CSE681 Software Modelling and Analysis

Key/Value Database OCD

Project#1

9/8/2015
Qinyun Li
Qli56@syr.edu

catalog

Project #1- Key/Value Database OCD

1 EXECUTIVE SUMMARY

1.1 Brief Introduction

Key/Value database, also called NoSQL, is an approach to data management and database design that is useful for very large sets of distributed data but support dynamic schema design, offering the potential for increased flexibility, scalability and customization compared to relational software .Key/Value is especially useful when customer needs to access and analyze massive amounts of unstructured data.

1.2 About the Users

The Key/Value database of this project intended users are Developers, Teaching Assistants, Instructor and project 4 and project5.

1.3 Structure

The Key/Value database consists of several modules which are responsible for different job in the flow of the program. There are TestExec Module, Item Factory, Item Editor, Query, DB Engine, DB Factory, Scheduler, Sharder, Persist Engine& Display Module. TestExec Module is the main function of the project, it call other modules to realize the function. First, it constructive the database by the Item factory. It uses those pairs to populate its NoSQLdb instance through an API provided by the DBEngine package Then the TestExec will send Query Engine to search for specified data set. Then the TestExec can call DBEngine to do add or delete the data. If the memory face the its limitation, the DBEngine will call Shard package to shard data into small pieces At last, the data will persist into XML file into the disk. . Additionally, the Schedule Module can also persists the data after 10 seconds for capacity of memory.

1.4 Critical Issues

There are five critical issues and solutions are considered in the project which are list below.

- How will the tool to ensure the data reliable in the whole operation?
- How will the tool to ensure the data security?
- How will the function shard do?
- How can the tool retrieve the data from disk or memory efficiently?
- How will the tool to handle the eventual consistency?

2 INTRODUCTION

As Conventional SQL databases are not well suited for these kinds of applications, so we turn to a non-relational databases-NOSQL database. It avoid selected relational functionality such as fixed table schemas and join operations. For example, instead of using tables, a NoSQL database might organize data into objects, key/value pairs. It can query, addition, deletion, update by the coding written to it. It persist the data in XML file in disk.

2.1. Objective and key idea

The objective and the key idea is to realize how a non SQL database can be constructed which is query, addition, deletion and shard in the project. The tool can Support a variety of queries, both simple and compound, like ‘SELECT’, ‘JOIN’ in SQLdatabase. The tool also can support addition to database and deletion from dadabase. After update the data, it need persist data in XML file. If memory is face limitation, the tool can shard the data to reduce the capacity in memory.

2.2. Obligations

- Implement a generic key/value in-memory database
- Edit the value
- Implement addition and deletion of key/value pairs
- Persist data in XML file
- Query the specified data or data set
- Implement through a series of discrete tests with display to the console

2.3. Organizing principles

The organizing principles to achieve the main goal of the tool using DB Engine module and separate other functionality into isolated modules which communicate with the TestExec module.

3 USES

3.1 A list of general users

The users can interact with this application and obtain valuable information. There are a variety of users interact with the Key/Value Database. Following is a list for some general users:

- Developer

The developer is a member of this software or application group. A developer designs software systems and modify software using mathematical models and scientific analysis to measure and predict the outcome or consequence of the design.

- Quality Assurance Team

This team goal is preventing mistakes or defects in manufactured products and avoiding problems when delivering solutions or services to customers.

- Software Architect

The person is the leader of the whole team who responsible for the customers and developers. The Software Architect need to make sure the development team is able to work as efficiently as possible to get a product out the door. Obviously, an architect is usually someone who has the most knowledge, skills, experience, and authority. A software architect is to personally responsible for the technical quality of the product.

- Customer

The potential buyer of the software or application after it is developed and tested.

3.2 Description of uses by different actors in this tool

- Developer-students

  The developer in this tool is our students. This time it is individual work instead of a team. Firstly, we design NOSQL data base according to the instructor requirements, and modify software using mathematical models. Then, test the tool to consider whether it is sufficiently usable, and also ensure the design can be installed and run in its intended environments to achieve the general result its desire. If there are something wrong in the coding, we should report the problem, fix the defects and improve performance or other attributes. Finally to achieve the general result its desire.

