|  |
| --- |
| TRACK Team Project Report |
| Distributed Database Query Engine Service (DDQES) |
| Frederic Colin [2012400565] Henry Loharja [2012280102] Ng Yi Ying [2012280098] |

Table of Content

[Introduction 2](#_Toc358667298)

[Project Scope 2](#_Toc358667299)

[System Environments 3](#_Toc358667300)

[Database System and Engine 4](#_Toc358667301)

[Global Table Structure 5](#_Toc358667302)

[Table Fragmentation 5](#_Toc358667303)

[Database System and Engines Communications 6](#_Toc358667304)

[Local Systems 6](#_Toc358667305)

[Master System 7](#_Toc358667306)

[Query Processing 7](#_Toc358667307)

[Communication 7](#_Toc358667308)

[Recovery Plan 8](#_Toc358667309)

[Master Site Strategy and Dictionary Storage 8](#_Toc358667310)

[Web Services 9](#_Toc358667311)

[Project Timelines 9](#_Toc358667312)

[Current Progression 10](#_Toc358667313)

[Summary 10](#_Toc358667314)

[Appendices 11](#_Toc358667315)

[Appendix I – Meeting Minutes 1 11](#_Toc358667316)

[Appendix II – Meeting Minutes 2 12](#_Toc358667317)

[Appendix III – Meeting Minutes 3 13](#_Toc358667318)

[Appendix IV – Meeting Minutes 4 14](#_Toc358667319)

[Appendix V – Meeting Minutes 5 15](#_Toc358667320)

[Appendix VI – Meeting Minutes 6 16](#_Toc358667321)

[Appendix VII – Meeting Minutes 7 17](#_Toc358667322)

[Appendix VIII – Meeting Minutes 8 18](#_Toc358667323)

# Introduction

Distributed Database Query Engine Service (DDQES) is a project system which is assigned to promote a better and deep insight on the knowledge of distributed database system through a hands-on software design and implementation experiment. In term of scope, the project system is covering a distributed system environment which involved four distributed sites and four database engines. In short, the entire distributed environment is designed to be capable of handling distributed query processing that involved distributed query requests, query analysis for decomposition and localization. With the concern of data communication cost, query optimization is within the considerations of the project scope for effective database retrievals and optimized returned result set. In addition, to ease crossed platforms data communication; DDQES is developed with the functionality of being published as a web service to support remote service call from all available sites for further processing, which is described with Web Service Description Language (WSDL). The entire project is handled by a group of three team members (with TRACK as the team name), namely Frederic Colin, Henry Loharja and Ng Yi Ying as the team leader. Throughout the entire final documenting report, elaborations on the topic of designed system environments, databases systems and engines communication, master site strategy and dictionary storage and the implementation of web services and identified project timelines are covered; to serve as a design and project proposal for review.

# Project Scope

Distributed Database Query Engine Service (DDQES) is designed to handle multiple remote calls from involved sites in the entire system network architecture; through a published web service. It is within the project scope to include four physical sites which involved four distributed databases which consisted of four tables: Publisher, Book, Customer and Order. The final submitting DDQES is expected to provide:

* Structure Query Language (SQL) Support: Capable to handle various SELECT statements through the submission of query request to the published web service
* High Availability (HA) Web Service: Provide a web-based access point for query request submission from various sites through remote call. Mainly to promote a high availability rate, the HA Web Service is scoped to have the capability of immediate handle by another on-line site (based on the service dictionary) to take over the web service providing responsibility if the current handling service site is down.
* Distributed Query Processing Application (DQPA): Upon the HA Web Service is being called by the site, an instance of DQPA which is online and already linked to the web service owner to analyze the submitted query, generate query retrieval plan with query decomposition and finally data location. Once the data is retrieval, merging the result set for return to caller site is also within the project scope. Specifically mentioned on the result of the query, the output will include the display of the following:
  + The size of query result set
  + The optimized query tree
  + The time cost of query
  + The communication cost of query
* Query Optimization: To reduce the data communication cost, various optimizations are expected to be included in the project design and implementations, including:
  + Query tree optimization and result set merging
  + Query tree reduction by fragmentation
  + Web service availability optimization

In term of project deliverables, it is expected to finally complete a reliable Distributed Database Query Engine Service (DDQES) with the mentioned scope and functionality, together with a final report and system operation manual for final submission. In the following section, the system environment for DDQES is documented which included the details of database system and engines, global tables structure and DDQES development and implementation environment.

