ISSN: 0975-766X
CODEN: IJPTFI
Available Online through
www.ijptonline.com
Research Article

A MULTI-TENANT WEB APPLICATION FRAMEWORK FOR SOFTWARE AS A SERVICE (SAAS)

[1]Naveen.M.J, [2]Dr.Muthukumar.B
Department-MCA, Faculty of Computing, Sathyabama University, Chennai-600119, Tamilnadu, India.
Department of Computing, Faculty of Computing, Sathyabama University, Chennai-600119, Tamilnadu, India.
Email: naveen.m@gmail.com

Received on 01-03-2016 *Accepted on 25-03-2016*

Abstract

To cater to the needs of the customers a vendor functions as a facility provider by providing databases as facility besides making this possible Multi-tenant (MT) data management is considered as the chief application of SaaS. With the help of internet the users will be in a position to use the SaaS applications when installed scheduled in a common setting from client-end software. In general terms, Multi-tenancy is considered as a norm in software design at which at a single instance, the software happens to run on a server which serves multiple client company (tenants). All the companies will be provided with a shared user interface (UI) by the Multi-tenant applications and it is a cost-effective method where it reduces the overall expenses of the owner by saving the data of the multiple tenants in a distinct database. The most of important of all problems in this system happens to be the scalability issue. The problem revolves around the capability to attend the raising total of tenants minus a noteworthy degradation of query performance. There are twofold state-of-the-art methods to design the schema namely Shared Tables Shared instances (STSI) and Independent Tables Shared Instances (ITSI). The above mentioned two approaches are associated with certain basic boundaries in assisting the MT database systems. The two problems associated with the STSI approach are firstly, the distributed tables are very sparse and secondly, Indexing on shared table is not operative. There is a poor scalability in ITSI as it has to take care of a great amount of tables. This study proposes a unique MT database schema scheme approach. The design is based on the requirements of the multi-tenant application, besides tenants' requests of data safety, segregation, queries enactment and reaction time speed. This unique method offers a trade-off between STSI and ITSI to attain storage area and performance. The core of discussion in the paper revolves around the know-hows of using the kernel matrix to figure out the amount of base tables, build the base tables through the application of graph-partitioning algorithms and gauge the significance of attributes through renowned Page Rank algorithm. The outcome of the experimental results proves that this method offers a decent scalability and great presentation through less space requirement.

Key words: Multi-tenant, SaaS, adaptive schema design, ITSI, STSI.

1. Introduction:

In the recent past, software industry has seen a number of developing software application services. In particular SaaS has garnered a lot of attention among the industry experts for its special features. In simple terms, SaaS can defined as Software organized as a accommodated facility and retrieved above the Internet [1].A user can use the SaaS by subscribing the software for a remunerated application rather than disbursing for the license of the software. For the initial stage in the month of February 2001, the word SaaS was published in a white paper as Software as a Service: Strategic Backgrounder [2]. Hey days of SaaS started in 2005-2006, with an increase of the internet face at a significant level. Also, from consumers point of view the internet was affordable and it keeps them comfortable in establishing their trade in the industry through internet.

This is chiefly due to the pressure many of the minor and average companies face to reduce the expenses in IT (information technology) through the cost cutting in purchasing a number of software certificates and hardware updating. Henceforth, now prevails a situation where many of the software vendors have looked up to the standard of distributing hardware means, software and facilities through the Internet to a huge customer base. An environment of this sort is widely known as cloud computing and the customers involved in this process are called as tenants. SaaS is nothing but the cloud software supply ideal and the key distinctive feature of SaaS is multi-tenancy [3]. This gives an option for the SaaS vendors, to help manifold tenants having different needs, by running a sole case of an application and a database.

