ISSN: 0974-2115
www.jchps.com Journal of Chemical and Pharmaceutical Sciences

Multiple-client data management using robust database pattern design

S. A. Ramesh Kumar[1], C.Ashwini[2], E. Rajeshwaran[3]
1. Asst.Prof, Karpaga Vinayaga College of Engineering &Technology.
2. Asst.Prof, Karpaga Vinayaga College of Engineering &Technology
3. P.G. Student, Karpaga Vinayaga College of Engineering &Technology
***Corresponding author: E.Mail: rameshkumarsa@hotmail.com**

ABSTRACT

Major application of Software as a Service is a Multiple-Clients data management. Data management service is offered by a third party vendor to manage the outsourced data with the aid of Multiple-clients database system. The Multiple-clients database system needs a high quality database pattern to have high execution, low space requirements and excellent scalability. II approaches to design the pattern are Independent Tables Shared Instances (ITSI) and Shared Tables Shared Instances (STSI). ITSI has poor scalability; cost of implementation is high when the table size increases and bottleneck occurs. STSI provides good scalability, but poor performance and it calls for a great number of NULLS & wastes large number of places. An adaptive database pattern design method is suggested for Multiple-clients applications to overcome the problems and limitations of ITSI & STSI. ITSI & STSI are used to find a balance to achieve good scalability and high performance with low space requirements. An appropriate number of base tables and supplemental tables are made by identifying the important attributes and remaining attributes. The turn of base tables is specified and constructed using Kernel matrix and Graph partitioning algorithms. A Page Rank algorithm is employed to measure the importance of attributes. The proposed method achieves high scalability, high performance and space demands.

Index Terms: Multiple-Clients, STSI, ITSI

1. INTRODUCTION

Cloud is something, which is present at the remote location. Cloud Computing refers to manipulating, configuring and accessing the applications online. It offers online data storage, infrastructure and application. Cloud computing in which large group of remote servers is networked to allow the centralized data storage, and online access to computer services or resources.

Software as a Service (SaaS) represents a model that companies do not have to purchase and maintain their own infrastructure; instead they can acquire the services by software from a third party. Requests can be made by Service on Demand for Service Providers and require only Internet access to use the avails.

The Service Provider must support Multiple-clients; it requires excellent public presentation, low space requirement and full scalability. In lodge to provide good performance the service provider must design a high-quality database pattern.

2. MULTIPLE-CLIENTS APPLICATIONS

Multitenancy is a principle in software architecture where a single instance of the software functions on a server waiting on multiple customers. A client is a group of users sharing the same view of the software they use. With Multiple client architecture, a software application is designed to provide every client a dedicated part of the instance, including its data, configuration, user management and client's individual functionality.

A Multiple-clients application allows customers or clients to share the same hardware resources, by providing them one shared application and database instance; clients can also configure the application based on their demands.

2.1 Characteristics of Multiple-Tenancy

2.1.1 Hardware Resource Sharing

2.1.2 High Degree of Configurability

2.1.1 Hardware Resource Sharing: The traditional method similar to Application Service Provider (ASP), is the single-clients software development method in which clients have their own server, where server utilization is low. Through virtualization the server utilization can be amended by placing multiple clients on the same host. Virtualization enables higher utilization of the existing servers and scales down the hardware resources needed.

Multiple clients have different shapes they are:

I. Shared application, Independent database

II. Shared application, Independent tables shared database instances

III. Shared application, Shared tables shared database instances

2.1.2 High Degree of Configurability: Customers or clients receive their own customized applications in a single client's environment so they can configure their applications based on their demands. When comes to Multiple-clients environment, Multiple customers or clients share the single application, then it is impossible to configure the application based on user demands. It is necessary to run Multiple versions of the same application next to each other for high degree of configurability.

2.2 Problem Formulation: Service Provider maintains Multiple-clients databases to store and handle the outsourced Multiple-clients data. Renters can apply SQL queries to view their information. Every client outsources a database with the number of tables (T1, T2, T3… Tn) they are called as source tables. It's the obligation of service provider to serve the clients with quick response. To offer such a service to large number of renters service provider requires a Multiple-clients database with good scalability and high performance with low space requirements. To reach this challenge the service provider needs to redesign the database pattern to manage the information efficiently. Redesigned tables are called physical tables.

Source Tables

a. Customer Table for Clients 3

Customer key	Name	Address	Salary	Nation
0005	Raju	UK	9000	London
0046	Sakthi	Goa	20000	India
0012	Gopi	New York	20000	US
0009	Arun	Japan	12000	France

a. Customer table for Clients 6

Customer key	Name	Address	Salary
1111	Ram	Chennai	25000
0215	Shoba	Africa	15000
1580	Jeni	Mumbai	10000

b. Customer table for Clients 10

Customer key	Name	Address	Age
0002	Sri	Coimbatore	25
0258	Srini	Kumbakonam	45
2580	Sathish	Chennai	25

3 ARCHITECTURE OF ADAPTIVE DATABASE PATTERN DESIGN

Client outsources their tables to service provider in their own name. The service provider reconstructs the physical tables from attribute level. Several base tables are constructed using the highly important attributes and by using the remaining unimportant attributes supplementary tables are generated.

