2015

A Digital Library for Plant Information with Performance Comparison between a Relational Database and a NoSQL Database (RDF Triple Store)

Md Arman Ullah
Grand Valley State University

A Digital Library for Plant Information with Performance Comparison between a Relational Database and a NoSQL Database (RDF Triple Store)

By

Md Arman Ullah

A project submitted in partial fulfillment of the requirements for the degree of

Master of Science in

Computer Information System

at

Grand Valley State University

April, 2015

Dr. Tao Yonglei **Date**

Abstract

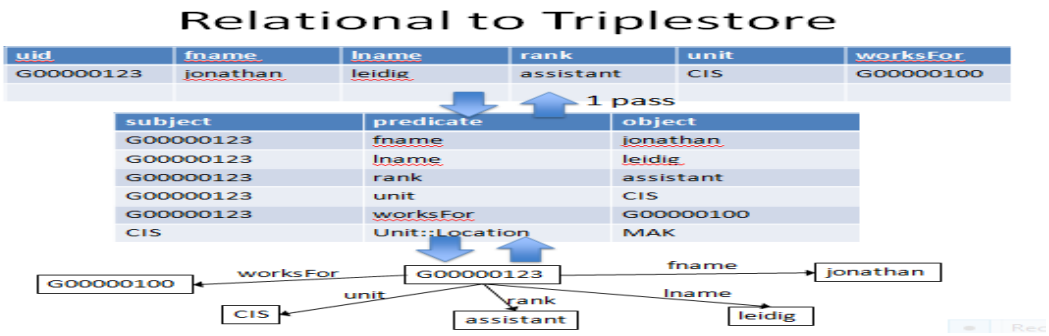
This project is to develop a digital library that allows the user to browse, search, and retrieve information about plants. It uses plant information acquired from the United States Department of Agriculture (USDA), which contains native and naturalized plants in North American territories. In addition, the user is allowed to contribute information and the administrator is able to add or edit plant information. This project is built by using a relational database and also a NoSQL database (RDF Triple Store), allowing to compare performance between the two databases.

1. Introduction

The project is a Plant Digital Library built in the form of windows desktop application to search and retrieve plant information. It also has recommendation feature that would recommends plants to the user based on their preference lists and the user is also able to contribute and add additional plants to the system. It allows the user to measure performance of its functions. I will compare between relational database and NoSQL RDF database based on their performance for this project.

A **triplestore** is a purpose-built database for the storage and retrieval of triples through semantic queries. A triple is a data entity composed of subject-predicate-object. Much like a relational database, one stores information in a triplestore and retrieves it via a query language. Unlike a relational database, a triplestore is optimized for the storage and retrieval of triples. In addition to queries, triples can usually be imported/exported using Resource Description Framework (RDF) and other formats [2]. SPARQL is a query language designed specifically to query RDF databases.

Relational database is one that presents information in tables with rows and columns. A table is referred to as a relation in the sense that it is a collection of objects of the same type (rows). Data in a table can be related according to common keys or concepts, and the ability to retrieve related data from a table is the basis for the term relational database. A Database Management System (DBMS) handles the way data is stored, maintained, and retrieved. The following figure shows the relation between relational database and RDF Triplestore.



2. Technologies used

The application is written using Microsoft .NET Framework, specifically using C# language. I used Microsoft Visual Studio 2012 as my code editor, BrightStarDB NoSQL database as data repository for NoSQL Application [1]. I chose BrightStarDB because it is an open source framework uses No SQL, specifically Resource Description Framework (RDF) Triplestore. I also used My SQL database for SQL Application.

3. Implementation

The application consists of five major functionalities namely Browsing, Searching, Contributing, Recommending, and Preference and one controller service name Admin panel. To implement the relational database Application I used My SQL database. This database consists of following schemas or tables.

Plants(id,name,scientificName,Durations,GrowthHabits,Growthrate,FoliageColor,FruiteColor,FlowerColor,BloomPeriod,MatureHeight,Shape,Toxicity,DroughtTolerance,MinimumPH,MaximumPH,MinimumTempInFarenheit).