- Quality Assurance Team- Teaching Assistants
  The person is the leader of the students who responsible for the quality of the project. He/she can check your project code to consider whether it is achieve the goal of this project. He/she also can assist students while they are working on their project, for example, give some hints to the students in order to guide them to the right direction, or directly answer the students' questions.

- Manager--Instructor
  The instructor is like Software Architect, who has the most knowledge, skills, experience, and authority. A software architect is to personally responsible for the technical quality of the product. He /she write the requirements for the students to do in their project, he/she constructive the whole ideas of the tool, he/she gives the interface of the functions. He /she taught the basic knowledge of this project, and make the students to open their minds to this knowledge and learn more about the project.

- Customer—project4&5
  For project4 and project 5, this project is a base which they will build on this to develop others functions.

4Partitions

4.1Module Partitioning

Base on the big key/value database activities, each task becomes a package candidate for code analyzing process, the following are module parts of this tool.

(1) Executive Module
(2) Item Factory Module
(3) Item Editor Module
(4) Query Engine Module
(5) DB Engine Module
(6) DB Factory Module
(7) Scheduler Module
(8) Sharder
(9) Persist Engine
(10)Display
(11)Back up

Among all the modules candidate, (4), (5), (6) and (7) is the main function of the tool which support a variety of queries, addition and deletion for this data base. The responsibility of each module will be described in detail below.

4.1.1Executive Module

The Executive module is like a controller, it control all the task to perform from the calling other modules, it is like the 'MAIN Function' in the program. However there are no detailed and logical code in this executive module. For here, the package name is TestExec which is directly in charge for the calling from function query, function addition, function deletion, the entry of database, the schedule of update data and also the display to complete the whole task for this tool.

4.1.2 Item Factory Module

The Item Factory is the entry for database, the TestExec module will send item to this module. It store pair of key and value data. To be specific, it implement a generic key /value in memory database where each value consists of a name string ,a text description of the item, a time-data string recording the date and time, and a relationship with other values. Additional, when it come to the type of the data. This might be a set of value of the same type, maybe a set of different data and so on. In short, in this module it implement various pair of key and value and holding instance of some generic type, if it create successful, it will sent back to TestExec to database.

4.1.3 Item Editor Module

There is an Item Factory which to create the item data, when it comes to manage and edit the item data, the Item Editor module achieve this goal. For here, the TestExec module will send item need to modify and also to send back to TextExec to add to database .'Item Editor', it is a module which top realize the capability to edit the values of a key. For instance, changing the text description of the item, changing the name string, changing the date time which written to the database and also it is able to changing the relationships of others. In summary, the Item Editor module is to achieve the goal for edit the pair of key and value data.

4.1.4Query Engine Module

For the entire tool, query is one of the main functions of database, so the Query Engine module is the one of significant role to look up the data form a big container. To be specific, the TestExec module will query for the specified set of data, and it will use DBEngine to search the data. If it is found,it will call DBfactory to create a new place to put the data in. Queries are retrieving information from a database and consist of questions presented to the database For instance, it support to retrieve the value of a specified key which narrow down other dataset, it support to retrieve the children value of a specified key. These two function is like 'SELECT' in SQL. It also support retrieve the data which they have some parts are same, specifically, it can retrieve all keys that contain the same value for date time or a specified string or pattern. These function is like the 'JOIN' in SQL. In brief, it will support look up whatever data you want base on the deep logical coding to do.

4.1.5 DB Engine Module

This package is the entire application program interface (API) for the noSQL database, The API specifies how software components should interact and APIs are used when programming. For here, following space include functions and interface.

- Dictionary
  Pair of Key/Values of datsbase store
- Addition
  Insert pair of key/values to database
- Deletion
  Delete pair of key/values from database

Literally, the Addition/Deletion is to realize the function addition and deletion of key/value pairs. It is easy to understand to add or delete data in the big database, but not like the module called 'Item Editor', it will add/delete the pair of key and value, the 'Item Editor' can only manage and edit the value of specified key, and editing only keys are forbidden. In a netshell, this module is to achieve the goal of adding or deleting pairs of data, both key and values.

