ACID Support and Fault-Tolerant Database Systems on Cloud: A Review

Pratiyush Guleria*1
*1NIELIT,Doecac Society, Chandigarh, Branch Office, Shimla, H.P, India

Abstract: Cloud computing represents a different way to architect and remotely manage computing resources. One has only to establish an account with Microsoft or Amazon or Google to begin building and deploying application systems into a cloud. These systems can be, but certainly are not restricted to being simplistic. Some applications require http services, some require relational database or might require web service infrastructure and message queues. With clouds, IT-related applications can be provided as a service, which can be accessed through internet. There are platforms on cloud which provide scalability and high availability properties for web applications but there are problems related to data consistency at the same time, and in case of server failures, it becomes major problem in applications related to payment services. Data needs to be properly managed in cloud environment and to achieve proper transaction processing and consistency, RDBMS techniques such as ACID transactions should be used. Web services in Azure ensure application availability by replicating stored data at least three times and offer optional geolocation of replicas in separate Microsoft data centres to provide disaster recovery services. Azure storage services provide scalable persistent storage of structured tables, blobs and queues.

Keywords: ACID, Azure, Scalability, Consistency, Relational, Persistent, Blobs, Queues

I. INTRODUCTION

There are platforms that enable to quickly build, deploy and manage applications across a global network. You can build applications using any language, tool or framework. There might be need to interoperate with CRM or e-commerce application services, necessitating construction of a custom technology stack to deploy into the cloud if these services are not already provided there. It might require the use of new types of persistent storage that might never have to be replicated because the new storage technologies build in required reliability. They might require the remote hosting and use of custom or 3rd party software systems. And they might require the capability to programatically increase or decrease computing resources as a function of business intelligence about resource demand using virtualization [1].

For example Windows Azure is an open cloud platform that enables you to quickly build, deploy and manage applications. Hosted services are easy to manage, scale up and down, reconfigure, and update with new versions of your application’s code. While the emergence of cloud computing has made it possible to rent information technology infrastructures on demand, it has also created new security challenges. The primary security concern is trusting data (or resources in general) on another organization’s system. This document seeks to examine the current state of security in cloud computing and presents a set of challenges to address the security needs of clouds. The end result is a framework to help the design and implementation of effective cloud security infrastructures [2]. Apart from it there is need to maintain and check ACID properties in software and web applications so that it guarantees error-free transactions.

II. INTEGRITY OF THE DATA

To ensure integrity of the data, we require that the database system maintain the following properties of the transactions [4, 5]:

1. Atomicity: Either all operations of the transaction are reflected properly in the database, or none are. Atomicity need to be maintained in the presence of deadlocks, database software failures, application software failures, CPU failures, disk failures etc. Sometimes transaction may not always complete its execution successfully and to ensure atomicity of software, an aborted transaction must have no effect on the state of the database. There is need of recovery schemes mechanism on the clouds such that it manages the aborted transactions and roll back all such operations.

2. Consistency: Execution of a transaction in isolation (that is, with no other transaction executing concurrently) preserves the consistency of the database. Handling data in clouds is still complicated - in particular as data size and diversity grows, pure replication is no viable approach, leading to consistency and efficiency issues.
3. **Isolation**: Even though multiple transactions may execute concurrently, the system guarantees that, for every pair of transactions Ti and Tj, it appears to Ti that either Tj finished execution before Ti started, or Tj started execution after Ti finished. Thus, each transaction is unaware of other transactions executing concurrently in the system.

4. **Durability**: After a transaction completes successfully, the changes it has made to the database persist, even if there are system failures. For data centre-type applications in cloud systems, it is considered as one of the main features. A transaction that completes its execution successfully is said to be committed. A committed transaction that has performed updates transforms the database into a new consistent state, which must persist even if there is a system failure.

### III. FAULT-TOLERANT DATABASE SYSTEMS

In a Platform as a Service (PaaS) like Windows and SQL Azure, however, selection of the proper engine for a particular dataset has implications ranging from cost to performance, and selecting the right engine is critical when leveraging data across different constructs. If the application needs to store data with ACID properties, is located in Windows Azure, and requires multiple indexes, then SQL Azure is a choice for the application’s data. Data can be accessed in SQL Azure using ODBC and ADO.NET classes, and manage it with traditional SQL Server tools[3]. ACID properties are essential while designing data for datasets that require a high degree of safety in the read/write pattern. The primary engine used for this type of data is often SQL Azure – an RDBMS in the same datacenters as Windows Azure [12]. In Windows Azure Table, there is support for Secondary Indexes i.e. Query and retrieve results will be sorted by other properties also. For single index, Query and retrieve results are sorted by Partition Key and RowKey. Likewise, for Entity Groups, Atomic transactions across multiple entities within same partition is there whereas for Single Entity Transactions, there is Single Entity Insert, Update, or Delete Atomically. Azure tables are Massively Scalable and simple familiar API are there which includes .NET-ADO.NET Data Services and LINQ(Language Integrated Query). Apart from it REST(Representational State Transfer) protocols can be used[13]. Fault-tolerant database systems are essential while working on cloud scale because Hardware and Software failures are inevitable or there can be failures due to anomalies.

