CEDAR: A Distributed Database for Scalable OLTP

Weining Qian
wnqian@dase.ecnu.edu.cn

華東師範大學

DaSE
Data Science
& Engineering

The Internet economy

- Content/traffic => money
- O2O : Online => Offline, or Offline => Online
- **O2O of mission-critical apps: 互联网+ (Internet+)**
- OLTP is inevitable

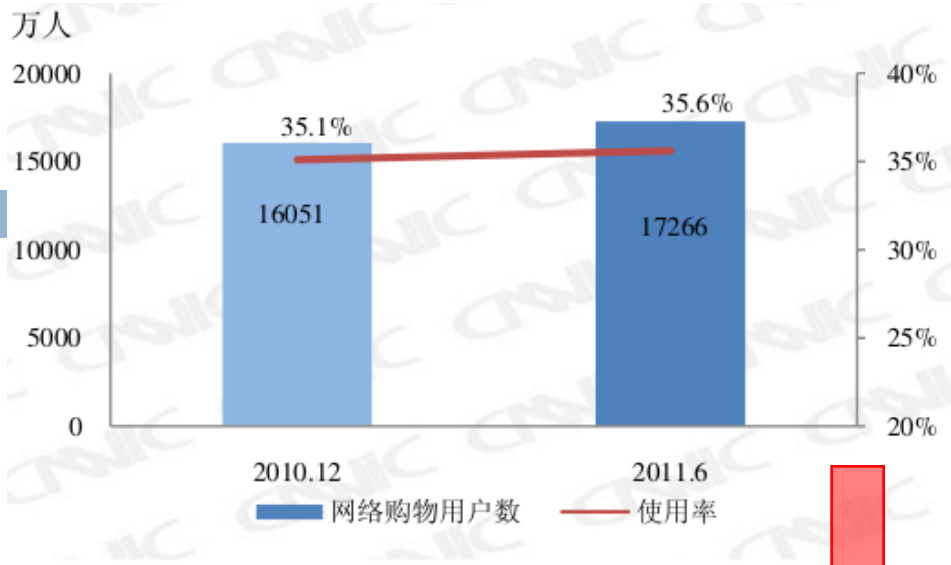
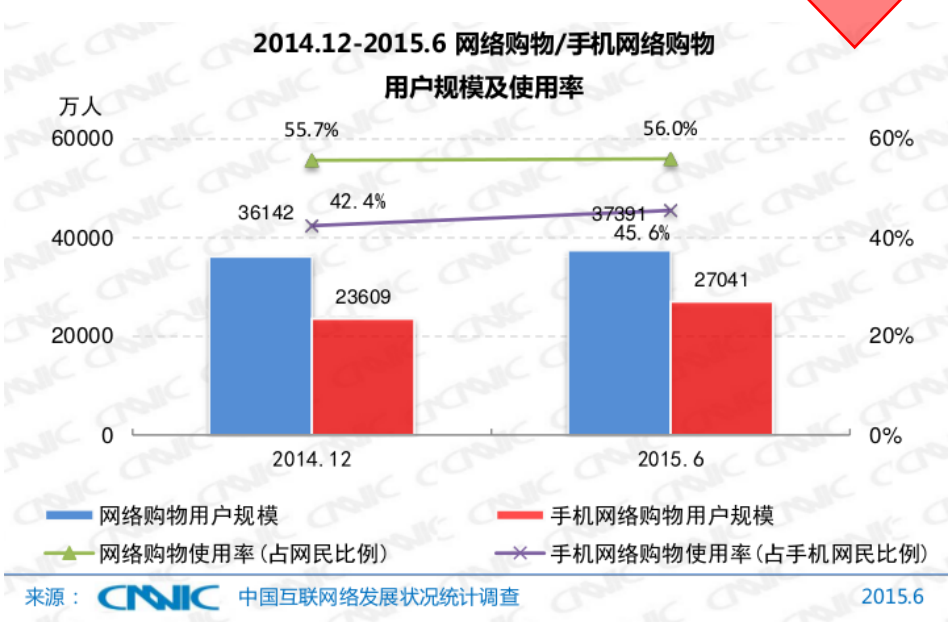


图 20 2010.12-2011.6 网络购物用户数及使用率



Phenomenal applications

- Phenomenal - very remarkable; extraordinary.

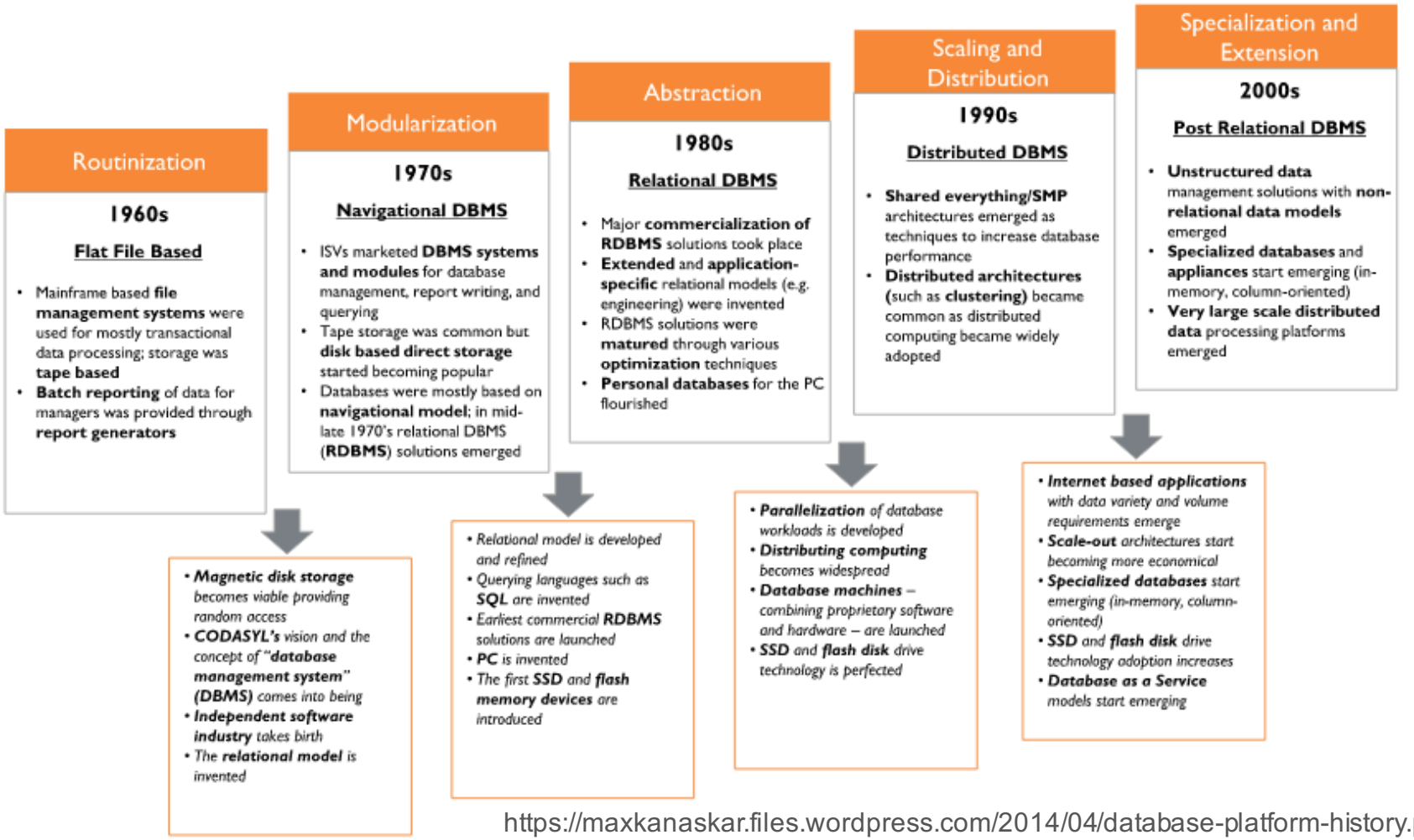
- 180,000 tps in year 2016

Phenomenal is common

- 12306 during Spring Festival
- Black Friday promotion/second kill, ...
- The pressure is on **backend (transaction | payment) systems**
- Be inevitable and day-to-day more common

- **All pressure will finally go to mission critical systems**
- **Essentially a High-throughput, scalable transaction processing problem**

