
Go in TiDB

Yao Wei | PingCAP

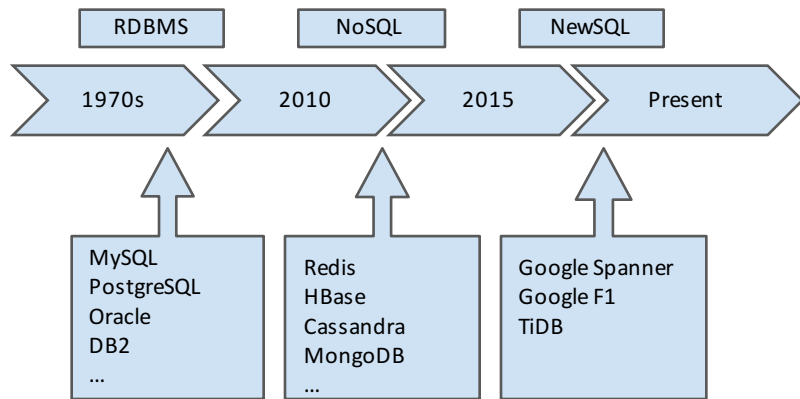
About me

- Yao Wei (姚维)
- TiDB Kernel Expert, General Manager of South Region, China
- 360 Infra team / Alibaba-UC / PingCAP
- Atlas/MySQL-Sniffer
- Infrastructure software engineer

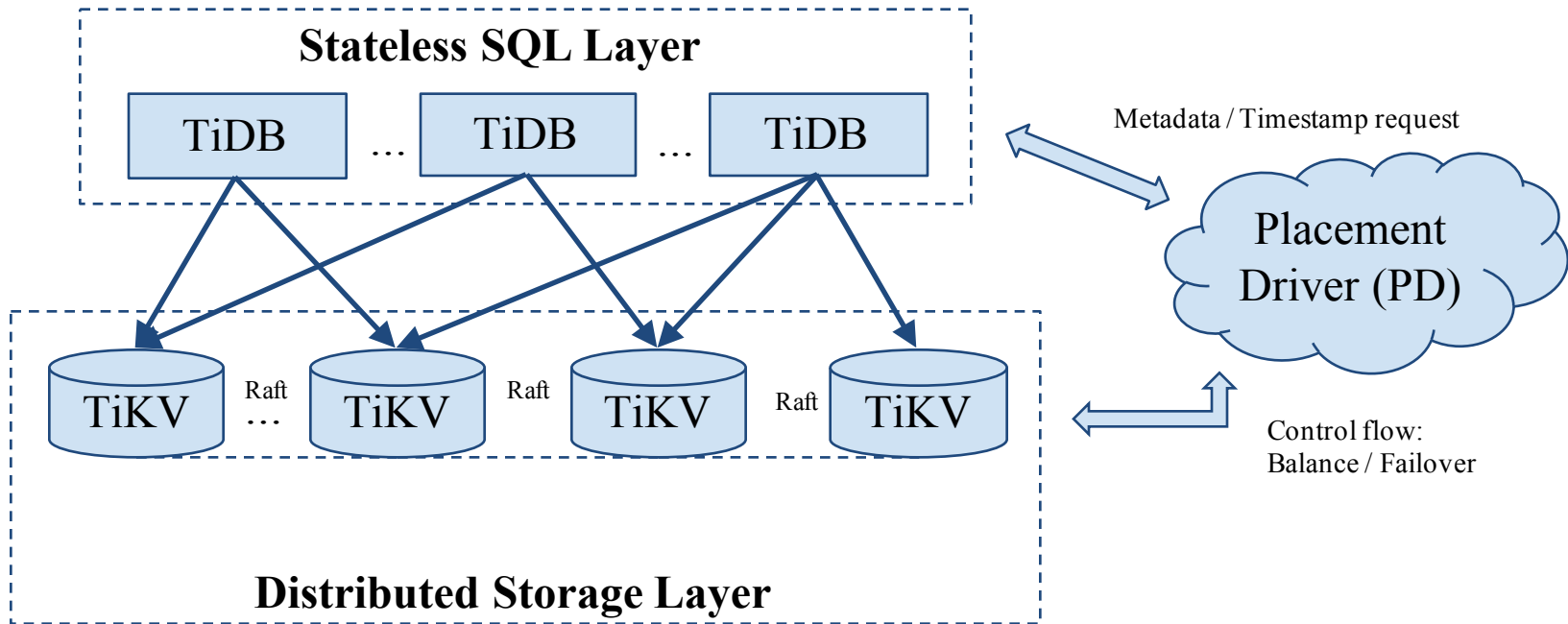
Why a new database?