Windows Azure Platform consists of fault-tolerant SQL databases at the highest level of the stack. For mitigating failures it consists of Database replication. At any one time, Windows Azure SQL Database keeps three replicas of each database— one primary replica and two secondary replicas. If any component fails on the primary replica, Windows Azure SQL Database detects the failure and fails over to the secondary replica. Replicas of each database are scattered across nodes such that no two copies reside in the same failure domain under the same network switch or in the same rack. Windows Azure SQL Database maintains a global map of all databases and their replicas in the Global Partition Manager (GPM). The global map contains the health, state and location of every database and its replicas [14]. In Windows Azure, SQL Azure provides ACID support similar to SQL Server, aside from distributed transactions. Atomic operations are provided within a given partition in Windows Azure Table Storage when multi-record writes are written to the same partition which may be done in one atomic operation. SQL Azure, as well as Windows Azure Storage (blobs, tables, and queues) are also considered Durable, as storage is triple-replicated for redundancy and these copies are replicated to a geographically separate datacenter again [15]. Windows Azure Table Storage provides snapshot isolation for each single query request to the service. A query maintains a consistent view of the partition from the start time of the query and throughout the transaction [3].

### IV. AVAILABILITY AND SCALABILITY

When application is deployed as a hosted service in Cloud, it runs as one or more roles. A role simply refers to application files and configuration. We can define one or more roles for our application, each with its own set of application files and configuration. For each role in our application, we can specify the number of VMs (Virtual Machines), or role instances, to run. For Example, Windows Azure is Microsoft's application platform for the public cloud which provides High Availability, Scalability and manageability, fault tolerance, geo-replication, limitless storage, and security of the cloud. Every Windows Azure application runs in one or more role instances, i.e., in one or more VMs. Each VM has local storage, which an application is free to use [11]. To achieve high availability and scalability, it is critically important that our application’s data be kept in a central repository accessible to multiple role instances. To help with this, Windows Azure offers several storage options such as blobs, tables, and SQL Azure [7]. However, there are transaction management products like CloudTP or Cloud Transaction Processing which enables applications running in distributed computing and cloud computing architectures to embed logical business transactions that adhere to the properties of ACID transactions. CloudTran enables a broad range of developers to implement highly scalable applications that run in cloud computing environments and distributed architectures[10]. Many Web applications need strong data consistency for correct execution and should support...
Atomicity, Consistency, Isolation, Durability (ACID) transactions without compromising the scalability of the cloud for Web applications, even in the presence of server failures and network partitions. However, although the high scalability and availability of the cloud makes it a good platform to host Web content, but scalable cloud database services provide relatively weak consistency properties [8]. Datastore support layer in the cloud platform are being extended with ACID transaction semantics and this is referred to as database-agnostic transactions (DAT). Such support is for a wide range of applications that require atomic updates to multiple keys at a time is independent of any datastore but that can be used by all datastores that plug into the database support layer. For scalability purpose, transactions should be performed across a set of related entities instead of global or table-level transactions. It results in avoiding slow, coarse-grain locking across large sections or tables of the datastore [9].

A. Concurrency related problems

When several transactions execute concurrently in database, however, the isolation property may no longer be preserved. But through concurrency control schemes the interaction among the concurrent transactions can be controlled and for that lock-based protocols are used. A lock is a variable that is associated with a data item in the database. A lock can be placed by a transaction on a shared resource that it desires to use, such that while one transaction is accessing a data item, no other transaction can modify that data item. The most common method used to implement this requirement is to allow a transaction to access a data item only if it is currently holding a lock on that item. In order to determine that the operations of two transactions do not result into conflict with each other, serialisation of schedules is required. For correctness of serialisation of schedules, binary locks and two phase locking protocol are used. A binary lock can have two states: locked and unlocked (or 1 and 0). A distinct lock is associated with each database item X. If the value of the lock on X is 1, item X cannot be accessed by a database operation that requests the item. If the value of the lock on X is 0, the item can be accessed when requested [16]. In 2PL protocol, locking takes place in two phases. In first phase, No conflicting locks are granted to a transaction. If transaction wants to read an object, then it needs to obtain the S (shared) lock else to modify an object, it needs to obtain X (exclusive) lock. In second phase, the locks are held only till they are required.

V. CONCLUSION:

Cloud Computing is forming the base towards better data management and offering us to produce highly-available and scalable applications. It offers persistence data storage and resource optimization through virtualization. Windows Azure SQL Database systems are providing rich query semantics, fault-tolerant data storage, and support for ACID transactions with complex data processing capabilities. Platforms like Windows Azure detects hardware failures and automatically moves application code to a new machine so that application remains available to clients. Geo-replication and easy manageability of data is there on Windows Azure Platform. Load Balancing like options are there in which roles can be divided into multiple instances in order to ensure data safety in case of failures. Replication of data is there on the cloud such that if one replica fails, others are still viable which maintains the reliability of data. SQL Azure helps in maintaining scalability by automatically moving databases from heavily accessed machines to other machines.

But still there are issues which need to be resolved like latency time issues while retrieving data, failure and unavailability of all replicas of storage. Performance can be slow when complex queries will be executed on distributed environment. Replication of data will not be a viable approach when the size of data increases. Application needs to be developed which must be capable of identifying the possibility of failures and techniques to adjust it through error handling or through other techniques.

REFERENCES

[7]. http://go.microsoft.com/fwlink/?LinkID=159357&lcid=409
[13]. www.microssoftpdc.com