- Schedule
  Accept a positive time interval or number of writes after which the database contents are persisted
- Sharding
  Breaking your database down into smaller chunks

- Query

To search the specified key for the values

This module provide application program interface which can populate its noSQLdb instance, indicate the intersection connect.

4.1.6DB Factory Module

DB Factory module is a instance of DB Engine. It uses those pairs to populate its noSQLdb instance through an API provided by the DBEngine package. It like a factory to generate database. For instance, there is a XML file to this DB factory, it will generate a database base on that XML file. The other example is which according to the customer require for retrieve a set of data, in DBFactory will generate those data.

4.1.7Scheduler Module

A schedule is a listing of activities and events organized by time, For here, the TestExec will call Scheduler to set the time . Scheduler module is to manage the database by time. There are some points to make a schedule such as:

- Sharding data schedule, persist in memory
  This is the main schedule use for this tool, becausing of the enormous growth in databases may cause some failure to the tool. It is essential to shard the data in some small pieces.

4.1.8 Sharding

Sharding is one of critical problem to the tool guarantee good running, specifically, database servers have a limit on how many connections they can accept and how much data they can process. When a memory reaches the limit of its capacity, it will appear to slow down .In extreme cases, the server can completely collapse under the strain .So this time it is crucial to sharding the data into small parts. For here, the DBEngine first to search the data in memory , then the XML file.Then call the Shard package to shard the data, finally to use the Persist Engine to save the data. It will also decide on the Schedule package, it sets a schedule to decide when to sharding the data, such as wirte 10 times to perist one time. Database sharding provides many advantages, such as faster reads and writes to the database and improved search response.

4.1.9 Persist Engine

The Persist Engine is to save the data to XML file. The Query Engine will be call to persist the data into XML file. It will call Persist Engine to do this.

4.1.10 Display

This is the last step which to show the result for the function. The output can be displayed printed on the standard output console. For here, for instance, the TestExec will call Display package to show the results on console.