The resource utilization will be increased by Multi-tenancy and also it shares the equivalent database case to MT. Nevertheless, there are chances for many customers to get affected due to the company's sharing of resources and its risk factors such as the outage of shared resource. Complexity of the resolution is also an addition of shared resources [4]. One of the biggest challenges and the most essential aspect in MT databases is to determine a quality high-end database schema. This section deals with the description of another multi-tenant architecture, Independent Tables and Shared Instances (ITSI) for a better memory distribution in database services. In the proposed method, the tenants share both the hardware and database instances. All the tenants are served by the service provider who keeps a huge common database. The tuples present in each tenant is loaded in its specific secluded tables designed after the base method and the storage is done in the private table in the common database case. The private tables that exist among various tenants are independent in nature. In this study, an adaptive database schema design technique for multi-tenant applications is proposed. A trade-off between STSI and ITSI is done and a fine balance is struck between these two methods to attain a high presentation and better scalability with little space requirement.

2. Related Work:

A number of MT database schema mapping methods were proposed in [5], [6]. The three chief methods of handling data in MT databases form a base from which most of the related techniques were formulated. This was mainly to generate a valid secluded database schema for each and each tenant in MT databases. The MT database schema mapping methods are categorized into three parts: Techniques that use a sole method to signify data in a MT database, techniques that mix two or three data separation methods, and practices that mix amorphous data as XML data type and arranged data as association database.

Techniques Using a Sole Approach:

In general, a sole MT data storage method to signify data in MT databases will be preferred by SaaS providers. They prefer this mainly to overcome the intricacy in database scheme. Private Tables Technique is formulated after the detached table's schema, at which each tenant happens to have a valid schema comprising of a set of extensions tables. This particular technique that provides a higher performance nevertheless ignores the storage space and scalability and it has been elucidated in [6].This is preferable to deploy while the applications have a little tenants or a few tables [7]. Universal Table Technique which is mentioned to as the flexible technique has been formulated through shared tables schema [8].SalesForce.com adopted this technique which happens to waste storage space due to the wide usage of null values. Besides, also harms the query performance as it does not upkeep hints.

Techniques Merge the different Tables Schema and the distributed Tables Schema methods:

A number of methods ware recommended instead of data in multi-tenant databases as in [7], [9], [10]. These techniques aim at striking a poise position among performance, scalability, , data segregation and storage space supplemented with cost-effective features. This aspect is achieved through the techniques mingling distinct tables and distributed tables schemas. Yet, these techniques accomplish a few features, at the charge of other features. Technique has been projected in [9]; it keeps storage space and avoids invalid values, at the cost of reconfiguration. This study proposes to solve null problem comparatively to tenant essential of execution. Chunk Table method and Chunk Folding technique ware has been proposed in [11]. Owing to the enormous joining operation, these techniques were considered to be flexible and also it reduces the total of tables, at the cost of growing the complexity of the queries. These techniques focal point lie on perpendicular apportioning into rational tables 'chunks', while the proposed study aims at perpendicular and parallel apportioning to save a particular performance level for each and every tenant.

3. Proposed Work:

3.1 Overview

MT data management is a type of SaaS where a service of third party giver provides databases as a service. And it also offers its consumers with a means to generate, stock and contact their databases in the site of host. To put forth in a different version, Multi-tenant databases are a trait that will allow a distinct case of an application to handle numerous consumers at one particular time. This scheme was investigated beforehand devoid of some clear connection with Multi-tenancy. In our study, we have used and Shared tables and Shared Database Instance (STSI) Independent Tables and shared Instance (ITSI),.The study has exchanged STSI and ITSI a fine balance has been struck amid them to attain a better scalability and higher performance with the provision of a little space. Kernel matrix has also been utilized to ascertain the total of the base tables, relate algorithm of graph-partitioning build the base tables and also evaluation of the significance of attribute is exercised with the help of the renowned Page Rank algorithm.

