***Review : Spanner Google’s Globally-Distributed Database***

***Problem:***

* Spanner is the first system to distribute data at global scale and support externally consistent distributed transactions.
* It is a Temporal database with multi-version support
* Non blocking reads
* Atomic schema changes

***Why the problem hard***

* For a distributed database system one of the most critical and difficult problem is to provide data consistency and accuracy across various nodes.
* Critical features that are difficult to implement in a distributed database:
  + Externally consistent reads and writes
  + Globally consistent reads across the database at a timestamp.

***Why is it important***

* As business expand with regional offices and presence in different geographical regions – there is a need for data centers that are locally present in these regions
* This is needed in order to process the clients request within the SLA.
* Need to replicate data across data centers and have robust failover mechanism

***Main idea:***

* At the highest level - its is a database that is sharded across global data centers
* replication used for global availability and geographical locality
* automatic failover between replicas
* automatically reshard’s data as the number of servers/ amount of data changes
* automatically migrates data across machines to load balance in response to failures.
* designed to scale upto millions of rows across hundreds to datacenters and trillions of database rows

***Design :***

* Consists of Spanservers - responsible for 100 to 1000 instances of tablet
* A tablet - maintains a bag of mappings: (key:String, timestamp:int64) -> String
* state maintained in B-tree like structures + WAL(Write Ahead log) : all on Colossus
* Replication is supported by maintaining Paxos state machine on top of the tablet.
* Paxo state machine is used to implement a consistently replicated bag of mappings

*Bucketing/Directory :*

* On top of KV mappings - an abstraction called "bucketing" is implemented
* Directory is a unit of data placement
* When data is moved between Paxos - it is moved by directory (can be moved to load balance, put freqently used directory in same group)
* Data moved as a background task to not impact performance of reads /write
* Two phase commits

*Data Model:*

* Driven by schematized semi-relational tables
* Support a Query Language
* General purpose transaction ( with 2-phase commits)
* data model is not purely relational , in that the rows must have names which makes it look like Key-value store.

*TrueTime API*

* Is the most critical component - which provides stronger time semantics
* This is the basis of Spanner distributed algorithm
* Designed to provide with bounded time uncertainty
* Underlying time references used by TrueTime are GPS and atomic clocks
* It uses two forms as they have different failure modes.
* Implemented by a set of timemaster machines per data center and time slave machines per machine

***Experimental Setup:***

* Use of Spanner in FI ( Google's Revenue critical database)
* FI originally had with MySql - with manual sharing
* knowledge of sharing in application business logic
* Any resharding exercise was impractical to do on a regular basis as it was too expensive

Using Spanner was helped as

* could automatically shard
* replication and automatic failure
* need for strong transactional semantics which means - it could not use NOSql db.
* Time stamp semantics helped to keep a in memory data structure based of the database state.

***Some Next steps:***

- Automate maintenance of

- secondary indicies

- resharding based of load

- Need to evaluate if the complexity/sensitivity of TrueTime API is really needed for all real world distributed systems.