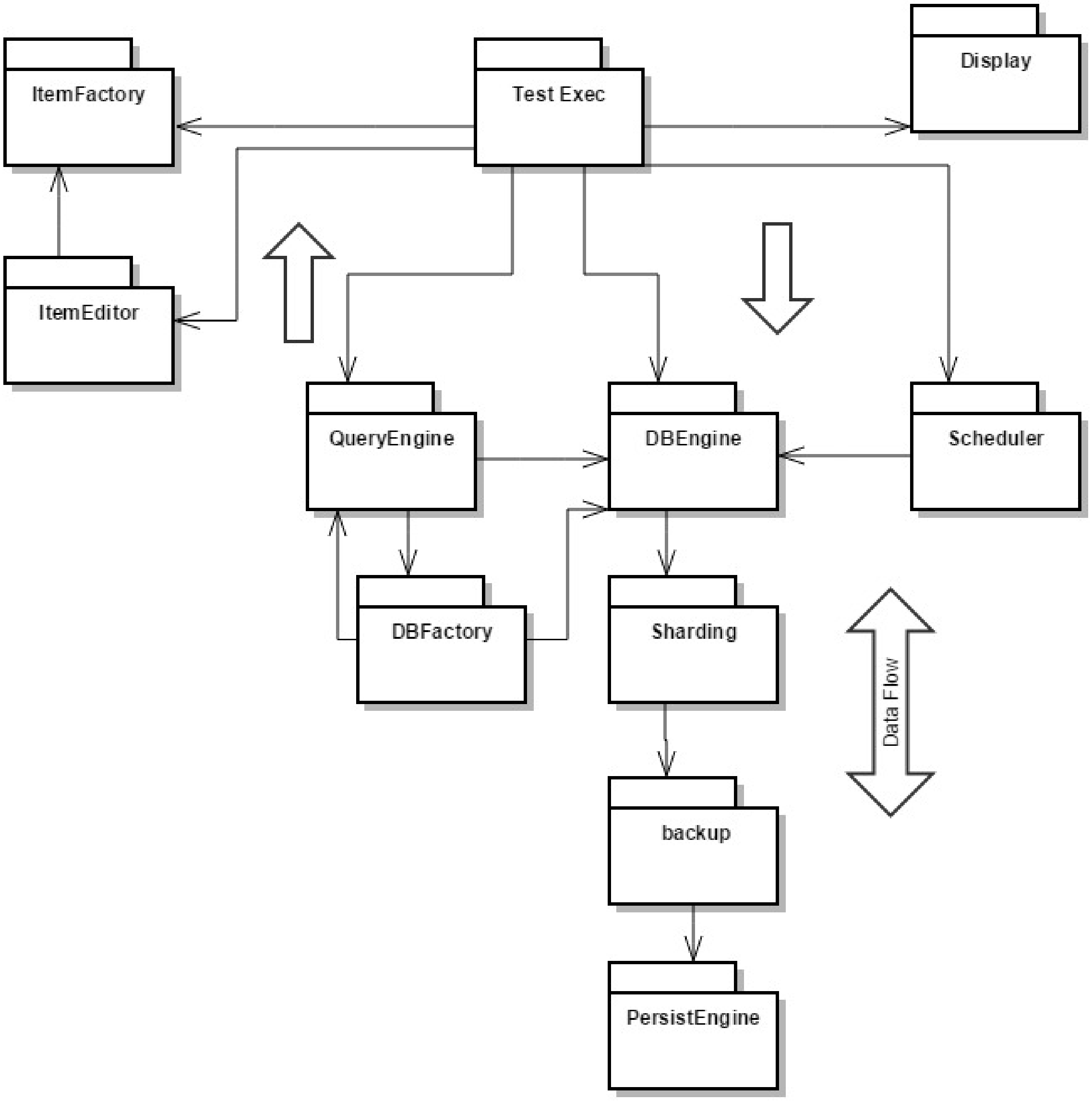
4.1.11 Backup

A NoSQL database should be highly resistant to corruption. If a data is corrupted, we should have some backup of the database. Servers must themselves have copies of the fleets of database shards.

4.2 Package Diagram

The Code Analyzer package diagram depicts the modules and their interaction for the functionality comprising the program.

