Shtern, M.; Gaikwad, P.; Litoiu, M. How do I Choose the Right NoSQL Solution? A Comprehensive Theoretical and Experimental Survey. *Big Data Inf. Anal.* **2016**, *1*, 185–216.
10. Gessert, F.; Wingerath, W.; Friedrich, S.; Ritter, N. NoSQL database systems: A survey and decision guidance. *Softw.-Intensiv. Cyber-Phys. Syst.* **2017**, *32*, 353–365. [CrossRef]
11. Davoudian, A.; Chen, L.; Liu, M. A Survey on NoSQL Stores. *ACM Comput. Surv. (CSUR)* **2018**, *51*, 40:1–40:43. [CrossRef]
12. Dimiduk, N.; Khurana, A. *HBase in Action*, 1st ed.; Oreilly & Associates Inc.: New York, NY, USA, 2012.
13. Lu, J.H. *Hadoop: Practical Technical Handbook*, 2nd ed.; TopTeam Information Co., Ltd.: Taipei, Taiwan, 2014. (In Chinese)

14. George, L. *HBase: The Definitive Guide*, 1st ed.; Oreilly & Associates Inc.: New York, NY, USA, 2011.
15. DB-Engines Ranking. Available online: https://db-engines.com/en/ranking (accessed on 4 March 2018).
16. Multi-Model Databases (Wikipedia). Available online: https://en.wikipedia.org/wiki/Multi-model_database (accessed on 15 June 2018).
17. Wu, R.H. *Object-Oriented System Analysis and Design: An MDA Approach with UML*, 4th ed.; BestWise Co., Ltd.: Taipei, Taiwan, 2013. (In Chinese)
18. Document-Oriented Database (Wikipedia). Available online: https://en.wikipedia.org/wiki/Document-oriented_database (accessed on 15 June 2018).
19. Multidimensional Databases. Available online: https://docs.oracle.com/cd/E12478_01/rpas/pdf/150/html/classic_client_user_guide/basic_rpas_concepts/multidimensional_databases.htm (accessed on 5 May 2018).
20. MultiValue (Wikipedia). Available online: https://en.wikipedia.org/wiki/MultiValue (accessed on 15 June 2018).
21. Introducing to Event Sourcing. Available online: https://msdn.microsoft.com/en-us/library/jj591559.aspx#sec1 (accessed on 16 January 2018).
22. Time Series Database (Wikipedia). Available online: https://en.wikipedia.org/wiki/Time_series_database (accessed on 16 January 2018).
23. Time Series (Wikipedia). Available online: https://en.wikipedia.org/wiki/Time_series (accessed on 16 January 2018).
24. Central Weather Bureau. Available online: https://www.cwb.gov.tw/eng/index.htm (accessed on 10 July 2018).
25. vsChart.com: The Comparison Wiki: Database List. Available online: http://vschart.com/list/database/ (accessed on 18 February 2019).
26. Chen, C.Y.; Chang, B.R.; Tsai, H.F.; Guo, C.L. Empirical Analysis of High Efficient Remote Cloud Data Center Backup Using HBase and Cassandra. *Sci. Progr.* **2014**, *2015*, 1–10.
27. Neo4j: Walmart Case Study. Available online: https://neo4j.com/case-studies/walmart/ (accessed on 10 December 2018).