INTERNATIONAL JOURNAL OF PURE AND APPLIED RESEARCH IN ENGINEERING AND TECHNOLOGY

A PATH FOR HORIZING YOUR INNOVATIVE WORK

RELATIONAL DATABASE SERVICE ON CLOUD

M JAVED MANSURI, PROF. R. R. KEOLE

Dept. of Computer Sci. & Engi., HVPMCOET, Amravati-444605.

Accepted Date: 15/03/2016; Published Date: 01/05/2016

Abstract: The usefulness of the new transactional service namely “database-as-a-service” (DBaaS) also called as Relational Database Service, can help in future development and maintenance of online databases. A DBaaS promises to reduce much of the operational burden of provisioning, configuration, scaling, performance tuning, backup, security, and easy access control from the database users to the service operator, offering much reduction in overall costs to users. Early DBaaS efforts include Amazon RDS and Microsoft SQL Azure, and also the salesforce which are promising in terms of establishing the market need for such a service. But they do lack in some features which they do not address like these three important challenges: efficient multiple usage, easily scalable, and database privacy. We try to overcome these three challenges before outsourcing database software and many users can easily manage their data and also proves to be economical for many service providers. The key technical features of Relational Database service include: (1) a workload sharing approach to multiple users that identifies the workloads among users that can be easily located on a database server, achieving higher unification and better performance than existing approaches; (2) An adjustable security scheme that enables SQL queries to run over encrypted data, including ordering operations, aggregates, and joins. An underlying theme in the design of the components of Relational Cloud is the notion of workload awareness: by monitoring query patterns and data accesses, the system obtains information useful for various optimization and security functions, reducing the configuration effort for users and operators.

Keywords: Cloud Application Distribution, Application Deployment, Relational Database, Cloud Services

PAPER-QR CODE

Corresponding Author: MR. M JAVED MANSURI

Access Online On:

www.ijpret.com

How to Cite This Article:

M Javed Mansuri, IJPRET, 2016; Volume 4 (9): 460-466

INTRODUCTION

Database management systems (DBMSs) are an important and basic component in most computing environments nowadays, and their importance will be remaining this high always. With the introduction of cloud computing services and storage, the facility to offer a DBMS as an outsourced service is gaining importance, as researched by Amazon's RDS and Microsoft's SQL Azure (1). Such a database as a service (DBaaS) is gaining attraction mainly for two reasons. First, due to economies of scale, the hardware and energy costs previously users were handling is likely to be get much lower as they are paying for a share of a service rather than running and maintaining everything by themselves. Second, the costs incorporated in a well-designed DBaaS will be equal to actual usage ("pay-per-use")—this is similar in both software licensing and administrative costs. The latter are often a significant expense because of the specialized expertise required to extract good performance from commodity DBMSs. By making a central server and automating many database managements related tasks, a DBaaS can drastically reduce operational costs and also performs well. From the DBaaS operator point of view, by taking advantage of the lack of interdependency between workloads of different applications, the service can be run on very few machines which otherwise could be run on independently provisioned machines for its peak performance.

There are three challenges that largely influence the design of Relational Database: useful multi-tenancy to minimize the hardware requirement for a given (or predicted) workload, flexible scale-out to handle increasing workloads, and database security.

1.1 efficient multi-tenancy: - The main aim is to reduce the number of machines required, while fulfilling application-level query efficiency performance goals. Relational Cloud serially determines which databases should be placed on which machines using a different non-linear optimization technique, combined with a model that calculates the combined resource utilization of multiple databases running on a machine. The design of Relational Cloud also includes a lightweight mechanism to perform live migration of databases between machines.

1.2 flexible distribution: - A efficient DBaaS must be able to support database and workloads of varying sizes. Relational Database, we use a recently developed workload adaptable [2], which uses graph partitioning to automatically understand complex query and handle and map data items to nodes to reduce the number of multi-node transactions/statements. Statements and transactions across multiple nodes obtain significant overhead, and are the main limitation to linear scalability in practice. Our approach makes few inference on the data or queries, and works well even for twisty workloads or when the data contains complex N-to-N relationships.

1.3 Privacy: - A significant limit to deploying databases in the cloud is the recognized lack of privacy, which in turn reduces the quality of trust users are willing to place in the system. In Relational Database, we have developed CryptDB, a set of techniques designed to provide privacy (e.g., to prevent administrators from seeing a user's data) with an acceptable impact on performance. CryptDB enforces different encryption levels for different types of data, based on the types of queries that users run. Queries are evaluated on the encrypted data, and sent back to the client for final decryption; no query processing runs on the client.

II. Working of Our 'Relational Database' System

Relational Database is constructed to allow many and independent users to connect directly to the untrusted cloud database without any common server. We assume that an occupant organization acquires a cloud database service from an untrusted Database provider. The tenant then deploys one or more machines and installs a Secure Database client on each of them. This user allows a user to connect to the cloud database to manage it, to read and write data, and even to create and modify the database tables after creation.

The information managed by this Secure database includes plaintext data, encrypted data, metadata, and encrypted metadata. Plaintext data is information a sender wishes to transmit to a receiver. To prevent an untrusted cloud service provider from violating confidentiality of user data stored in plain form, Secure database adopts multiple cryptographic techniques to transform plaintext data into encrypted tenant data and encrypted tenant data structures because even the names of the tables and of their columns must be encrypted. Relational database users produce also produce a set of metadata containing information required to encrypt and decrypt data as well as other administration information. Even metadata are encrypted and stored in the cloud database.

