Megastore: Providing Scalable, Highly Available Storage for Interactive Services

J. Baker, C. Bond, J.C. Corbett, JJ Furman, A. Khorlin,
J. Larson, J-M Léon, Y. Li, A. Lloyd, V. Yushprakh
Google Inc.

CIDR 2011, Jan. 12 2011

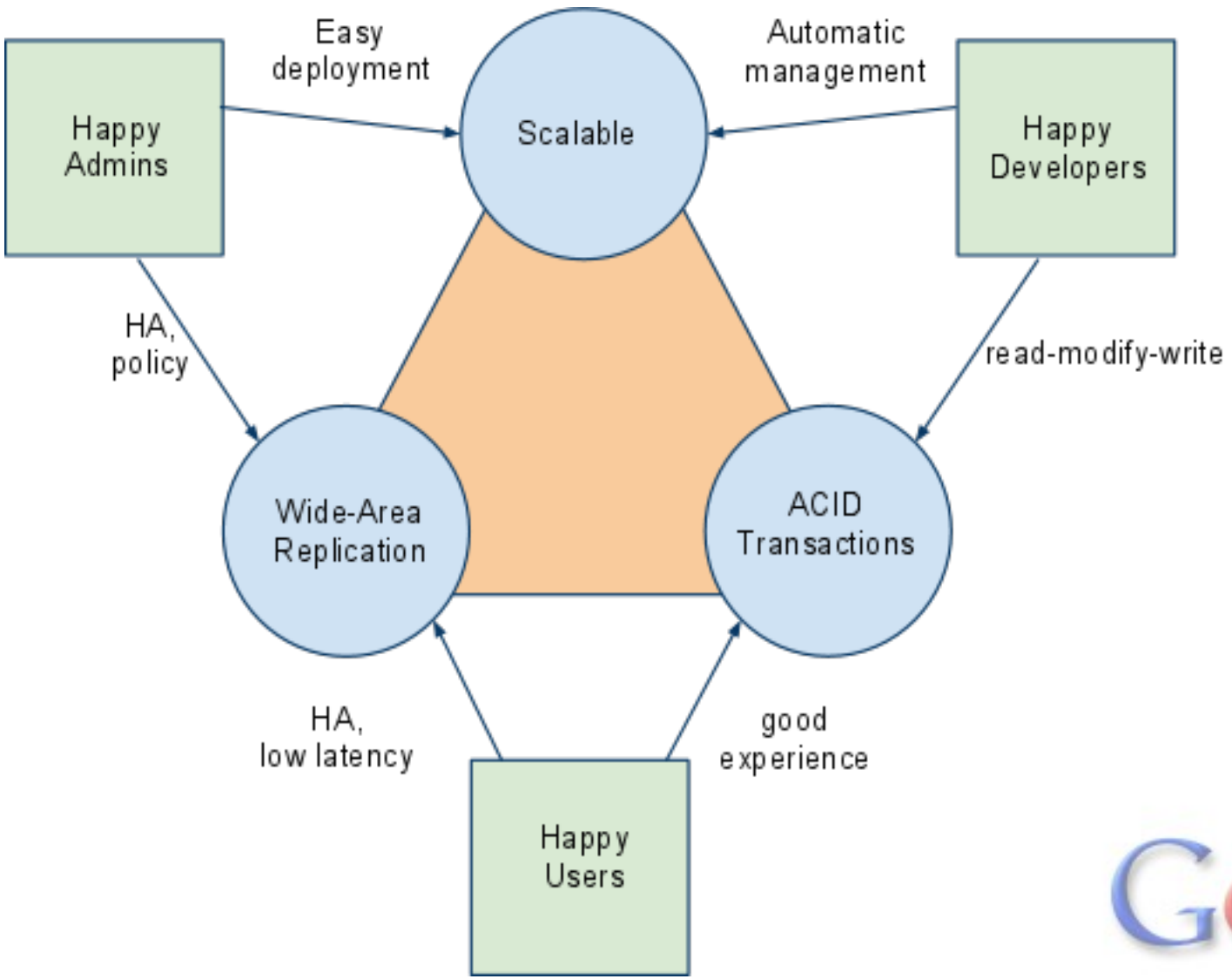
Google™

With Great Scale Comes Great Responsibility

- A billion Internet users
  - Small fraction is still huge
- Must please users
  - Bad press is expensive - never lose data
  - Support is expensive - minimize confusion
  - No unplanned downtime
  - No planned downtime
  - Low latency
- Must also please developers, admins