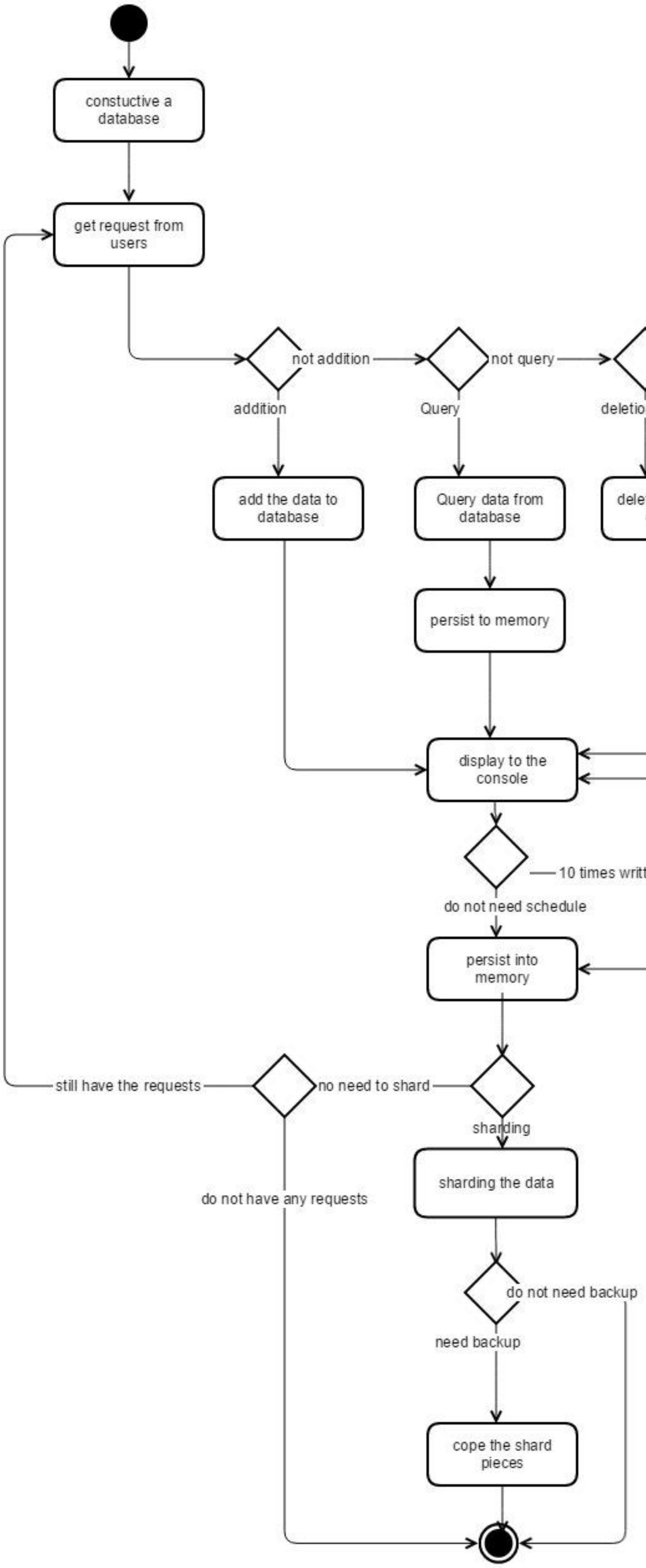
Project #1- Key/Value Database OCD



From the package diagram, it is obvious that all other modules are created and invoked via the TestExec, either directly or indirectly.

5 Application Activities

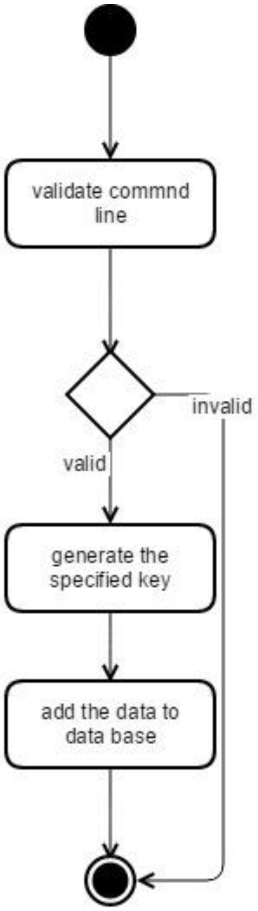
5.1The high leve diagram of whole function



5.2The description of the high-level diagram

- Constructive a database
  It is the first things to do, to create key and key corresponding values described by metadata and holding an instance of some generic type.
- Get the requests form users
  To get the function require from the user, which function need to do
- Addition deletion query edition
  Then go to the realize the function which need to do
- Display on the console
  All the function need to show on the console
- Scheduler
  Decide is it the time to persist in memory, my judgement is consider is it 10 times written
- Persist in memory
  Always to persist the update in memory
- Shard
  Consider if the memory is full to shard the data in piece