A brief history of DBMS



https://maxkanaskar.files.wordpress.com/2014/04/database-platform-history.png

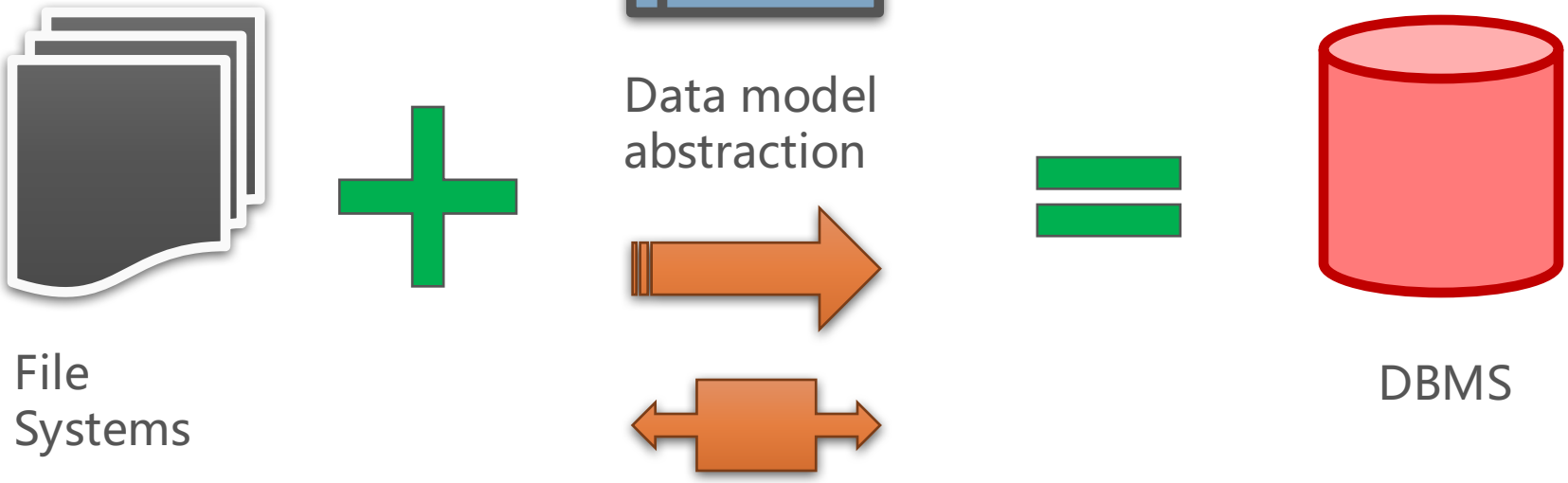
One size fits all => One size fits none!

2015 - One Size Fits None!!

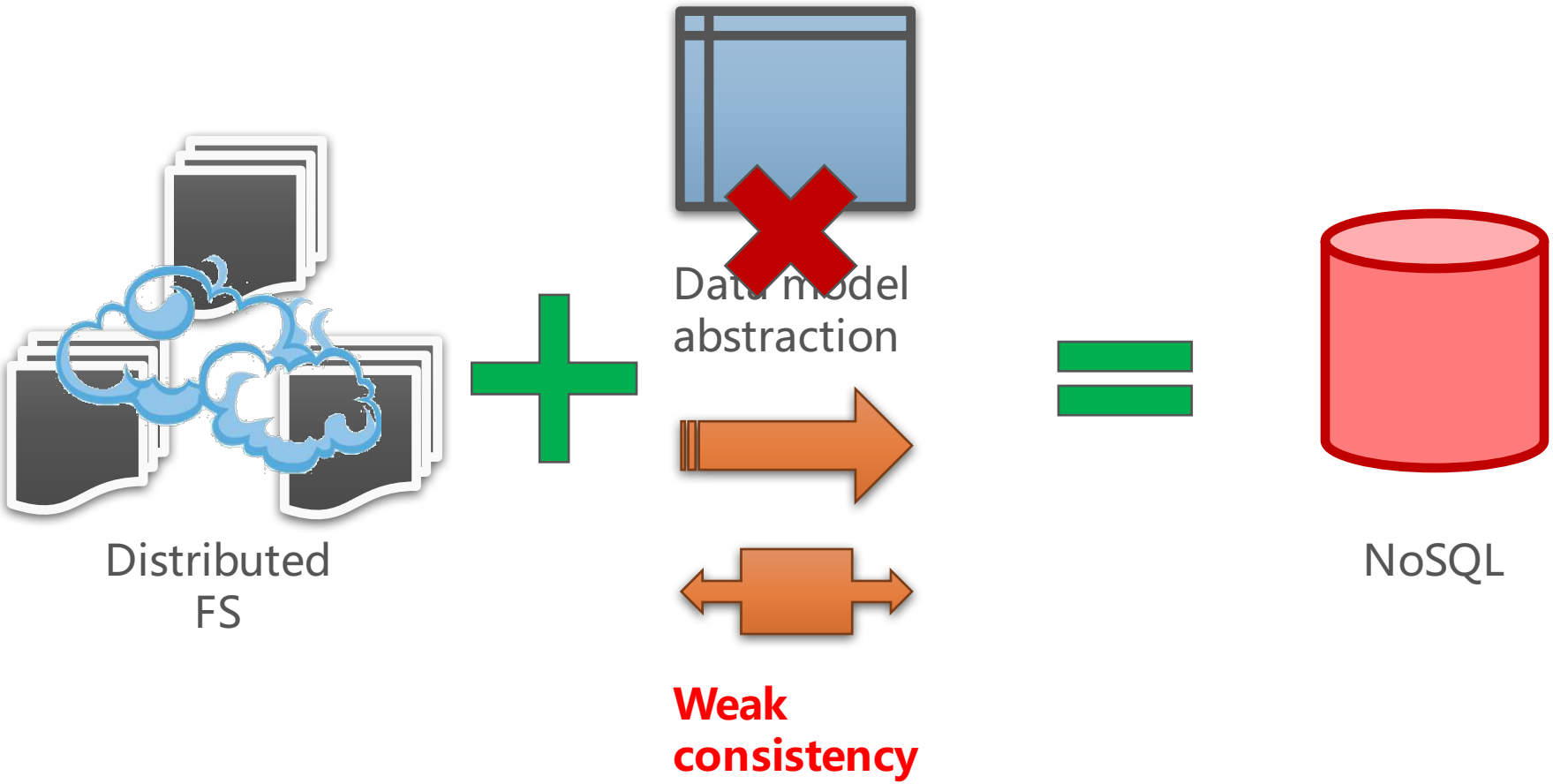
- Data Warehouse market
- OLTP market
- NoSQL market
- Complex Analytics
- Streaming market
- Graph analytics market

