However, migration from SQL to NoSQL structure appears to be challenging task because it requires validating the system requirements and maintaining same functionality and data integrity of the new NoSQL DB schema

The migration process has two requirements: First, changing and developing the design. Second, migrating the data .

The migration of the data is relatively easier task that the design which has to assure validity, integrity, and performance. NoSQL design can be achieved either by an expert or by using tools.

Because NoSQL DBMS does not support join queries and has no notion of foreign keys, the efficacy of NoSQL design depends on how to model efficiently the relationships in the relational database (SQL). There are three ways to model such relationships, namely, embedding, referencing, and mix of embedding and referencing

Given a well-designed relational database r and a transformation function T that converts a relational database into a NoSQL database, select the transformation that achieves the best performance in term of retrieval time.

Where r is a relational database, T is the transformation function from relational schema to NoSQL structure such that T(r)= s∈S, and S is the set of all different NoSQL structures.

We use a standard benchmark database named Employee. The entity-relationship diagram (ERD) is shown in Figure 1.

//slide 6

1. Relational Database

The ERD shows that the DB contains four relations and four relationships. Each department has many employees and runs many projects. The many to many (M-N) works-in relationship between project and employee entities depicts the fact that each employee in a department might be involved in more than one project in their department only. The 1-N relationship represents dependence relationship between Employee and Child relations. Primary and foreign keys are indexed in all the tables. The mapping of the ERD to tables has resulted to 5 tables, see Table 1. The relationships and number of records inserted in each table are summarized in table 1. The database was created through a java API and filled with random data using the same API.

In converting a relation database into NoSQL model, MongoDB has two ways of modeling relationships between relations/entities, namely, embedding and referencing. In this work, we have transformed the Employee database into three MongoDB-based NoSQL structures. The first structure is based on embedding all the data about employees in one collection called Employee Collection. The department information and children information of the employee are represented as an inner embedded documents. The project details that the employee is involved in are the second level embedded document, see Figure2. In our experiments, the Employee collection contains 2 million employee documents.

//slide 9

The last NoSQL structure represents each relational table into separate collection, see Figure 4. All relationships are implemented using references. The Department collection doesn’t have any reference relationship with any other collection. The employee collection has a reference relationship with department collection. The project collection has a reference relationship with department collection. Child document has a reference relationship with employee collection. The works-on collection has a reference relationship with both employee and project collections.

To compute the performance of each NoSQL structure, we developed a benchmark retrieval queries. In this benchmark queries, we considered different complexity levels of retrieval including indexing, joining tables, and aggregations.

IV. Experiments A. Setup

We have used MySQL server version 5.7.17 MySQL Community Server (GPL) for relational database and MongoDB version v3.4.1. Both databases are running on MacBook Pro (13-inch, Mid 2010) with 2.4 GHz Intel Core 2 Duo processor, 4 GB 1067 MHz DDR3. We use Java to develop the API for managing the communication with the DBs through NetBeans IDE 8.2. The library used to access MySQL server DB for java API is “mysql-connector-java-5.1.40-bin.jar”, and the other one used to access MongoDB is “mongo-java-driver-3.4.1.jar”.

B. Results and Analysis

Query 1. MySQL database achieved better retrieval time compared with all MongoDB structures. Query 1 retrieves all the information about one employee. Retrieval times of MongoDB structure 1 (M1) and structure 2 (M2) are very close while structure 3 (M3) took 28 hours because MongoDB structure 3 uses references to lookup data.

Query 2 (Q2). MongoDB structure 2 (M2) outperformed other structures and MySQL with 1.92 milliseconds. The retrieval time of MongoDB structure 1 (M1) execution time was very close to MongoDB structure 2 (M2) execution time. We observe that a noticeable increase in the execution time of the same query in MySQL structure, as the query will access projects and works\_on tables to find employee id then it will get all the information related to that employee. All of the three previous structures execution time was very short compared with the execution time of MongoDB structure 3 (M3) which reached about 24 hours to get the results. The reason behind the very high execution time in structures 3 (M3) is the reference relationship between the collections and the need to access all the collections after finding the employeeID required.

For Query 3, MySQL database structure recorded the longest time. The time needed to retrieve data was about 65.95 milliseconds because in MySQL the process will join two tables namely, Works\_on table and Project table on an non-key fields. In MongoDB structures M1, M2, and M3, results were close to each other. We can conclude that any representation in MongoDB is better than MySQL representation of data in executing this kind of queries that does not rely on key data.

Overall, results show that using embedded structures along with references would make retrieval times reasonable and can easily outperform relational databases. Additionally, designing NoSQL with only references or embedded documents makes searching/retrieval worse than relational databases

In this paper, we investigated different NoSQL structures to represent relational databases. We used MongoDB for NoSQL and MySQL database as a relational database. Our experiments are based on three NoSQL structures that include embedding documents only, referencing documents only, and both. We used large number of records and a set of five queries as a benchmark to measure retrieval time. Our benchmark database contains five tables and variety of relationships that include many-to-one, and many-to-many. The final finding of this research indicates that using both embedding and referencing of documents is the best way balance the tradeoff between replications and retrieval time.