Brief History

- Standalone RDBMS
- NoSQL
- Middleware & Proxy
- NewSQL

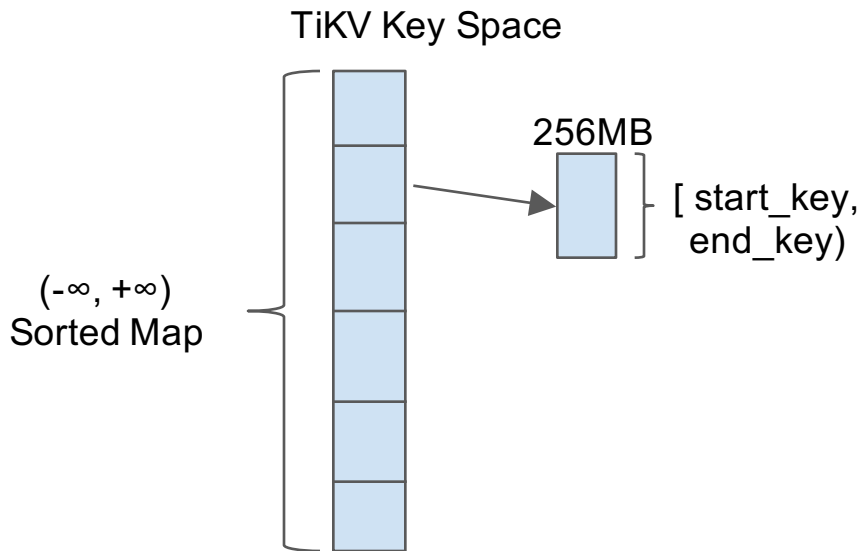
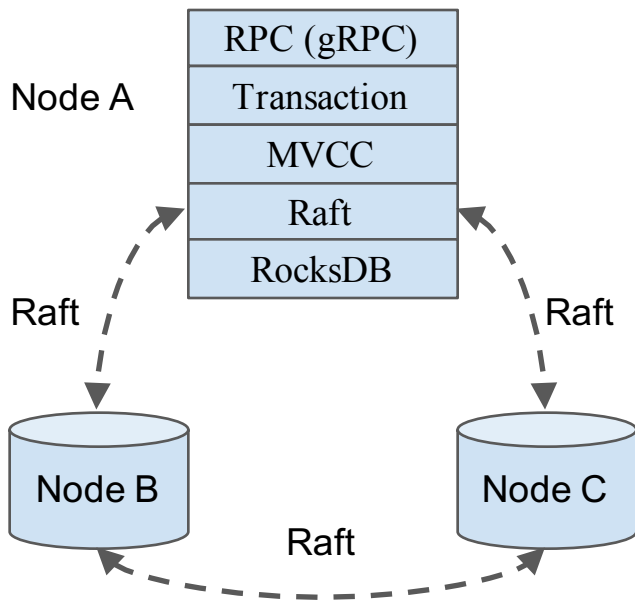