Development Environment

Specify the development tools and computer system

# System Environments

Under this section, it aimed to document the descriptions of relevant system environment of Distributed Database Query Engine Service (DDQES). Mainly to ease the integration efforts of having multiple developers (three team members) for the project, Java is the identified programming language for building DDQES. Another reason for the decision for the selected programming language is due to its capability to provide web service with Servlet and Java Server Page which fulfilled the project development and implementation needs. Hence, it is reasonable to visualize that all four sites within the DDQES are running the same Java application instance on their physical location for handling all necessary operations. To simulate the real world environment and to own a closer estimations for the communication cost, four physical personal computers will be used to each represents a site rather than using a single virtual computer for demonstration. Regarding the data communication protocol, socket communication is preferred which relies on the personal computer open ports. The overall Distributed Database Query Engine Service (DDQES) architecture paradigm can be viewed as the following:

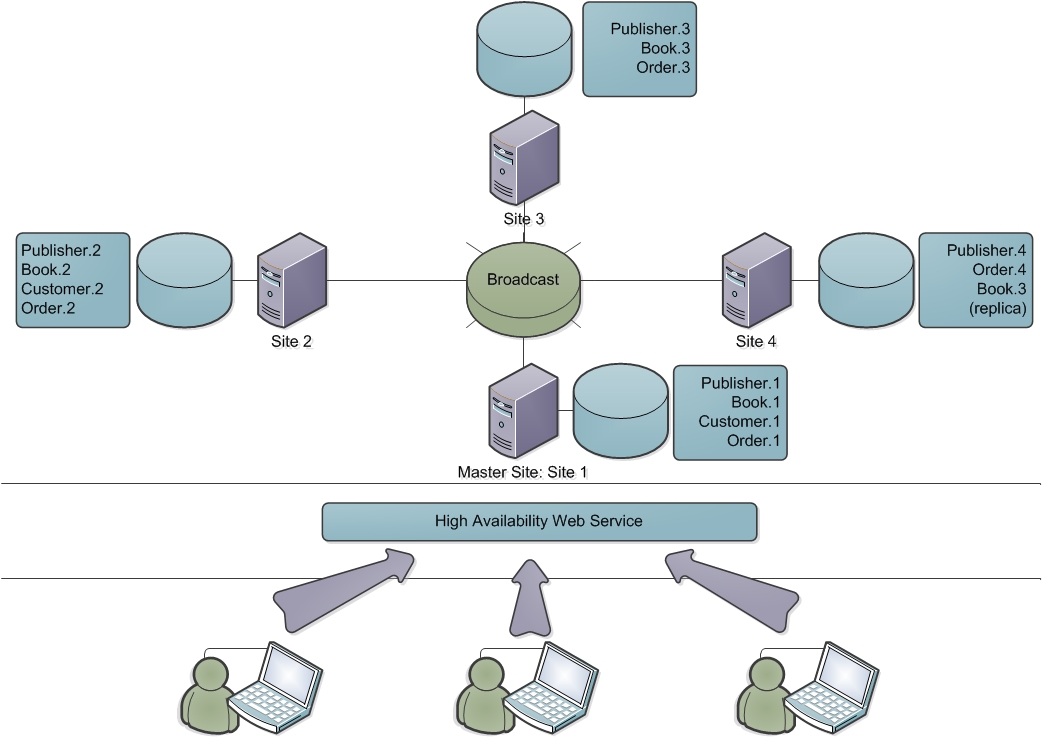


Figure 1 Distributed Database Query Engine Service (DDQES) Architecture Paradigm

Based on the above illustration, it can be further explained as the users who intended to grab information from the databases own no direct knowledge of the physical storage location of the intended data. The users simply submit their query requests to the published web service through an URL. The High Availability (HA) Web Service is responsible of further branching of proceeding operations which will be discussed in the coming section [Section: Web Services].

## Database System and Engine

For times it is relatively hard to standardize all the distributed systems to be equipped the same database systems and/or engines for data storage and handling; the project is designed to handle the communication of multiple heterogeneous database systems, including PostgreSQL, MySQL and SQLite; where each database system can be handled equally by each team member. It is believed that such implementation may higher increase the practicality of the entire project.

