Comprehensive Analysis of Cloud based Databases

Comprehensive Analysis of Cloud based Databases

[1]Dr. E. Saravana Kumar, [2]Dr. Selvaraj Kesavan, [3]Dr. R. Ch. A Naidu, [4]Dr. Senthil Kumar R, [5]Latha
[1]Associate Professor,
[2]Advisor Solution Architect in DXC Technology, India,
[3]Professor & H.O.D ,
[4]Associate Profesor,
[5]Assistant Professor
[1, 3,4,5] Dept. of CSE, The Oxford College of Engineering, Bommanahalli, Bangalore-560068
saraninfo@gmail.com, selvarajkesavan@gmail.com, chanr789@gmail.com, senkr.raj@gmail.com, latha.shreeram@gmail.com

Abstract

With standard change in the computing architecture and distributed, wide range processing approach leverage the large data to be analyzed and extract the content. The recent advancement in Web technology enables the user to produce process and ingest content of any structure. Cloud computing helps to provide computing infrastructure, centralized storage and seamless services for organizations and users. In a digital world, data is core aspect of the economy, become part of government, enterprise, social sectors and every individual in a day to day life. Data analysis is gathering of strength in both research and production communities. There are many social networking sites, web, enterprise and IoT data require being stored in secured, centralized data store. It also demands data to be stored without any predefined schema and able to retrieve in near real time. When demand goes up application needs to be scaled to sustain for the demand. Similarly, data store should support scalability to support applications demand. Cloud providers and third-party organization offers numerous data storage, database options. This article highlights various available data storage options, suitability and limitations.

Keywords: Open-Source Database, Cloud storage, Data Storage

1. Introduction

Data science becoming the integral part of every business and individual life. Organizations are spending huge amount in data science to keep up the heat in today‟s competition and improve their business outcome. It helps to derive business insights, key metrics and predictive analysis from the data captured. Data science and its relevant fields are highly valuable; it helps management teams to make strategic business decisions. Many companies around the world are exploring the data science and trying to build the data science platforms.

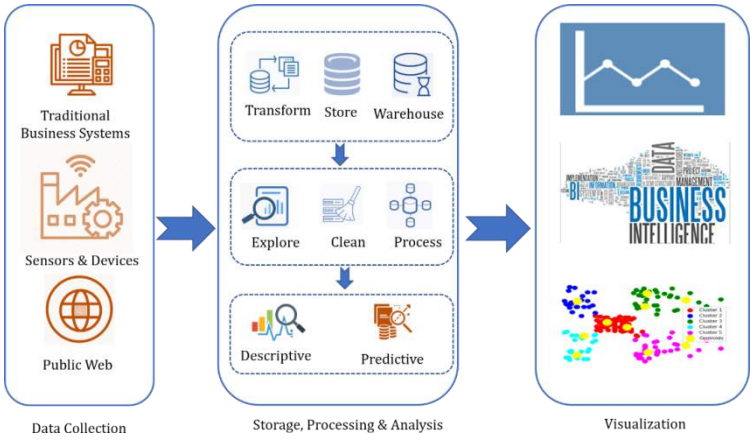

Figure 1. Data science pipeline

With the invent of Web technologies, data with volume, velocity and variety coming from major sources such as Internet of things/sensors, web &social network and traditional business systems. The data source may be structured varieties or unstructured varieties or combinations of both. Data science uses the numerous approaches to clean, explore, transform the data before applying techniques like machine learning deep learning data visualization text analytics to analyze the data. It is tightly integrated with artificial intelligence to make reality of decision-making robots. The Fig.1 shows the architecture of data science Pipeline with detailed manner.

Data science extracts knowledge and provides insights of various form of data for business process efficiency data is important and key part of cloud computing. Data captured from many industry verticals/users such as business & commerce, Entertainment, Science, social network, medical, manufacturing etc.. Data in raw or extracted format brings lot of value to organizations and individual. It helps to expedite the business decisions and improve quality of life. The evolving technologies help to capture Terabytes of data, store in large repositories, and analyze the data with complex data analysis approach using distributed/parallel processing. Data store to Data warehousing helps to store, process, analyze commutative and historical data from single or multiple data sources. Data analysis tools, algorithms and framework helps to derive descriptive and predictive analytics from data and aids to quick decision. The critical aspect of the pipeline is to collect and store the data from diverse sources which produce different data formats.