Architecture



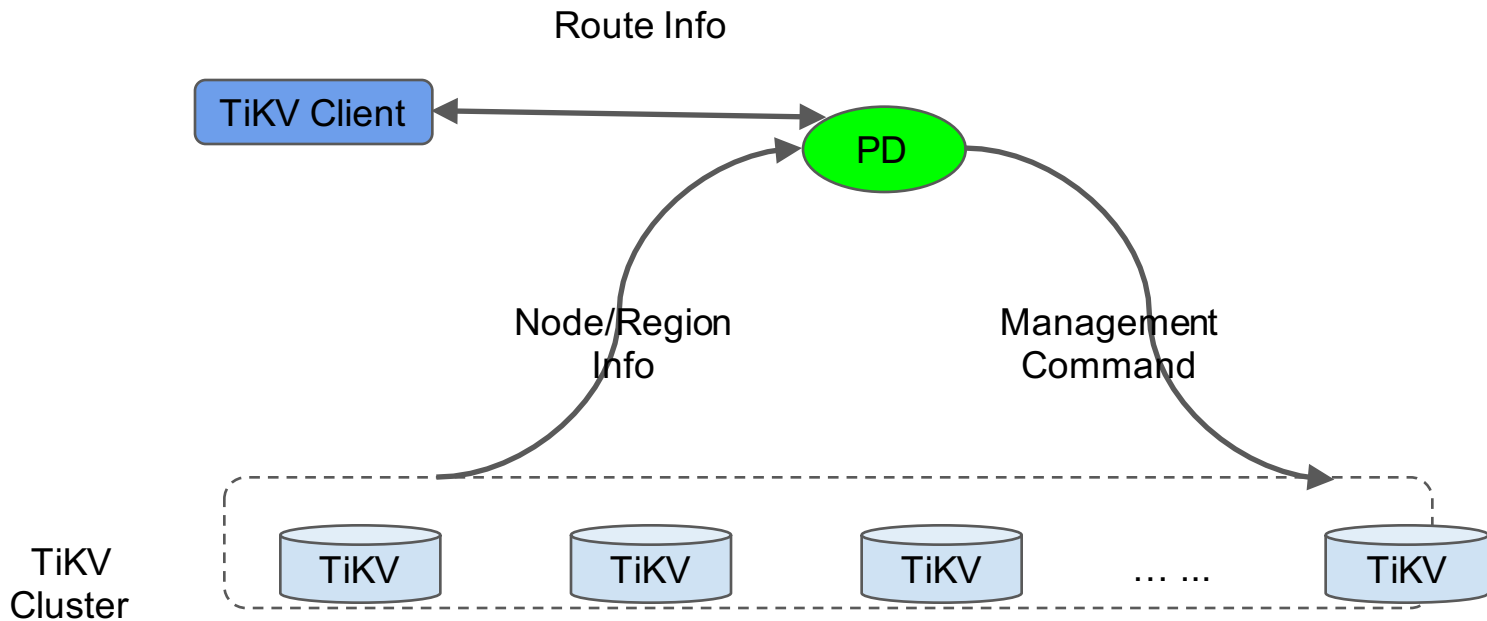
TiKV - Overview

- **Region: a set of continuous key-value pairs**
- **Data is organized/stored/replicated by Regions**
- **Highly layered**