## Global Table Structure

In details, each database system in the sites are based on the following table structure as mentioned previously; including Publisher, Book, Customer and Orders table:

|  |
| --- |
| Table Structure |
| Publisher (pid integer PRIMARY KEY, pname char(100), pnation char(3))  Book (bid integer PRIMARY KEY, btitle char(100), bauthors char(200), bpid integer, bcopies integer)  Customer (cid integer PRIMARY KEY, cname char (25), crank integer)  Orders (ocid integer, obid integer, oquantity integer) |

Table 1 Global Table Structure of Publisher, Book, Customer and Orders

## Table Fragmentation

As can be noticed from the Distributed Database Query Engine Service (DDQES) architecture paradigm [Figure 1], the number of tables and ID are different from one site to another. It is in fact drawn based on the fragmentation rule which aim to optimize data communication cost and performance. The fragmentation details can be summarized as below, grouped by table name:

* Publisher

|  |  |  |
| --- | --- | --- |
| Fragmentation | Fragmentation Condition | Site |
| Publisher.1 | id < 104000 AND nation=’PRC’ | Database 1 [Site 1] |
| Publisher.2 | id < 104000 AND nation=’USA’ | Database 2 [Site 2] |
| Publisher.3 | id >= 104000 AND nation=’PRC’ | Database 3 [Site 3] |
| Publisher.4 | id >= 104000 AND nation=’USA’ | Database 4 [Site 4] |

Table 2 Fragmentation of Publisher Table

* Book

|  |  |  |
| --- | --- | --- |
| Fragmentation | Fragmentation Condition | Site |
| Book.1 | id < 205000 | Database 1 [Site 1] |
| Book.2 | id >= 205000 AND id < 210000 | Database 2 [Site 2] |
| Book.3 | id >= 210000 | Database 3, 4 (R) [Site 3, 4] |

Table 3 Fragmentation of Book Table

* Customer

|  |  |  |
| --- | --- | --- |
| Fragmentation | Fragmentation Condition | Site |
| Customer.1 | (id, name) | Database 1 [Site 1] |
| Customer.2 | (id, rank) | Database 2 [Site 2] |

Table 4 Fragmentation of Customer Table

* Orders

|  |  |  |
| --- | --- | --- |
| Fragmentation | Fragmentation Condition | Site |
| Order.1 | customer\_id < 307000 and book\_id < 215000 | Database 1 [Site 1] |
| Order.2 | customer\_id < 307000 and book\_id >= 215000 | Database 2 [Site 2] |
| Order.3 | customer\_id >= 307000 and book\_id < 215000 | Database 3 [Site 3] |
| Order.4 | customer\_id >= 307000 and book\_id >= 215000 | Database 4 [Site 4] |

Table 5 Fragmentation of Orders Table

On top of the basic database configurations to support the implementation of Distributed Database Query Engine Service (DDQES), Hibernate Framework will be used which is an object relational mapping library for mapping an object-oriented domain model to a traditional relational database. In the following section, detailed discussion on the framework will be described.

# Database System and Engines Communications

The database system is composed of 2 different levels that are the local systems and the master system.

## Local Systems

The management of the local database systems is regulated by Hybernate. Hybernate is a Java library which provides an Object-Relational Mapping (ORM). This allows us to define the local tables with an XML file. Then, during the execution, it can generate objects according to the XML description by using a DOM4J session. So that the sub-queries processed locally are handled with a generic code

## Master System

Located at the master site, it correspond to the web-service that will be in charge to split a request, transmit the sub-requests to the local systems, make the unions, joins and semi-joins of the result-sets, and finally return the final result-set to the user.

## Query Processing

Here is an example step by step of the processing of a query:

1. User input its query by using the web-service.
2. The master site generates the query tree.
3. From the query tree, the master site creates 1 sub-query per site involved in the query.
4. Each sub-query is sent to the corresponding site.
5. Each local site processes its own sub-query by using the Hybernate framework.
6. Each local result-set is sent to the master site.
7. By referring to the query tree, the master site realizes unions, joins and semi-joins to get the final result-set.
8. The final result-set is sent to the user through the web-service.

## Communication

The communication between the different sites is done by sending packets over the network.