3.2 Overall Architecture

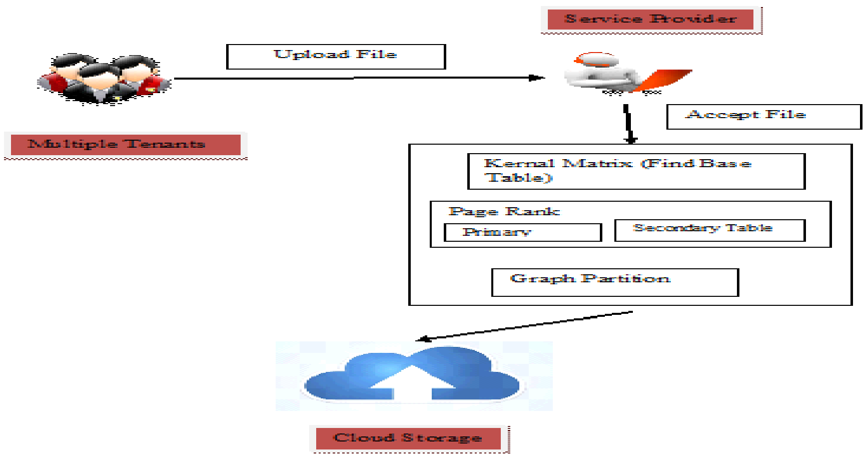


Figure1: Multi tenant data management system.

3.3 Independent Tables and Shared Database Instances (ITSI):

In this study ITSI has been considered for memory distribution in database services. This approach supports the tenants not only to share hardware and database cases. The facility supplier is capable enough to serve all tenants as they hold a large shared database. All of the tenant load their tuples to their individual tables configured after the base schema and it also stocks the individual table in the distributed database instance. The individual tables among various tenants are independent. The particulars of ITSI architecture are briefly illustrated which are as follows: As distinguished to IDII, the system has simply one common database and also a common query processor and buffer. The storage of data as tables sets from all tenants by the shared database is in the form {{T1R1...,T1Rn},{T2R1,...T2Rn},,{TmR1,TmRn}}. Here Ti Ri represents the private table of tenant Ti with relation Ri. On account of spare table name from various tenants, the label of all and every private table is affixed with a Tenant ID to demonstrate who possess the table. For example, tenant 1s private Worker table is named asEmployee1. To return the right responses, the queries are reformulated to distinguish the customized table names.

For example, to recover tuples from table of Employee, the basic query issued by tenant 17 is as follows. In difference to IDII, the database case, query processor, and cache buffer of all the facilities in various tenants are distributed in ITSI. The data of the similar tenant is shared while the data among various tenants is independent. The positive aspect of this technique is a known fact as here is not at all a necessity for manifold database instances. This is an indication of a cost-effective feature particularly while a huge figure of tenants remain processed and this leads to the reduction of large maintenance cost on account of various database instances management.

Yet, ITSI happens to feature a problem at which all table is stored and optimized. Separately. And the growth of private tables in the shared database is linear by the amount of tenants. Hence, it takes large memory for sharing the tables to tenets.

3.4 Shared Tables and Shared Database Instances (STSI)

In STSI, there is a provision for only one query processor in the system besides entirely the cases and tenants share the similar processor. Furthermore, contrast to ITSI, the whole of the tenants share both databases and tables for the better a management purpose of the whole data. STSI is unique since the relation of universal where the last is an extensive virtual schema that places whole of the entities and associations in the similar rational table, at the similar period the previous is a schema for bodily illustration and stocks a total of units that goes to the similar unit set in the similar table. For instance, in this study all worker statistics would be stored in tenant into an extensive table, nonetheless the worker, clienteles and yields would not be put in the similar corporal table.

This study, trades-off the twofold approaches and strike a poise among the twofold approaches so as to attain a better scalability, higher performance and a little space occupancy. In real time applications, various subcontracted tables after various tenants are theme oriented and the total of the characteristic configured through all the tenants are not that large. On these lines of observation, the study builds the corporal tables from the attribute level rather than the tenant level. At first highly important attributes are extracted after the various base tables and tenants are built with the help of such an important attributes. For the left out insignificant attributes, supplementary tables are built for each one of them in the form of column-based tables.