31st IEEE ICDE SEOUL KOREA, 2015

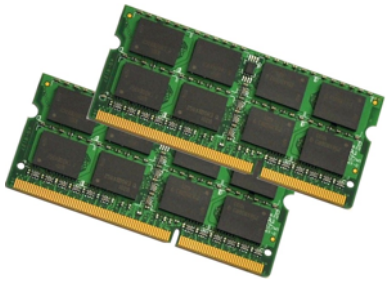
DBMS


Data model abstraction
File Systems
Data processing abstraction
DBMS

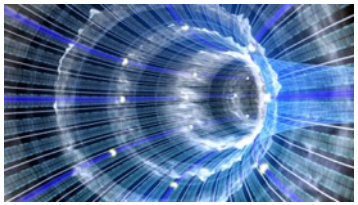
NoSQL


Data model
abstraction
Distributed
FS
NoSQL
Weak
consistency

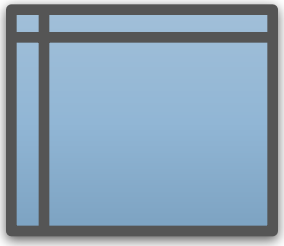
NewSQL



In-memory computing

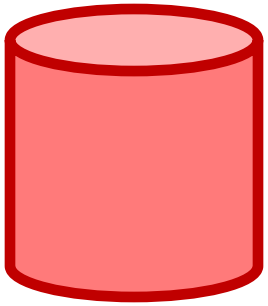


Fast networking



Relational model

Transaction processing

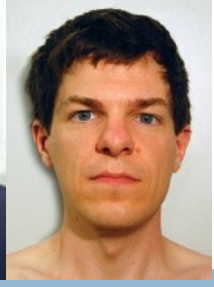


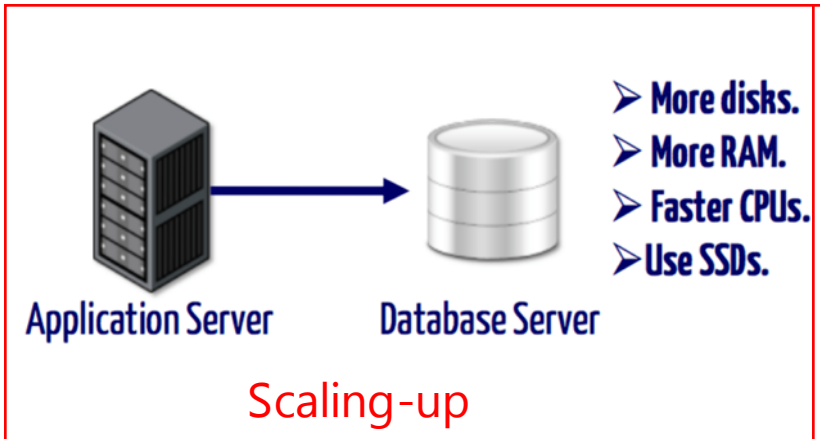
NewSQL

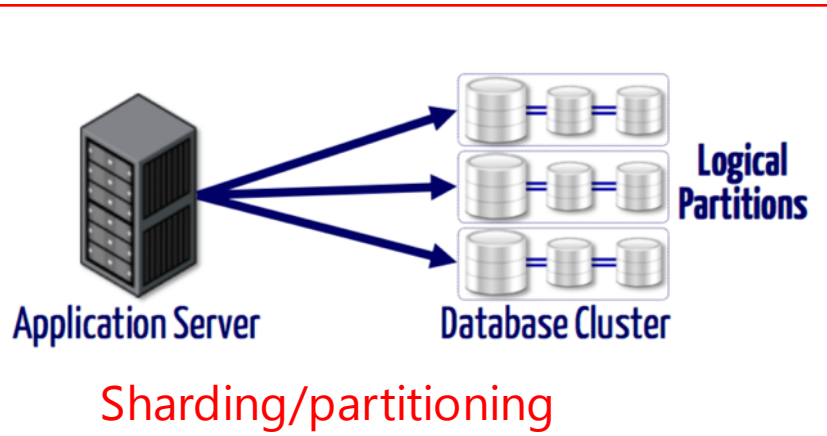
OldSQL vs. NoSQL vs. NewSQL

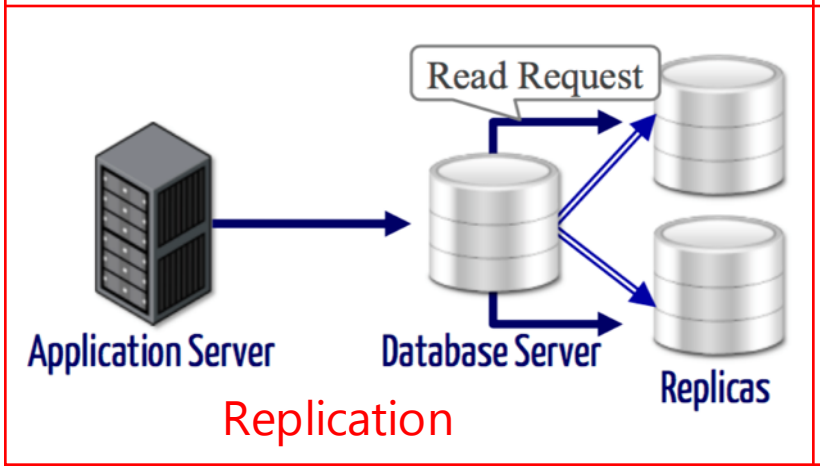
	OldSQL	NoSQL	NewSQL
Data model	Relational	---	Relational
Interface	SQL	Variance	SQL
Consistency/Concurrency control	Strong	Weak	Strong
Fault tolerance	Strong	Fine	Strong
Performance	Poor	Good	Very good
Scalability	Poor	Good	Fine

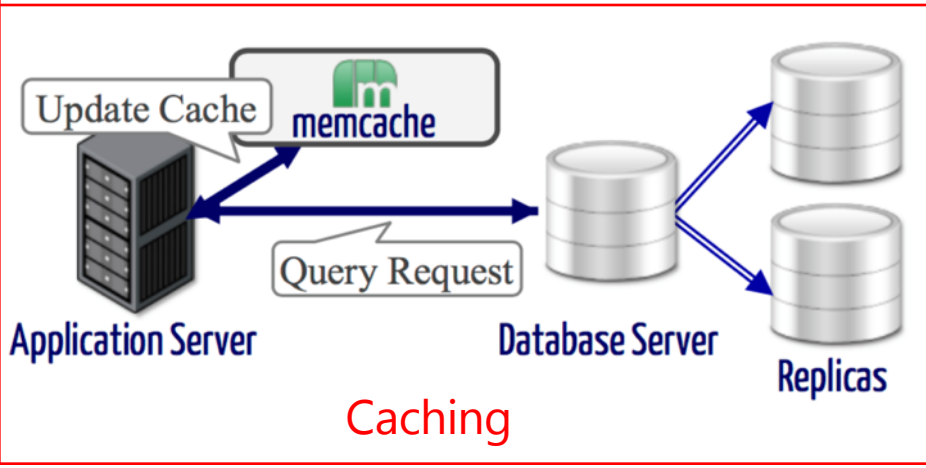
How to scale a database system?





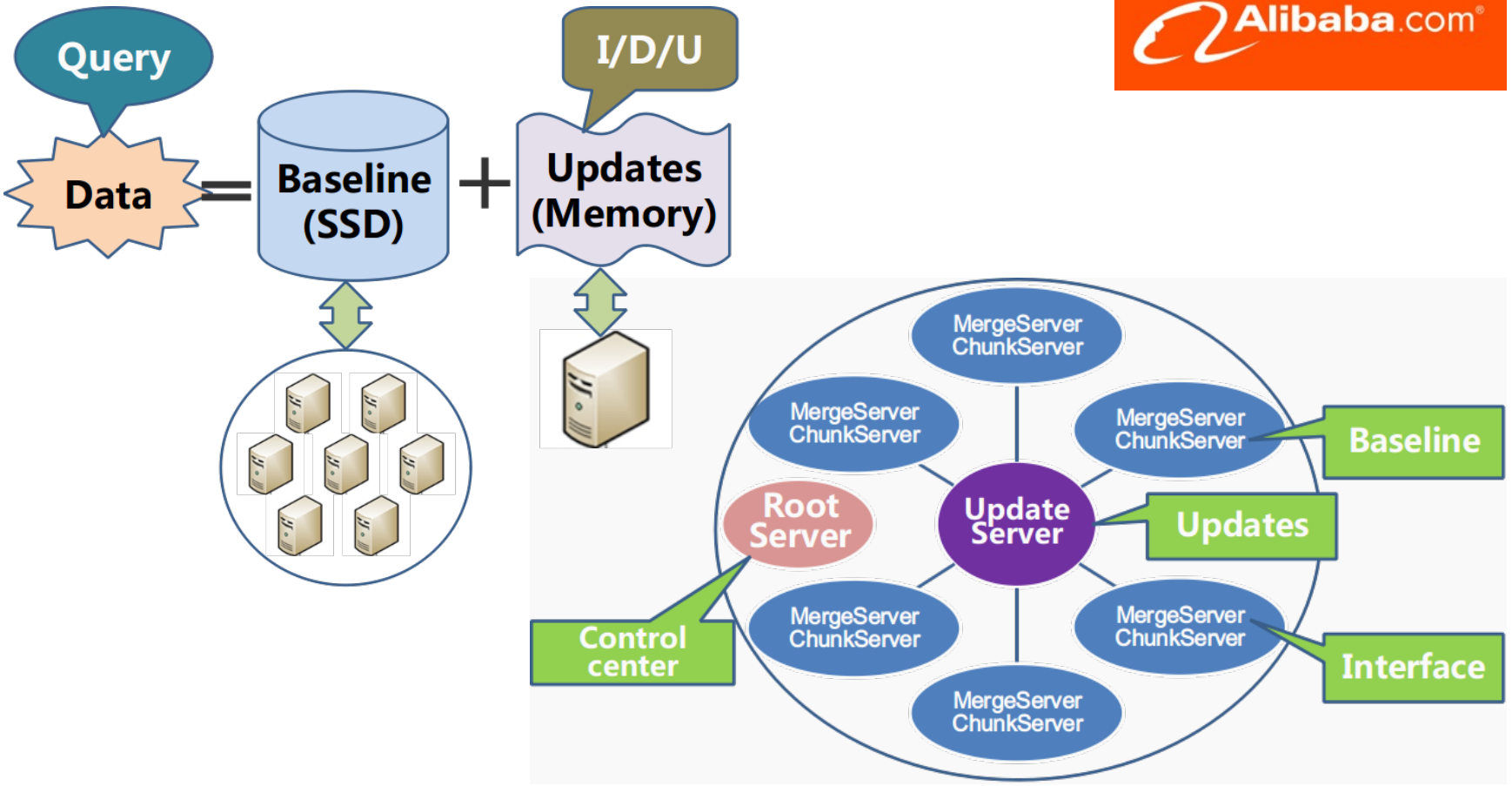






https://www.cs.cmu.edu/~christos/courses/dbms.S14/slides/29scaling.pdf

The open-source OceanBase (0.4)



OceanBase 0.4 is not enough

- Simple transactions only
- Weak availability support
- Single-point transaction

