NoSQL v/s Relational Database Systems

1960's, the time when the computer industry was only used for Enterprise systems, do help in business but the impact of it was not that significant. As the data storage capabilities were primitive if compared to today's abilities. Before relational databases were a thing the industry was working on network and hierarchical models. CODASYL which is the acronym for Conference/Committee on Data Systems Languages and the hierarchical model called IMS which is the acronym for Information Management System. There was one more database system which belonged to the ***'Big Blue'*** and was the only commercially successful which was used to help American Airlines with their reservation's data, it was name SABRE acronym for Semi-Automated Business Research Environment. As computers became more and more cheap there was a need for a data storage system which was efficient and quick and also cost was an important factor.

1970, at this point of time in the computer industry IBM was the biggest and most dominant corporation in manufacturing computers for the enterprises. IBM has no competitors and whatever IBM put out in the market became the cornerstone of the industry no matter if it was not up to the mark. Edgar F. Codd was a mathematician working in IBM's research lab, where he published a paper titled 'A Relational Model of Data for Large Shared Data Banks' which introduced the world to the relational databases that we know of today. Codd came up with the idea of relational databases gaining inspiration from the relational calculus of mathematics. He used relational calculus as the model for the database. One of the main reasons relational database got popular is because of its ease in the implementation but as implementation was the boon, lack of efficiency and storage issues were the bane. One question that has always been around is '*Even with a number of flaws, why has relational database been so dominant?*' In this time, as we know was the start of the computer industry, and a lot of people had less knowledge about the data systems. Firstly, a lot of professionals misinterpreted the name 'Relational Database' as someone who isn't fully knowledgeable of the actual structure of the relational database people thought that relational database actually expressed relationships. But the truth was that the relational database was data described in grids or tables as we call them now with columns and rows which has their identifiers or name which in reality had no true relationship information but using the structure can be manipulated to derive a relationship whether it is significant or not. As this was the time where non-technical people had a lot of say in technical things the name sounded catchy for business people they related them to relationships in organizations. The rise of relational databases had a lot to do with IBM's dominance in the computer industry as it was the only legit computer company as Hewlett Packard was not a big player till then, so whatever IBM did became the default industry standard and the best solution for the problem as it had no competitors.

NoSQL is the industry's answer to the problems of relational databases. NoSQL as the name suggests was founded on the idea of database engines that do not work on SQL. As one of the main reasons relational database is not the best its low performance issues and unreliability. Relational Database engines based on SQL provide a primary functionalities of what SQL can perform with some extra additional features. Features such as JOIN which is used to add two tables, TRANSACTION which is small chunk of program an can contain several tasks, LIMIT which returns a limit of records where queried, these set of functions or extra features are not available in any NoSQL engines as these features are one of the main causes of the performance and reliability issues. It needs to be noted that NoSQL is not the perfect alternative of SQL or relational databases in general, using a NoSQL or SQL system totally depends on the requirements of the systems it cannot be argued that NoSQL can provide great service in every scenario but the more rational argument is that NoSQL works better than SQL in a lot of huge applications or datasets while SQL fails to keep up with the performance when the volume is increased significantly. NoSQL as a whole is very different from the traditional relational database systems as in NoSQL the data is not stored in fixed table schemas. MongoDB is one of the most popular NoSQL management systems in the industry and many huge companies have implemented it. It was released in 2009, NoSQL database systems serve well to the large scaled web applications as in these scenarios they outdo the traditional relational database systems. One of the main reasons Mongo does well in these scenarios is the way it stores database, they have adopted BSON which is an acronym for Binary Encoding of JSON in this Mongo stores data as dynamic schemas of JSON like documents. MongoDB aims at four major and important things in new age application development and delivery the first one being flexibility, as the applications are growing faster and the scale of these applications and the reach of the customers is increasing minute by minute a database for such an application needs to be highly scalable and flexible as MongoDB supports replicated servers and also allows us to index it becomes easy for the application providers to scale the application without any concerns. The second one is power or power of convenience, this point is connected to the first one as business have a lot of issues with this the legacy procurement pattern leads to undervalue convenience. MongoDB focuses to solve this problem and create a whole system which works well for the developer and the customer or the user. The next one is speed, and this one has been associated with databases a lot as speed has been looked as one of the primary parameter to judge a good database management system. The upload and retrieval speed often is the distinction between good and bad service and as Mongo's counterpart which the relational database management systems are low on speed when they reach higher volume, speed has been one of the important pillars of MongoDB or NoSQL database management systems in general. The last point is ease of use, and it one of the most effective points as it dictates how the users are going to interact with database management system, it is important for any system to be easy to use for developers and user to implement their applications on. MongoDB primarily has ability to accept large amount of data it is because of the way it is designed when we compare the maximum value size of the two database there is a huge discrepancy in both of them. Oracle database which is the prominent relational database management system used in the market it has a maximum value size of 4 KB while on the other hand, MongoDB has the maximum value size of 16 MB as you can see the start point of both the databases are way off and the relational database is evidently on the lower side. Oracle database uses the integrity model which is named ACID which is the acronym for Atomicity, Consistency, Isolation and Durability. Where atomicity means, ensuring that the whole transaction takes place at once or it doesn't taking out midway actions as there will be no incomplete transactions as they can lead to discrepancies in the database.

