The study of application and evaluation with NoSQL databases in Cloud Computing

Tse-Chuan Hsu[1], Dong-Meau Chang[2], Hsin-Jan Lee[3]

[1]Present Office
Digilink Corporation
Taipei City, Taiwan(R.O.C.)
hsu@digilinktw.com

[2]Department of Journalism
Hsuan Chuang University
Hsinchu City, Taiwan(R.O.C.)
morgan@hcu.edu.tw

[3]Department of Information Management
Hsuan Chuang University
Hsinchu City, Taiwan(R.O.C.)
s8304192008@gmail.com

***Abstract* - Access reliability based of information is an important factor affecting cloud applications. In this research investigates on how Heroku cloud systems necessity file system database publishing technology, with different conditions to test the read and write data access capabilities. To facilitate comparison, will use the commercial database MongoDB and PosgresSQL as a test database and the study will establish the local PostgresSQL database as a control group. The purpose of this study was to investigate the cloud computing environment, the credibility of information use in the different database structures. About the test case, we will conduct the contains data writing, reading, and query tests. The results of this study support through a new type of cloud resource salesforce Heroku subordinate mode to explore, study the behavior of software services used in commercial construction and deployment validation tests will help in the understanding of cloud architecture, design structure in different databases, data use analysis reliabilities.**

Keywords*：*Cloud Computing, Database, Design Structure, Cloud Architecture

I. INTRODUCTION

On the cloud computing platform, database systems have gradually developed into DaaS (Database-as-a-Service), which have been combined into the database access service to the cloud service. In the development cloud of the entire evolution database, gradually have two different directions, one direction is that because there are many existing enterprise application systems are still using relevant databases, relational and therefore must be retained database mechanism. So it should be managed into a traditional relational database, moved under the cloud computing architecture. Therefore, some commercial database vendors have their products to the cloud architecture inside, the essence is still the original structure.[1][2]

Another trend is the development of cloud computing since it considered a distributed architecture, use this platform features abandoning the conventional relational database architecture, the way back to save the file table format, not only can enhance the use of performance, but also reduce the complexity of database management. Such a database is generally referred to as NoSQL, said they were not sophisticated database system structure, often only one database file can be in operation.[3][4]

In the former, for example, database system vendors should consider how different cloud framework for the proper functioning of the existing system, after all, the bottom of the storage unit has been completely different. Another important issue is that if the database is in the cloud, How to sync the user data and local database? The different architectures have different problems to be overcome in orderto effective solving this problem, some experts and scholars study how to make the user having to change the habits;therefore, the database can be transferred to the cloud computing platform, it would be a simple solution.

In this study, we will be implemented in two database validation, Section II. Compare about different computing database systems. Section III. The NoSQL database architecture. Section IV. A case study of NoSQL database implements. Section V. Conclusion.

II. COMPARE TO different computing database systems.

Cloud computing set up via NoSQL data base, by following paragraph will verify the different of SQL database and NoSQL data base.

A. SQL database

The traditional databases migrate to cloud platform like Amazon or Virtual machine such as Oracle, Microsoft SQL server and MySQL, there are different type of database, it can be in the cloud (either as a virtual machine image, or as a service, depending on the manufacturer). SQL databases are difficult to scale, which means that they are not suitable for a cloud environment, although cloud-based SQL database services are trying to solve this problem. [18]

978-1-4799-6566-3/14 $31.00 © 2014 IEEE
DOI 10.1109/TSA.2014.18

I. Relational Database inception internet applications made for such high scalability requirements, therefore, the main consideration is designed to simplify operations, generate SQL database language contributed to the standardization of interfaces. Oracle database company and led the development of upstream and downstream industry chain.[17]

II. SQL database, the basic concept is that, it has Relational database. Relational database strictly uses relations (frequently called as tables) to store data. A relational database matches data by using common characteristics found in the dataset. And the resulting group is termed as Schema.[11]