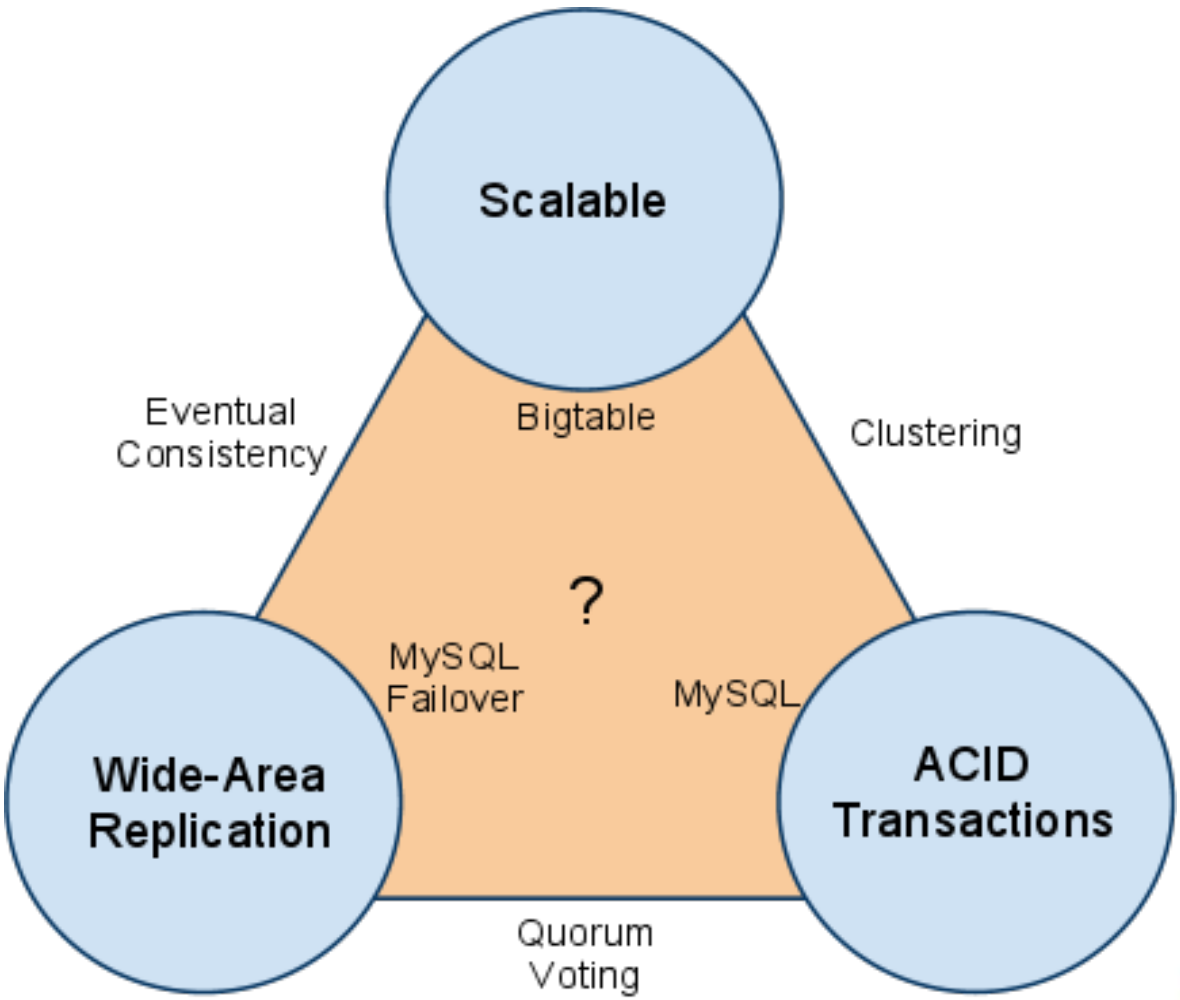
Google™

Making Everyone Happy



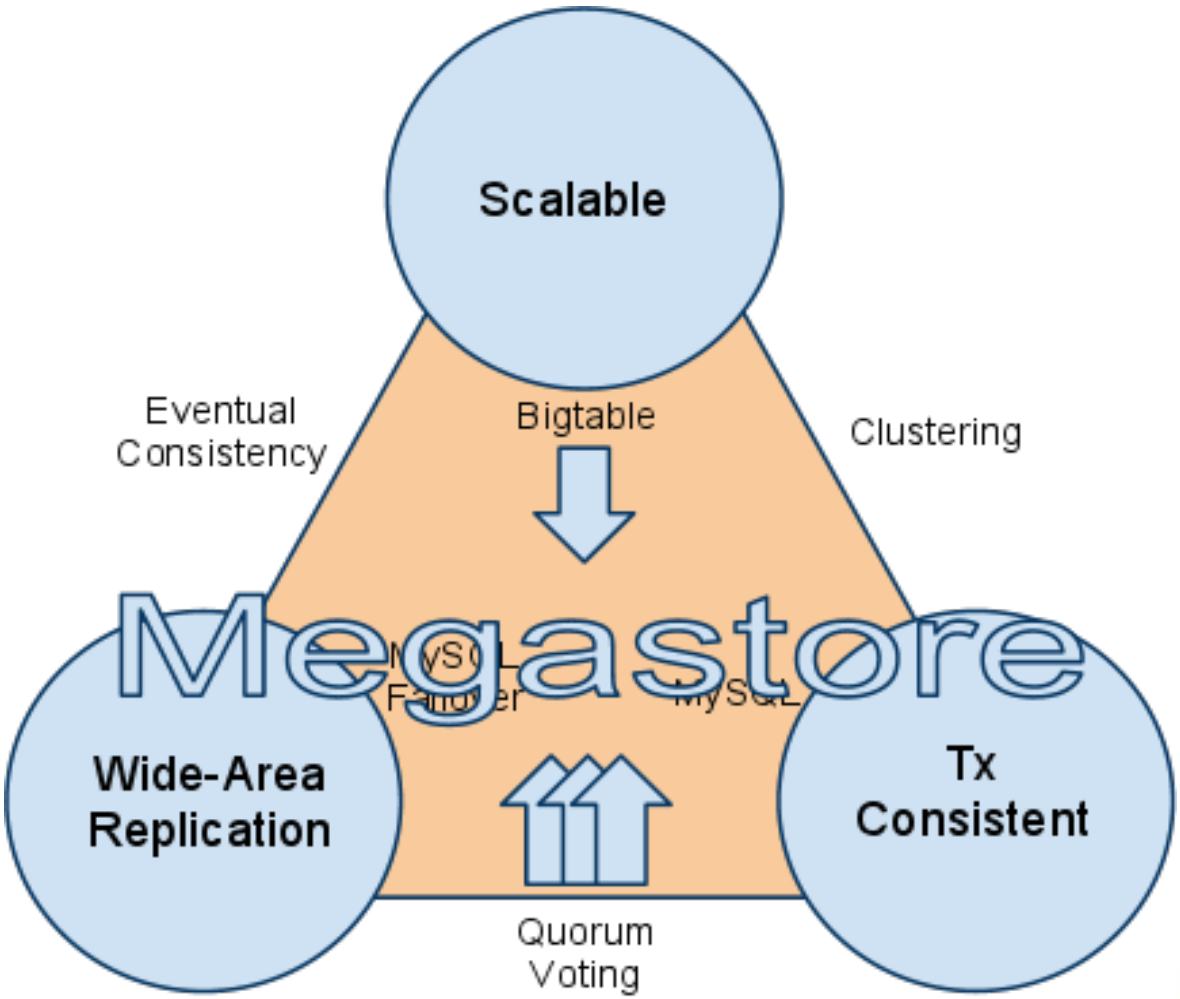
Google™

Technology Options



Google™

Technology Options



Google™

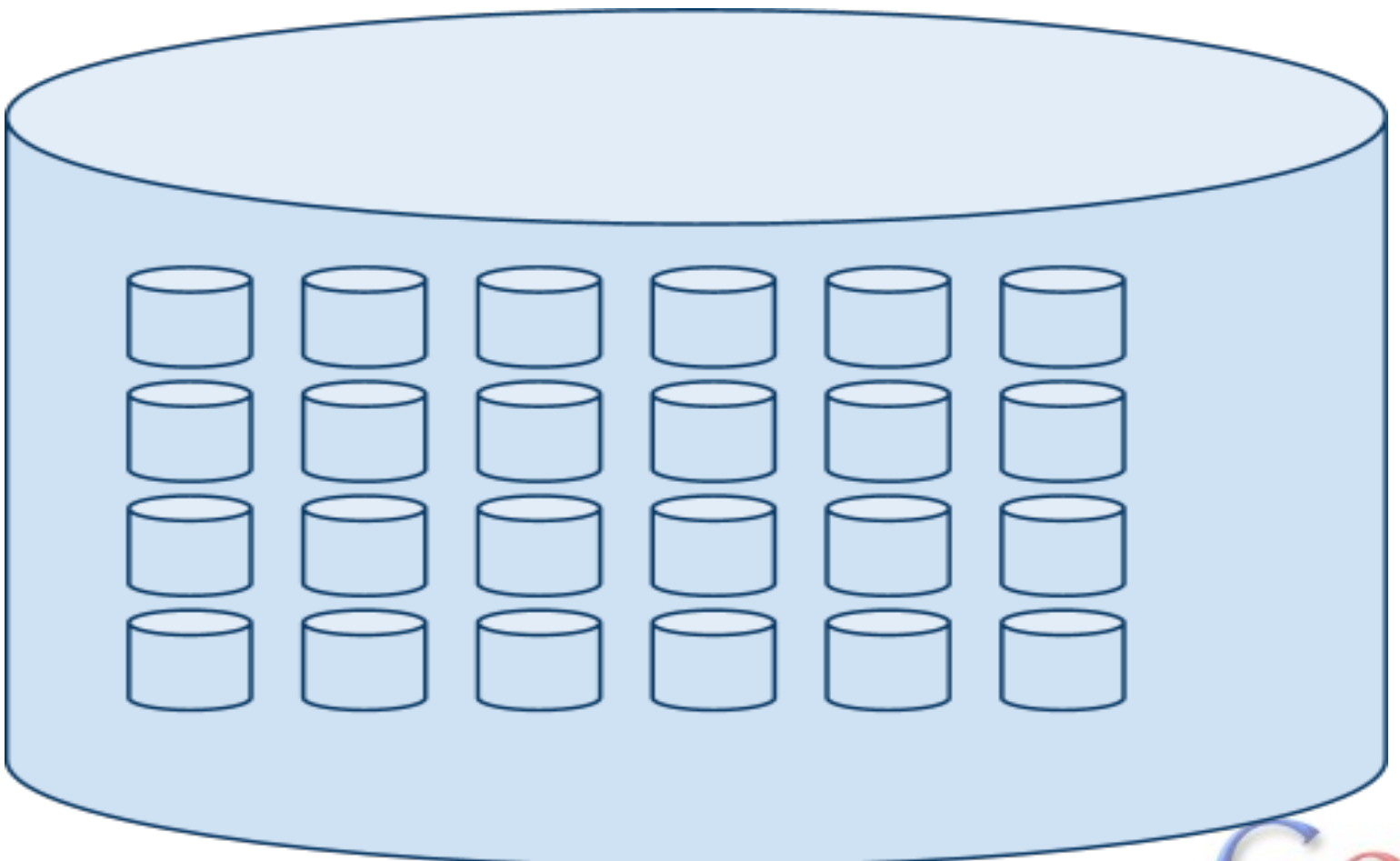
Megastore

- Started in 2006 for app development at Google
- Service layered on:
  - Bigtable (NoSQL scalable data store per datacenter)
  - Chubby (Config data, config locks)
- Turnkey scaling (apps, users)
- Developer-friendly features
- Wide-area synchronous replication
  - partition by "Entity Group"