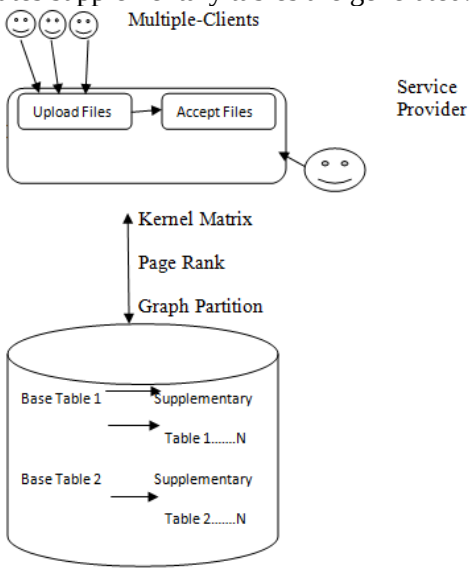

Figure.1.

Common Attribute: An attribute from source table that is highly shared is known as a common attribute. When the usual properties are highly shared by a number of clients then they are called as important attributes.

Star Attribute: An attribute from source tables that is primary key and also referenced by some other source tables is called star attribute.

Uncommon Attribute: An attribute is called an uncommon attribute if it is neither a common attribute nor a star attribute. In the above case of the Customer table in STSI, the data of several clients are stored below the single board using the properties. As we discussed earlier STSI method occupies number of NULL values in empty attribute places. In the above example Nation & Age attribute occupies several NULL values that waste a large sum of quads.

Adaptive Database Pattern design method is presented to overcome the problems of ITSI & STSI. In this method Customer table is separated into a Base Table and Supplementary Table based on the important attributes and insignificant attributes.

3.1 Experimental Work

3.1.1 Data Upload to Service Provider: Many companies outsource their data to a third party service supplier that supplies information management service which hosts a Multiple-clients database. Each society is called clients. Clients manage and query their information by posing SQL queries to the database scheme.

Renters must register their information to a service provider for a service request. Once the service is granted the clients companies can log on to their profile and can upload their data to the avail provider. Service Provider authenticates the data for cloud storage. Renters can view their data from cloud storage once the files have been uploaded.

Company login details can be viewed from the Company Register table. Status table can be utilized to consider the status of data that has been uploaded. Service Provider details can be viewed from the Service Provider Login table.

3.1.2 ITSI & STSI: The traditional method developed to design a Multiple-clients database is Independent Databases and Independent Instances (IDII). In this method the service provider will build an independent and separate database to store the client files, but the single hardware is shared by multiple clients. Deployment of IDII is easy and it has good data isolation and security. Maintenance cost of IDII is very expensive; the service provider has to do so many constellations in order to manage different database instances.

As advancement to IDII, Independent Tables Shared Database Instances (ITSI) has been introduced, by this method clients can share both hardware and database instance. The service provider utilizes the same database instance to store all the source tables that are uploaded by multiple clients in this method maintains private tables for each renter. A TENANT ID is applied to separate the tables from different clients with similar names. Queries from clients are translated using clients ID to return correct answers to the corresponding clients. Maintenance cost of ITSI is low when compared to IDII and also offers better scalability because of its single database instances. The scalability is limited when the number of private tables in the shared database grows in proportion to the number of clients. When the number of tables is large the memory cost will increase.

Shared Tables and Shared Database Instances (STSI) was introduced to overcome the limitations of IDII &ITSI. STSI shares both the database instances and the control panels. The service provider employs a great table to stash away the Multiple-clients data these tables contain all the tuples of clients. A column client ID is used for the purpose of dividing the tuples of each client. NULL is filled for clients with no attributes. STSI has reduced maintenance cost and provides better scalability as the size of the big table varies based on the number of source tables uploaded by clients. Of course STSI also have more or less of the limitations, large quantities of space are wasted with NULLS by a low number of clients with less attributes. The performance of STSI is obviously degraded when database tables become too sparse.

3.1.3 Multiple-Clients DB Design: Multiple clients outsource their data to the service provider; the service provider evaluates the tables and extracts the common, uncommon, star, important and unimportant attributes. Kernel matrix is used to determine the number of base tables and adopt graph partitioning algorithm to construct the base tables. Page Rank algorithm is used to analyze the importance of attributes. A cost based model is used to automatically generate high-quality database pattern.