The data captured from various sources includes structured, single dimension, multidimensional, tabular, spreadsheet, time series, documents formats. The storage mechanism is simultaneously evolving to address the need and support for multiple applications and data formats. The selection of data base is depending on multiple factors such as

- Requirement of Relational between elements
- Amount of data to store
- Real Time Streaming of Data
- Scalability, reliability and availability requirement
- Multi-tenancy
- Cost

Structured Query Language (SQL) is mainly designed for the data requires relational with elements. It deals structured data with defined attributes. The scalability of SQL recognized by Web2.0 companies with large, increasing data and framework needs Facebook, Amazon and Google etc. Because of its sheer size and lack of structure, traditional relational database management system (RDBMS) unable to handle very large big data. These kinds of very large data require a flexible non-relational model that supports fast access to data for processing.

Relational databases are predefined, expensive to scale, and have high storage costs. They are also transactional, consistent, and use mature SQL and a relational mindset. There is no SQL database that can meet this need. On the other hand, non-relational (NoSQL) databases are flexible, scalable, and resilient. They feature commodity storage, are not transaction-oriented, and are eventually consistent. Non-relational databases are also programmable with SQL-like and use a new mindset. No-SQL databases are playing vital role in recent applications in multiple fields. It provides free flow operations without any bottleneck in the performance and use a different types of data models, which includes text document, graphical models, dictionary and columns, helps unique way to query the data. Horizontal scaling and flexible data model definitions. Distributed and developed for very large-scale forms of data storage and enormous parallel processing data across a different number of servers and achieve speed at scale, overcoming RDBMS limitations and inefficiencies. With built-in high-resiliency architecture, it helps to store dynamic schema, semi-structured data for low-latency applications and update, retrieval in real time. This increasing interest among NoSQL and DBMS‟s evolved with a focus on improved performance and consistence with reliability. The major No-SQL data bases are Mongo DB(Document), Redis (Key-value),Amazon Redshift (Columnar), Cassandra (Columnar), HBase (Columnar),Dynamo DB(Document DB-stores JSON/XML) and GraphDB .

The major contributions of the proposed paper covers three main points.

- Provides list of various available data storage options, highlights the criteria to choose appropriate database, cache and limitations.
- Elaborately discuss the No-SQL architecture, retrieval mechanism and Caching strategy
- Detailed comparison of various No-SQL storage options with key metrics

Apart from the main points the remaining the paper is organized in other section. Section 2 presents the various related research work in the area of data base and data storage mechanisms and options. The evolution of database paradigm and classifications are discussed in Section 3. The various aspects of No-SQL Database Storage architecture, retrieval mechanism and selection strategy are captured in Section 4. In section 5, the comparison of key No-SQL databases with key metrics are discussed in detail. Finally, section 6 concludes the paper.

2. Related work

Database helps applications to store, update and retrieve information. SQL and No-SQL databases are widely used by the organizations for storing data collected from various sources. Cloud and IoT evolving exponentially, it is critical to have appropriate databases for store, retrieve, and process data on real time. Sensors/Things connected to server via network and deliver connected industry solutions for efficient control and improved human experience. Billions of connected devices are an indicator of IoT and transmits data in high velocity, variety and volume. The real value of IoT is on data which helps business insight and enable data-driven economy.