These packets contain the following information:

1. A fixed hexadecimal code to ensure that the packet is sent by the system.
2. A packet ID
3. A command ID to specify the type of message:
   1. QUERY (0 x 1)
   2. SUB-QUERY (0 x 2)
   3. ANSWER (0 x 3)
   4. RESULT (0 x 4)
   5. PING (0 x 5)
   6. PONG (0 x 6)
4. Query (XML formatted)
5. A timestamp for further optimization such as avoiding request collision.
6. A checksum to make sure the packet has not been corruption during the transfer.

## Recovery Plan

[Need modification]

Each site owns a routine that will restart the local system if the software is crashing. By using the PING of the sites, the distributed system can detect if the master site is done and then give the lead to another site. If a query needs some data from a down site, it will return that the corresponding site is not available. Of course this should happen only in case of a physical crash.

## Master Site Strategy and Dictionary Storage

The master site is the site where most important processes such as the web service for interface, query distribution, and query optimization will take place. This master site will be the central “brain” for the system to operate and so this master site will be very important.

One of the main concerns is when the master site is not available because of several possibility reasons. Since it is very important then it is a must to elaborate some strategies to anticipate the possible conditions. One of the strategies is to actually make every site to have a potential to be master site. In another words, every site can be a master site even though at one time, there will be only one site which will act as the master site. With this strategy, when the current master site is not available, another site will be ready to replace as the master site so it will not disturb the service to the user. A mechanism similar to “ping” will be used to check and make sure that the current appointed master site is working.

The concept here is that it’s like a domain name and hosting service. We have a domain of www.abc.com for example which is an access point for the user to submit query. If the current master site is down (failed to ping it); then another site will take over to host the www.abc.com. This will be the main strategy which our team will implement. There is possibility when we actually working in building the system there will be several changes to this strategy.

Another important issue regarding the master site in the system is the dictionary. The dictionary can be considered as one form of the meta data for all the data which will be used in the system. This dictionary will be an irreplaceable component for the master site to do the processing especially for the query processing. There are many strategies regarding how to store the dictionary. In our implementation, we will store dictionary on all the sites. There are several considerations for implementing this strategy; one of the most important is that by storing it on all sites the dictionary will always be available to access as long as one of the sites is working.

# Web Services

The main purpose of web service will be to “connect” the user with the distributed database system. As the mentioned purpose, this web service will be the interface in which user mainly interact in order to use the distributed database system. From that, it is obvious that the web service is a service which will run on the master site. The web service will be build using JSP/Servlet and HTML technology.

For the interface, it can be separated into several main parts. The first part will be the part where the input will be placed. In this part, user can input the query for the system in which this this input will be processed later.

The second part will be the result part. This part will be the place where the result of the processing of the query. In another words, the query result set retrieved by the system based on the query will be displayed in this part.

The third part will be the process part. This part will be the place where the explanation of the process, mainly the initial query tree and the optimized query tree will be shown. There will probably be another important information regarding the process will be shown in this part.

# Project Implementation

## Web User Interface

Discussion of the web user interface and the responsibility of web user interface

## Web Service

Discussion of the functionality of the web service

## Query Management

Introduction regarding query management

## Query Decomposition

How to obtain user input and perform decomposition

## Query Tree Optimization

How to generate optimization, discussion about the algorithm and the creation of optimized query tree; and how to pass the data to next step?

## Hibernate Data Retrieval and Communication

After obtaining optimized query tree, how does it work to grab your intended data and returned back to caller

## Data Communication and Format

The using protocol and the data format? Such as XML?

# Project Timelines

Since the project was initiated, the team was having a weekly meeting to plan and follow up the project progression [Appendix I - III]. Good project management starts with proper scheduling of a project which may increase the project success rate. As indicated the deadline submission of the project is mid of June, below is the created milestones for the team proper progression and monitoring after the last group discussion. While the person responsible simply means the person who owns a higher knowledge towards one particular subject, the entire team will be paying full attention and coordination to contribute to the project success:

|  |  |  |
| --- | --- | --- |
| Milestone Title | Date | Person Responsible |
| Mid-Point Project Review  Finalize Launcher and Hibernate Connectivity | 29th April, 2013 | Ying  Fred |
| Perform Query and Data Communication Cost  Finalize Optimization Strategy Plan | 6th May, 2013 | Fred, Henry, Ying  Ying |
| Database Building and Hibernate Linkage  Final Documentation Template Creation | 13th May, 2013 | Fred  Ying |
| Individual Application Development (Java) | 20th May, 2013 | Henry |
| Application and Connection Integration | 27th May, 2013 | Fred, Henry, Ying |
| Application Wrap and Documentation | 3rd June, 2013 | Fred, Henry, Ying |
| Final Review and Presentation Wrap Up | 10th June, 2013 | Henry |