3.5 Kernel matrix and partitioning algorithm:

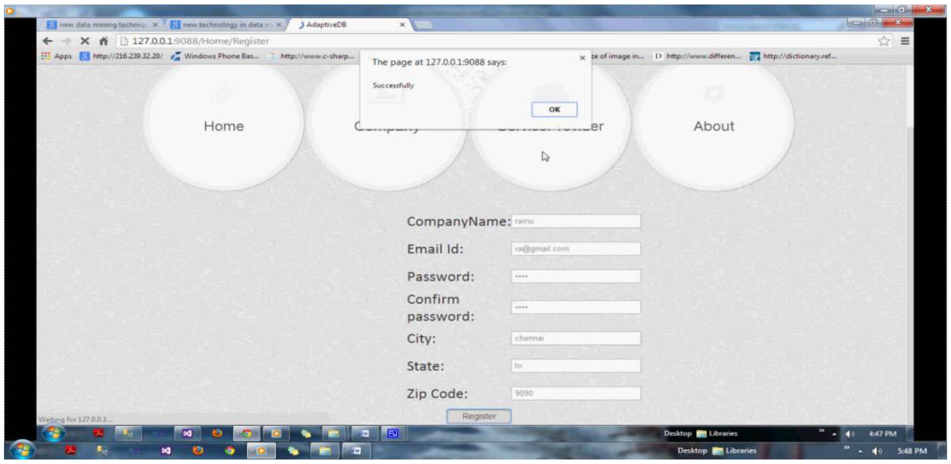
In this projected study, algorithm of kernel matrix is introduced to identify sum of base table. The Kernel-based learning functions by inserting the data into a Hilbert space and it looks out for linear relations in such kind of space. This embedding is carried out in an implicit manner by stating the inner products among all due of points to certain extend than by providing their matches clearly. A number of advantages are present in this approach.

The most significant findings from the information are that the internal product in the inserting space could frequently be computed. This can be done at ease when compared to the correlates of the points themselves. At this juncture, the Partitioning algorithm is used to augment locality of query and reduce response time of query, the primary partitioning has to be optimized in terms of possessing nearly reasonable supplies (effective results) through little amount of edge-cuts.

4. Result and Discussion

As a form of software as a service, multi-tenant database system brings great benefits to organizations by providing seamless mechanisms to create, access and maintain databases at the host site. However, the way of providing high-quality services for multiple tenants becomes a big challenge.

To address the problem, we adaptive database schema design for multi tenant data management and analyze their features in terms of query performance and system scalability.



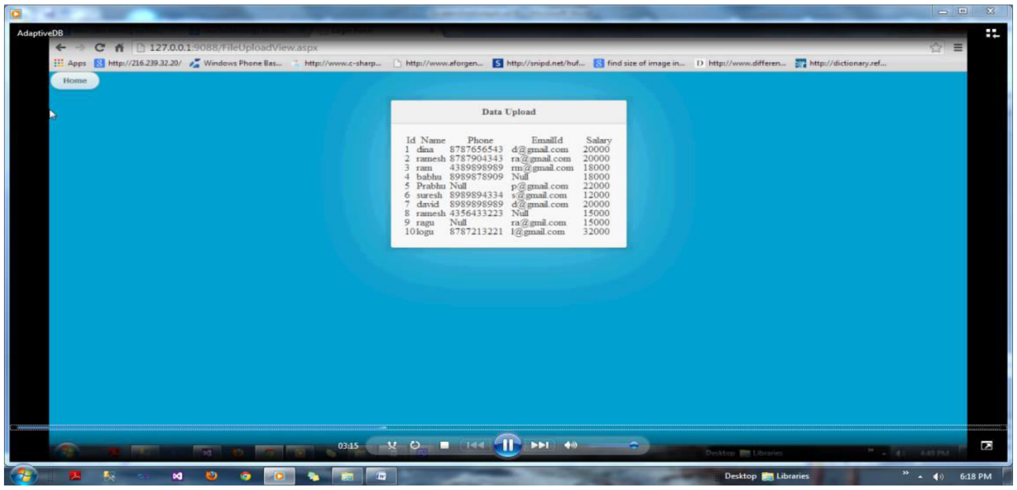
The page at 127.0.0.1:9088 says:

Successfully

CompanyName:	ramu
Email Id:	ra@gmail.com
Password:	****
Confirm password:	****
City:	chennai
State:	tn
Zip Code:	9090

Figure2: Registration process of Company or Tenant.

The above screens show the registration process for every company. This process is mandatory for all company for uploading the data to cloud database.