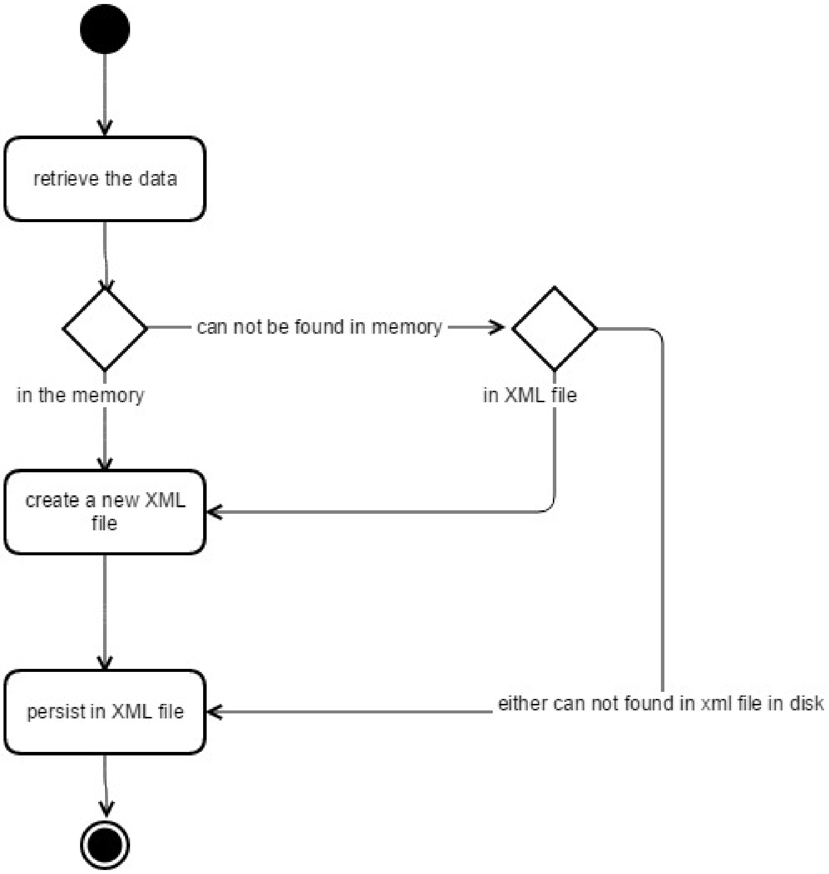
5.3 The addition activity diagram



5.4 The description of addition diagram

- Validate command line
  To type the correct information, if it is invalid, it won't work.
- Generate the key
  It is a new data, it only have the values, we need to generate a key for the values, make the data are pair of key and values
- Add to database
  This is time to add to database