B. NoSQL database

NoSQL databases are built to service heavy read/write loads and are able scale up and down easily and therefore they are more natively suited to running on the cloud. NoSQL databases, such as Apache Cassandra, CouchDB and MongoDB, are another type of database which can run on the cloud. However, most contemporary applications are built around an SQL data model, working with NoSQL databases often requires a complete rewrite of application code.[12] [5][6][7][8]

NoSQL database development, it's quite fit in the cloud computing environment to build; in current years, several emerging database systems such as CouchDB, MongoDB and the like, are of such of the databases [9]; which Google BigTable proposed repository can be said was the best representative of such a database. Basically, NoSQL databases means that there are no complex data structures, but is closer to a single file-based database.[13][14] Although the simple structure, similar to the way the table storage format; NoSQL database architecture do not have the correlation, does not mean that there is no concept of the field; but should you want it as a super-large table, on which the user can define the fields you want to use.

Traditional relational database is a database schema (database schema), table definitions and relationships between tables do declaration. So whether it is stored units, field index and Query plan optimization techniques are quite mature. Each vendor in a special data fields, data access on the performance characteristics of the development of its own database. PostgresSQL [19]field provided as follows:

Name	Aliases	Description
bigint	int8	signed eight-byte integer
bigserial	serial8	autoincrementing eight-byte integer
bit [(*n*)]		fixed-length bit string
bit varying [(*n*)]	varbit	variable-length bit string
boolean	bool	logical Boolean (true/false)
box		rectangular box on a plane
bytea		binary data ("byte array")
character [(*n*)]	char [(*n*)]	fixed-length character string
character varying [(*n*)]	varchar [(*n*)]	variable-length character string
cidr		IPv4 or IPv6 network address
circle		circle on a plane
date		calendar date (year, month, day)
double precision	float8	double precision floating-point number (8 bytes)
inet		IPv4 or IPv6 host address
integer	int, int4	signed four-byte integer
interval [*fields*] [(*p*)]		time span
json		JSON data
line		infinite line on a plane
lseg		line segment on a plane
macaddr		MAC (Media Access Control) address
money		currency amount
numeric [(*p*, *s*)]	decimal [(*p*, *s*)]	exact numeric of selectable precision
path		geometric path on a plane
point		geometric point on a plane
polygon		closed geometric path on a plane
real	float4	single precision floating-point number (4 bytes)
smallint	int2	signed two-byte integer
smallserial	serial2	autoincrementing two-byte integer
serial	serial4	autoincrementing four-byte integer
text		variable-length character string
time [(*p*)] [without time zone]		time of day (no time zone)
time [(*p*)] with time zone	timetz	time of day, including time zone
timestamp [(*p*)] [without time zone]		date and time (no time zone)
timestamp [(*p*)] with time zone	timestamptz	date and time, including time zone
tsquery		text search query
tsvector		text search document
txid_snapshot		user-level transaction ID snapshot
uuid		universally unique identifier
xml		XML data

Cloud database is an emerging field; many technologies are in the development stage. More representative system has Google's BigTable and in the open source world to get considerable support for MongoDB. But the two companies in the storage and access to technical information are not the same. Google's BigTable is a key / value pairs as a storage unit, and MapReduce and GFS Chubby technology to construct their own storage systems. Based on a custom attribute class, description type of the relevant art field cannot be used. These special fields different from the general database field attributes are object-oriented classes, and other derivatives of the fields can be used to make the extended class Expando dynamic properties or ListProperty collection category.

MongoDB is based on documents (document), as the smallest unit of processing; the so-called file is just a "collection" concept that can contain nested. In general, the file is stored JSON format, if the data size of the file exceeds 16 MB, the system will use a system similar to Google GFS, to deal with such large data. Basically, MongoDB has no concept of the field, but in order to make a traditional database applications successfully transferred, the system provides two tools references and embedded documents to establish relationships between the data. This reference the tool used to create one or more relationships.