Database management systems (DBMS) and databases are critical to everyday business and work

[1].Relational Database Management Systems (RDBMS) is old, pioneer system used many years [2]. The need of new databases is inevitable to develop and use new generation digital applications such as social networking, business visualizations, web 3.0 and connected sensors/devices [3]. Authors [4] performs independent investigation on the NoSQL and SQL performance of databases based on dictionary of key-value stores. The various operations such as comparing reading, writing, deleting and instance operations on key-value stores of SQL and NoSQL. Apart from that, authors investigated the operation of iteration across all the keys. A framework of abstract key-value pair [5] designed implemented and tested using the above basic operations. A detailed comprehensive survey [6] of various approaches and deployment mechanisms of data-intensive areas in the cloud gains major attention in both research and industry. An analysis of different design decisions of every approach and its fitness to support various areas of applications and different end-users. Different views of some of the open issues and future questions related to scalability of large scale database in economical processing on the cloud is addressed. Bartholomew [7] explains the differences of SQL and NoSQL databases and its history. The authors [8] compared functional and non-functional features of graph database; dictionary based key value stores, column store database and documents. In each category, authors selected one database on document store (MongoDB), column database (Cassandra) stores, key value stores and graph based databases (Redis and Neo4j)

A qualitative research [9] has been conducted on SQL and No-SQL data base and detailed, intensive analysis and comparison performed. The NoSQL and SQL performance compared experimented and analyzed by the authors [10].IoT and IIoT are one of the major sources in providing data. The authors [11] have analyzed and performed comparisons of SQL, NoSQL in the context of Internet of Things. Authors [12] performed analysis on the effectiveness problem of data format for the purpose of storing and processing in RDBMS technology. SQL and NoSQL databases have been compared [13] to use for an small IoT application of water sprinkler system and analyses the merits, fitness, performance of NoSQL and SQL under different scenarios.

Even though, the various research papers discuss the functionalities and applications of different databases, it does not provide complete view and comparison of multiple databases and applications with various metrics. However, it is important to have all-in-oneview of different databases along with features, architecture, flexibility, limitations and applications. This paper gives comprehensive view of data storage in the market, selection strategy and comparison which helps for developers, organizations to understand the functionalities and choose right data base for their need.

2.1 Evolution of Database and classification

Information Systems and data bases are the two components concurrently existing in the computer science technology. Appropriate data base management system required for storing and retrieval of data in a meaningful manner. RDBMS databases applies the ACID model in transactions management strong atomicity, consistency with Isolation and Durability property. Unfortunately the transaction properties cannot be applied on NOSQL databases because the priorities are different. The NOSQL applies the BASE model which ensures the Basic Availability of the database, Soft state of the model and the Eventual consistency of the database. Over the years, many researchers have conducted various researches in the mentioned field to justify BASE in NOSQL and ACID properties of RDBMS database.

Various structures were applied and improved to enhance the read operation searching techniques. Big companies developed Proprietary NoSQL to focus on their needs. Google developed BigTable and Amazon developed DynamoDB. The successful outcome of these proprietary softwares started the open source community to develop such a software, Hbase, MongoDB , Cassandra, DynamoDB, Hypertable, and Redis are the popular NoSQL database developed by these communities.

2.2 Data quality and governance

2.2.1 Data store

Persistently store and manage repository of data, files, organized and unorganized data. It includes file system, SQL, No-SQL databases etc. Quick access of data on very large number of nodes allowed in non relational database management systems like distributed DBMS. Distributed DBMS has very good access performance and rich query abilities which are limited key-value semantic in other databases. These limitations are noted in Bigtable , Azure and Dynamo models performs comparatively better in peer-to-peer network.

2.2.2 Data Lake

Data lake is a repository to hold large amount of structured, unstructured data accumulated from various sources at any scale. It is act as a raw repository which holds data in original format as it received from the sources. It stores all data types whether structured or unstructured. Storing data in data lake is comparatively cheap .It is suitable for Machine Learning, Predictive analytics, data discovery and profiling

Ex: Amazon S3, Amazon Glacier

2.2.3 Data Warehouse

A data warehouse is a database optimized to store data which are formatted / processed to give shape to the data. The data structure, schema are defined in advance in order to optimize read / write operations. The data helps to derive insights, analytics which helps organization to take management decisions. Storing data in data warehouse is comparatively costlier than Data Lake. It is suitable for Batch reporting, BI and visualizations

Ex: Amazon Redshift

2.2.4 Database

Technology stores data in an organized manner using Data base management system. It is classified as SQL and No- SQL Databases. Entire row is stored Row-oriented databases in a physical block. Secondary indexes used to achieve the high performance in read operations. Instead of packing the whole rows into a block each column organized into set of physical blocks in Column-oriented databases. Better performance achieved in master/slave architecture type with an overhead that single point of failure presented in master node. Every node is identical in peer-to-peer cluster and same responsibilities assigned to each node. Every node achieves fault tolerance with overhead of consistency maintenance is high.