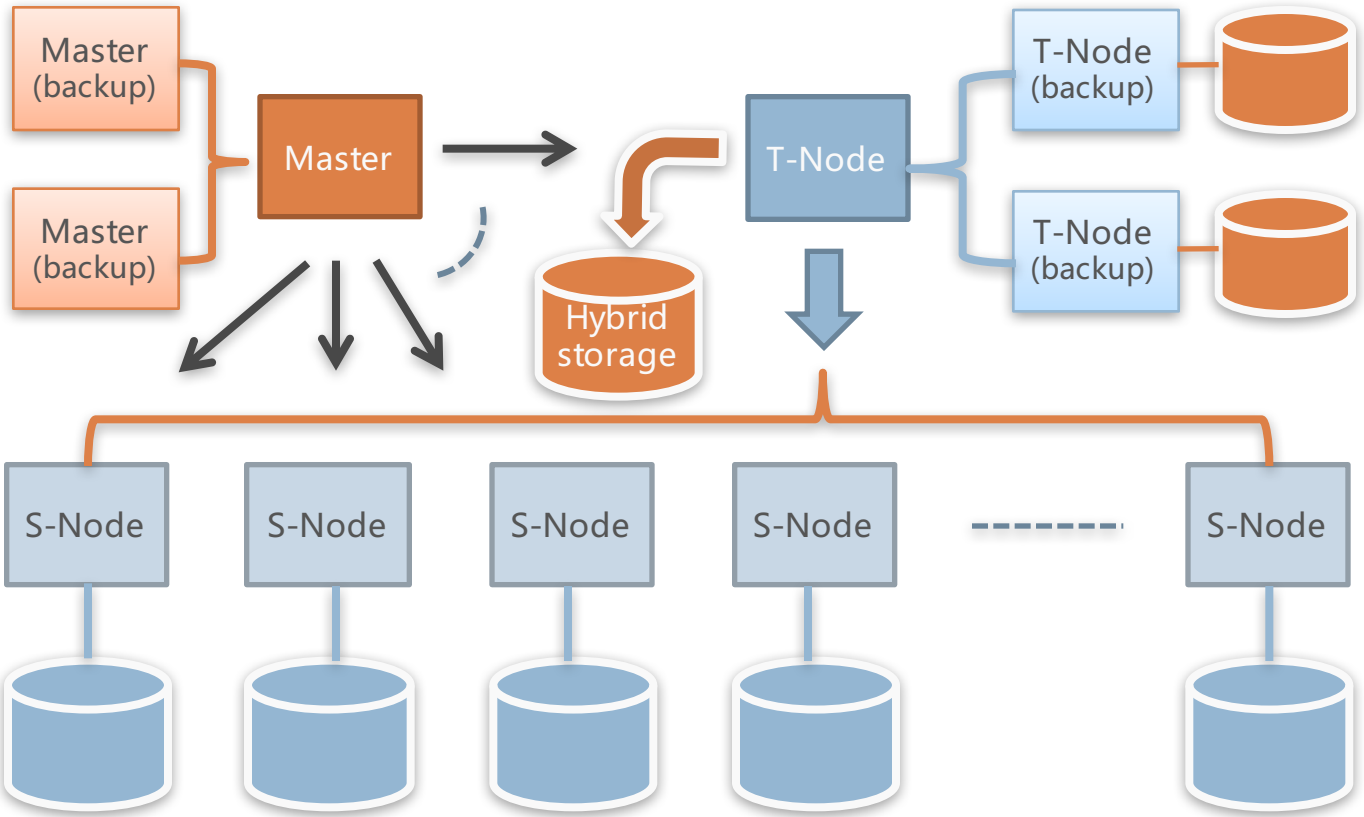
Not enough for mission critical apps in banks with strong-consistency, high-availability, and high-throughput complex transaction processing requirements

- More optimization needed for query/storage
- Interface adaptibility

Features we need

- Complex transactions
- High-performance:
  - High-throughput
  - Low latency
- High-availability
- Scalability
- Elasticity

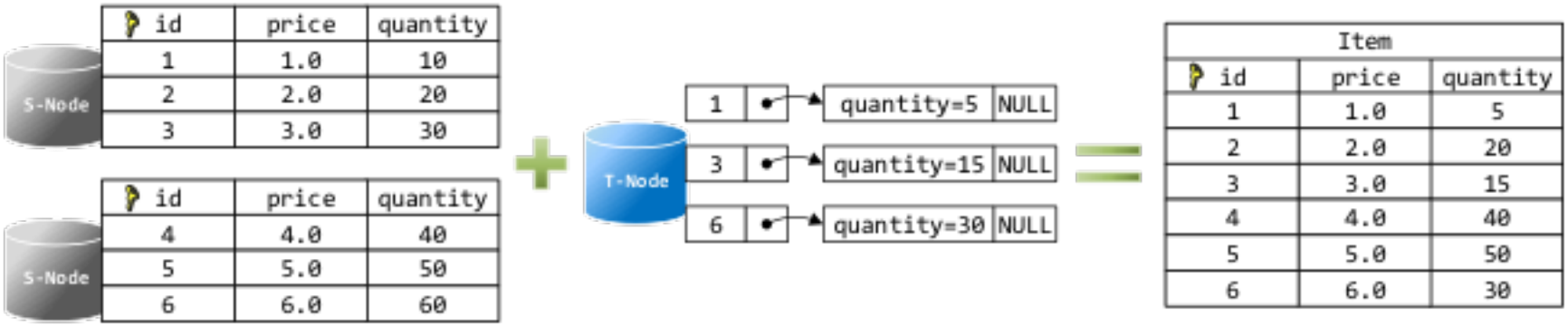
Overview (CEDAR core)



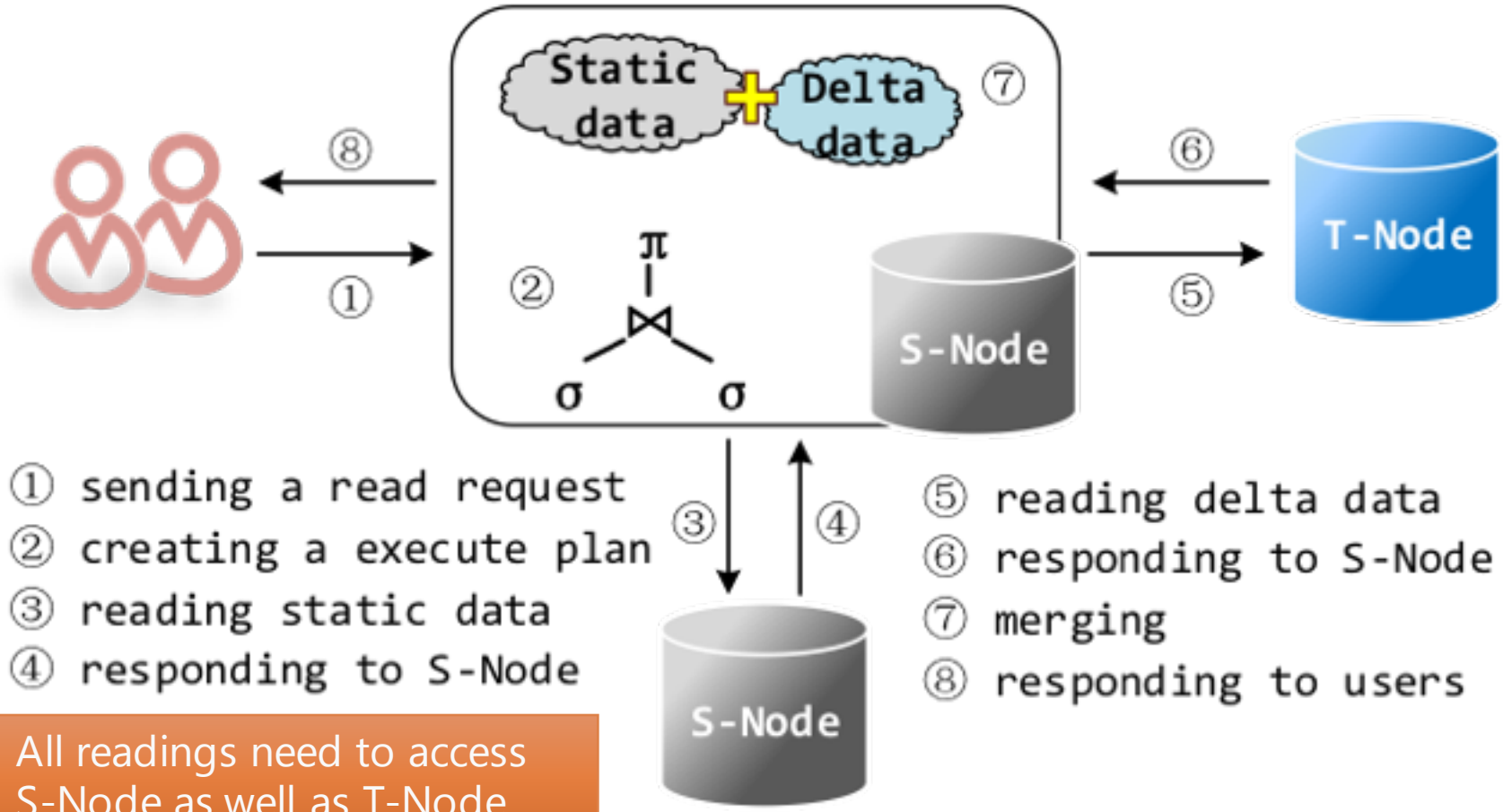
Design choices

- Separating read and write operations on different nodes
- Scalable reading on multiple nodes
- Writing to memory in one node only
  - No expensive distributed concurrent control or synchronization
- "Deep" optimization for transactions
  - To optimize data transmissions, query execution plans, and executions
- High-availability is guaranteed by log synchronization