III. THE NOSQL DATABASE ARCHITECTURE.

Since each NoSQL database still has its special field, basically, in order to meet various storage formats and special access ways; therefore, like Google particularly defined property category, to describe the field type that cannot be used . These special fields different from the general database field attributes, the following table lists the comparative value of its underlying data storage type description Property category. Type any value in the table, can be used in the extension type Expando dynamic properties or ListProperty compiled class. As for the actual value of the field access, then relies on Google's API (API) to complete.

In table 1 shows the current GAE database type, its non-traditional SQL schema architecture [15], GAE the use of technology as cloud database NoSQL data structures, using NOSQL need first to establish SQL field type, by creating GAE field, re-create structure, GAE provides fields, many are self-Google-defined fields; table "value type" refers to a defined value compared to traditional database fields, as well as practical in the corresponding underlying in BigTable value.[16]

TABLE I, GAE NoSQL data structure.

	Property	type	
1	StringProperty	str unicode	Unicode (str ASCII)
2	ByteStringProperty	ByteString	
3	BooleanProperty	bool	False < True
4	IntegerProperty	int long	values
5	FloatProperty	float	
6	DateTimeProperty DateProperty TimeProperty	datetime.datetime	

Primary Key:

In this case, we use GAE environment to described in Django, basically still respect on the basis of traditional database design, so the primary key must be set, in addition also provides automatic tired increasing function. GAE is a table for each of them, automatically provides two fields id and key, basically it do not have an additional set of primary key values, unless the special reasons, users can still set their own master key.

Relational tables

In BigTable architecture, the tables are stored in the cloud, so in the past called the database schema design method, which is relatively unimportant. That any time you can change the definition of form fields, and even the presence of existing data table can also be adjusted, while GAE also offers two special fields: ReferenceProperty and StringListProperty, through ReferenceProperty this field can be established from the associated primary key / foreign key; Likewise, can also be used in the same way, the establishment of many to many relationship table

```
class Author(db.Model):
  name = db.StringProperty()

class Story(db.Model):
  author = db.ReferenceProperty(Author)

story = db.get(story_key)
author_name = story.author.name
```

The multi-table definition and access operations:

```
class Contact(db.Model):
  # User that owns this entry.
  owner = db.UserProperty()

  # Basic info.
  name = db.StringProperty()
  birth_day = db.DateProperty()

  # Address info.
  address = db.PostalAddressProperty()

  # The original organization properties have been replaced by
  # an implicitly created property called 'companies'.

  # Group affiliation
  groups = db.ListProperty(db.Key)

class Company(db.Model):
  name = db.StringProperty()
  description = db.StringProperty()
  company_address = db.PostalAddressProperty()

class ContactCompany(db.Model):
  contact = db.ReferenceProperty(Contact,
                        required=True,
                        collection_name='companies')
  company = db.ReferenceProperty(Company,
                        required=True,
                        collection_name='contacts')
  title = db.StringProperty()
```

This code illustrates the form PYTHON syntax defined, the code can learn from traditional relational tables defined in the important items: correlation between tables, such as one-to-one, and many-to-many relationship in the table structure in GAE only reference property to construct, but did not carefully distinguish what kind of association. This is why repeatedly stressed that such a cloud database system is completely different from a traditional RDBMS.

IV. A case study of NoSQL database implements.

We chose this case study conducted MongoDB database design application, first, this study illustrate the advantages of MongoDB

- High performance, very fast
- There is no fixed table structure, not in order to modify the table structure and data migration
- Query language is simple, easy to use

In this case, we use heroku architecture through new types of conduct practical cases verify, in this case, after the establishment of a commercial website modular shelf space cloud environment, using heroku mongo database architecture platform support, and support for station management interface design. Via noSQL test, create a non-relational database module, as NOSQL information form for the design itself, without the advantages of a relational database schema therefore the present study, in Mongo database, to be re-defined class structure, providing calls on the front page table program, through class redefined for data form cross comparison, its substantive verification.