Google™

Entity Groups

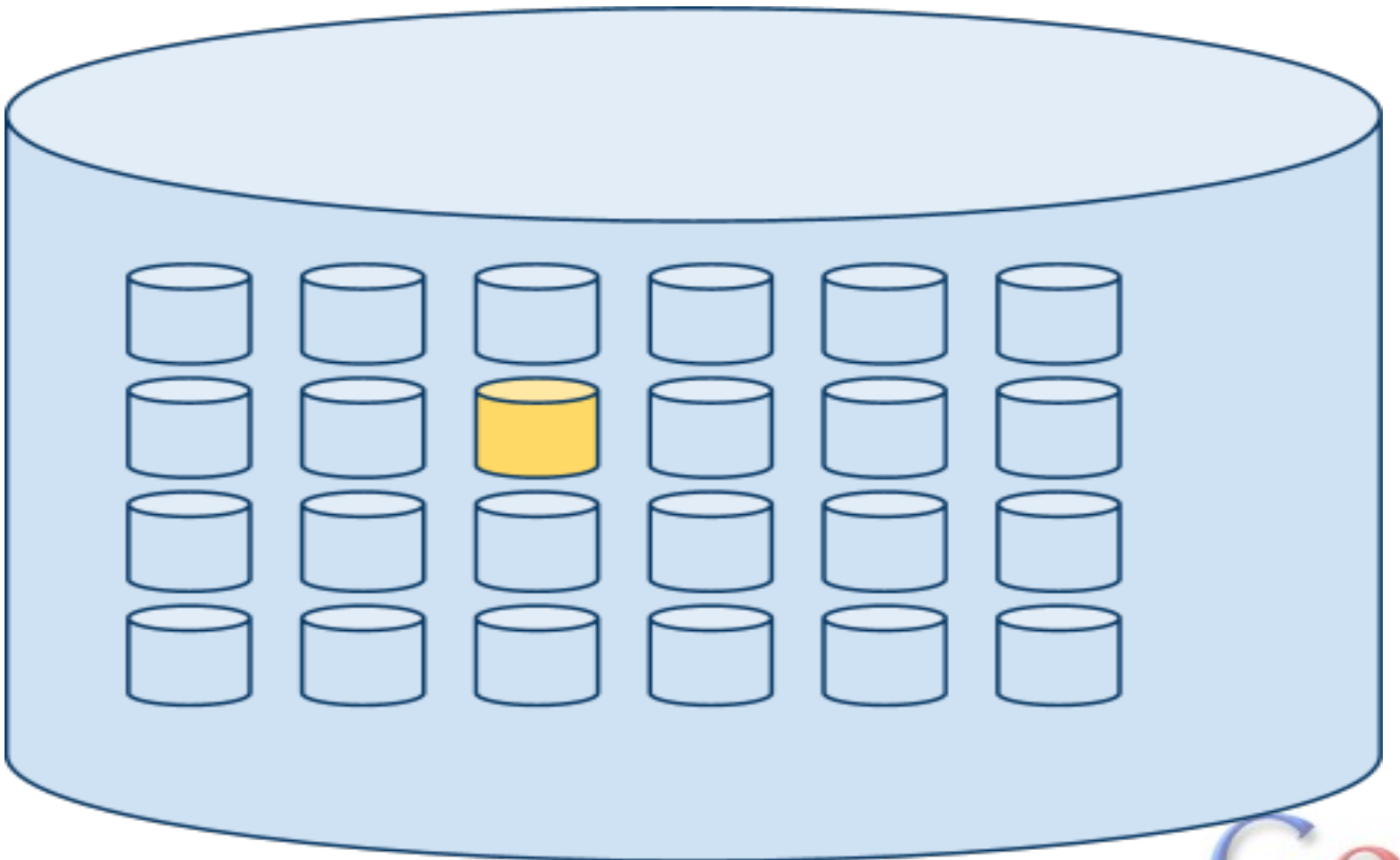
Entity Groups are sub-databases



Google™

Entity Groups

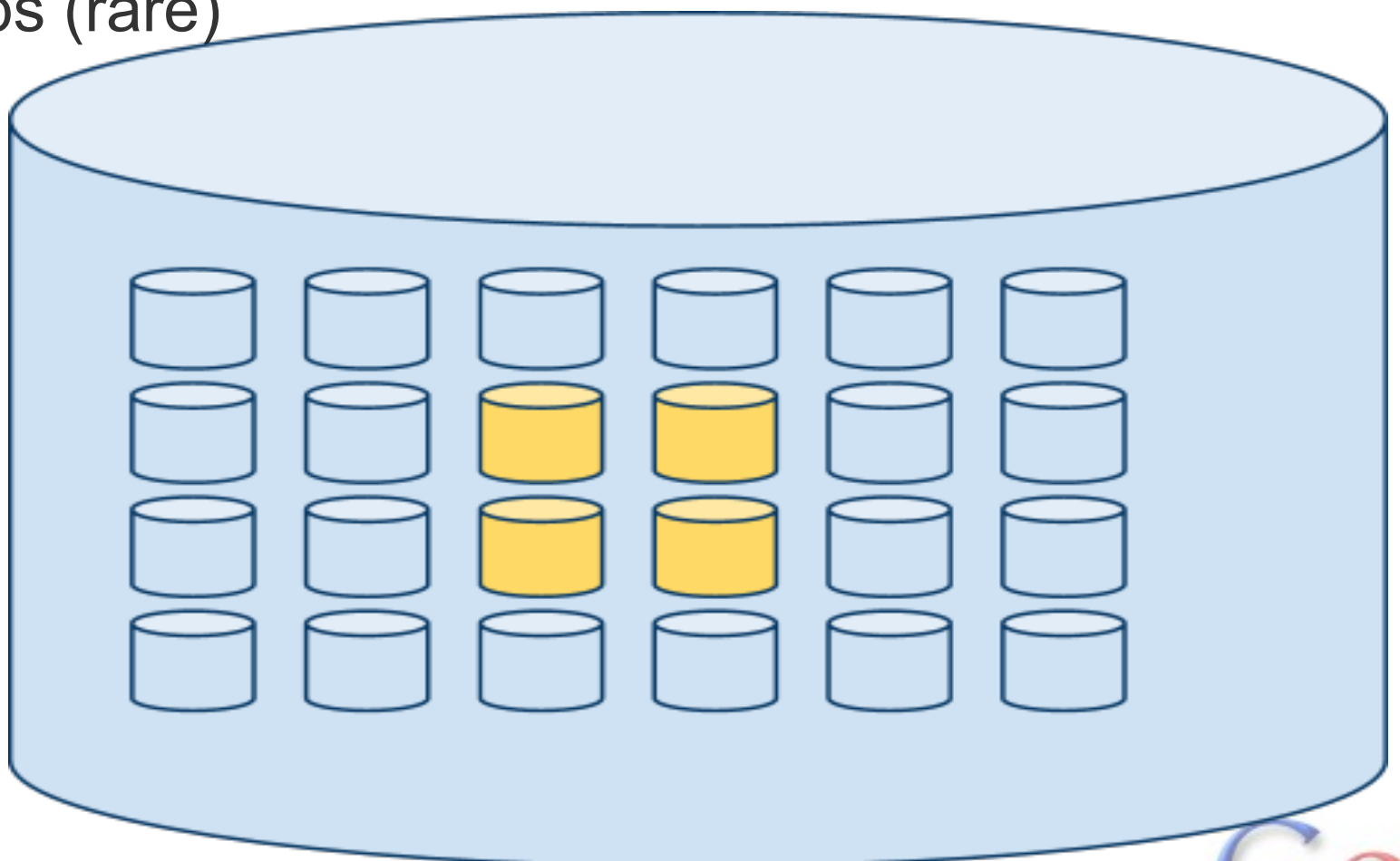
Cheap transactions within an entity group (common)



