

# TiDB: HBase分布式事务与SQL实现

# About me

- TiDB & Codis founder
- Golang expert
- Distributed database developer
- Currently, CEO and co-founder of PingCAP

[liuqi@pingcap.com](mailto:liuqi@pingcap.com)

<https://github.com/pingcap/tidb>

weibo: @goroutine

# Agenda

- HBase introduction
- TiDB features
- Google percolator and omid
- Internals of TiDB over HBase

# Features of HBase

- Linear and modular scalability.
- Strictly consistent reads and writes.
- Automatic failover support between RegionServers.
- Block cache and Bloom Filters for real-time queries.
- Query predicate push down via server side Filters
- MVCC

# What did they say ?

“Nothing is hotter than SQL-on-Hadoop, and now SQL-on-HBase is fast approaching equal hotness status”

Form **HBaseCon 2015**

**We want more !**

**SQL + Transaction(ACID)**

# TiDB Features

- **Consistent distributed transactions**
  - TiDB makes your application code simple and robust.
- **Compatible with MySQL protocol**
  - Use TiDB as distributed MySQL.
  - Replace MySQL with TiDB to power your application without changing a single line of code in most cases.
- **Focus on OLTP**
  - There are lots of OLAP system( Spark, Presto, Impala...)

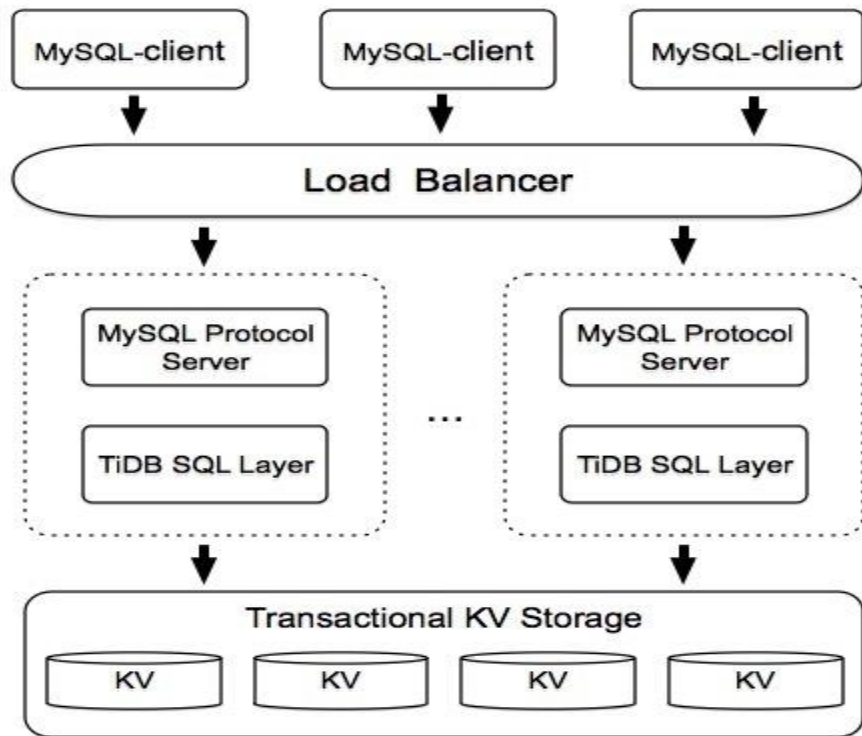
# TiDB Features

- **Multiple storage engine support**
  - TiDB supports most of the popular storage engines in single-machine mode. You can choose from goleveldb, LevelDB, RocksDB, LMDB, BoltDB and even more to come.
- **Written in Go**
  - Faster develop
  - Run fast



# Why called TiDB ?

# TiDB Architecture



## AH. HBase

- First things first
  - Need to build a transactional layer over HBase

# Google percolator

- Design
  - BigTable
  - Transactions
  - Timestamps

# Percolator

- **Three components**
  - Percolator worker
  - BigTable tablet server
  - GFS chunkserver

# Percolator

- **Transactions**

- ACID semantics
- **Snapshot-Isolation** (too weak for RDBMS)
- must maintain locks explicitly

# Percolator's Transactions

Bob wants to transfer 4\$ to Joe

Key	Bal: Data	Bal: Lock	Bal: Write
Bob	6: 5: \$10	6: 5:	6: data @ 5 5:
Joe	6: 5: \$2	6: 5:	6: data @ 5 5:

# Percolator's Transactions

Key	Bal: Data	Bal: Lock	Bal: Write
Bob	7: \$6 6: 5: \$10	7: I am Primary 6: 5:	7: 6: data @ 5 5:
Joe	6: 5: \$2	6: 5:	6: data @ 5 5:



# Percolator's Transactions

Key	Bal: Data	Bal: Lock	Bal: Write
Bob	7: \$6 6: 5: \$10	7: I am Primary 6: 5:	7: 6: data @ 5 5:
Joe	7: \$6 6: 5: \$2	7: Primary@Bob.bal 6: 5:	7: 6: data @ 5 5:

# Percolator's Transactions

Key	Bal: Data	Bal: Lock	Bal: Write
Bob	8: 7: \$6 6: 5: \$10	8: 7: I am Primary 6: 5:	8: data @ 7 7: 6: data @ 5 5:
Joe	8: 7: \$6 6: 5: \$2	8: 7: Primary@Bob.bal 6: 5:	8: data @ 7 7: 6: data @ 5 5:

# Percolator's Transactions

Key	Bal: Data	Bal: Lock	Bal: Write
Bob	8: 7: \$6 6: 5: \$10	8: 7: 6: 5:	8: data @ 7 7: 6: data @ 5 5:
Joe	8: 7: \$6 6: 5: \$2	8: 7: Primary @ Bob.bal 6: 5:	8: data @ 7 7: 6: data @ 5 5:

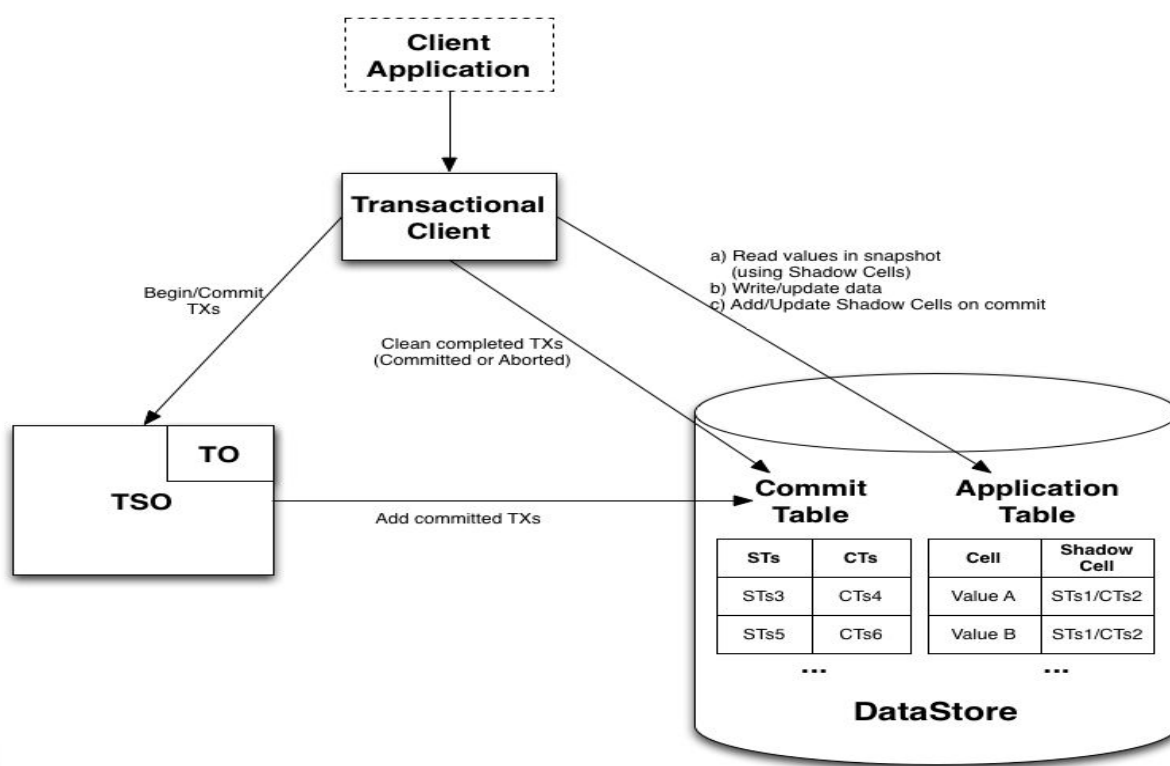
# Percolator's Transactions

Key	Bal: Data	Bal: Lock	Bal: Write
Bob	8: 7: \$6 6: 5: \$10	8: 7: 6: 5:	8: data @ 7 7: 6: data @ 5 5:
Joe	8: 7: \$6 6: 5: \$2	8: 7: 6: 5:	8: data @ 7 7: 6: data @ 5 5:

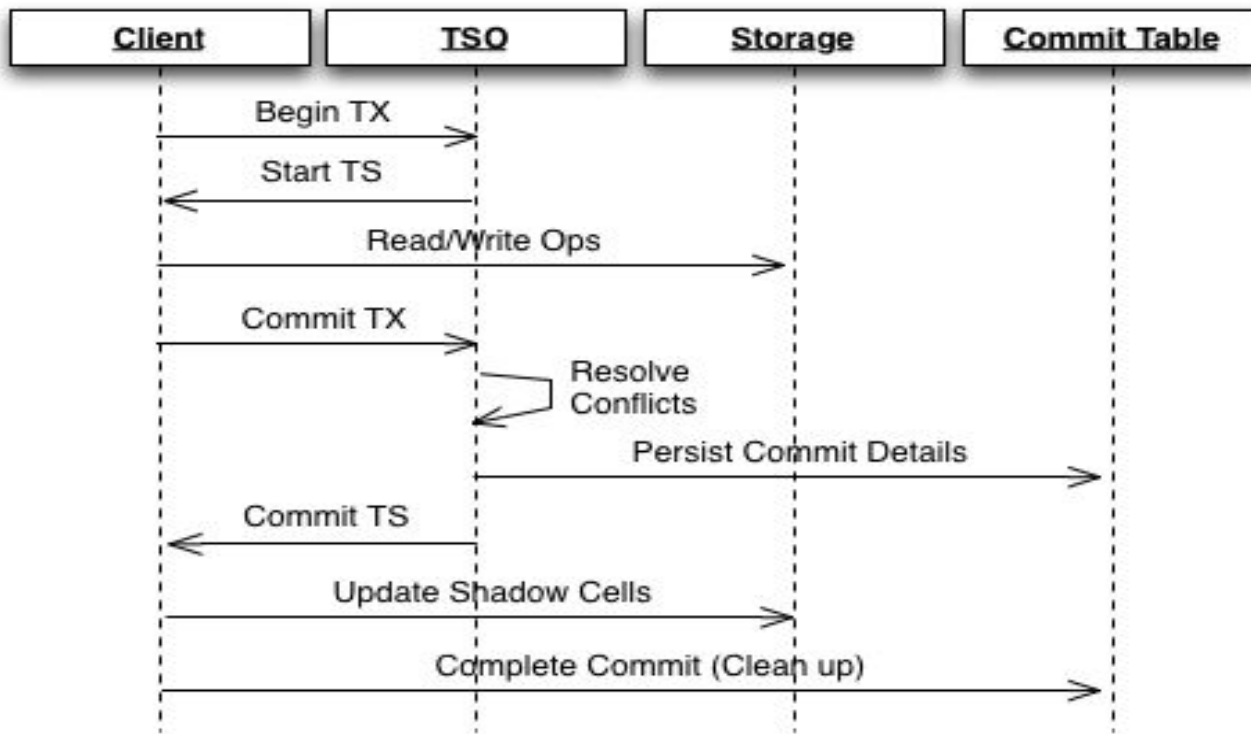
# Timestamp

- Timestamps in strictly increasing order.
- For efficiency, it batches writes, and "pre-allocates" a whole block of timestamps.
- How many timestamps do you think Google's timestamp oracle serves per second from 1 machine?
  - 2,000,000 / s

# Yahoo's OMID



# Yahoo's OMID



# Google Spanner

- ‘We wanted something that we were confident in. It’s a time reference that’s owned by Google.’
  - — Andrew Fikes



# Google Spanner

- With Spanner, Google discarded the NTP in favor of its own time-keeping mechanism
- TrueTime API
  - Atomic clocks
  - GPS (global positioning system) receivers

# Google F1

- Architecture

- Sharded Spanner servers
- data on GFS and in memory
- Stateless F1 server
- Pool of workers for query execution

- Features

- Relational schema
- Extensions for hierarchy and rich data types
- [Non-blocking schema changes](#)
- Consistent indexes
- Parallel reads with SQL or Map-Reduce

# Let's talk about SQL

# How does TiDB map SQL to KV

User table

RowID(hidden column)	name	email
1	bob	bob@gmail.com

Inside TiDB, each table, column has an unique ID

# How to map SQL to KV

Let assume ID of user table is 1, ID of name is 2, ID of email is 3

key (TableID : RowID : ColumnID)	value
1 : 1 : 1	nil
1 : 1 : 2	bob
1 : 1 : 3	bob@email.com

# How to map SQL to KV

**Example SQL:**

```
select name, email from user;
```

**Map to Key-Value (TableID : RowID : ColumnID):**

```
name := kv.Get( " 1 : 1 : 2 " )
```

```
email := kv.Get( " 1 : 1 : 3 " )
```

# How to map SQL to HBase

Example:

Key	列族:标识符	Value	列族:标识符	Value	列族:标识符	Value
row_1	cf:q	value	L:cf#q	lock_info	P:cf#q	startTs
row_2	cf:q	value			P:cf#q	startTs

```
lock_info {  
    startTS(version)  
    primary lock or secondary lock  
    pointer to primary lock (for secondary lock)  
}
```

# MySQL protocol support

- **Why does TiDB support MySQL protocol?**
  - **Testing**
  - **Community**
  - **Plenty of tools, easy to use**



# Current status

- **Able to run many famous applications**
  - WordPress, phpMyAdmin
  - ORM: Hibernate, SQLAlchemy ...
- **Asynchronous schema changes**
- **Active and growing community**
  - ~2800 star, 21 contributors within two month

# Thank you

## Q&A

<https://github.com/pingcap/tidb>

[liuqi@pingcap.com](mailto:liuqi@pingcap.com)

**We are hiring**