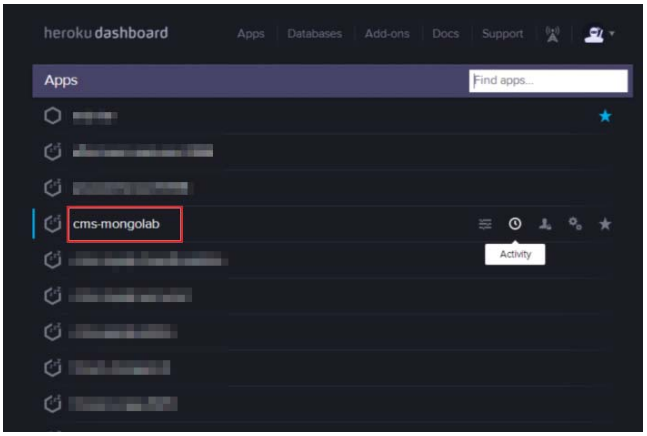

Fig. 1 Mongo databases project developer

By the main panel design, Mongo database supports user-defined data structures, in this case we have to group data table, for example, first with traditional relational databases the same attributes required to define the form in advance of each data category, figure 2, 3 shows that we need to build the catalog, and can use the collection metrics.

Fig. 2 Mongo databases database metrics (1)

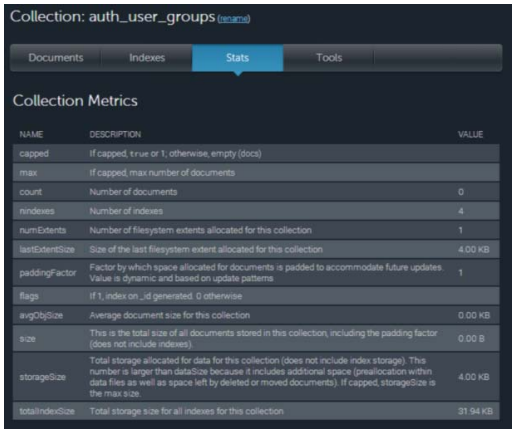

Fig. 3 Mongo databases database metrics (2)

In figure 4and 5, when MongoDB indexed for each document, and the establishment of the state when the index value becomes collection, can check the status value. In simple terms, collection library similar to relational tables, but it is more like a document field. In Figures 3 and 4 is author_user_groups the table of contents, in MongoDB management interface, the content of any collection, are in JSON format to display the user if not used to this notation, you can also change the way in the form of the list to be displayed. As the part of the data manipulation, there is no standard SQL syntax is available, the system must be provided in accordance with API MongoDB, and do queries and information literacy movement. In fiugre 5 is a sample query and the results.

Fig. 4 Databases command (NoSQL)

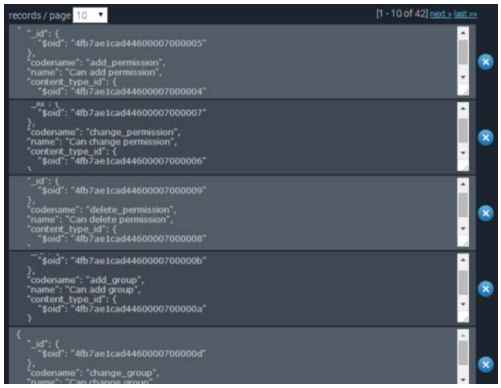

Fig. 5 Databases command (NoSQL)

In this study, through the Heroku platform test, including the back-end database, and are used PostgresSQL MongoDB system, the following description separately.

To make the database can be used for the actual operation of the site, particularly the introduction of the concept of middleware, that is, between the application server and the back-end database, especially with a layer of a database API, as processing data access interface. The advantage of this approach is that you can let the program developers, not against each other database systems, one to write the appropriate code to reduce application development time away. In this study, using Django application development framework provided by the Object Relational Mapping (ORM) as middleware, combined with API provided by various manufacturers, you can use the same command for different data in the database to do the movements access .