Google™

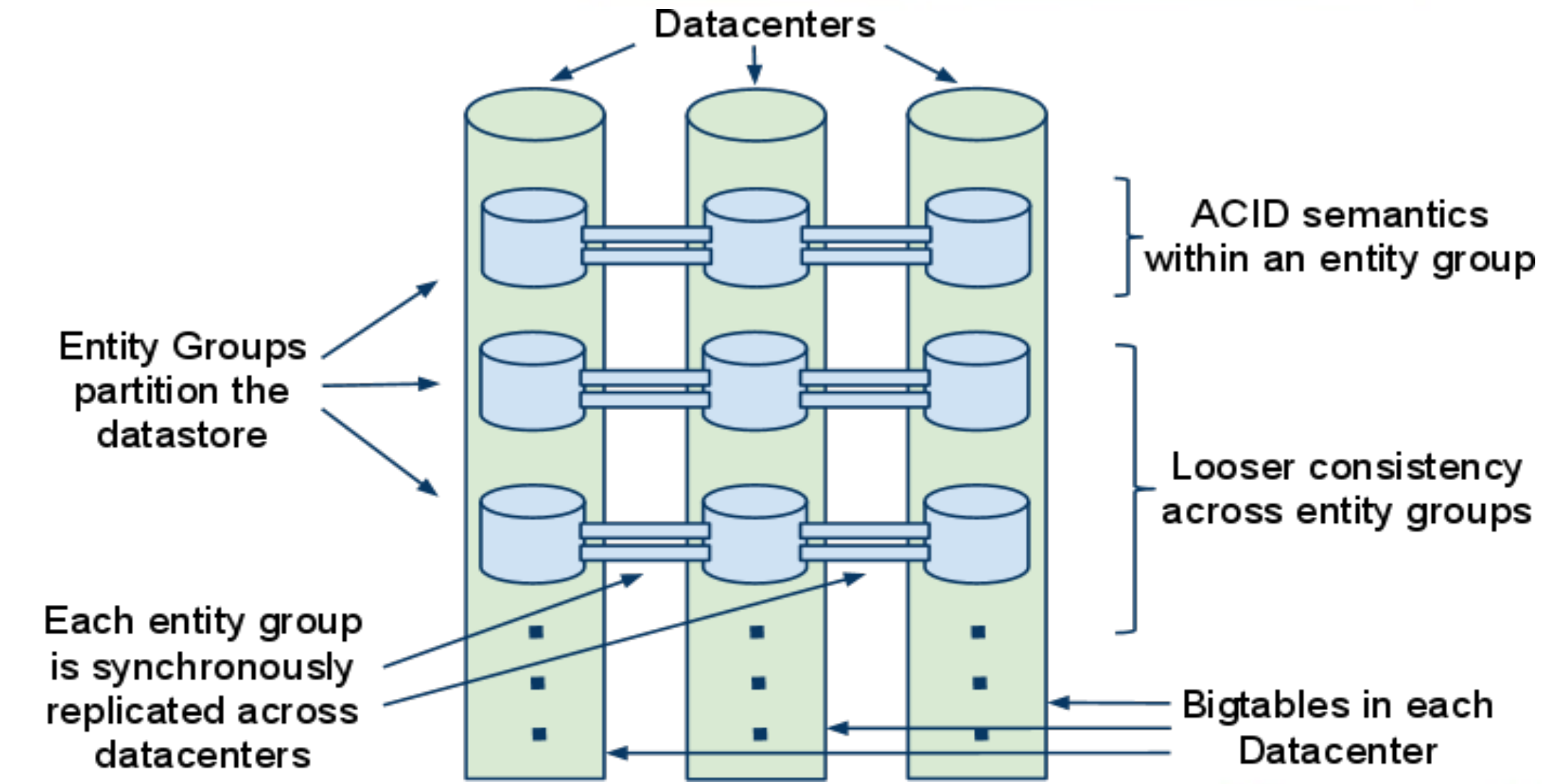
Entity Groups

Expensive or loosely-consistent operations across Entity Groups (rare)



Google™

Scale Axis vs. Wide Replication Axis



Google™

Entity Group Mapping Examples

- Applications must choose their partitioning
- Common operations within an EG

Application	Entity Groups	Cross-EG Operations
Email	User accounts	none (out-of-system)
Blogs	Users, Blogs	Access control, notifications, global indexes
Mapping	Local patches	Patch-spanning ops (2PC)
Social	Users, Groups	Messages, bi-directional relationships, notifications
Resources	Sites	Shipments

Achieving Technical Goals

Scale

- Bigtable within a datacenter
- Easy to add Entity Groups (storage, throughput)

ACID Transactions

- Write-ahead log per Entity Group
- 2PC or Queues between Entity Groups

Wide-Area Replication

- Paxos
- Tweaks for optimal latency

Google™

Paxos: Quorum-based Consensus

"While some consensus algorithms, such as Paxos, have started to find their way into [large-scale distributed storage systems built over failure-prone commodity components], their uses are limited mostly to the maintenance of the global configuration information in the system, not for the actual data replication."

-- Lamport, Malkhi, and Zhou, May 2009

Google™

Paxos: Megastore Tweaks

- Replicates transaction log entries on each write
- Writes: one WAN round-trip (avg.)
- Strong Reads: zero WAN round-trips (avg.)
  - per-replica bitmap invalidated on faults
- Reads/Writes from any replica (no master)
  - no pipelining: limited per-EG throughput
  - batching will improve throughput
- Background scanners finish all operations

Google™

Comparison with Other Approaches

NoSQL	**Megastore**	**RDBMS**
Minimal features	Scalable features	Full-featured
Highly scalable	Highly scalable	Medium scale with effort
PK lookup and scan	Indexes, scans, physical clustering	Storage abstraction, complex query planning and execution
Limited/eventual consistency	Partitioned consistency	Global consistency

Google™

Features

- Declarative schema
- Serializable Transactions (within Entity Group)
- Queues and 2PC (between Entity Groups)
- Indexes
  - declared fields
  - full-text
- Online backup and restore
- Built-in encryption and compression

Google™

Omissions (current)

- (currently) No query language
  - Apps must implement query plans
  - Apps have fine-grained control of physical placement
- (currently) Limited per-Entity Group update rate

Google™

Is Everybody Happy?

