Megastore is scalable storage for online services. The system, distributed over several data centers, has a

very large capacity, 1PBin 2011, and it is highly available.Megastore is widely used internally at Google;

it handles some 23 billion transactions daily: 3 billion write and 20 billion read transactions [37].

The basic design philosophy of the system is to partition the data into entity groups and replicate

each partition independently in data centers located in different geographic areas. The system supports

full ACID semantics within each partition and provides limited consistency guarantees across partitions

(see Figure 8.12). Megastore supports only those traditional database features that allow the system to

scale well and that do not drastically affect the response time.

Another distinctive feature of the system is the use of the Paxos consensus algorithm, discussed in

Section 2.11, to replicate primary user data, metadata, and system configuration information across data

centers and for locking. The version of the Paxos algorithm used by Megastore does not require a single

master. Instead, any node can initiate read and write operations to a write-ahead log replicated to a

group of symmetric peers.

The entity groups are application-specific and store together logically related data. For example, an

email account could be an entity group for an email application. Data should be carefully partitioned

to avoid excessive communication between entity groups. Sometimes it is desirable to form multiple

entity groups, as in the case of blogs [37].

The middle ground between traditional and NoSQL databases taken by the Megastore designers is

also reflected in the data model. The data model is declared in a schema consisting of a set of tables

composed of entries, each entry being a collection of named and typed properties. The unique primary

key of an entity in a table is created as a composition of entry properties. A Megastore table can be

a root or a child table. Each child entity must reference a special entity, called a root entity in its root

table. An entity group consists of the primary entity and all entities that reference it.

The system makes extensive use of BigTable. Entities from different Megastore tables can be mapped

to the same BigTable row without collisions. This is possible because the BigTable column name is a

concatenation of the Megastore table name and the name of a property. A BigTable row for the root

entity stores the transaction and all metadata for the entity group. As we saw in Section 8.9, multiple

versions of the data with different time stamps can be stored in a cell. Megastore takes advantage of

this feature to implement multi-version concurrency control (MVCC); when a mutation of a transaction

occurs, this mutation is recorded along with its time stamp, rather than marking the old data as obsolete and adding the new version. This strategy has several advantages: read and write operations can

proceed concurrently, and a read always returns the last fully updated version.

A write transaction involves the following steps: (1) Get the timestamp and the log position of

the last committed transaction. (2) Gather the write operations in a log entry. (3) Use the consensus

algorithm to append the log entry and then commit. (4) Update the BigTable entries. (5) Clean up.