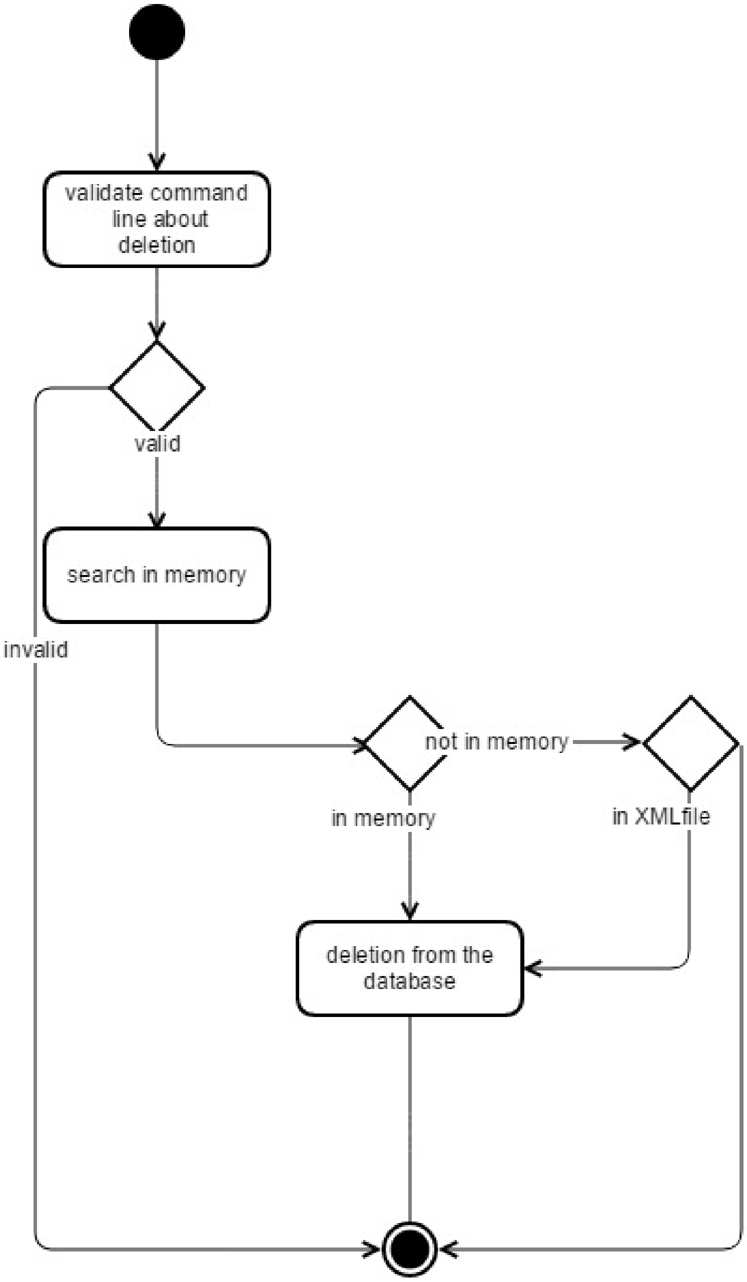
5.5 The query activity diagram



5.6 The description of query diagram

- Retrieve the data
  First depend on the data you need to query to retrieve the data
- In the memory
  First search is in the memory, if not search for XML file
- Create the XML file
  If it is found, next is to create a new XML file to put the new data set
- Persist XML file
  Always persist the updated data

5.7 The deletion activity diagram


validate command line about deletion
valid
invalid
search in memory
not in memory
in memory
in XMLfile
deletion from the database

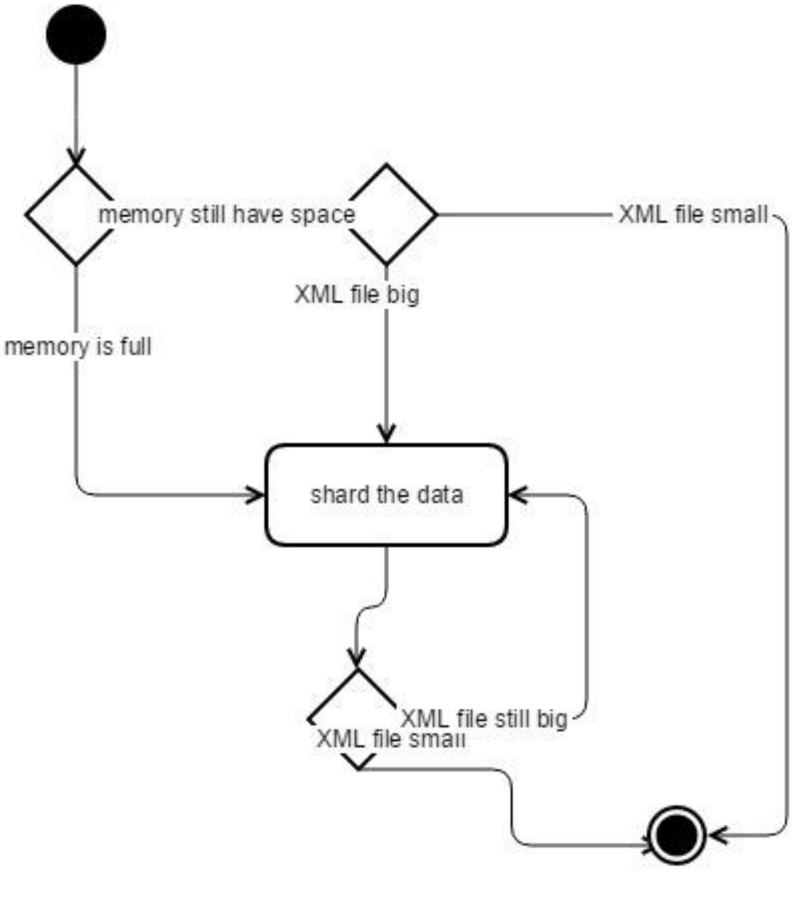
5.8 The description of deletion diagram

First type the valid command then search in the memory, if not exist, then search
In the disk. Then find the corret XML file, then delete the key and value from that XML file.

- Validate command line
  To type the correct the information, if it is invalid, it won’t work.

- In the memory
  First search is in the memory, if not search for XML file
- Delete from the database

5.9 The sharding activity diagram



5.10 The description of sharding diagram

First we check the memory, judge it is full or not. If not full, we can search in XML file. Judging the XML file is big or not. If the XML file is big , it need to shard in small piece.

6 Critical issues

These are the issues which are critical to the design of the key/value database. Some of the critical issues have been discussed below.

6.1Sharding

When the tool memory reaches the limit of its capacity, it will appear to slow down or become unresponsive to any applications that are using it. So this time it need shard the data in small pieces.