Consistency as the name suggests ensure that the database management system is consistent before or after the transaction it can also be noted as the correctness of the database. In a huge database, the database can be accessed from different services and they can be called consecutively, Isolation ensures to maintain the consistency of the database in high transaction situations as inconsistency can lead to a number of problems or the application. Durability is one of the major aspects of a relational database management systems it ensures that once the transaction is completed all of the updates and the modifications to the database are stored in and written to the disk and that they reside in the disk even if a system failure occurs. The MongoDB works on the BASE properties which is an acronym for Basically Available, Soft State, Eventual Consistency all of these properties stem from the earlier important aspects that were the pillars for the creation of a NoSQL database.

As the time passed by there have been a lot of new NoSQL database management systems that have reached the market, following are some examples

- Key – value stores (Distributed Hash Tables):
  - Major applications: Project Voldemort, Tokyo Cabinet, Riyak, Redis.
- BigTable clone(Inspired by Google's BigTable):
  - Major applications: HBase, Hypertable, Cassandra.
- Document oriented Databases:
  - Major applications: MongoDB, couchDB.
- Graph databases(Inspired by Graph theory):
  - Major applications: Neo4j, VertexDB, AllegroGraph, InfoGrid

Redis, is a popular in-memory database system which is used by a lot of giants in the industry the one being Twitter they use Redis to bring new tweets as quick as possible. Key value stores can be used in a number of places the first use case that is very popular is for caching sessions and Redis has an upper hand over Memcached as Redis offers persistence. One of the main features of Redis is its consistent service it can provide a relatively easy FPC platform and due to the disk persistence provided by Redis there will be no fluctuations in the request speeds in the application. As mentioned earlier, twitter uses Redis to bring users the latest tweets it is because of the queuing provided by Redis.

Cassandra, is a unique database management system in this list of various database systems. It is a culmination of Amazon's Dynamo's fully distributed database design with Google's BigTable's data model. Cassandra if put in simple words is a multi-dimensional map. Cassandra has several use cases, but it is also noted that Cassandra cannot be used in any situations as it is designed to perform well in some special situations using it without complete understanding of its plus points will not result in the best performance. Cassandra can work effectively well in tracking applications, and a lot of giant companies such as Google or Facebook or Amazon use Cassandra on a significant level as it ensures great results. Cassandra can also be used in logging systems as in Cassandra write exceeds the data reads by significantly higher margins.

MongoDB, as mentioned earlier it is a very popular oriented database and its use cases are evident. One of the major uses cases is in analytics, the amount of data produced and which needs to be analyzed has increased exponentially and as the data is created from various sources and most of them have different schemas the complexity in collaborating all of these types of schemas becomes a huge task and the basic need to perform significant analysis is that the schema of the data should be similar. Here MongoDB provides the perfect solution, as MongoDB is more flexible and scalable as compared to the traditional database management systems the collaborations of the data becomes relatively easy the development of analytical tools becomes easier and it can result into more dynamic experiences. MongoDB can also prove effective in Real – Time Data Integration as its core aspects provide the necessary infrastructure to build on it.

Concluding this comparison, we have learnt that each type of database system proves to be effective in certain areas, relational database systems are still used because there are some use cases where they are effective. But NoSQL has been a wildcard in this race, it has evolved rapidly as it has formed itself based on the drawbacks of the earlier one it has been successful in eliminating the known problems and has been successful in providing solutions for various needs of the industry.

References:

1) Boicea et.al “MongoDB vs Oracle - database comparison”
2) Burtica et.al’s article “Practical application and evaluation of no-SQL databases in Cloud Computing”
3) History of RDBMS
4) Mongo Use Cases
5) Cassandra use cases
6) Redis Use cases