Table 6 Table of Project Timelines

# Workload Matrix

The workload matrix

# Project Challenges

What are the encountered challenges and the solutions

Future Enhancement

The future enhancement which can be done

# Conclusion

Though TRACK team had decided to adopt the default scenario for implementing Distributed Database System Query Engine Service (DDQES), the following are the strengths and/or challenging points which worth a note:

* Four physical computers: The team is increasing the project difficulty level by pushing the DDQES environment to be run from four different physical computers rather than running on a single computer with four virtual sites.
* Heterogeneous database engines: Instead of putting all four databases into a single computer which most likely to be the same database system, the team is challenging the scope by using heterogeneous database system; including MySQL, PostgreSQL and SQLite by using Hibernate framework library.
* High Availability (HA) Web Service: To reduce the limitation of the nature of distributed database system, the publishing web service is aimed to provide a high availability rates which is capable of service recovery to reduce the possibility of having an offline service.

# Appendices

## Appendix I – Meeting Minutes 1

Date : 11th March, 2013

Time : 10:00 am – 12:30 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Identify team leader
* Identify as to create new scenario or use the default one
* Identify development environment
  + Number of sites
    - Using same application or different sites uses different application environment, such as site A for finance and site B for warehousing
  + Programming language
  + Data communication and transmission medium
  + Database management system
* Gather all team members’ student ID
* Desirable project task distributions and responsibilities
* Identify schedule for proper progression tracking of the project
* Next step to proceed after discussion

**Meeting Minutes**:

* The student ID of the team members were noted to be filled up to the discussion board in the website and Yi Ying was selected as the team leader.
* The team decided to use back the default scenario and add features to the existing one after the basic requirements are met.
* The following are the discussion results regarding the development environment
  + Number of sites: The team decided to stick back to the default 4 sites and expand it if there’s a time.
  + Programming language: The team decided to start with the same programming language first and JAVA is preferred.
  + Data communication and transmission medium: It is preferable to use socket and TCP/IP or UDP which able to expose the data through web services.
  + Database Management System: The team had shortlisted Hybernate and MySQL.
* The team decided to work together for all tasks and will distribute other unknown tasks in future for better project management.
* Every Monday will be the weekly meeting for the team and during the meeting, the team members will review the progression since last week, discuss on the tasks for this week and how additional working hours are required for this week.
* Fred suggested storing the scenario condition into an XML, further creating an application to generate and launch the sites by referencing the XML “dictionary”.

**To-do List**:

* Submit group to discussion board in <http://learn.tsinghua.edu.cn>
* The student ID for team members are as follow:
  + Frederic Colin – 2012400565
  + Henry Loharja - 2012280102
  + Ng Yi Ying – 2012280098 [Team Leader]
* Register for GitHub and download and install Git
* Create a scenario for the assignment by specifying the limitation and general picture of the entire project
* Create a prototype launcher based on the XML
* (Optional) Draw the relationship among sites and possible working/communicating logic

## Appendix II – Meeting Minutes 2

Date : 18th March, 2013

Time : 10:00 am – 12:30 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Brief discussion on last week lecturing materials
* Review on the draft of the created first scenario
  + Architecture and drew diagram
  + XML for environment variables
  + Scalability
  + Potential modifications
* Review on the created launcher prototype
* Shortlisting database systems to further prototyping
* Distribution of tasks
* Scheduling and milestones for better project monitoring
* Next step to proceed after discussion

**Meeting Minutes**:

* A brief review of last week lecturing was carried out by the team.
* The initial draft of the case scenario was finalized to be just fine for the starting and will definitely enhance the model when the project goes on. The potential solution is to make each and every site a master site of its own so that when one of the sites was down, the other may still progress while the other will perform the recovery.
* Fred and Henry were agreed to check out on Hybernate and its connectivity to first launcher prototyping while Yi Ying will be working on MySQL to further explore the database connectivity.
* After checking the schedule for mid-term presentation, it was agreed by the team that there will be sufficient time to prepare and work on this project to achieve desire goals. For now, weekly meeting and report are agreed to be kept going as more prototyping can increase the completeness of the project.