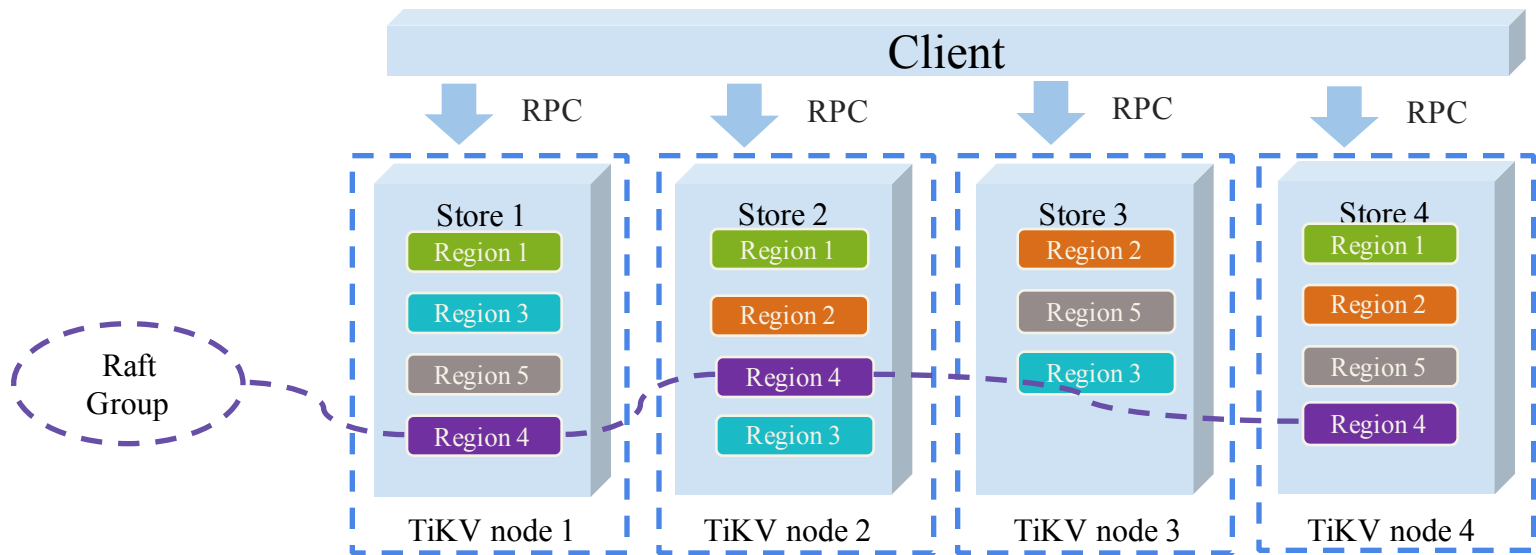
PD - Overview

- Meta data management
- Load balance management

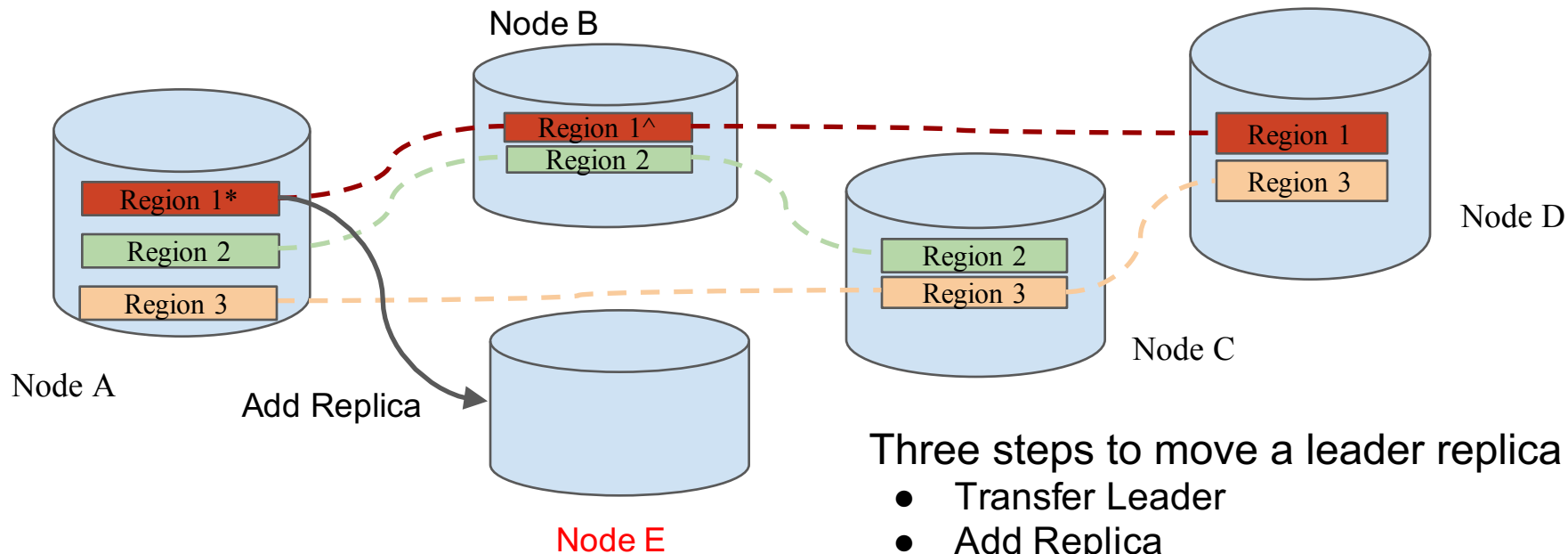


TiKV - Multi-Raft

Multiple raft groups in the cluster, one group for each region.