Relational database proposes a totally different way where in all data and metadata are stored in the cloud database. Relational database users can obtain the necessary required metadata from the untrusted database by using SQL statements, so that multiple instances of the relational database client can access to the untrusted cloud database independently with the guarantee of the same availability and scalability properties of typical cloud DBaaS. Encryption strategies for user's data and productive solutions for metadata management and storage are described in the following two sections.

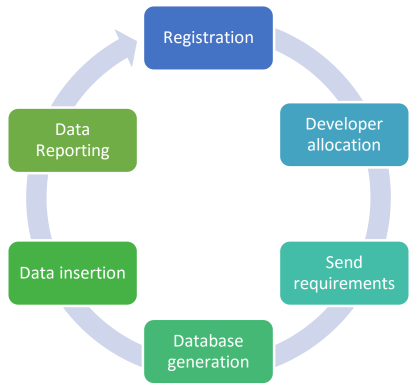

Fig: Work Flow of Relational Database System

2.1 data management: -

Encrypted user's data are stored by using secure tables. SQL statements, each plaintext table is converted into a secure table as the cloud database is untrusted one. The name of a secure table is generated by encrypting the name of the relating plaintext table. Table names are encrypted with the help of same encryption algorithm and an encryption key that all the relational database users have information about. Hence, the encrypted name can be computed from the plaintext name. On the other hand, column names of secure tables are randomly generated by relational database; hence, even if different plaintext tables have columns with the same name, the names of the columns of the corresponding secure tables are different. Relational database provides three layered security attributes:

1) Column (COL) is the default secure level that should be used when SQL statements work on one column; the values of this column are encrypted through a randomly generated encryption key that is not used by any other column.

2) Multicolumn (MCOL) should be used for columns referenced by join operators, foreign keys, and other operations involving two columns; the two columns are encrypted through the same key.

3) Database (DBC) is recommended when there are operations involving multiple columns; in this instance, it is easy to use the special encryption key that is generated and completely shared among all the columns of the database characterized by the same secure type.

2.2 metadata management

Metadata obtained by relational database contain all the information that is required to manage SQL statements over the encrypted database in a way easily understandable to the user. Metadata management strategies depict an original idea because relational database is the first architecture storing all metadata in the untrusted cloud database together with the encrypted tenant data. Relational database uses two types of metadata.

1) Database metadata are related to the whole database. There is only one instance of this metadata type for each database.

2) Table metadata are associated with one secure table. Each table metadata contains all information that is necessary to encrypt and decrypt data of the associated secure table.

III. Facilities given by Relational database service System

a. communication

- Customer-developer communication
- Customer send requirements to developer
- Customer can send request to get complex report to developer
- Developer send report in excel sheet etc.
- Inbox
- Outbox
- Customer-cloud admin communication

b. reporting

- Customer can generate simple report
- Send request to developer for complex reports
- Complex report generation
- Getting complex reports in excel format
- Get reports in various format like Excel

c. query processing

- Criteria selection
- Dynamic query generation
- Query processing
- Report creation in excel sheet
- Delivery

IV. Uses of Relational Database System

Meaningful use of our Database-as-a-Service system can be broadly summarized as follows [6].

- As using relational database all record of users is stored in one single database then this database can be used to retrieve all data according to the
- Customer can maintain any number of databases on cloud using developer's support as well as GUI editor.
- No need to worry about maintenance.
- Excellent reporting facility.
- Customers can maintain their database online in appropriate cost.
- Multitenant applications help our system to run throughout the web so as to provide equal amount of features to each and every customer.

V. CONCLUSION

We propose an ingenious architecture that guarantees confidentiality of data stored in public cloud databases. Unlike may advance approaches, our solution does not depend on an intermediate proxy that we consider a single point of failure and a bottleneck limiting availability and scalability of typical cloud database services. A large part of the research includes solutions to support concurrent SQL operations (including statements modifying the database structure) on encrypted data issued by heterogeneous and possibly geographically dispersed clients. The proposed architecture does not require changes to the cloud database, and it is immediately applicable to already present cloud database, such as the experimented PostgreSQL Plus Cloud Database [4], Windows Azure [5], and Xeround [3]. There are no theoretical and practical limits to extend our solution to other platforms and to include new encryption algorithms.

REFERENCES

1. S. Das, D. Agrawal, and A. E. Abbadi. ElasTraS: An elastic transactional data store in the cloud. HotCloud, 2009.
2. C. Curino, E. Jones, Y. Zhang, and S. Madden. Schism: A Workload-Driven Approach to Database Replication and Partitioning. In VLDB, 2010.
3. "Xeround: The Cloud Database," Xeround, http://xeround.com, Apr. 2013.
4. "Postgres plus Cloud Database," EnterpriseDB, http://enterprisedb.com/cloud-database, Apr. 2013.
5. "Windows Azure," Microsoft Corporation, http://www.windowsazure.com, Apr. 2013.
6. Carlo Curino, Evan P. C. Jones, Raluca Ada Popa, Nirmesh Malviya, "Relational Cloud: A Database-as-a-Service for the Cloud".
7. Carlo Curino, Evan P. C. Jones, Raluca Ada Popa, Nirmesh Malviya, "Relational Cloud: A Database-as-a-Service for the Cloud".
8. Luca Ferretti, Michele Colajanni, and Mirco Marchetti, "Distributed, Concurrent, and Independent Access to Encrypted Cloud Databases", *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS,* VOL. 25, NO. 2, FEBRUARY 2014.