Status = Baseline + Delta

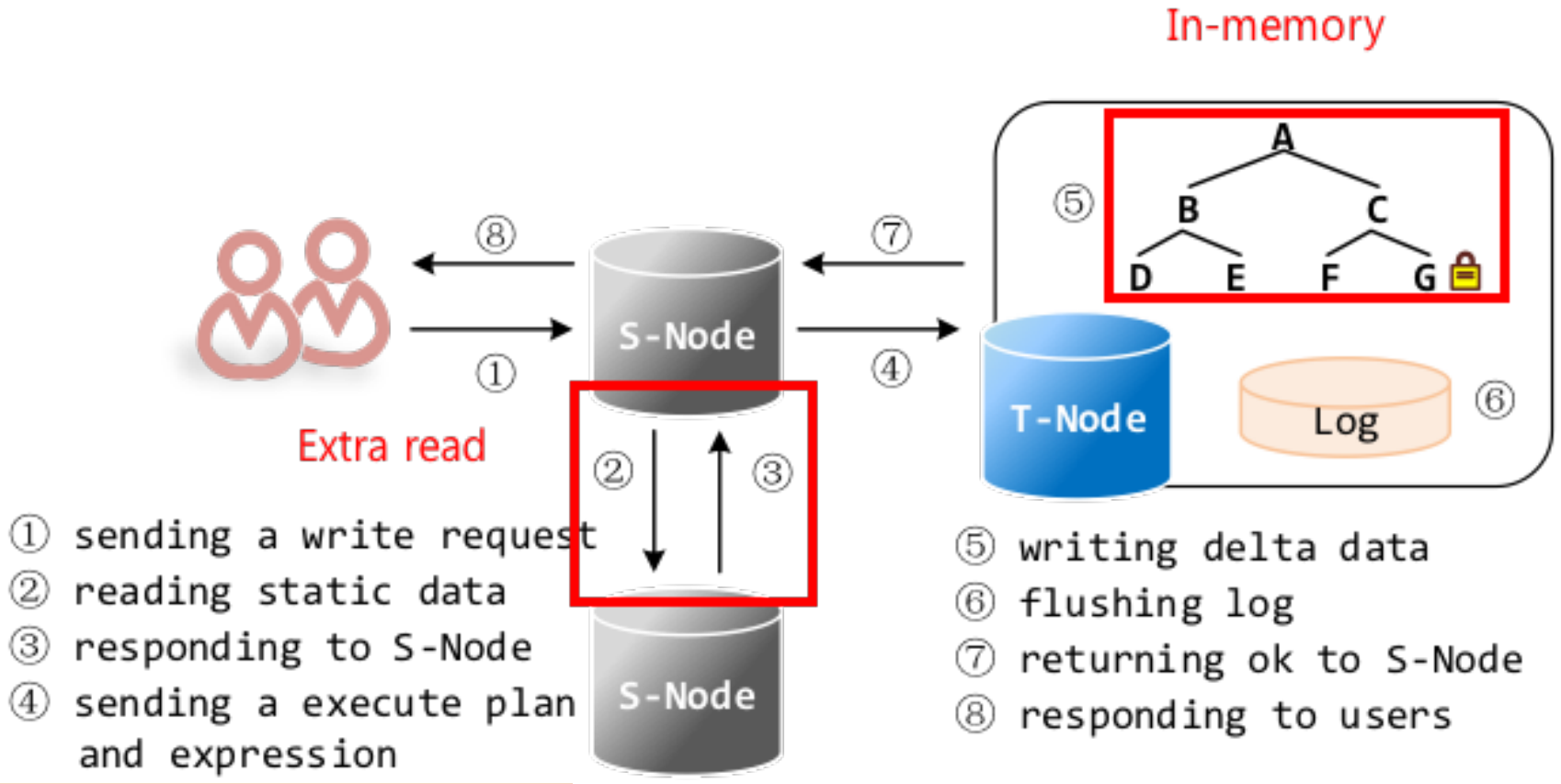


Reading



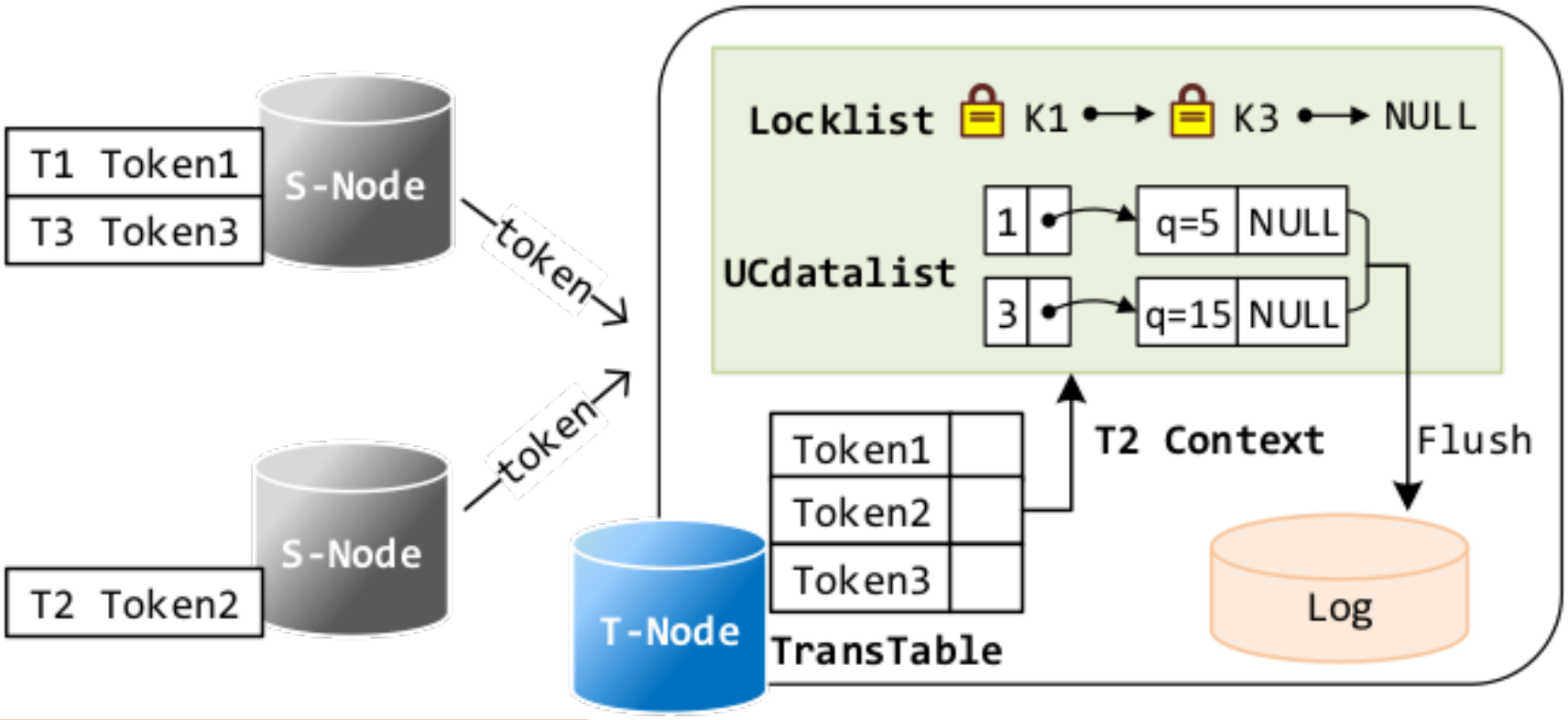
- All readings need to access S-Node as well as T-Node

Writings



- All writings need to access S-Node as well as T-Node

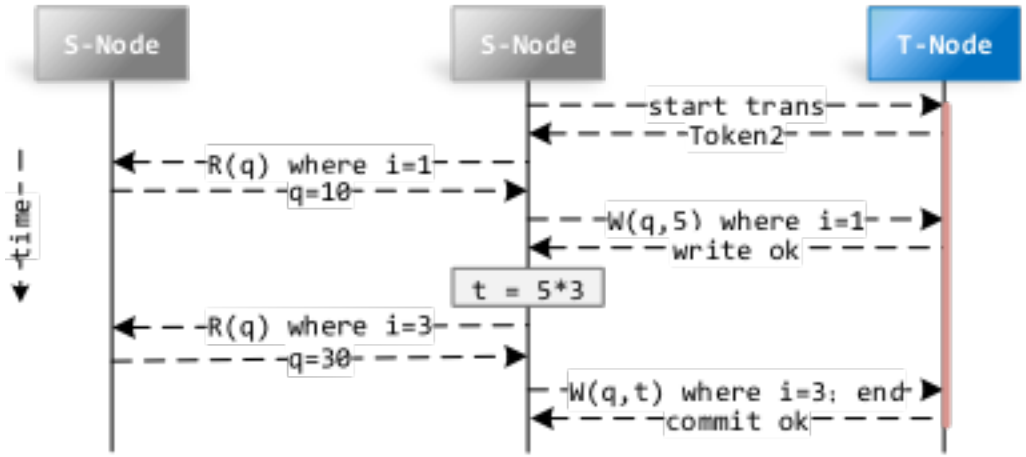
Transaction management



- Transactions are only processed on the T-Node

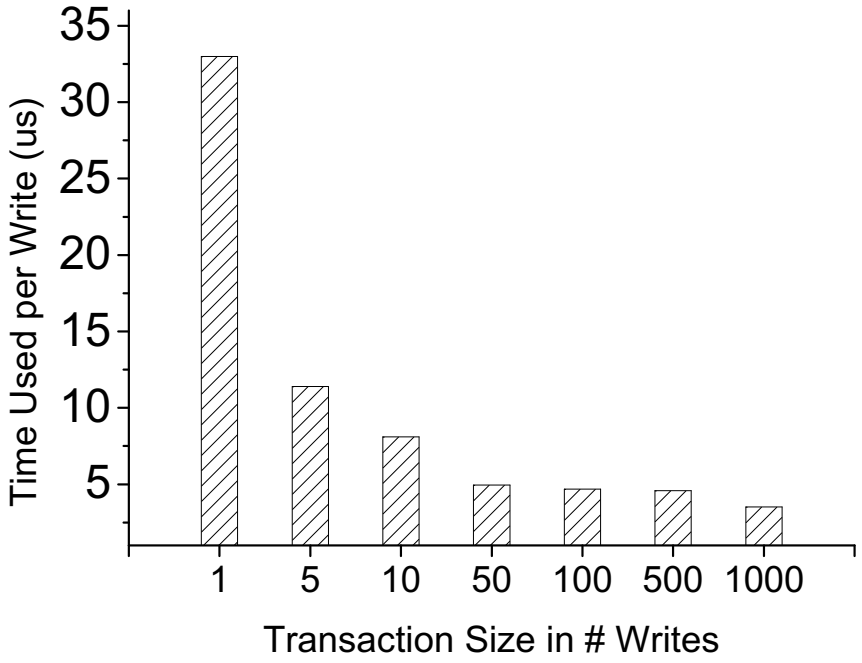
Pros and cons

- Pros
  - Massive storage
  - Scalable read
  - Efficient transaction management
- Cons
  - Expensive data transmission