Admins

- linear scaling, transparent rebalancing (Bigtable)
- instant transparent failover
- symmetric deployment

Developers

- ACID transactions (read-modify-write)
- many features (indexes, backup, encryption, scaling)
- single-system image makes code simple
- little need to handle failures

End Users

- fast up-to-date reads, acceptable write latency
- consistency

Google™

Take-Aways

- Sync WAN replication on each write
- Constraints acceptable to most apps
  - EG partitioning
  - High write latency
  - Limited per-EG throughput
- Turnkey scaling achieved
  - >100 apps
  - >3 billion writes/day
  - >20 billion reads/day
  - ~1PB data (before index, replication)
  - Most apps get carrier-grade (five 9's) availability
- In production use for over 4 years

Google™

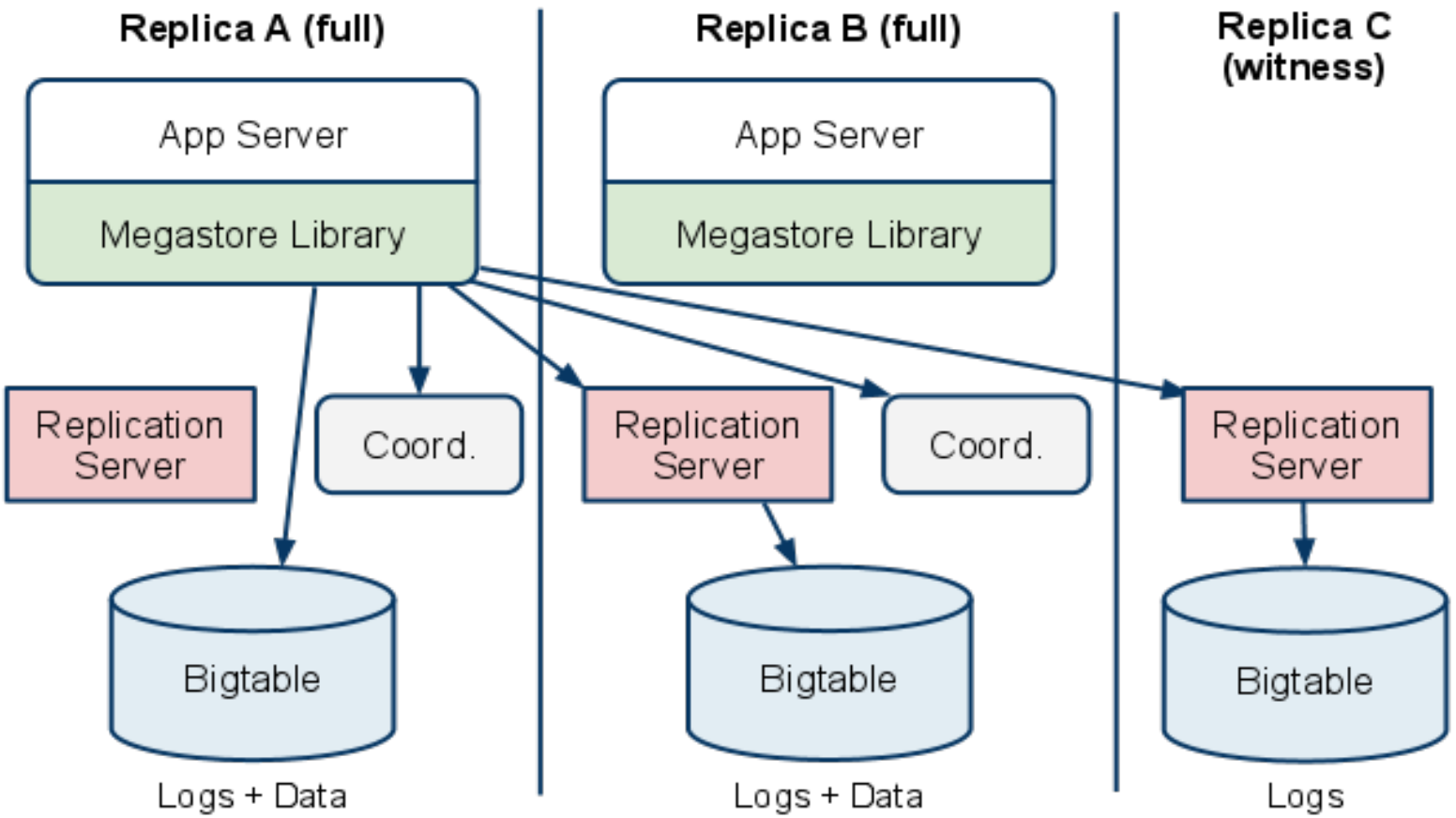
For more information

- Read our full paper
- Become a Megastore customer:
  - Use Google App Engine ("high replication")
- Ask a question...

Google™

Extra Slides

Megastore Architecture



Google™

Why Not Lots of RDBMS's?

- Functional
  - Need a place to store global and full-text indexes
- Space and Time
  - Create new local EG in ~10ms
  - Overhead of <1KB per EG
- Administration
  - Load-rebalancing
  - Fault recovery
  - Monitoring
  - Operational team

Google™

Schema

```
CREATE SCHEMA PhotoApp;

CREATE TABLE User {
 required int64 user_id;
 required string name;
} PRIMARY KEY(user_id), ENTITY GROUP ROOT;

CREATE TABLE Photo {
 required int64 user_id;
 required int32 photo_id;
 required int64 time;
 required string full_url;
 optional string thumbnail_url;
 repeated string tag;
} PRIMARY KEY(user_id, photo_id), IN TABLE User,
  ENTITY GROUP KEY(user_id) REFERENCES User;

CREATE LOCAL INDEX PhotosByTime ON Photo(user_id, time);
CREATE GLOBAL INDEX PhotosByTag ON Photo(tag) STORING (thumbnail_url);
```