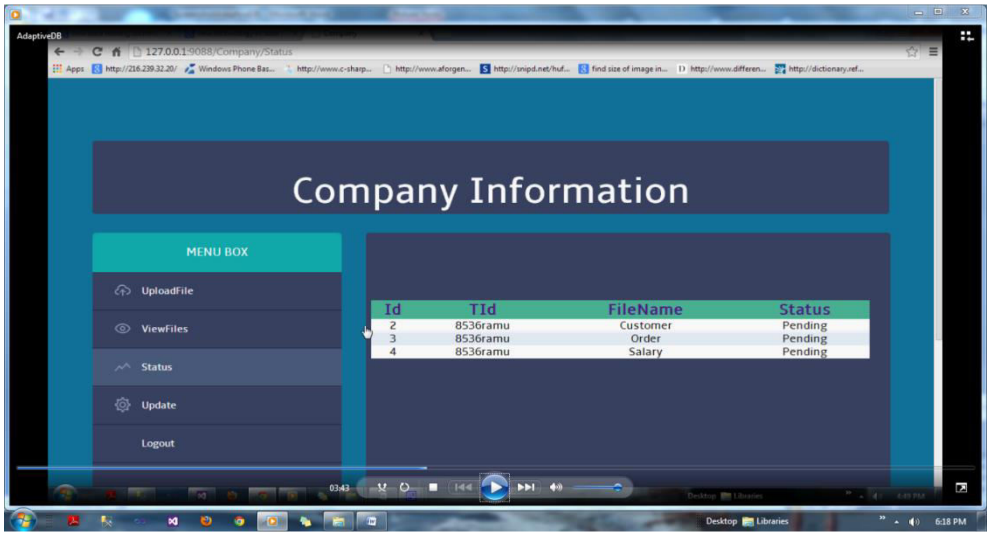


Data Upload

Id	Name	Phone	EmailId	Salary
1	dina	8787656543	d@gmail.com	20000
2	ramesh	8787904343	ra@gmail.com	20000
3	ram	4389898989	rm@gmail.com	18000
4	babhu	8989878909	Null	18000
5	Prabhu	Null	p@gmail.com	22000
6	suresh	8989894334	s@gmail.com	12000
7	david	8989898989	d@gmail.com	20000
8	ramesh	4356433223	Null	15000
9	ragu	Null	ra@gmil.com	15000
10	logu	8787213221	l@gmail.com	32000

Figure3: Display data as table format on database.

The above screen shows the data on cloud database in table format. Once registration is finished the tenants start to upload their data on data base schema. Now days more company needs to outsource their data for data management and cost management.



Id	TId	FileName	Status
2	8536ramu	Customer	Pending
3	8536ramu	Order	Pending
4	8536ramu	Salary	Pending

Figure4: Assigning Tenant ID to Each Tenant.

The above figure explains the Tenant ID for each Tenant (Company). Once the registration process has done, each tenant will get tenant id for accessing their data. In our multi tenant database schema system, tenant data's are stored in outsourced database based on the multi tenant system. If tenant want to access the data, they can access by tenant id.