Preference_plants(name,Durations,GrowthHabits,FruiteColor,FlowerColor,MinimumPH,MaximumPH,MinimumTempInFarenheit).

Contribute_plants(id,name,scientificName,Durations,GrowthHabits,Growthrate,FoliageColor,FruiteColor,FlowerColor,BloomPeriod,MatureHeight,Shape,Toxicity,DroughtTolerance,MinimumPH,MaximumPH,MinimumTempInFarenheit).

Admin(username, Fullname, Email, Password).

I also used the following constraint to update information in related tables.

ALTER TABLE `database`.`preference_plants` ADD CONSTRAINT `fk_casecade` FOREIGN KEY (`name`) REFERENCES `database`.`plants` (`name`) ON DELETE CASCADE ON UPDATE CASCADE

I used BrightstarDB .NET framework to implement No SQL Application because it is an open source framework uses No SQL, specially RDF Triplestore.

3.1 Browse

The Browse feature allows the user to browse the plants based on predefined categories. The plants are categorized by Durations and Growth Habits. I used tree view list to organize plants in a hierarchy. Each of these nodes has sub nodes of various categories. The "Durations" indicates the lifespan of the plants. Annual plants will cease to exist after a year. Biennial has lifespan of two years and perennial comes back to life every year. The Growth Habits further categorize different plants based on their growing habits such as shrub, tree, forb/herb, graminoid, vine, etc. Figure 1 demonstrates the view for the Browse feature.

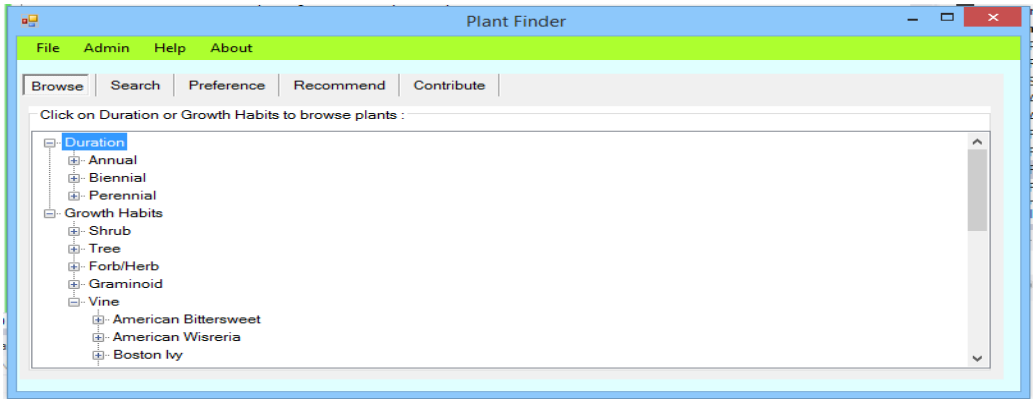


Fig 1: Shows Browse feature

For example user could further expand the Durations category to its sub categories of Biennial, Perennial and Annual and each of these sub categories could further expanded to display all the plants. User also can expand the Growth Habits category to sub category to its sub categories such as Shrub, Tree, Vine, Graminoid, Herb, etc which further could be expanded to all the plants related to the corresponding sub categories.

3.2 Search

This service provides Google-Like Search Engine to retrieve the plants from the system based on the search term entered by the user. When the user enters his search terms, the search will be performed for each of these terms against the data repository using the RDF Graph or using relational database. The resulting outcome will contain the properties of the plants which will contain the search term either in full match or partial match.