RDBMS, usually executes query language (SQL), is combination interrelated tables. The tables contains rows and columns. The schema represents the relationships among tables and column represents the attribute. These RDBMS databases are highly consistent by design. RDBMS queries are used in banking applications for online transactions. Cloud providers which are popular are IBM DB2 , SQL Server , MySQL and Oracle. MySQL platform is open source cloud platform.

Table model is not used in non-relational databases which is called NoSQL because it stores the content as a document regardless of any structure. Unstructured data is well-suited in non-relational databases technology like image and video contents of social media.

2.2.5 SQL Databases

SQL data bases have predefined schema. Database stores data in table row format in the databases

such as MS SQL Server Oracle DB Server open source MySQL, PostgreSQL and SQL databases. Traditionally data warehouse was used in these systems in which online transactional processing (OLTP) implemented better than for analytical purposes.

In a SQL database, all of the columns are read by query for all of the rows to satisfy the query condition It creates a bottleneck in the performance especially analysis related use cases. But in relational database systems it is well suited for transaction processing across many tables . The multiple tables joined using complex join operations.

2.2.6 No-SQL Databases

A different scalable paradigm instead of relational databases are No-SQL data base. The cloud computing currently uses scalable applications but the challenge of scalable database is storing models. NoSQL data base based on cloud can be considered as an alternative. No fixed scheme is required for distributed data bases which used No-Sql. It is well suitable for unstructured data which need to fit into schema based database or the design changes frequently. Different data model varieties utilized such as key- value based data model, columnar data model , graphs and document model

The familiar, widely used No-SQL DB are

- Document -Mongo DB,
- Columnar - Amazon Redshift, Cassandra, HBase, Dynamo DB
- Key-Value – Redis, Riak, Dynamo DB, Berkeley DB
- Graph - Neo4J

2.2.7 Key - Value -Redis, riak

Implementation of NOSQL database is based on Key-value based databases , which are simple . It uses hash method of dictionary . By using hash it matches the key to the values A hash table will be implemented to separately store the unique keys with pointers refers to the corresponding values. Scalar data types can be used for values , for example integer and complex structures. BLOB, List structure and JSON are the complex structures. In NoSQL databases design and implementation can be performed using key value stores. The API such as get, put , delete can be used by clients for reading, writing using NoSQL. These operations are uses primary key concept. It enable scalability and high performance can be achieved.

2.2.8 Document based – Mongo DB, CouchDB

Mongo DB, CouchDB database systems are not relied on schemas which is storing the document. Each record in document databases is an associated data which treats the document as semi-structured data.

A database object encapsulate everything related to document database. It gives less dependency and more agility. The encapsulated and encoded database objects are provided in some formats like JSON an XML . JSON provided object notation of java script.

The JSON , BSON and XML formats uses stored data structure. It uses Key-Value pair in document store and each key has a corresponding value provides the data structure. This is called as *document.* An object containing metadata value with array , string, binary and date. It gives a method to index and apply query based on features in document

2.2.9 Graph based- Neo4j, GraphDB

Graph based databases are entirely different when compared with NOSQL database. Graph based databases that rely on graph structure. In graph structure nodes and edges are connected with others through structure relations like tree. Property Graphs contains *nodes* . The *relationships* among the nodes represented in property graphs. Properties are the attributes which can be store nodes along with relationships. Nodes represented as entities of graph, Node metadata tagged with *labels* used to represent the contexts. Direction provided by relationships which can be bi- directional. The connections are named among two nodes. More than one relationship can be specified between two different nodes and each relationship with a direction, type, start and an end node.