Performance is affected by

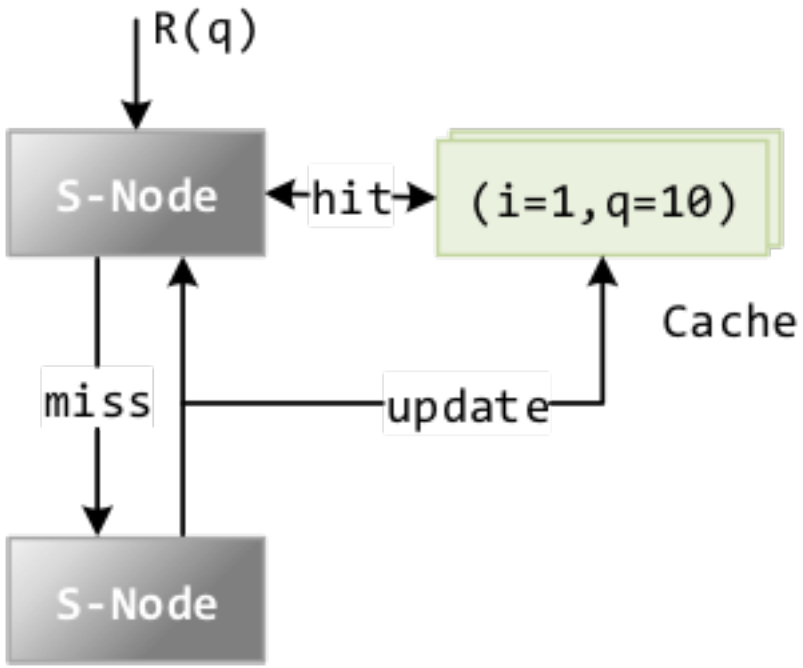
- The lengths of locks affect
  - Degrees of parallelization
  - Latency
- Capability of S-Nodes
  - Throughput of readings
- Capability of the T-Node
  - Throughput of writings



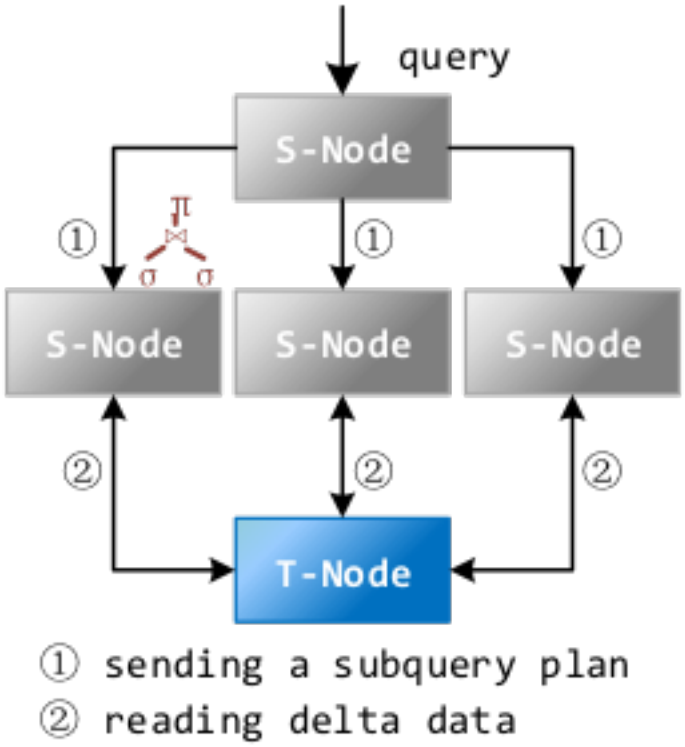
- Most cost is for short/simple transactions
- They are easier to be scheduled

S-Node optimizations

Static data caching



Parallel readings



T-Node

- Storage node?
  - All computation resources are for TP
  - High communication cost
- Computation node?
  - Low communication cost
  - How much work should it do?

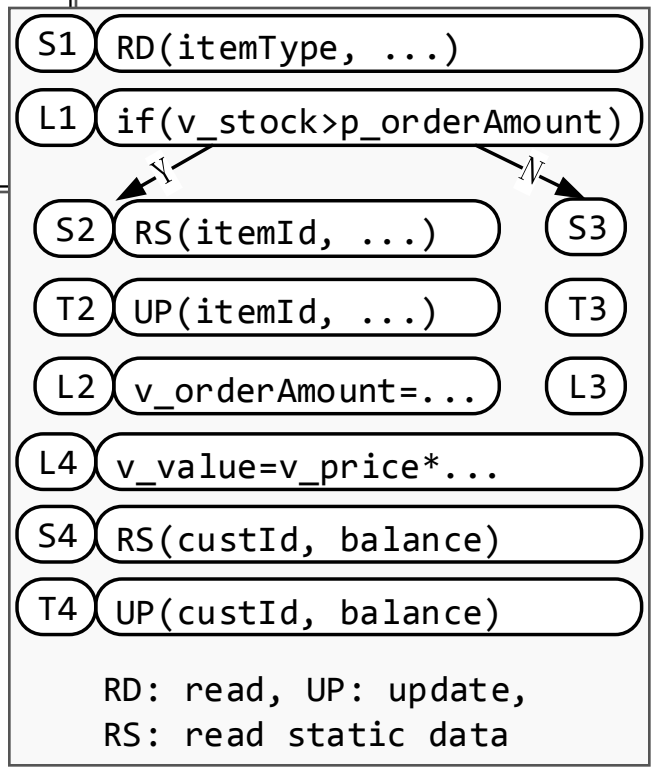
- Balancing T-Node's computation for queries and transaction processing

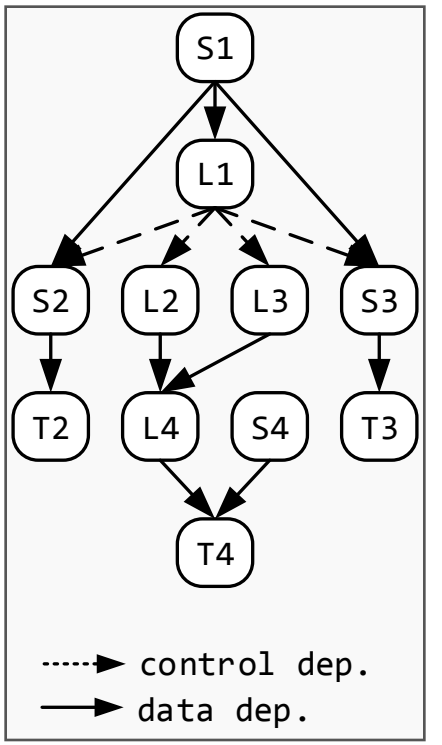
Transaction compilation

```
Procedure Order(p_itemType int, p_custId int,
                      p_orderAmount int)
declare v_price, v_value double;
declare v_itemId, v_stock, v_orderAmount int;
select itemId, price, stock
     into v_itemId, v_price, v_stock
     from item where itemType = p_itemType
     order by price desc limit 1;
if( v_stock > p_orderAmount )
  update item set stock = stock - p_orderAmount
        where itemId = v_itemId;
  v_orderAmount = p_orderAmount;
else
  update item set stock = stock - v_stock
        where itemId = v_itemId;
  v_orderAmount = v_stock;
end
v_value = v_price * v_orderAmount;
update customter set balance -= v_value
      where custId = p_custId;
```