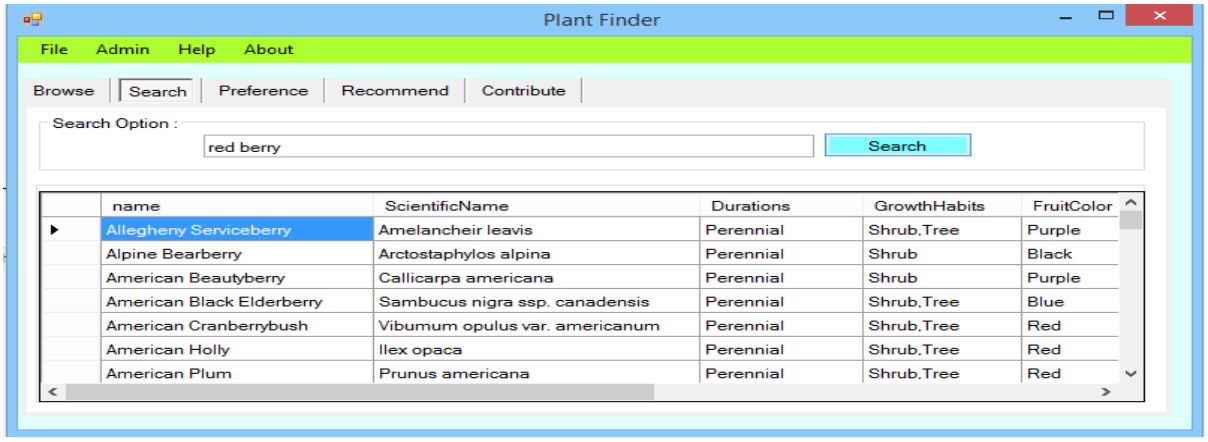


Figure 2 : Search function

Referring to Figure 2 as an example, the user performed a search based on two terms such as “red berry”. The search engine would parse the term into two portions red and berry. Search engine would search for any plant that has a full match or partial match of “red” in any of its properties. This search will be repeated for other term similarly.

3.3 Contribute

Using this service, users can add plants to the system. This service also allows user to maintain the contributed plant(s).User can add plant(s) to contribute plant(s) list by using “Add Plant” button .In addition, it allows the user to delete plant(s) from the list and Administrator can add contributed plant(s) to main plant database by using “Add to Plant” link button.

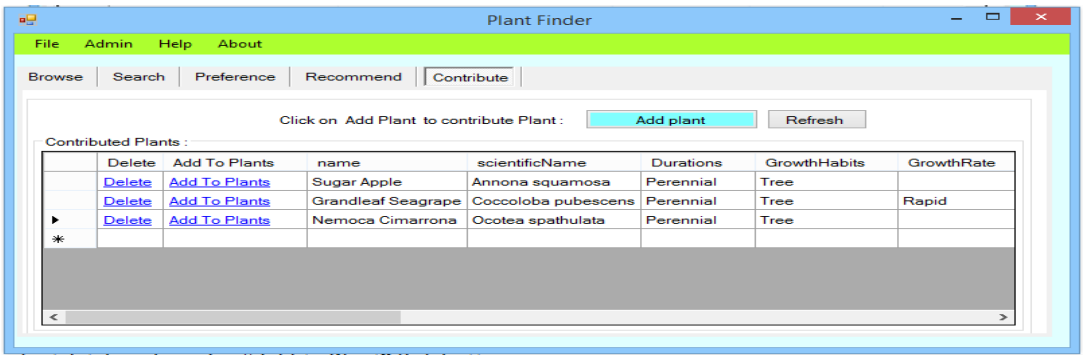


Fig 3 : User contributed plants in the system

As depicted in Figure 3, the user has added three plants to the system. Now if the user desires to edit the plant information that he provided, he can use the edit feature provided by the system on that screen and update the plant information. Alternatively user can also remove the contributed plant using the remove feature on the same screen.

3.4 Preference

Preference is another service in this application which will be used by the user to enter their preferences in the system. The preferences will be used by the recommendation system. The user can add existing plants to his preference list by searching for plants by name against the database. If user click on "Add Plant" button another window will be appear which will allows the user to search plants from plant database and then user can add plant(s) to their preference list. User also has the capability to remove the plants from his preferred list. Likewise he can also modify this preference for the plant properties. Figure 4 shows the preference service.

