BigTable is a distributed storage system developed by Google to store massive amounts of data and to

scale up to thousands of storage servers [73]. The system uses the Google File System discussed in

Section 8.5 to store user data as well as system information. To guarantee atomic read and write

operations, it uses the Chubby distributed lock service (see Section 8.7); the directories and the files in

the namespace of Chubby are used as locks.

The system is based on a simple and flexible datamodel. It allows an application developer to exercise

control over the data format and layout and reveals data locality information to the application clients.

Any read or write row operation is atomic, even when it affects more than one column. The column

keys identify column families, which are units of access control. The data in a column family is of the

same type. Client applications written in C++ can add or delete values, search for a subset of data, and

look up data in a row.

A row key is an arbitrary string of up to 64 KB, and a row range is partitioned into tablets serving as

units for load balancing. The time stamps used to index various versions of the data in a cell are 64-bit

integers; their interpretation can be defined by the application, whereas the default is the time of an

event in microseconds. A column key consists of a string defining the family name, a set of printable

characters, and an arbitrary string as qualifier.

The organization of a BigTable (see Figure 8.11) shows a sparse, distributed, multidimensional map

for an email application. The system consists of three major components: a library linked to application

clients to access the system, a master server, and a large number of tablet servers. The master server

controls the entire system, assigns tablets to tablet servers and balances the load among them, manages

garbage collection, and handles table and column family creation and deletion.

Internally, the space management is ensured by a three-level hierarchy: the root tablet, the location

of which is stored in a Chubby file, points to entries in the second element, the metadata tablet, which,

in turn, points to user tablets, collections of locations of users’ tablets. An application client searches

through this hierarchy to identify the location of its tablets and then caches the addresses for further use.

The performance of the system reported in [73] is summarized in Table 8.2. The table shows the

number of random and sequential read and write and scan operations for 1, 000 bytes, when the

number of servers increases from 1 to 50, then to 250, and finally to 500. Locking prevents the system

from achieving a linear speed-up, but the performance of the system is still remarkable due to a fair

number of optimizations. For example, the number of scans on 500 tablet servers is 7,843/2   103

instead of 15,385/2 103. It is reported that only 12 clusters use more than 500 tablet servers, whereas

some 259 clusters use between 1 and 19 tablet servers.

BigTable is used by a variety of applications, including Google Earth, Google Analytics, Google

Finance, andWeb crawlers. For example, Google Earth uses two tables, one for preprocessing and one

for serving client data. The preprocessing table stores raw images; the table is stored on disk because

it contains some 70 TB of data. Each row of data consists of a single image; adjacent geographic

segments are stored in rows in close proximity to one another. The column family is very sparse;

it contains a column for every raw image. The preprocessing stage relies heavily on MapReduce to clean and consolidate the data for the serving phase. The serving table stored on GFS is “only” 500 GB,

and it is distributed across several hundred tablet servers, which maintain in-memory column families.

This organization enables the serving phase of Google Earth to provide a fast response time to tens of

thousands of queries per second.

Google Analytics provides aggregate statistics such as the number of visitors to aWeb page per day.

To use this service, Web servers embed a JavaScript code into their Web pages to record information

every time a page is visited. The data is collected in a raw-click BigTable of some 200 TB, with a row for

each end-user session. A summary table of some 20 TB contains predefined summaries for a Website.