2.2.10 Columnar – Casandra, H-Base

Data stored n Column databases, is cells of columns. These are grouped into families of column. Multidimensional nested sorted map of maps format used in these databases. data identified in the innermost map contains a timestamp. A cell is used to store the data This will be mapped to column and mapped with column family. A Collection of column families found using a row key. The row key on sets of columns implemented for read and write operations. It will be stored as a continuous entry in the storage which is improving the performance.

2.3 Choosing Appropriate Database

Scenario	Database Type
Session Storage, user preference, shopping cart data	Key value
Blogging, web Analytics, real time analytics, ecommerce applications	Document Database
Content management, IoT,log analysis	Columnar DB
Spatial Data ,Social network, transport, routing Applications	Graph DB
- Distribute Static and media contents Low cost ,highly durable - Data store for computation and large-scale analytics	Object Storage
Structured data with Query	Relational DB
Meta data store	Key value and Caching
Uninterrupted Key-Value	Object and key value storage

2.3.1 No-SQL Database Storage and retrieval mechanism

The storage and retrieval of No-SQL contains strong consistency. All clients performs two-phase commit protocol on updating the dataset which uses XA transactions, ACID properties. It provides high availability: The availability of requested data to all the clients found at least one copy of the data, even if some of the machines in a cluster is down, Partition-tolerance: the total system keeps its characteristic even when being deployed on different servers, transparent to the client. Flexible schema supported in NoSQL databases which enables developers to quickly adopt the databases. The schema supports structured and unstructured data enables efficient processing which supports high performance in executing queries in the vary large scale environment.

2.3.2 Master less Ring Architecture

Identical role played by all the nodes in master less ring architecture; Without the master node concept all the other nodes are communicating with each other nodes through scalable and

distributed protocol called as "gossip."across datacenters architecture based on a peer-to-peer ring can be deployed which can be done by Cassandra.

The Cassandra performs logical division of racks and the required snitches inside the cluster, this installation determines the best node to be stored and rack for replicas of the node

Mongos query routers are used to build the hierarchal architecture MongoDB. The architecture contains one or more routers and one or more shards used to run the mongod. The replica node built the mongodb and it sets consist of a primary node and secondary nodes.

A master-less peer-to-peer architecture architecture along with many data centers is supported by Riak Commercial version architecture. One datacenter acts primary and many different data centers supports with primary node which will be synchronized.

A peer-to-peer architecture Couchbase, consists of cluster manager, index, query in each and every node. The multiple datacenter in Couchbase support, updation flowing from one datacenter to other datacenters. These flowing can be done bilaterally where conflicts can be resolved by each cluster which is the owner of certain set of partitions.

2.3.3 Peer To Peer Architecture Systems

The upshot of the discussion is that the client-server model generates a service hot-spot at the server and increasingly hot network utilization the closer one gets to the server. Traffic that is diffuse toward the client is concentrated near the serve. The system is represented in fig.2

Distributing the service helps. But, as long as clients out number servers, there is a natural imbalance. The solution to this imbalance is a peer to peer architecture where services are provided (hopefully, but not likely) in proportion to their use.

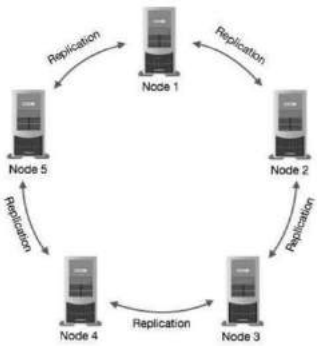

Figure 2: Peer-to-Peer System

3. Challenges of Peer-to-Peer systems

The peer-to-peer architecture systems present several challenges:

- Describing and searching for content
- Naming
- Finding objects and directory services
- Stability of the peers
- Trust of the peers

It becomes difficult to search for object in peer-to-peer systems because high-level searches don't

localize well. For this type of thing, we really do want a distributed map-reduce or other parallel search or large, in memory monolithic database, or both. We don't want to have to ask a large number of distant peers and then need to coordinate the results.