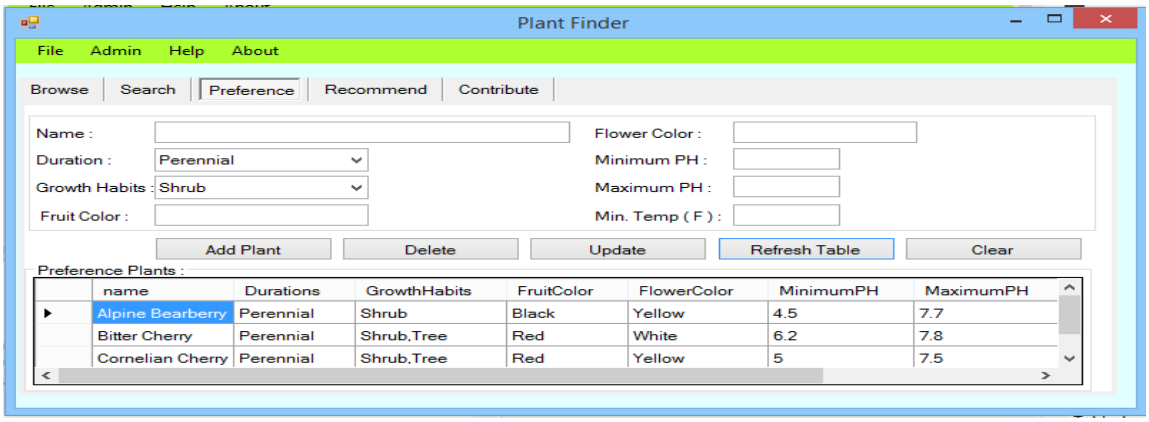


Fig 4: User added plants to preference list.

3.5 Recommend

The recommendation service recommends plants to the users based on his Preferences. Figure 5 demonstrates the recommend service which provided by system. User added 3 plants in his preference list and all plants are perennial and growth habits shrubs or tree or both and flower color is black or red and minimum ph is 4.5 and maximum ph is 7.8 . So all recommended plants has same properties as preference plants have.

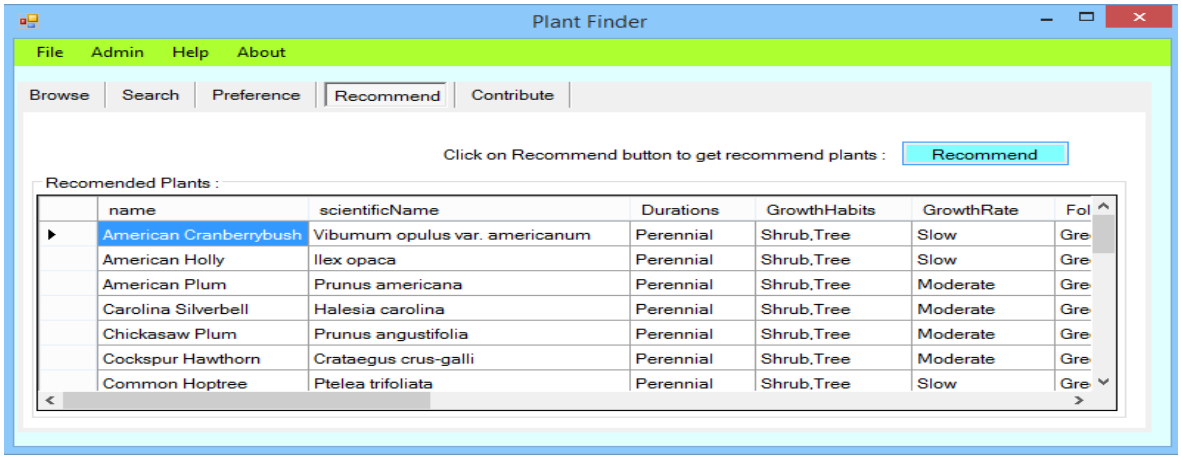


Fig 5: System recommended user based on his preference list.

4. Administration

Administrator can add plant(s) or edit plant(s) from plant database. In addition, Administrator can check for new plants by using “check update” link and also can maintain user account.