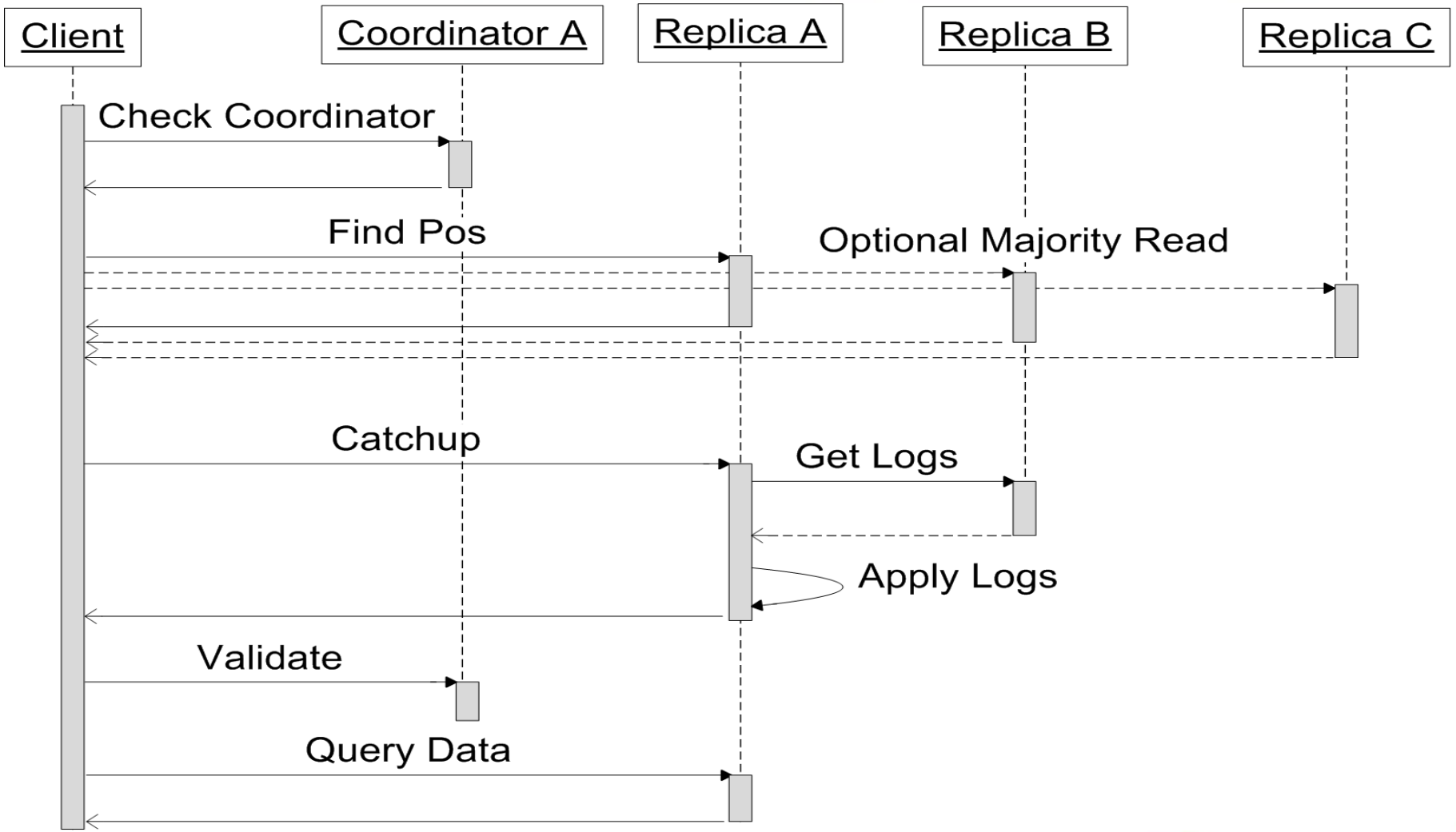
Google™

Locality

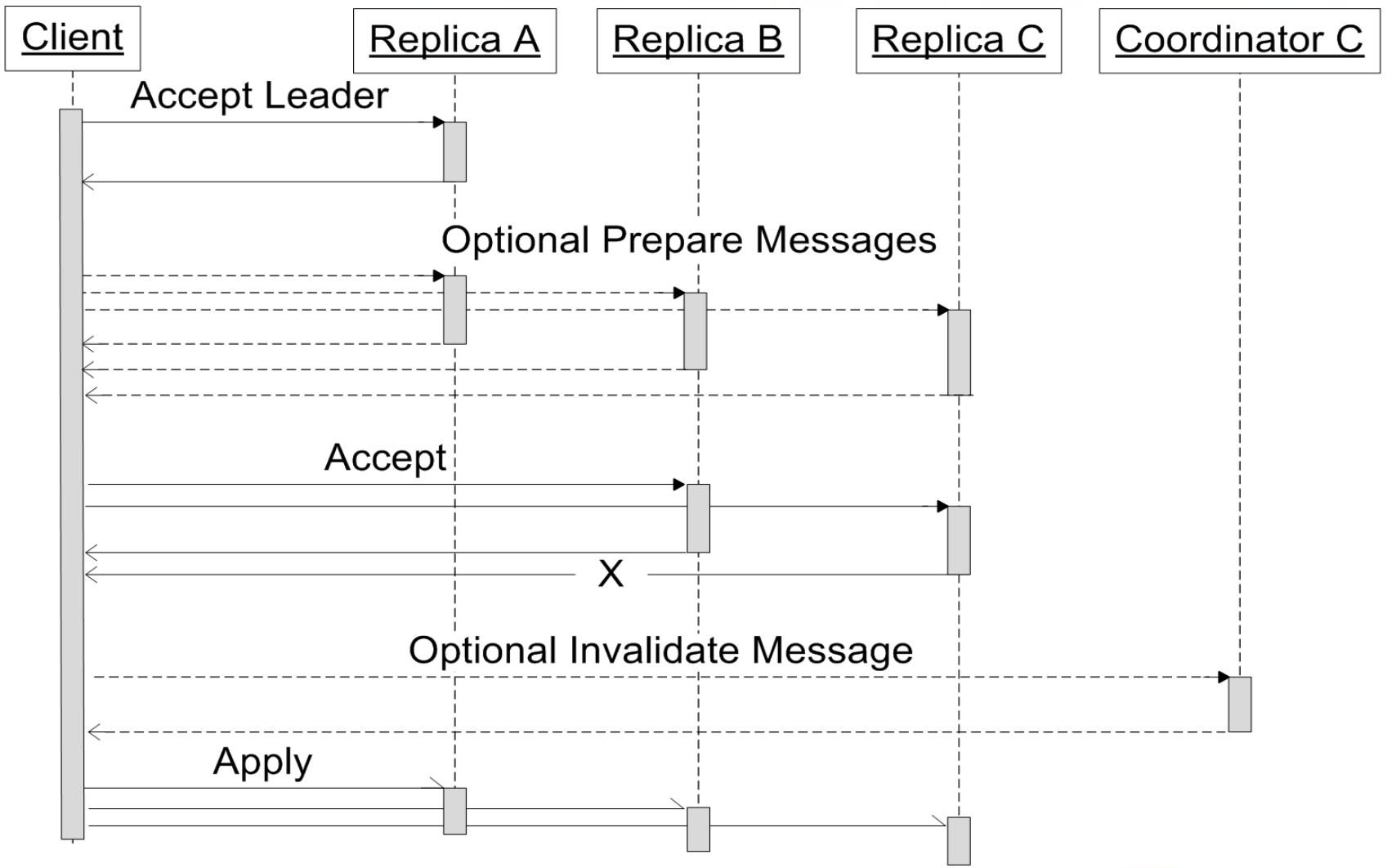
- Bigtable
  - column-oriented storage
  - faster access to nearby rows