Once can identify each object we want, we need to find it. One option is a fully distributed directory service, another is a directory service distributed among select peers, and a third is a distributed hash. Stability of peers is obviously an issue. Some peers will be well resourced and stable, others will be thin and brittle. The stability of the system depends upon having enough stable resources to mitigate the impact of a smaller quantity of brittle resources. One common solution here is to appoint willing, richer, more stable, longer-serving peers as "super peers" and giving them more responsibility, perhaps incentivizing them by proving more or better service (or just by the good feeling from being a good citizen).

Trust is the nearly impossible part. Insert the whole discussion about public key infrastructure here. It is really challenging to trust the identity of hosts. With luck, we can get a hash of what we want from enough sources trusted enough to trust the answer -- and then we can check that what we get matches the hash. Thus, the must brittle part of this is really trusting the search results and/or human-name to hash-name mapping.

3.1 Distributed Hashing: Consistent Hashing and Chords

Another idea for a peer to peer system is to implement a huge distributed hash table. The problem with traditional hash tables, though, is that they don't handle growth well. The hash function produces a large number, which is then taken modulus the table size. As a result, the table size can't change without needing to rehash the entire table -- otherwise the keys can't be found.

A hash value which is independent of the hash table size is used by a hashing scheme in distributed hashing. Keys can be found from the result even if the table size changes. This is not supported of a distributed hash table, The keys found even the nodes enter the system.

Chord protocol is a technique for doing this hashing. A logical ring used to view the world in this protocol. For the selected „m „number of bit key, the bit contains the logical positions 0 to 2m-1. Think of them as hours on a clock. Some of these positions have actual nodes assigned to them, others do not. Every node node "need" the successor only like the token ring, but the topology is known for the entire ring which will be used to handle failures.

For more than one key each and every node is responsible because there are only fewer nodes than actual addresses which is hours on the clock. Mapping is performed on keys to actual nodes by identifying the keys to the "closest" node which can be equal or greater than in number.

a brute force searching technique is use to find a key of the circle, but instead each node keeps a "finger" which points to the next node, 2, or 4 or 8 nodes away, etc. In other words, every node points to nodes exponential manner farther and farther away. These pointers are stored in a table such that the i^{th} entry of the table contains a pointer to a node that is 2i away from it, e.g. at position node number + 2i. The keys, if suppose any node is not present at the exact location then the next passing greater node will be used. This modified arrangement enables the possibility of search for a bucket with the time complexity O(log n) . The time complexity found with each step, which either find the right node, or performs cutting the search space into half.

In order for a node to join, it simply is added to an unrepresented position (hour on the clock) within the hash table. It gets its portion of the keys from its successor, and then goes live. Similarly, disappearing from the hash simply involves spilling ones keys to one's successor.

3.2 Schema less Architecture

NoSQL DB provides Schema less architecture which helps to create schema on-the-fly and it does not need any predefined schema.

Database	Architecture	When to Use
Cassandra	peer-to-peer	**SQL-style data types** **Steep learning curve when switching from SQL**
MangoDB	**master-slave**	**Lots of highly unstructured data** **Loosely coupled objectives**

3.3 Caching Design Patterns and Strategy

- Consistent Caching (Sharding)
  - Issue with Modulo (CRC32) ,keys mapped to new node when scaled
  - Consistent hashing – Ring methodology
  - Client libraries
- Lazy population (updated cache when app reads , cache miss lead to direct query)
- Write on through
  - Cache upgraded to real time when database updated
  - Cache miss avoided
  - Minimize app fetch delay
  - Cache always up-to date
  - Filled unnecessary objects
  - Lot of cache churn due to repeated update
  - No strategy to repopulate cache node when it fails
- Expiration Date
  - TTL to cache keys
  - Short TTL for rapid changing data types (news, leaderboards etc)
  - **Russian doll caching** – nested records with own cache keys , delete only cache keys which needs to be updated (suspect that it will affected by database update)
- Thundering Herd (Dog piling)
  - Cache miss , millions of user hit the DB in parallel increases DB throttling
  - Setting TTL to popular data will impact DB performance (if data not updated but TTL expire will trigger millions to request to be routed to DB)
  - When new cache node added , many requests routed to DB
  - Run script to populate cache before app make requests
  - For new node , run script to populate data in new cache node
  - Prewarming node (incase regular addition/deletion of cache node?
- Cache everything
  - Heavily hit queries and expensive calculations
  - Caching data is stale data (no longer fresh and not appropriate in some scenarios)