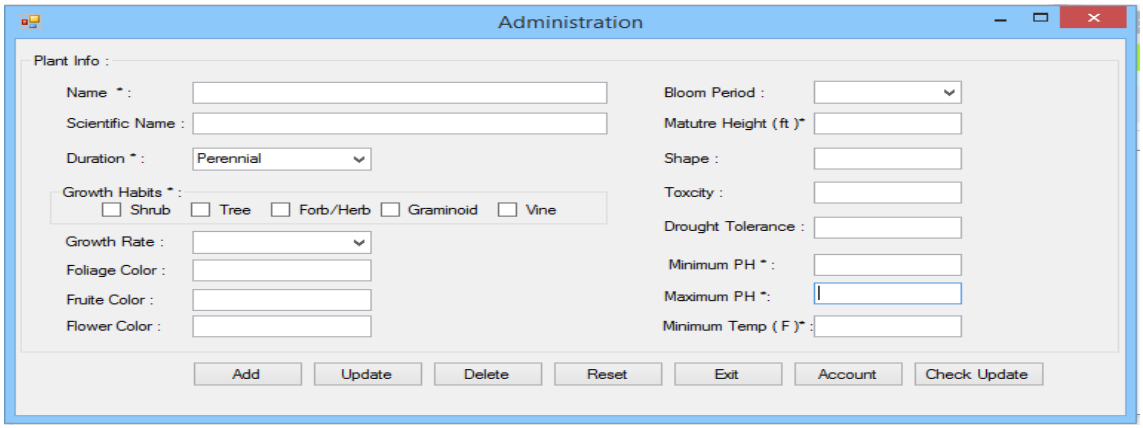


Fig 6: Administration service

5. Performance comparison between MySQL and RDF Triplestrore for this project :

Although NoSQL database has its advantages such as storing unstructured data but it also has disadvantage when it comes to memory usage because RDF Triple store has three fields such as Subject, Object and Predicate, these functions use lot of memory space. It requires a large storage to store thousands of data let alone millions of them which in turn makes the processing of the data slow hence affecting the performance The tabulated results presented here aim to provide sufficient information on how this project perform important tasks such as loading data [3]. The following table shows performance of both databases for this project.

TABLE 1: PERFORMANCE OF RDBMS AND RDF FOR THIS PROJECT.

Functions or services	Time in RDBMS Application	Time in RDF Application
Load Browse Tab	399 ms	9689 ms
Search “red berry”	165.43 ms	7 ms + loading time= 5255 ms
Insert plant in Plant DB	87 ms	82 ms
Delete plant from Plant DB	169 ms	314 ms
Update plant in Plant DB	271 ms	109 ms
Recommend	77 ms	1508 ms

By getting data from Table 1 I can make following decisions for this project.

- Loading time for RDF is more expensive than RDBMS (MySQL).
- For Search plant(s) RDF is cheaper than MySQL but by using RDF requires a large amount of loading time that is why searching in RDF looks very slow. So I prefer MySQL for searching in this project because it contains a lot of plants.
- For insertion both database almost same.
- For deletion RDF is more expensive than MySQL.

- For update RDF is cheaper than MySQL.
- For recommendation MySQL is cheaper than RDF because of RDF having large amount of loading time to display data

6. Usability Evaluation

During the testing phase, some novice users provided feedback about the system.

1. Simple and easy to use.
2. User Control and Freedom
3. Consistency and Standards
4. Error Prevention
5. Recognition Rather Than Recall
6. Flexibility and Efficiency of Use
7. Aesthetic and Minimalist Design

Lesson Learned

- NoSQL database has its advantages such as storing unstructured data but It requires a large storage to store thousands of data.
- Loading data from NoSQL database such as RDF Triplestore is extremely higher than loading data from SQL database.
- Remove data from RDF Triplestore is expensive. To add and update data to RDF database is good.

Conclusion

This system provides a very simple way for user to access the plant information and user can contribute plants to the system and add his preferences to be used for recommendations. This system can be used by any novice user. In spite of RDF Triple store having flexibility and few advantages over MySQL , I prefer MySQL because of its performance, having low loading time and consistency of retrieve data from database.

References

[1] www.brightstardb.com

[2] http://en.wikipedia.org/wiki/Triplestore

[3] https://www.w3.org/wiki/LargeTripleStores