- **Problem:** Due to the simple problem with database is that tables are so big – often with billions of rows, this will cause the tool slow down or even break down. How do the shard do to deal with that?
- **Solution**: Sharding, also known as horizontal partitioning, reduces the burden on individual database servers by spreading the rows of tables across multiple machines. Each server contains the table structure, but only a small subset of the total data that is contained in it. And the database is only dealing with part of the data, they do not suffer from indices that are too large to be useful.

  For here, it will offer a function called sharding in the coding, which decide on which server it should be stored. For example, we have a database about what we buy from the store in two days, a sharding function might look at a key/value pair that contains the things first day buy on one server, and the things second day buy on another. And save to XML file.

  Additionally, sharding can accompany with schedule, which can charge the time to persist the data to ensure the capacity of memory. For instance, it can offer ten seconds to persist data once or persisting data after write 10 times.
- **Problem:** if we using sharding it will increased complexity of noSQL

  Solution: Increased bugs because the developers have to write more complicated SQL to handle sharding logic. The solution to it is only the developer pay great attention on test the coding function to guarantee its goal.

6.2 Reliable

A NoSQL database should be highly resistant to corruption. If an application crash, or an operating-system crash, or even a power failure occurs in the middle of a transaction, the transaction should be automatically rolled back the next time the database file is accessed. The recovery process is fully automatic and does not require any action on the part of the user or the application.

- **Problem:** what should we do with if the data is corrupted?

  Solution: To this point, it should have a failover to do with that. Failover servers must themselves have copies of the fleets of database shards. Additionally, Systems that run automatic backups in the background might try to make a backup copy of an SQLite database file while it is in the middle of a transaction. The backup copy then might contain some old and some new content, and thus be corrupt.

6.3 Query

Since querying is generally a useful feature, this have come up with a number of ways to make queries possible. When we want to retrieve the data from XML file, but we know there are numbers of files in the disk, if we go through the value in data, it will become a nightmare.

- **Problem:** How can we retrieve the data from disk or memory efficiently?
  Solution: One way is to structure the data using. We can store the data information in three parts in the memory: key、metadata、address. Key is the data specified key. Metadata is only include the name strings, timestamps, relationships. Address is a link address that include other details. And these link address will be located in disk. When we want retrieve somethings, we search the metadata firstly. It will reduce the time on searching instead of wasting time on search every detailed things one by one. For instance, we have a database about the things purchased from stores, there are T-shirt, dresses, skirts, shoes. I want the information about the size of the red shoes. We can search the shoes first instead of searching the size or red. It is more efficiently for retrieving data.

6.4 Eventual consistency

We must ensure Nodes that have executed the same updates eventually reach an equivalent state .that is to say, once updated done, everything should have the same state.

- **Problem:** As we persist the data according to the scheduler, once scheduler first persist the data in the xml file, if the TextExec want to access the data before the scheduler, it is not able to find the data.
  Solution: The solution is easy to achieve, we can set the searching not only for memory, if the data is not located in memory, the next step is to search the XML file in disk.

6.5 Safety

Databases can sometimes be open to privacy risks as well as fire and security threats. Care providers need to keep certain confidential information about their clients on file; if such information got into the wrong hands the consequences could be extremely harmful.

- **Problem:** How can you be sure that your data is safe, secure and protected?

Solution: Following are some steps to prevent the database from others.

i) ***Identify, locate and discover your most important data***

It is the first thing to know what is the most important data to protect and also you need to identify which particular fields in those databases are the ones you need to protect.

ii) ***segregation of duties***

The next step is to ensure that only people who absolutely require access to each portion of data have it. Access should be blocked to everyone else.

iii) ***Monitor Who's Doing What with the Sensitive Data***

It is critical to deploy a comprehensive Database Activity Monitoring solution in organization. These systems make it easy to define audit trail policies for sensitive tables and individual data columns, including the ability to review "before and after" reports of all changes made.

iv) ***Protect Databases from Internal SQL Injection Attacks***

More reliable and less costly approach is to deploy an "application firewall" between the database and all applications which access it.

v) ***Enforce Strong Password Policies***

Because companies often fail to change/remove the standard administrator user names and today's hackers have access to advanced password-cracking software, it is critical that every user account should have a strong password policy.