Comparative Exercise on MongoDB, CouchDB and MySQL databases

Kevin Lloyd H. Bernal, Randall Romeo T. Castro, Dyanara M. Dela Rosa and Janet P. Ordillano

The exercise was to make a web application with a basic CRUD interface for managing different types of fruits.

Each fruit has to contain details like:

1) name
2) quantity
3) distributor
4) price (this changes everyday and history of changes must be saved)

I. Implementation and Approach

A front-end boilerplate was made to have a uniform interface across all databases, as well as to speedup the coding process. A CodeIgniter PHP-framework was used in the Back-end.

Each database implementation has their own dedicated project folder, which are independent from one another.

A. MySQL

Two separate tables were made, namely: 'fruit' and 'fruit-price'. The 'fruit' table has the attributes: name, distributor, quantity and an ID. The fruit.ID refrences itself to the 'fruit-price' table to determine a particular fruit's price history.

B. MongoDB

Same as the MySQL approach, two Collections were made, namely: 'fruits' and 'prices'. The same attributes from MySQL's 'fruit' are in the MongoDB's 'fruits' Collection, with the addition of currentPrices attribute. The Collection 'prices' contain documents that pertain to a particular fruit's price history (same structure as with MySQL's).

C. CouchDB

A database named 'myfruit' was made, with each of its document representing a distinct fruit having the details mentioned above as its data items.

However, the price contains a dictionary which contains the date (represented inthe format: Y-m-d) as the key, while the price for that specific date as its value.

II. Relational vs. NoSQL

Relational Databases takes much more time to design and implement compared with NoSQL databases, such that a lot of overhead times are needed for conceptualizing and planning the database structure, tables and attributes. After the Relational Database has been made, it's relatively harder for data to change, scale up and be up-to-date with newer database changes.

On the other hand, NoSQL databases takes no need for planning their structure and approach in designing and implementation. NoSQL databases can scale up relatively easier as their data increases in size and variety.

III. Execution Time: CouchDB vs. MongoDB

[A single computer was used to record the tests]

For this analytics, the microtime() PHP function was used to record a relatively accurate time up to the millisecond. A total of 10 runs were made for each test and each runs is rounded up to the 4th decimal digit:

(smaller time is better)

- ADD FRUIT [10 runs]
  - MongoDB: 0.01405 seconds
  - CouchDB: 0.03658 seconds
- EDIT FRUIT: [10 runs]
  - MongoDB: 0.00451 seconds
  - CouchDB: 0.02076 seconds
- DELETE FRUIT: [10 runs]
  - MongoDB: 0.00100 seconds
  - CouchDB: 0.01622 seconds
- VIEW ALL FRUITS [10 fruits with 1 price history]
  - MongoDB: 0.001 seconds
  - CouchDB: 0.094 seconds

IV. Big Data: MySQL vs. CouchDB vs. MongoDB

Given the assumption that the data is increasing as well as having rapid updates, MongoDB would suit the job perfectly. Not to mention MongoDB's faster execution time in terms of simple queries.

Being a Non-Relational Database, MongoDB has no problems when scaling up its structure and adding up heterogenous data types.

V. Misc

The price changes for each day is represented by the javascript library called Highcharts.