In PostgresSQL and MongoDB is an entirely different database systems, so the definition of a table, or to do some fine-tuning, especially in the traditional database design, will gain automatic tried to do the main key fields (primary key), but not such MongoDB field, it is particularly necessary to change the field properties, the following two code for the page record form ‘tbl_pages’ defined in different database systems.

```
#in PostgresSQL:
-----------------------------------------------------------------
class tbl_pages(models.Model):
        pg_id = models.AutoField(primary_key=True)
        pg_link = models.CharField(max_length=150)
        pg_title = models.CharField(max_length=255)
        pg_cont = models.TextField()
        pg_nav = models.IntegerField()
        pg_order = models.IntegerField()
        tbl_nav = models.ForeignKey(tbl_nav,null=True)
-----------------------------------------------------------------
```

```
#in MongoDB
-----------------------------------------------------------------
class tbl_pages(models.Model):
        pg_id = models.IntegerField(primary_key=True)
        pg_link = models.CharField(max_length=150)
        pg_title = models.CharField(max_length=255)
        pg_cont = models.TextField()
        pg_nav = models.IntegerField()
        pg_order = models.IntegerField()
        tbl_nav = models.ForeignKey(tbl_nav,null=True)
-----------------------------------------------------------------
```

Data insert and comparative inquiry

(1) Data Insert

Insert data to the RDBMS tables where we can use standard SQL syntax commands to accomplish:

insert into tbl_name (col_name, ...) values (fld_value, ...)

As for the real action is written, API developed by the vendor to complete. In MongoDB, the system provides two different ways to the user, the first one is by the system for different programming languages such as C #, Java, Node.js, PHP, Python and Ruby-written-driven programs (drivers) to complete writing data. The following example is an example of using the Python language:

```
SEED_DATA = [
    {
        'decade': '1970s',
        'artist': 'Debby Boone',
        'song': 'You Light Up My Life',
        'weeksAtOne': 10
    },
    {
        'decade': '1980s',
        'artist': 'Olivia Newton-John',
        'song': 'Physical',
        'weeksAtOne': 10
    },
    {
        'decade': '1990s',
        'artist': 'Mariah Carey',
        'song': 'One Sweet Day',
        'weeksAtOne': 16
    }
]
```

Another way is to use the REST API MongoDB provides to complete the insert data. REST is due to pass information between pages in common formats; the following explains how to use java script to add "Documents".

```
POST /databases/{database}/collections/{collection}
Content-Type: application/json
Body: <JSON data>

Example (using jQuery):

$.ajax( { url:
"https://api.mongolab.com/api/1/databases/my-db/collections/my-coll?a
piKey=myAPIKey",
        data: JSON.stringify( { "x" : 1 } ),
        type: "POST",
        contentType: "application/json" } );
```

The above methods in different systems, we must use a different syntax and programming language to insert the data. To overcome these differences and reduce development difficulties, with the following Django ORM way you can write data to a different database systems:

```
wObj = tbl_pages()
wObj.pg_link = request.POST['pg_link']
wObj.pg_title = request.POST['pg_title']
wObj.pg_cont = request.POST['pg_cont']
wObj.pg_nav = navid
wObj.pg_order = request.POST['pg_order']
#in mongoDB need to prepare the primary key
import time
wObj.pg_id = int(time.time())
#add the F key
navObj = tbl_nav.objects.get(nav_id =navid )
wObj.tbl_nav = navObj
wObj.save()
```

(2)Data Select

SQL is a query language syntax itself, so the search function only can combine different operators can also use JOIN, UNION way to do arithmetic table between tables, the following formula is the query command:

```
select "column1"
  [,"column2",etc]
  from "tablename"
  [where "condition"];
  [] = optional
```

As seen from the above methods in different systems, we use a different syntax and programming language to insert data. To overcome these differences and reduce development difficulties, with the following Django ORM way you can insert data to a different database systems:

```
cursor = songs.find({'weeksAtOne': {'$gte': 10}}).sort('decade', 1)
```

Followed by the REST API usage, the following formula is the query:

```
GET /databases/{database}/collections/{collection}
```

Optional parameters

[q=<query>][&c=true][&f=<fields>][&fo=true][&s=<order>][&sk=<skip>][&l=<limit>]

We can according to the above formula, the corresponding parameters into the system will return the results of the query.