Transactions

Execution plan

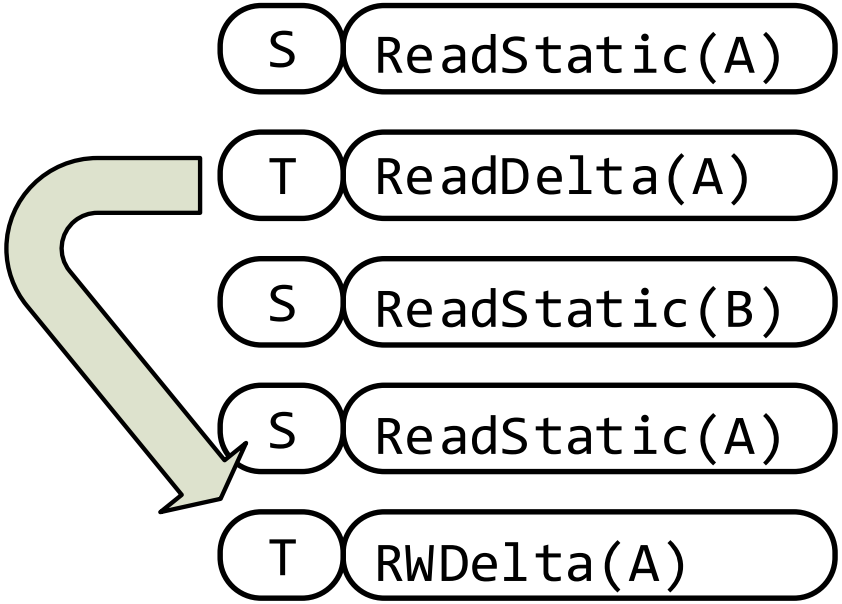




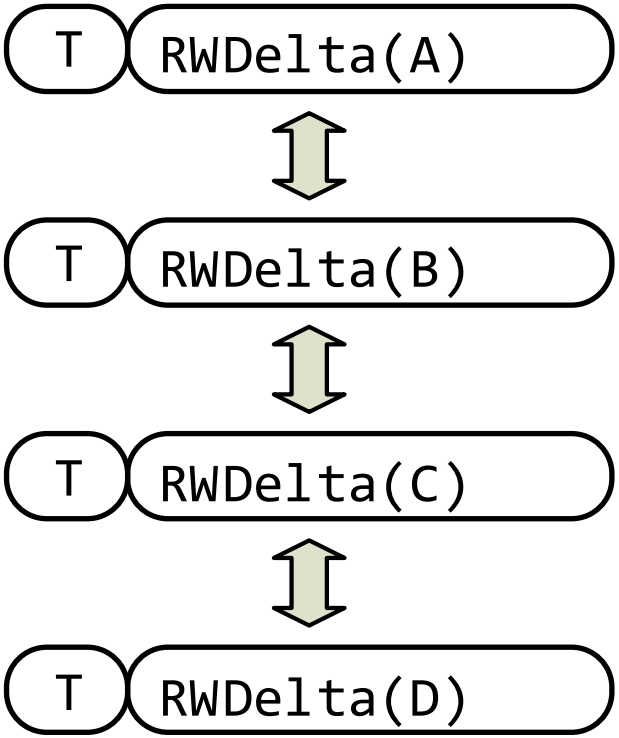
Dependency graph

T-Node optimization

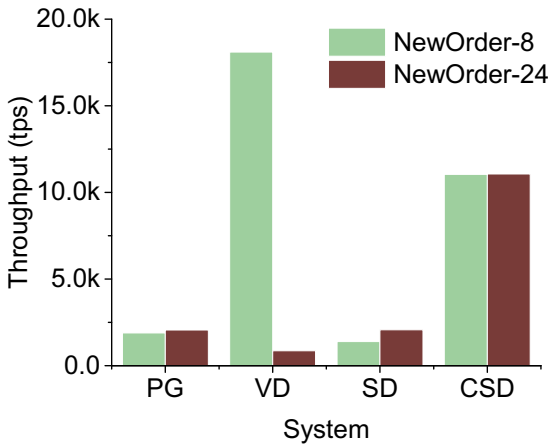
Re-order T-Node ops

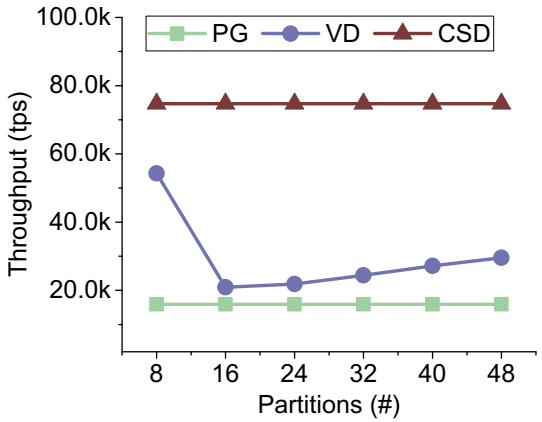


Postpone conflict ops

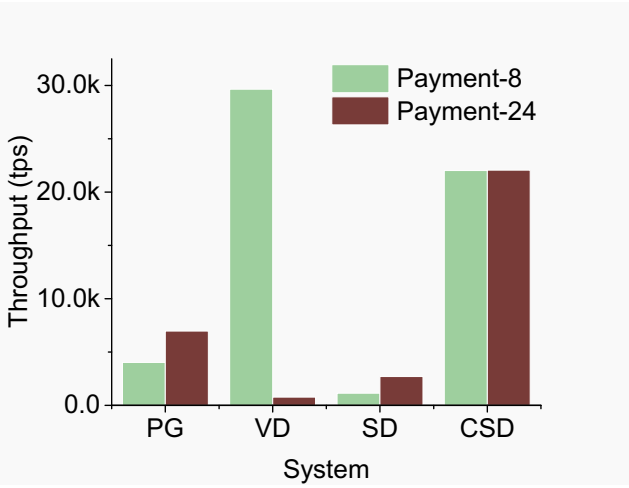


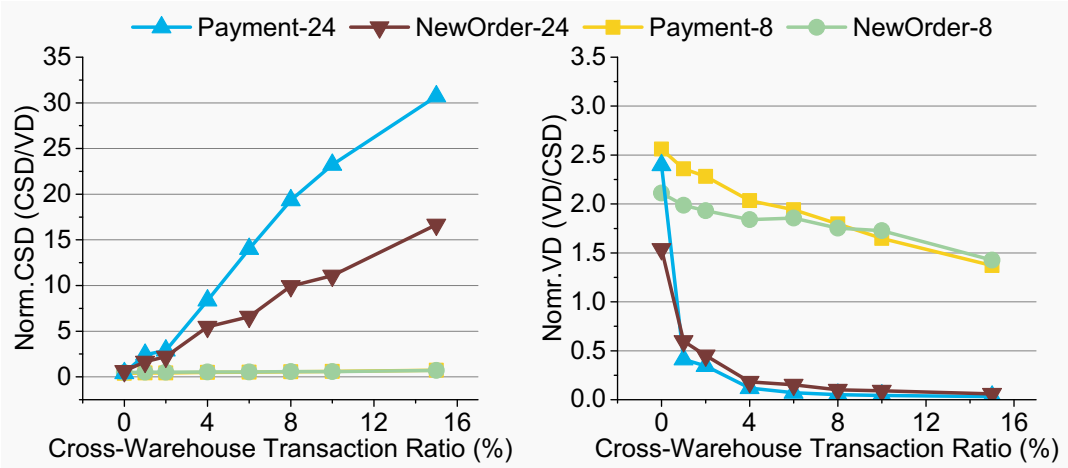
TPC-C, Smallbank, TATP Benchmarks





- Better than PG
- Better than VoltDB under complex workload
- With significant advantage when data cannot be naturally partitioned

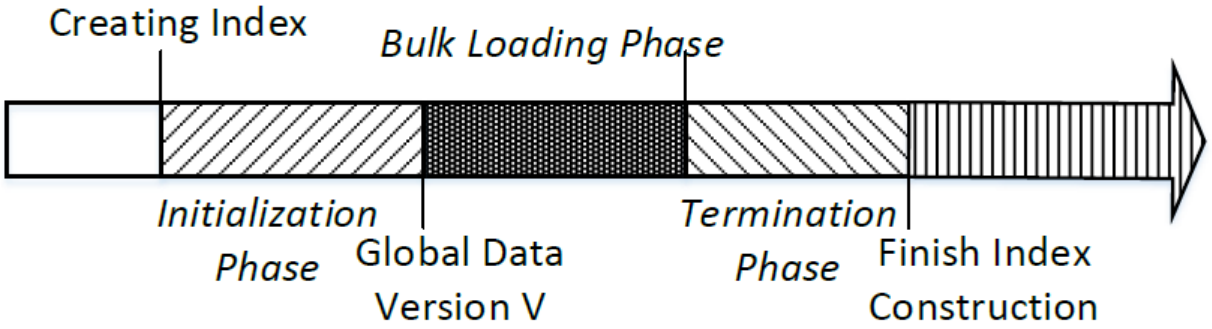




Indexing

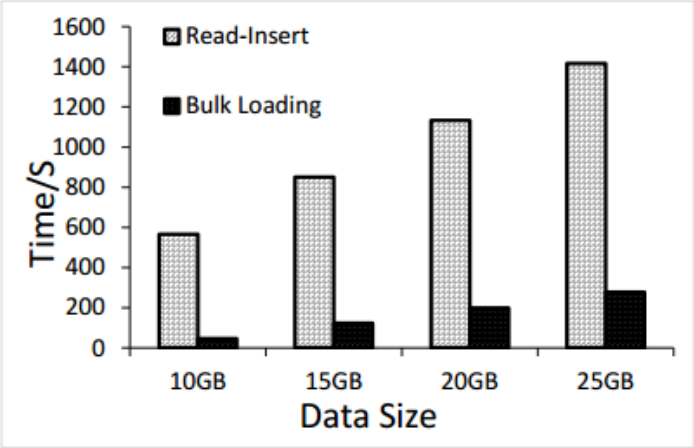
- Index is organized as a table.
- No distributed transactions.
- Taking advantage of the load balancing, availability of system.