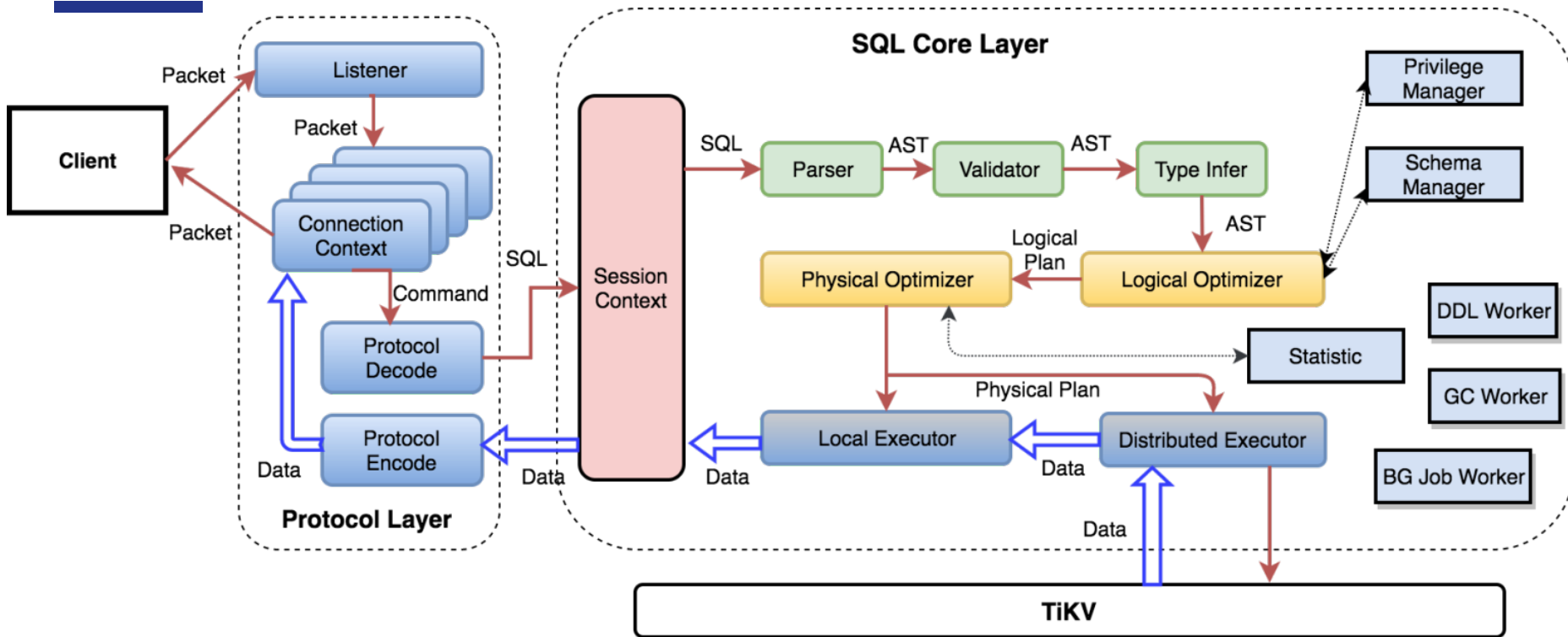
TiKV - Horizontal Scale



Three steps to move a leader replica

- Transfer Leader
- Add Replica
- Remove Replica

SQL Layer

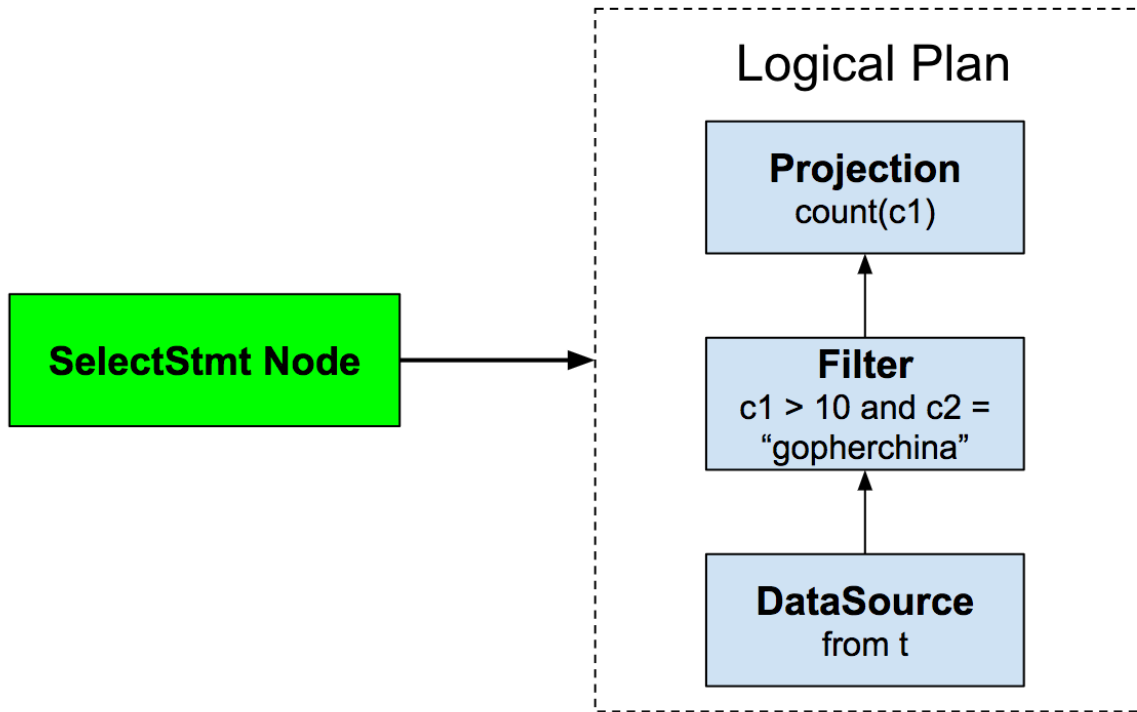


Example - SQL

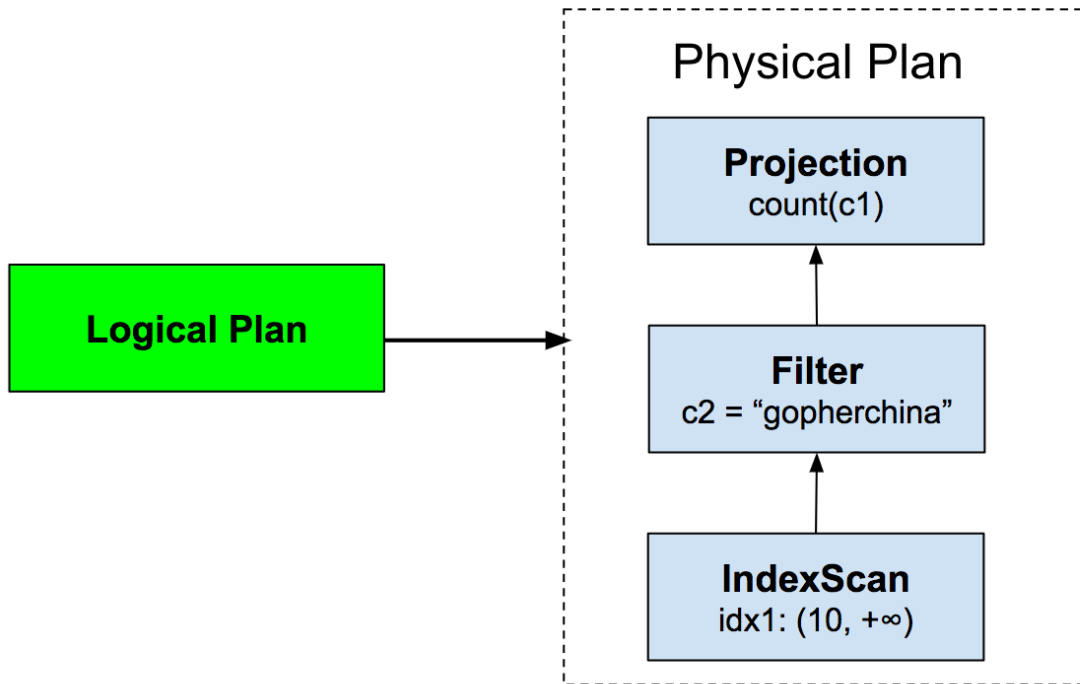
```
CREATE TABLE t (c1 INT, c2 VARCHAR(32), INDEX idx1 (c1));
```

```
SELECT COUNT(c1) FROM t WHERE c1 > 10 AND c2 =  
“gopherchina”;
```

Example - Logical Plan



Example - Physical Plan



Challenges of distributed ACID database?