4. GRAPH PARTITIONING ALGORITHMS

The graph partition problem is defined on data presented in the shape of a graph G = (V, E) with V vertices and E edges, such that it is possible to partition G into smaller components with specific attributes. For illustration, a

k-way partition divides the vertex set into k smaller components. A good segmentation is defined as one in which the number of edges running between separated components is small. Uniform graph partition is a type of graph partitioning problem that consists of splitting up a graph into components, such that the components are of roughly the same size and there are few connections between the parts. Significant applications of graph partitioning include scientific computing, partitioning various stages of a VLSI design circuit and task scheduling.

Customer Table Layout in STSI

TID	Customer key	Name	Address	Salary	Nation	Age
3	0005	Raju	UK	9000	London	NULL
3	0046	Suri	Goa	20000	India	NULL
3	0012	Gopi	New York	20000	US	NULL
3	0009	Arun	Japan	12000	France	NULL
6	1111	Ram	Chennai	25000	NULL	NULL
6	0215	Shoba	Africa	15000	NULL	NULL
6	1580	Jeni	Mumbai	10000	NULL	NULL
10	0002	Sri	Chennai	NULL	NULL	25
10	0258	Srini	Andhra	NULL	NULL	45
10	2580	Sai	Chennai	NULL	NULL	25

Expensive Attribute List

ATTR_NAME	BTable_NO	OCC_TIME	NULL_NUM
C_SALARY	1	10	3
C_AGE	1	5	7
C_NATION	1	2	6

Customer Table Layout in ADAPT

a. Base Table

TID	Customer key	Name	Address	Salary
3	0005	Raju	UK	9000
3	0046	Suri	Goa	20000
3	0012	Gopi	New York	20000
3	0009	Arun	Japan	12000
6	1111	Ram	Chennai	25000
6	0215	Shoba	Africa	15000
6	1580	Jeni	Mumbai	10000
10	0002	Sri	Chennai	NULL
10	0258	Srini	Andhra	NULL
10	2580	Sai	Chennai	NULL

b. Supplementary Tables

TID	Customer key	Age
10	0002	25
10	0258	45
10	2580	25

TID	Customer key	Nation
3	0005	London
3	0046	India
3	0012	US
3	0009	France

5 PAGE RANK ALGORITHM

Page Rank is a link analysis and it specifies a numerical weighting to each component of a hyperlinked set of text files, such as the World Wide Web, with the design of assessing its relative importance within the circle. The algorithm may be applied to any collection of entities with reciprocal quotations and citations. The numerical weight that it ascribes to any given element E is referred to as the Page Rank of E denoted by PR (E).

6 ADVANTAGES OF ADAPTIVE DATABASE PATTERN

6.1 High Scalability: Scalability is the ability of a system, network, or process to handle a growing amount of work in a capable manner or its ability to be enlarged to accommodate that growth. Our proposed system the number of tables does not grow linearly and the scalability is limited.

6.2 High Performance: In Adaptive database pattern method the index method is very effective and it delivers high performance with reduced NULLs but STSI degrades the performance with a number of NULLS in the board.

6.3 Low Space Requirements: The attributes from all the clients are put into a single large table without any importance in STSI and it lets in a great amount of Nulls. In proposing method the attributes are divided into important and unimportant attributes and stored in the base table and auxiliary tables

7. CONCLUSION

In this report, we propose an Adaptive database pattern design by placing the important attributes such as common properties, star attributes and dependent attributes to get the base tables using the important properties. Supplementary tables are built utilizing the other properties. In this method base table is set and built using Kernel matrix and Graph partitioning algorithms. A Page Rank algorithm is used to evaluate the importance of attributes. The proposed method achieves high scalability, high performance and space requirements.

REFERENCES

B C Ooi, B Yu and G Li. One table stores all: Enabling painless free-and-easy data publishing and sharing, In CIDR, 2007, 142-153.

C P Bezemer and A Zaidman, Multiple-clients saas applications: maintenance dream or nightmare? In EVOL/IWPSE, 2010, 88-92.

H Cai, B Reinwald, N Wang and C Guo, Saas Multi-tenancy: Framework, technology and case study. IJCAC, 1(1), 2011, 62-77.

K Haller, Web services from service provider prespective: Client's management services for Multitenant information systems. ACM SIGSOFT Software Engineering Notes, 36(1), 2011, 1-4.

M Hui, D Jiang, G Li and Y Zhou. Supporting database applications as a service, In ICDE, 2009, 832-843.

W Lang, S Shankar, J M Patel and A Kalhan, Towards Multitenants performance slos, In ICDE, 2012, 702-713.

W Lee and M Choi. A Multiple-clients web application framework for saas. In IEEE CLOUD, 2012, 970-971.