

Spanner

Spanner: Google's Globally Distributed Database

JAMES C. CORBETT, JEFFREY DEAN, MICHAEL EPSTEIN, ANDREW FIKES, CHRISTOPHER FROST, J. J. FURMAN, SANJAY GHEMAWAT, ANDREY GUBAREV, CHRISTOPHER HEISER, PETER HOCHSCHILD, WILSON HSIEH, SEBASTIAN KANTHAK, EUGENE KOGAN, HONGYI LI, ALEXANDER LLOYD, SERGEY MELNIK, DAVID MWAURA, DAVID NAGLE, SEAN QUINLAN, RAJESH RAO, LINDSAY ROLIG, YASUSHI SAITO, MICHAL SZYMANIAK, CHRISTOPHER TAYLOR, RUTH WANG, and DALE WOODFORD, Google, Inc.

Spanner is Google's scalable, multiversion, globally distributed, and synchronously replicated database. It is the first system to distribute data at global scale and support externally-consistent distributed transactions. This article describes how Spanner is structured, its feature set, the rationale underlying various design decisions, and a novel time API that exposes clock uncertainty. This API and its implementation are critical to supporting external consistency and a variety of powerful features: nonblocking reads in the past, lock-free snapshot transactions, and atomic schema changes, across all of Spanner.

Categories and Subject Descriptors: H.2.4 [Database Management]: Systems—Concurrency, distributed databases, transaction processing

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Distributed databases, concurrency control, replication, transactions, time management

ACM Reference Format:

Corbett, J. C., Dean, J., Epstein, M., Fikes, A., Frost, C., Furman, J. J., Ghemawat, S., Gubarev, A., Heiser, C., Hochschild, P., Hsieh, W., Kanthak, S., Kogan, E., Li, H., Lloyd, A., Melnik, S., Mwaura, D., Nagle, D., Quinlan, S., Rao, R., Rolig, L., Saito, Y., Szymaniak, M., Taylor, C., Wang, R., and Woodford, D. 2013. Spanner: Google's globally distributed database. *ACM Trans. Comput. Syst.* 31, 3, Article 8 (August 2013), 22 pages.

DOI: <http://dx.doi.org/10.1145/2491245>

Spanner: globally consistent

A ground-breaking attempt at building a truly worldwide database system.

“Spans” the entire globe: data physically located in any data center anywhere on the planet should be seamlessly accessible, and **consistent**.

NoSQL approaches like BigTable can store data across multiple datacenters, but ensuring the data is consistent in all locations (i.e., that someone in data-center A examining the local copy of cell C will always see the same as someone looking at the local copy of cell C in data-center B) is very hard.

Consistency an issue due to transmission latencies between data centers.

Spanner: globally consistent

A ground-breaking attempt at building a truly worldwide database system.

“Spans” the entire globe: data physically located in any data center anywhere on the planet should be seamlessly accessible, and **consistent**.

NoSQL approaches like BigTable can store data across multiple datacenters, but ensuring the data is consistent in all locations (i.e., that someone in data-center A examining the local copy of cell C will always see the same as someone looking at the local copy of cell C in data-center B) is very hard.

Consistency an issue due to transmission latencies between data centers.

Synchronising servers over a wide area is notoriously difficult.

Timestamps usually done via Network Time Protocol (NTP): online service connecting servers to official/certified atomic-clock global time signals.

July 2012: several web services go down due to (routine, expected) NTP subtraction of a leap-second: Java and Linux systems did not anticipate this.

Spanner

A g
“Sp
any
No
dat
son
the
Co
Syn
Usu
ser
July
sub

Browser address bar: 'Leap Second' Bug Wreaks | x
www.wired.com/2012/07/leap-second-bug-wreaks-havoc-with-java-linux/

WIRED GEAR SCIENCE ENTERTAINMENT BUSINESS SECURITY DESIGN OPINION MAGAZINE VIDEO INSIDER SUBSCRIBE

SCALABLE. RELIABLE. SECURE.
EMAIL INFRASTRUCTURE FROM MANDRILL
SUBSCRIBE GIVE A GIFT RENEW INTERNATIONAL ORDERS

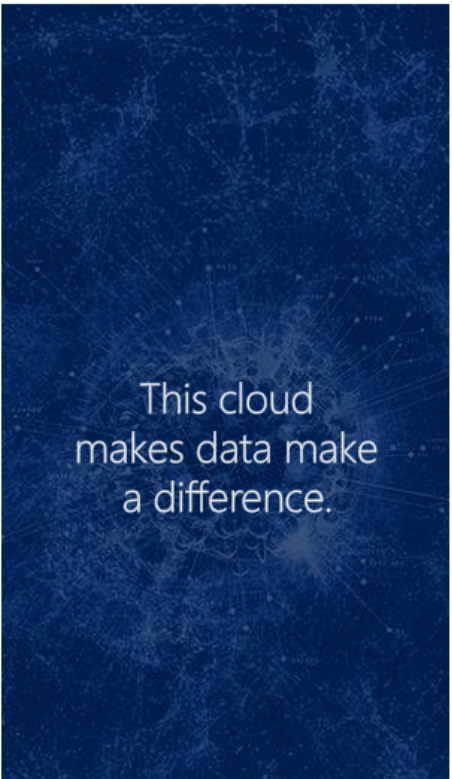

ENTERPRISE | Foursquare Gawker leap second LinkedIn

FOLLOW WIRED [Twitter] [Facebook] [RSS]

'Leap Second' Bug Wreaks Havoc Across Web

BY CADE METZ 07.01.12 | 3:48 AM | PERMALINK

[Facebook Share] 6 [Twitter Tweet] 0 [Google+ +1] 432 [LinkedIn Share] [Pinterest Pin it]



This cloud makes data make a difference.