**To-do List**:

* Fred and Henry: Try out launcher prototype and test out the connectivity between the application and Hybernate.
* Yi Ying: Try out the connectivity between the application and MySQL and try to expand the launcher prototyping.

## Appendix III – Meeting Minutes 3

Date : 25th March, 2013

Time : 10:00 am – 12:30 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Brief discussion on last week lecturing materials
* Review on the draft and progression of the created second scenario
  + Connection from Launcher to MySQL
  + Connection from Launcher to Hibernate
* Discussion on the concept of web services (Since one of the requirements expects the distributed database query engine is capsuled into a web service) and WSDL
* Review on the required content of mid-term design report and presentation
  + Design of the distributed database query engine
  + Timeline for project work plan
* Next step to proceed after discussion

**Special Notes:**

* Mid-term presentation will be on 26 April, 2013

**Meeting Minutes**:

* A short discussion on the homework assignment was carried by linking to the project needs in the future
* The use and publish of service to the web may need further verification with the teaching assistant before proceeding
* It is agreed to ready the launcher before next coming meeting for evaluation and preparation for later mid-point presentation

**To-do List**:

* Meet up the teaching assistant for concept verification
* Build up the launcher connectivity to MySQL, Hibernate and Postgre

## Appendix IV – Meeting Minutes 4

Date : 19th April, 2013

Time : 15:00 pm – 18:00 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Review on the draft and progression of the created second scenario
  + Connection from Launcher to MySQL
  + Connection from Launcher to Hibernate
* Review on the required content of mid-term design report and presentation
  + Design of the distributed database query engine
  + Timeline for project work plan
* Preparation of midterm presentation and report writing section distribution
* Next step to proceed after discussion

**Meeting Minutes**:

* A short discussion on last class of distributed database system.
* The current progression of the Launcher was reviewed by specifying existing limitation of not yet including Hibernate as part of the prototype.
* However, the connection from Launcher to database system is relatively stable for now and should plot over to include Hibernate right after midterm presentation.
* The discussion of the entire scenario was carried out to double confirm the discussed details in the past month.
* The content of the midterm report was discussed and section writing was distributed and agreed to complete before Thursday for integration.
* The demonstration of Launcher was agreed not to push up to the midterm presentation since it is not quite complete yet.
* Presentation slide will be prepared by Yi Ying and content will be added by Fred and Henry.
* Another meeting was expected to be carried out before the midterm presentation for rehearsal of having better presentation performance.

**To-do List**:

* Write on midterm report
* Create midterm presentation slides and content

## Appendix V – Meeting Minutes 5

Date : 29th April, 2013

Time : 10:00 am – 12:30 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Brief discussion on last week presentation lesson and feedbacks
* Review on the draft and progression of the created second scenario, finalizing:
  + Connection from Launcher to MySQL, MySQLite and PostgreSQL through Hibernate
* Discussion on the immediate step in implementing next steps as of prioritizing which tasks first:
  + Stable programming solution
  + Cost estimation calculation
* Task distribution for the following tasks
* Next step to proceed after discussion

**Meeting Minutes**:

* The team agreed to finalize and come out with a working solution in next week meeting for all the mentioned three database connections:
  + MySQL
  + SQLite
  + PostgreSQL
* It was agreed to work on the programming part first as the stability of a working solution should be in place and optimization plan can come in later throughout the process
* Fred was assigned to handle connectivity to MySQL, Henry assigned to PostgreSQL connectivity while Yi Ying was assigned to handle connection to SQLite.
* To test out the latest Hibernate Framework, version 4.1.12 was suggested and accepted to be used for the building.