3.3.1 Comparison

- HBase (ordered keys, semi-structured data), Cassandra, BigTable,
- CouchDB (name or value in text)
- Sherpa or PNuts (JSON, unordered keys)
- MongoDB (based on JSON)

Name of the tools	**Cassandra Database model**	**GraphDB**	**MongoDB**	**Neo4j**	**Redis**
Descriptions of the comparative model	Supports wide-column store based on ideas of BigTable and DynamoDB	The graphs uses Enterprise RDF and with reasoning, external index cluster and synchronization support	One of the most popular document stores	The toos supports Open source graph database	**In-memory data structure store, used as database, cache and message broker**
Supported programming languages	Java, JSS,PHP, Python,Ruby,Scala,.Net	C#,C++,Clojure, Erlang,Go,Haskell,Java, JSS,PHP,Python,Ruby,Scala	Java, JSS,PHP,Python,Ruby,Scala,.Net,coldFusion	Java, JSS,PHP,Python,Ruby,Scala,.Net,Go, Groovy	**Java, JSS,PHP,Python, Ruby,Scala, .Net, Crystal**
The scripting of server-side	no	Java Server Plugin	JavaScript	yes	**Lua**
Partitioning techniques	Sharding supported	none	Sharding supported	none	**Sharding supported**
Replication techniques	Supports selectable replication factor	Supports Master- master replication	Supports Master-slave replication	Causal Clustering using Raft protocol	**Multi-master replication, Master-slave replication**
Consistency concepts of the models	Supports Immediate Consistency, Eventual Consistency	Supports Immediate Consistency, Eventual consistency	Supports Immediate Consistency Eventual Consistency,	Eventual and Causal Consistency configurable in Causal Cluster setup Immediate Consistency in stand-alone mode	**Strong eventual consistency with CRDT**

Transaction techniques	Not supported	ACID Properties supported	ACID Transactions on Multi-document with snapshot isolation	ACID Properties supported	**Atomic execution of commands blocks and scripts and Optimistic locking,**
User concepts supported in models	**Per object the access rights for users can be defined**	**Supported**	**Allows the access rights for users and roles**	**Supports standards wtih Users, roles and permissions. Pluggable authentication (Active Directory, Kerberos, LDAP)**	**Supports password-based access control**

4. Factors impacts database Performance

4.1 Merits

- The databases support elastic scalability which is designed for low-cost commodity hard wares.
- Massive volumes of Big Data Applications can be handled by NoSQL
- Economy: NoSQL databases installation is cheap in commodity hardware even with data volumes and transaction increase. But installation of RDBMS is expensive storage in proprietary servers. Processing and storage is less cost.
- Dynamic schemas: No schema design required for NoSQL databases. In RDBMS, a schema has to be defined first, it makes more difficult because the schema has to be changed every time when the requirements are changed. It ensures that data quality control should be performed on the application. NoSQL has no schema, of the data must be organized properly.
- Auto-sharding: In RDBMS scalability supported vertically, which makes a lot of databases spread among many servers and takes the disk space from multiple servers needed t work. Auto sharding is normally supported in NoSQL databases. Naturally NoSQL automatically spread the data across multiple arbitrary servers. These servers does not required even the application used composition of server pool
- Replication: Automatic database replication is supported in most of the NoSQL databases. This replication used to maintain availability of data all the time. Event maintenance or event of outages maintained using replication. Fully self-healing supported by most of the sophisticated NoSQL which provides automated failover and data recovery. Replication provides the ability to distribution of database across many geographic regions to avoid the regional failures and supports localization.
- Integrated caching: Integrated caching is supported in most of the NoSQL technologies, the frequently-used data kept in the system memory for storing and removing.