- Distributed Database is very complex
- Lots of RPC work
- Keep high performance
- Tons of data
- Huge amount of OLTP queries
- Very complex OLAP queries
- External Consistency
- SQL is much more complex than KV



Why TiDB choose Golang?

- Easy-learning
- Productivity
- Concurrency
- Easy to trace bugs and profile
- Standard libraries and tools
- Tolerant GC latency
- Good performance
- Quick improvement

Go in TiDB

- More than 160K lines of Go code and 138 contributors.

github.com/AlDanial/cloc v 1.72 T=4.00 s (137.4 files/s, 52274.8 lines/s)

Language	files	blank	comment	code
Go	523	17655	18761	163173
yacc	1	389	220	6236
XML	7	0	0	999
JSON	1	0	0	502
Markdown	9	238	0	470
YAML	3	6	4	227
make	1	38	2	132
Bourne Shell	3	12	5	52
Bourne Again Shell	1	9	18	47
Dockerfile	1	5	0	8
SUM:	550	18352	19010	171846

Memory && GC

- Query may touch a huge number of data.
- Memory allocation may cost a lot of time.
- Put pressure on GC worker.
- Degrade the performance of SQL.
- OOM sucks!
- `runtime.morestack`

Reduce the Number of Allocation

- Get enough memory in one allocation operation

```
a := []int{1, 2, 3, 4, 5}
```

```
b := []int{}
```

```
// a much better way:
```

```
// b := make([]int, 0, len(a))
```

```
for _, i := range a {
```