**To-do List**:

* Finalize and develop final version of Launcher to Hibernate and to corresponding database system

## Appendix VI – Meeting Minutes 6

Date : 8th May, 2013

Time : 10:00 am – 12:30 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Brief discussion on last week assigned work progression
* Review on the progression of the created second scenario, finalizing:
  + Connection from Launcher to MySQL, MySQLite and PostgreSQL through Hibernate
* Discussion on the next application development plan and achievements
* Task distribution for the following tasks
* Next step to proceed after discussion

**Meeting Minutes**:

* There were issues encountered while finalizing the Hibernate connection from the Launcher application upon adopting the latest Hibernate library files:
  + SQLite connection was not well-established from Hibernate
    - Solved by moving the configuration file to source folder
  + PostgreSQL connection was not well-established yet from Hibernate due to lack of tutorial resources
  + Issues of upgrading as the preferred feature in version 3+ Hibernate Framework was wiped off from the latest Hibernate Framework
* It was agreed to work to downgrade the Hibernate Framework down back to the latest version of 3+ due to the shortcoming of latest Hibernate Framework. The downgrading version is 3.6.10 which can be downloaded from

<http://sourceforge.net/projects/hibernate/files/hibernate3/3.6.10.Final/>

* To quickly pull the project on-track, Fred was assigned to work on exploration of the intended features which will add value to the entire application project, tested and conduct knowledge transfer.
* Henry was assigned to handle the connection between Hibernate and PostgreSQL with Hibernate Framework 3.6.10.
* Yi Ying was assigned to handle the structure of proper programming file management format and the online socket connection between computers.

**To-do List**:

* Downgrade Hibernate Framework back to 3.6.10 which supporting XML mapping and format identification.
* Connection between Hibernate and PostgrSQL
* File management structure and programming naming convention
* Socket connection between two computers

## Appendix VII – Meeting Minutes 7

Date : 20th May, 2013

Time : 10:00 am – 12:30 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Review on the progression of the created second scenario, finalizing:
  + Connection from Launcher to MySQL, MySQLite and PostgreSQL through Hibernate
* Discussion on the current progression and achievement since last meeting
* Finalizations on the modification of version of Hibernate
* Task distribution for the tasks in development
* Next step to proceed after discussion

**Meeting Minutes**:

* The demo of the socket network communication was brought up for team sharing knowledge; the use of networking class was presented by Yi Ying
* It was finalized by the group where it’s no longer necessary to downgrade the version of Hibernate as it is already possible to achieve the intended functionality.
* Connection to Hibernate and PostgreSQL was successful; same goes to MySQL and SQLite.
* Discussion on the remaining task and schedule was carried out.
* The tasks were split among team members and each of the member will be handling each part as follow:
  + Yi Ying: Interface, web services and query decomposition
  + Henry: Based on the decomposed query, generate optimized query tree
  + Fred: Capture optimized query from Henry’s part and perform data retrievals within site and send it back to calling site
* Discussion in term of overall working plan of the entire application, query decomposition, calculating and identifying optimized query were carried out.
* It was agreed by the team members as all the works will be at least drafted with confidence before coming next meeting.

**To-do List**:

* Core application development
  + Web user interface, web services, query decomposition, query tree optimization, data communication and retrievals
* Final report structure identification
* Final presentation slide identification

## Appendix VIII – Meeting Minutes 8

Date : 20th May, 2013

Time : 10:00 am – 12:30 pm

Venue : 8-304

Attendees : Frederic Colin, Henry Loharja, Ng Yi Ying

**Intended Discussions**:

* Review on the progression of the created second scenario, finalizing:
  + Connection from Launcher to MySQL, MySQLite and PostgreSQL through Hibernate
* Discussion on the current progression and achievement since last meeting
* Finalizations on the modification of version of Hibernate
* Task distribution for the tasks in development
* Next step to proceed after discussion

**Meeting Minutes**:

* The demo of the socket network communication was brought up for team sharing knowledge; the use of networking class was presented by Yi Ying
* It was finalized by the group where it’s no longer necessary to downgrade the version of Hibernate as it is already possible to achieve the intended functionality.
* Connection to Hibernate and PostgreSQL was successful; same goes to MySQL and SQLite.
* Discussion on the remaining task and schedule was carried out.
* The tasks were split among team members and each of the member will be handling each part as follow:
  + Yi Ying: Interface, web services and query decomposition
  + Henry: Based on the decomposed query, generate optimized query tree
  + Fred: Capture optimized query from Henry’s part and perform data retrievals within site and send it back to calling site
* Discussion in term of overall working plan of the entire application, query decomposition, calculating and identifying optimized query were carried out.
* It was agreed by the team members as all the works will be at least drafted with confidence before coming next meeting.

**To-do List**:

* Core application development
  + Web user interface, web services, query decomposition, query tree optimization, data communication and retrievals
* Final report structure identification
* Final presentation slide identification