Row key	User.name	Photo.time	Photo.tag	Photo.url	Photo.-I. PhotosByTime
101	John				
101,500		12:30:01	Dinner, Paris	http://...	
101,502		12:15:22	Betty, Paris	http://...	
101,12:15:22,502					X
101,12:30:01,500					X
102	Mary				

Timeline of read algorithm

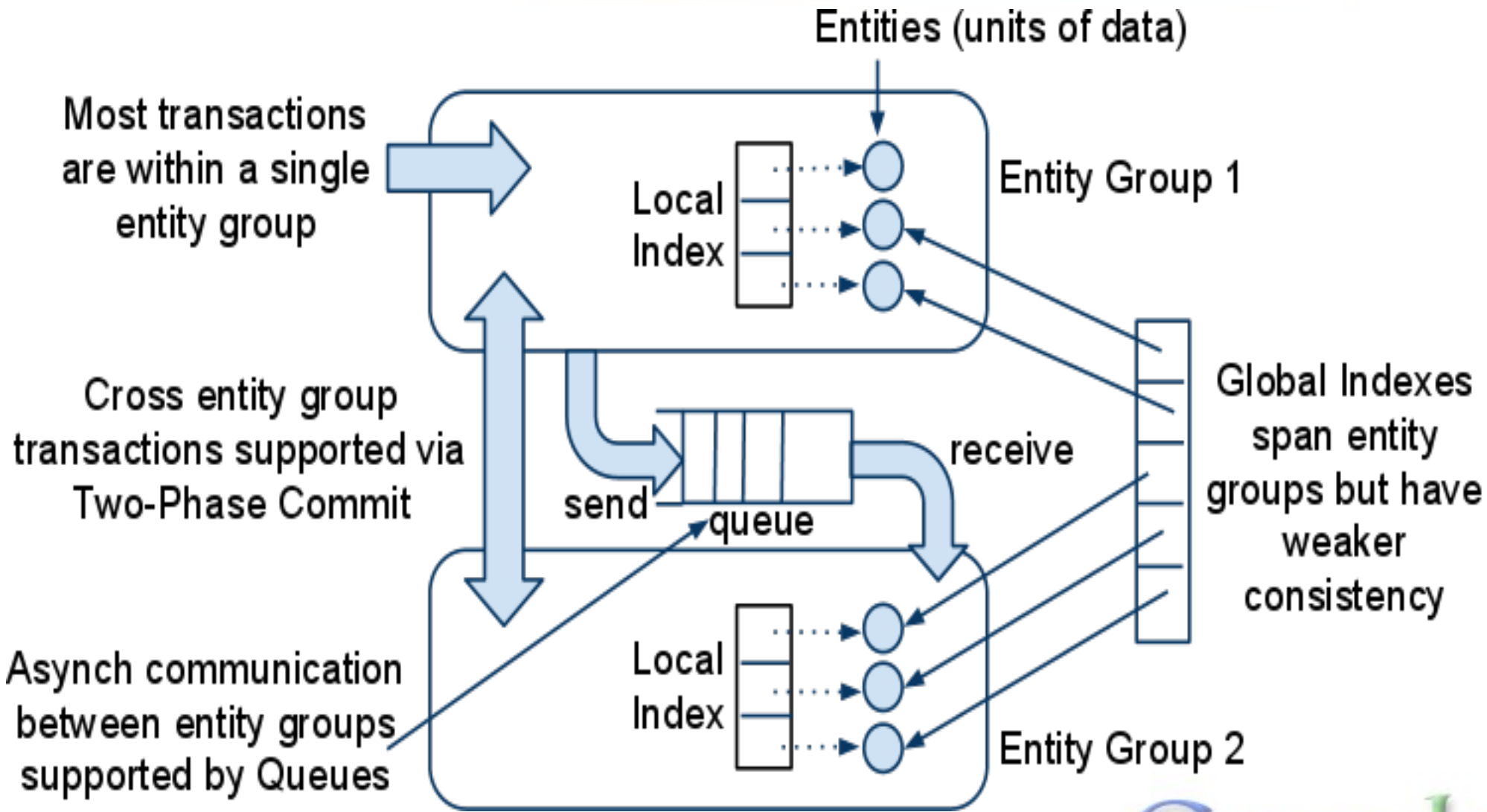


Timeline of write algorithm



Google™

Operations Across Entity Groups



Google™