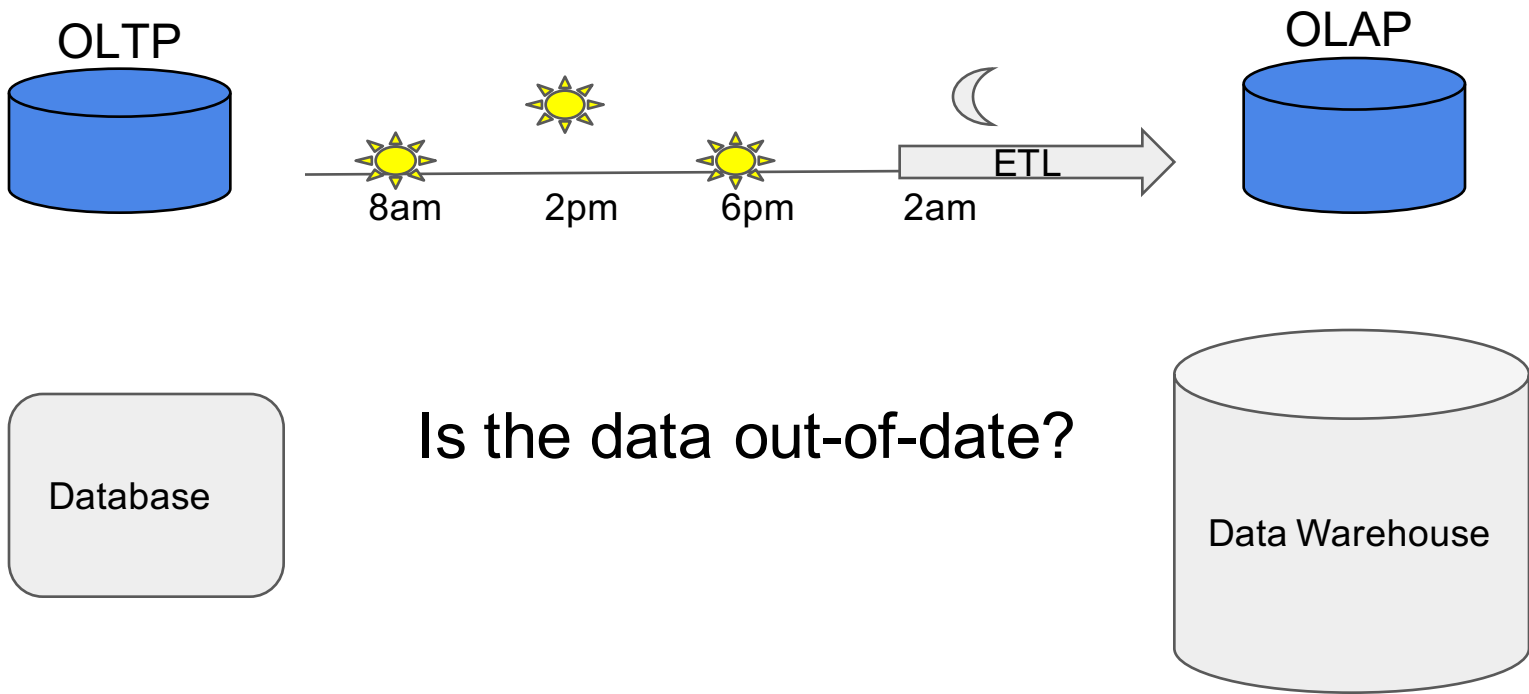
```
    b = append(b, i)
```

```
}
```

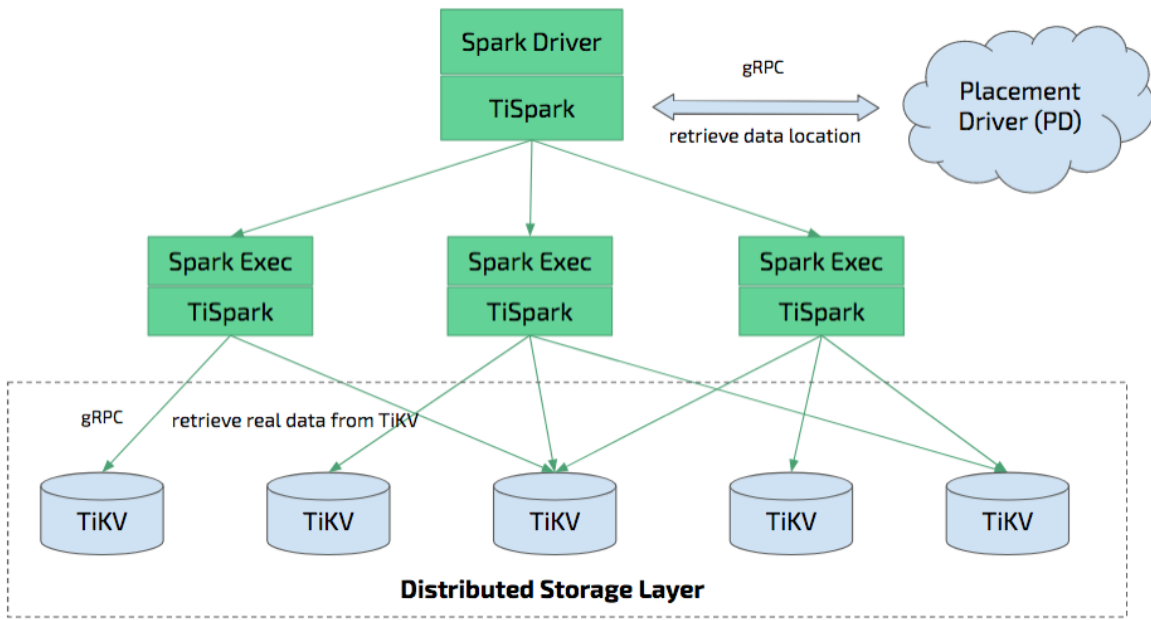
Reuse Object

- Share a stack for all queries in one session
- Introduce a cache in goyacc
- Resource pool

OLTP & OLAP



TiSpark



Thanks!

Contact me:

wink@pingcap.com

www.pingcap.com

Wechat: mrwink



PingCAP & 广州 9.9
gopher meetup