Spanner: Google's own global time signal

For Spanner, Google ignored NTP and developed **TrueTime**, its own global time-reference service.

Spanner data-centers are equipped with their own atomic clocks, and with GPS receivers to get time reference from GPS signals.

The clocks and GPS Rx feed time data to a number of master servers; the masters disseminate time data to all other server machines in the DC.

Each server runs a daemon that constantly checks with the masters in the local DC, and in other DCs, to get time reference signals and to compute from this a consensus view on what the real time is.

This means that data can be committed to DCs at two different locations – potentially on opposite sides of the planet – and the common clock means that there is no dispute about which one happened first.

Global coordination with minimal inter-DC communication.

Traditional approaches involved a **lot** more inter-process communication.

Spanner

Using Spanner, Google can accurately replicate data across multiple data-centers, with globally synchronised timestamps: with TrueTime the data replicas, the mirror images, are time-ordered and globally consistent too.

So Spanner gives greater system resilience as any outage can quickly be addressed by firing up the most recent replica/mirror image.

Can also reduce latency: if one replica is getting a lot of hits and a backlog is building up, requests can be diverted to another replica; and/or choose to use the most local replica (in terms of distance, or network latency).

Spanner automatically shards data across data-centers around the world, and automatically reshards data as the amount of data or number of servers changes; it automatically load-balances, moving data across machines and across data centers to balance load and deal with failures.

Up-front costs for clocks and GPS tech are in the thousands of dollars: spread across the data-center on a per-server basis they are very cheap.

Spanner

“Spanner’s main focus is managing cross-datacenter replicated data, but we have also spent ... time in designing and implementing important database features on top of our distributed-systems infrastructure. Even though many projects happily use BigTable... we have also consistently received complaints that BigTable can be difficult to use for some kinds of applications: those that have complex, evolving schemas, or those that want strong consistency in the presence of wide-area replication. ... Spanner has evolved from a BigTable-like versioned key-value store into a temporal multiversion database... data is stored in schematized semirelational tables; data is versioned, and each version is automatically timestamped with its commit time; old versions of data are subject to configurable garbage-collection policies; and applications can read data at old timestamps. Spanner supports general-purpose transactions, and provides an SQL-based query language.”

(Corbett *et al.* 2013, 8:2)

Spanner

Novel features:

- Applications given fine-grain dynamic control of data replication configs
 - (control where data is located, which data centers, and how many replicas).

Spanner

Novel features:

- Applications given fine-grain dynamic control of data replication configs
 - (control where data is located, which data centers, and how many replicas).
- Externally consistent reads and writes.
- Globally consistent reads across the database at a timestamp.

Spanner

Novel features:

- Applications given fine-grain dynamic control of data replication configs
 - (control where data is located, which data centers, and how many replicas).
- Externally consistent reads and writes.
- Globally consistent reads across the database at a timestamp.

Spanner

Novel features:

- Applications given fine-grain dynamic control of data replication configs
 - (control where data is located, which data centers, and how many replicas).
- **Externally consistent** reads and writes.
- Globally consistent reads across the database at a timestamp.

External Consistency. External consistency guarantees that a transaction will always receive current information. Using the concepts we have just introduced, we can provide a formal definition of external consistency. The actual time order in which transactions complete defines a unique serial schedule. This serial schedule is called the *external schedule*. A system is said to provide external consistency if it guarantees that the schedule it will use to process a set of transactions is equivalent to its external schedule.

Spanner

Novel features:

- Applications given fine-grain dynamic control of data replication configs
 - (control where data is located, which data centers, and how many replicas).
- Externally consistent reads and writes.
- Globally consistent reads across the database at a timestamp.

“These features enable Spanner to support consistent backups, consistent MapReduce executions, and atomic schema updates, all at global scale, and even in the presence of ongoing transactions.” (Corbett et al. 2013, 8:2).

Spanner

“The key enabler of these properties is a new TrueTime API and its implementation. The API directly exposes clock uncertainty, and the guarantees on Spanner’s timestamps depend on the bounds that the implementation provides. If the uncertainty is large, Spanner slows down to wait out that uncertainty. Google’s cluster-management software provides an implementation of the TrueTime API. This implementation keeps uncertainty small (generally less than 10ms) by using multiple modern clock references (GPS and atomic clocks). Conservatively reporting uncertainty is necessary for correctness; keeping the bound on uncertainty small is necessary for performance.” (Corbett et al. 2013, 8:2-8:3).

FI