4 .2 Limitations of Cloud Database

- Multiple-Data has no relationships among all the data.
- Multi-operation Transactions: Failure to save any one of key among many keys and if any transactions aborted then implementing roll back operation is very difficult
- Query Data by 'value': It support the searching by keys, which is based on information found in the 'value' of the key.
- Operation by groups: Simultaneously several keys cannot run as operations are grouped to one key at a time

5. Conclusion

If the application consists of a small amount of traffic Cloud Datastore will work well and performs better. if database is scaled up to very large number of requests then performance decreases. No wasted capacity will be made for the request given. It is expensive than using data store and required to know the capacity in advance. These will be charged for the additional capacity. Amazon announced auto scaling will be supported in cloud database in near future. Getting the provisioned capacity will help to write the own scripts to handle the scalability issue. Depending on the more capacity requirement it is provisioned, then start to deplete accumulated credits. If credits depleted, then the request will be throttled or failed some time. Depending on the options, the scalability available databases is alluring. The tradeoffs and selecting the right pricing model suitable for the application, enable to free the user from constraints of RDBMS.

References:

1. K. L. Berg, T. Seymour and R. Goel, “History of databases,” International Journal of Management & Information Systems, vol. 17, no. 1, pp. 29-35, 2013.
2. U. Bhat and S. Jadhav, “Moving Towards Non-Relational Databases,” International Journal of Computer Application, vol. 1, no. 13, pp. 40-46, 2010.
3. R. P. Padhy, M. R. Patra and S. C. Satapathy, “RDBMS to NoSQL: Reviewing Some Next-Generation,” International Journal of Advanced Engineering Science and Technologies, vol. 11, no. 1, pp. 15-30, 2011
4. Y. Li and S. Manoharan, "A performance comparison of SQL and NoSQL databases," *2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, Victoria, BC, 2013, pp. 15-19.
5. J. C. Anderson, J. Lehnardt, and N. Slater, CouchDB: The Definitive Guide. O'Reilly Media, January 2010.
6. S. Sakr, A. Liu, D. Batista, and M. Alomari, "A survey of large scale data management approaches in cloud environments," Communications Surveys Tutorials, IEEE, Vol. 13, no. 3, pp. 311-336, 2011.
7. Mahrishi M., Shrotriya A., Sharma D (2012). “Globally Recorded binary encoded Domain Compression algorithm in Column Oriented Databases” in Global Journal of Computer Science and Technology, [S.l.], ISSN 0975-4172.
8. A. Gupta, S. Tyagi, N. Panwar, S. Sachdeva and U. Saxena, "NoSQL databases: Critical analysis and comparison," *2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN)*, Gurgaon, 2017, pp. 293-299, doi: 10.1109/IC3TSN.2017.8284494.
9. K. Sahatqija, J. Ajdari, X. Zenuni, B. Raufi and F. Ismaili, "Comparison between relational and NOSQL databases," *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, 2018, pp. 0216-0221, doi: 10.23919/MIPRO.2018.8400041.
10. Kumar, A., Vengatesan, K., Vincent, R., Rajesh, M., Singhal, A. : A novel Arp approach for cloud resource management; International Journal of Recent Technology and Engineering (IJRTE) at Volume-7 Issue-6, March 2019
11. K. Fraczek and M. Plechawska-Wojcik, "Comparative Analysis of Relational and Non-relational

Databases in the Context of Performance in Web Applications", *Proceedings 13th International Conference Beyond Databases Architectures and Structures (BDAS 2017)*, pp. 153-164, May 30 - June 2, 2017.

12. F. Haleemunnisa and W. Kumud, "Comparison of SQL NoSQL and NewSQL Databases for Internet of Things", *IEEE Bombay Section Symposium*, pp. 1-6, Dec. 2016.
13. K.B. Kumar, SrivydiaSundhara and S. Mohanavalli, "A performance comparison of document oriented NoSQL databases", *Computer Communication and Signal Processing (ICCCSP) 2017 International Conference on*, 2017.