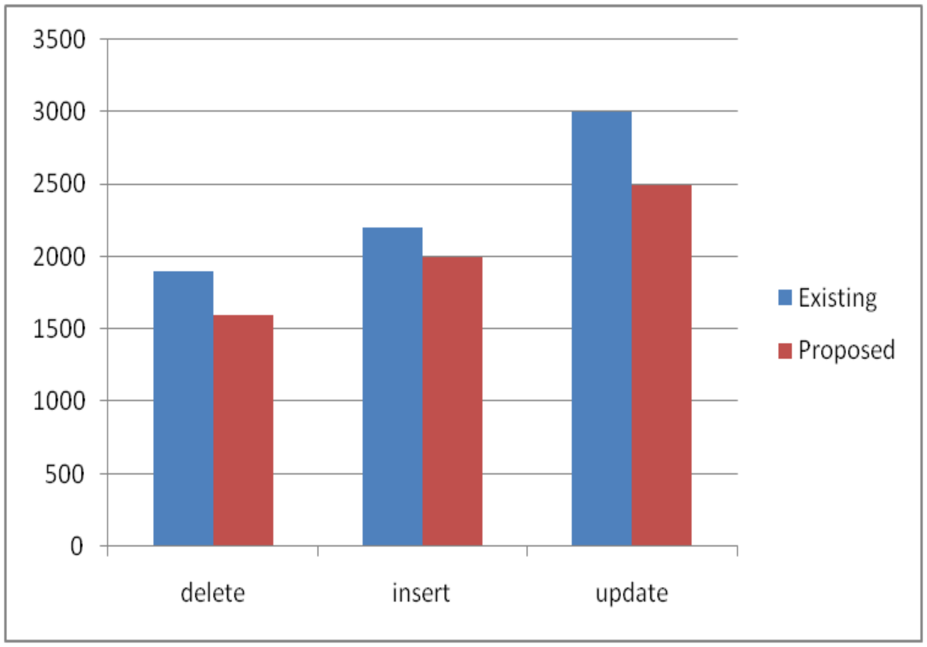


Figure 5: Performance Comparison source table.

The above figure 5 shows the comparison of existing and proposed system. The cost of the operation is low in proposed system compare than existing. Because the number of attributes to be updated is usually not many and the updated operation always specifies the filter condition in the WHERE clause. While the delete operation of STSI is the best since other methods will have more deletions and locating the tuples to be deleted will cost some time.

5. Conclusion

For together the presentation and data the Scalable multi-tenant applications on the community cloud needs a scalable architecture. Through this system, the proposal of an Adaptive multitenant data management framework is projected. This could be used to expand accessible cloud applications so that high scalability of the storage resources can be attained. The study has proposed three techniques within the background. The grouping of STSI and ITSI is applied to reduce the storage magnitude of the tenants' tables and make each of the operation cost-effective. To figure out the base tables, Kernel matrix is one of the technologies applied and partitioning of algorithm helps to build the base tables which are recognized through the Kernel matrix. The outcome of the experimental results proves that trades off of the STSI and ITSI notably decrease the utilization of disk space.

6. References

1. H. Yaish, M. Goyal, and G. Feuerlicht, “An Elastic Multi-tenant Database Schema for Software as a Service,” 2011 IEEE Ninth Int. Conf. Dependable, Auton. Secur. Comput., pp. 737–743, Dec. 2011.

2. R. F. Chong, “Designing a database for multi-tenancy on the cloud Considerations for SaaS vendors,” pp. 1–12, 2012.

3. F. Burno. "Exeuting an IP Protection Strategy in a SaaS Environment", http://www.slideshare.net/Rinky25/saas-environment, Jul.22, 2011.

4. Nitue, "Configurability in SaaS (software as a service) applications", ISEC , 2009, pp. 19-26.

5. S. Aulbach, M. Seibold, and S. a P. Ag, “A Comparison of Flexible Schemas for Software as a Service,” Proc. 35th SIGMOD Int. Conf. Manag. data, pp. 881–888, 2009.

6. H. Yaish, M. Goyal, and G. Feuerlicht, “An Elastic Multi-tenant Database Schema for Software as a Service,” 2011 IEEE Ninth Int. Conf. Dependable, Auton. Secur. Comput., pp. 737–743, Dec. 2011.

7. Liao, K. Chen, and J. Chen, “Modularizing Tenant-Specific Schema Customization in SaaS Applications,” AOAsia "13 Proc. 8th Int. Work. Adv. Modul. Tech., pp. 9–11, 2013.

8. M. H. M. Hui, D. J. D. Jiang, G. L. G. Li, and Y. Z. Y. Zhou, “Supporting Database Applications as a Service,” 2009 IEEE 25th Int. Conf. Data Eng., pp. 832–843, Mar. 2009.

9. S. Aulbach, T. Grust, D. Jacobs, A. Kemper, and J. Rittinger, “Multi-tenant databases for software as a service: schema-mapping techniques,” Proc. ACM SIGMOD Int. Conf. Manag. Data, pp. 1195–1206, 2008.

10. S. Foping, I. M. Dokas, J. Feehan, and S. Imran, “A new hybrid schema-sharing technique for multitenant applications, ” 2009 Fourth Int. Conf. Digit. Inf. Manag., 2009.

Corresponding Author:
Naveen.M.J,
Email: naveen.m@gmail.com