Similarly, we use Django ORM framework to complete the inquiry data:

```
pg = tbl_pages.objects.filter(pg_nav = 2,pg_order=id)
```

V. CONCLUSION.

In this research, through the cloud NoSQL technology, commercial website development validation, discussion on NoSQL architecture, how to create a cloud database management subordinates save mode, this study found that NoSQL databases need to establish a data form their own class module past the traditional database, without creating a form associated form their own links. Due to the technical use of NoSQL, breaking the existing database written form, through the brief instructions written form for data read write process and reduce the complexity of the form established, while subordinating the cloud space quickly, establish a database design, a large number of items in the data alone. The data are too scattered, how to deal with system performance real-time data is written to the commercialization of reading applications, will be one of the future discussion can be extend inside this research project.

REFERENCES

[1] A. Cuzzocrea, "Analytics over large-scale multidimensional data: the big data revolution!," in Proc. of the ACM 14th international workshop on Data Warehousing and OLAP, DOLAP'11, New York, USA, 2011, pp. 101-104.

[2] V, Gopalkrishnan, D. Steier, H. Lewis, and J. Guszcza, "Big data, big business: bridging the gap," in Proc. of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications, BigMine'12, New York, USA, 2012, pp. 7-11.

[3] J. Han, "Survey on NoSQL database," in Proc. of the 6th International Conference on Pervasive Computing and Applications, ICPCA'11, Beijing, China, 26-28 Oct. 2011, pp. 363-366.

[4] J. Polorny, "NoSQL databases: a step to database scalability in web environment," in Proc. of the 13th International Conference on Information Integration and Web-based Applications and Services, iiWAS'11, New York, USA, 2011, pp. 278-283.

[5] MongoDB. [Online] http://www.mongodb.org/.

[6] MongoDB Production Deployments. [Online] http://www.mongodb.org/display/DOCS/Production+Deployments

[7] MongoDB Drivers. [Online] http://www.mongodb.org/display/DOCS/Drivers.

[8] NoSQL. [Online] http://nosql-database.org/.

[9] R. Cattell, "Scalable SQL and NoSQL data stores," ACM SIGMOD Record, vol. 39, no. 4, Dec. 2010.

[10] PC3. [Online] http://twiki.ipaw.info/bin/view/Challenge/ThirdProvenanceChallenge.

[11] SQL No SQL compare http://www.thewindowsclub.com/difference-sql-nosql-comparision

[12] Database models http://www.creativeworld9.com/2013/05/cloud-computing-hosting-database-model.html

[13] R. Burtica, "Practical application and evaluation of no-SQL databases in Cloud Computing," in proc. of the IEEE International conference on System Conference, SysCon'12, Bucharest, Romania, 19-22 Mar. 2012, pp. 1-6.

[14] M. P. Zhu, "Querying combined cloud-based and relational databases" in proc. of the International Conference on Cloud and Service Computing, CSC'11, Uppsala, Sweden, 12-14 Dec. 2011, pp. 330-335.

[15] Google App Engine. [Online] https://developers.google.com/appengine/.

[16] GQL. [Online] http://code.google.com/appengine/docs/python/datastore/gglreference.html.

[17] G. Sanders, S. Shin, "Denormalization Effects on Performance of RDBMS," in proc. of the 34th Annual Hawaii International Conference on System Sciences, HICSS'01, Vol. 3, NY, USA, 3-6 Jan. 2001, pp. 3013.

[18] Nidhi Khurana , Dr Rattan Datta "Logical Data Model For Cloud Computing " International Journal of Advanced Research in Computer Science and Software Engineering. April 2013.

[19] PostgreSQL 9.2 Manuals Documentation chapter 8 Data Types [Online] http://www.postgresql.org/docs/9.2/static/datatype.html