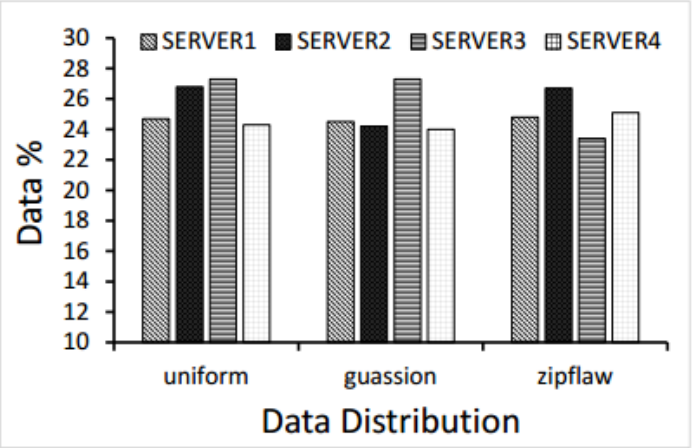
Indexing overview

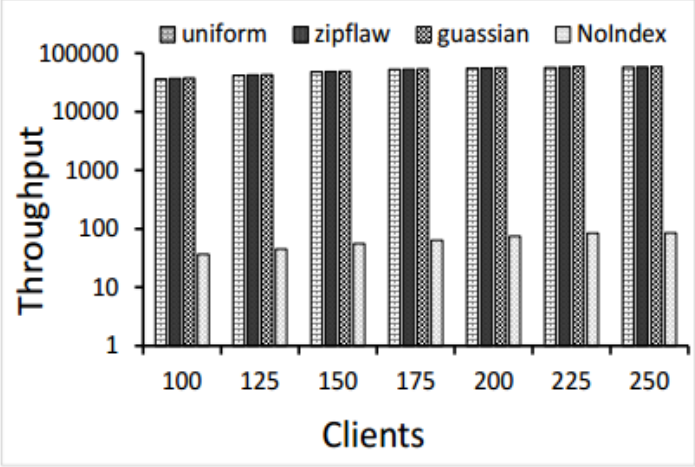


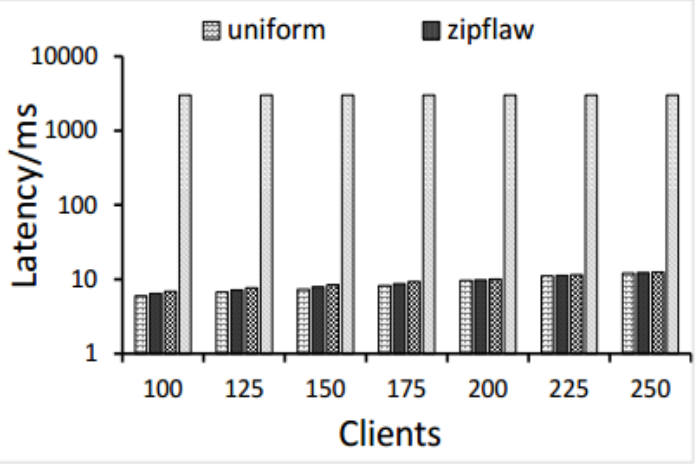
- Initialization: Preparing for the start of index construction.
- Bulk Loading
  - Local processing: collecting statistical information
  - Global processing: achieving load balancing based on an equi-depth histogram
- Termination: Scheduling the task for replication of the index for high availability.

Experimental results







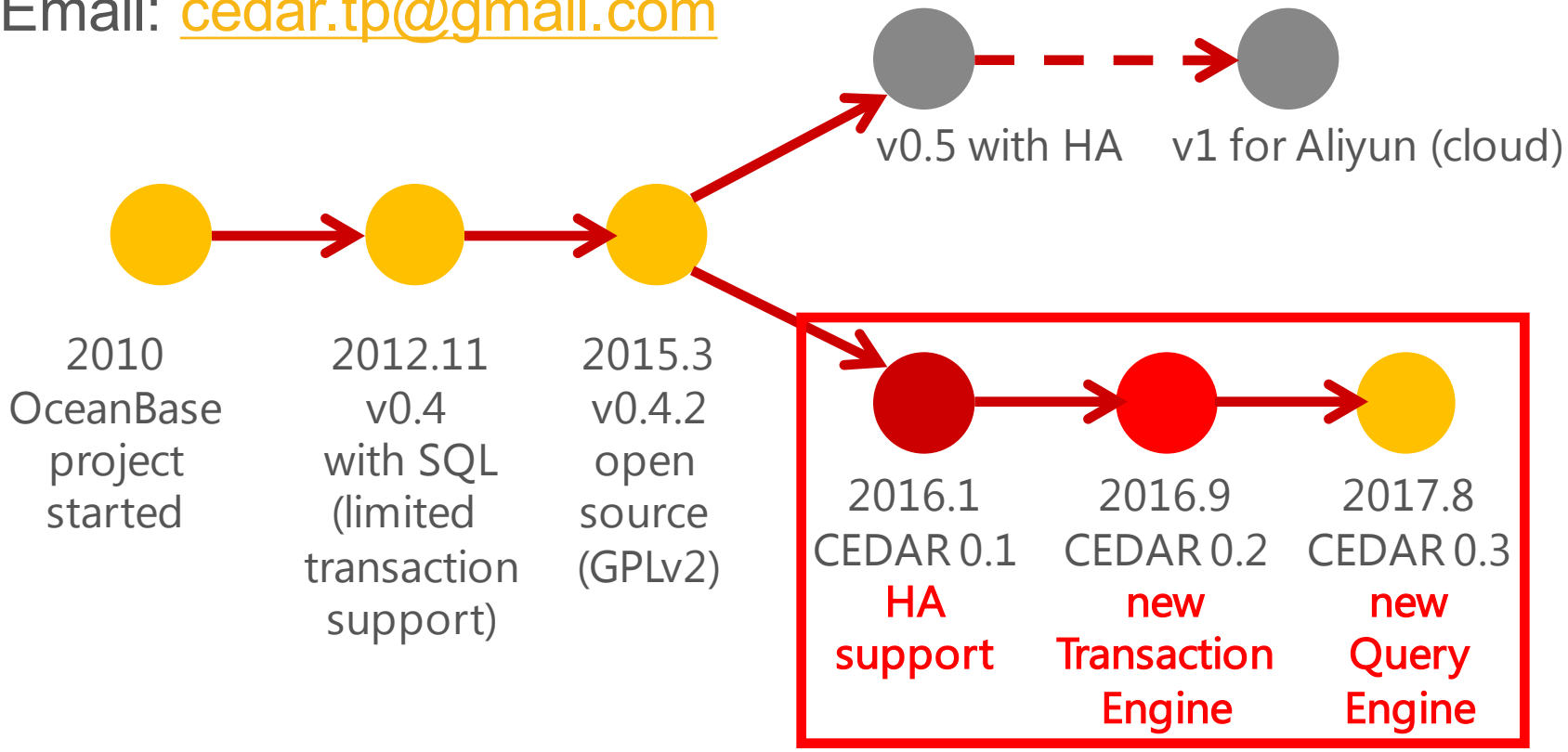


Other works on the CEDAR

- Scalable range optimistic concurrency control
- Global snapshot isolation with Paxos replication
- Scalable commit log recording/synchronization
- Data transmission optimization
- Distributed statistics monitoring
- Rule-based and cost-based query optimization

Release

Homepage: https://github.com/daseECNU/CEDAR
Email: cedar.tp@gmail.com



Homepage: https://github.com/daseECNU/CEDAR

> CEDAR是华东师范大学数据科学与工程学院（简称"DaSE"）基于 OceanBase 0.4.2 研发的可扩展的关系数据库。2016年2月1日，CEDAR项目组完成了CEDAR 0.1 版本的开发与测试，2016年9月26日，CEDAR 0.2 版本发布。
>
> ## 版本特性
>
> CEDAR在OceanBase 0.4.2 的基础上新增了如下11个功能模块：

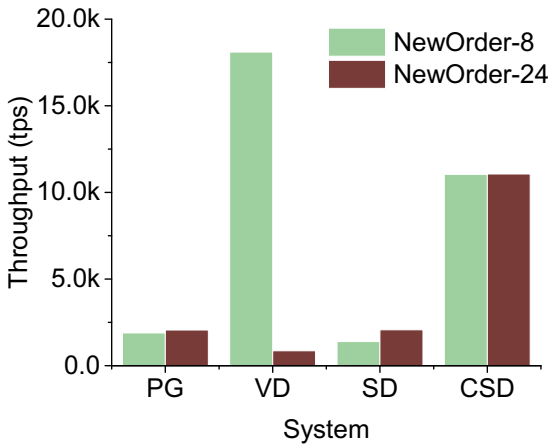
CEDAR 0.1 版本新增的功能有：

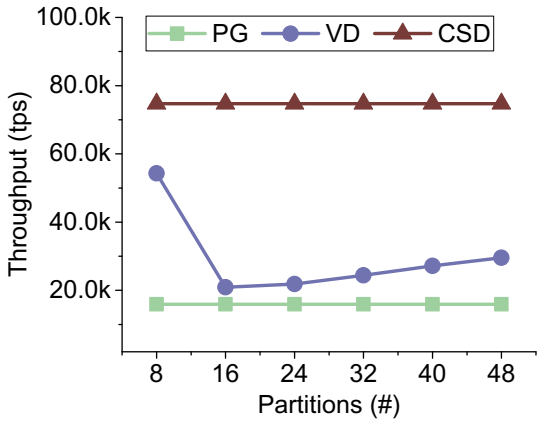
- 高可用的三集群架构（集群间选主、集群角色自动切换、日志强同步及恢复等机制）
- 多线程网络IO处理框架Libonev
- 游标
- 存储过程
- 二级索引
- 非主键多行更新
- 半连接

CEDAR 0.2 版本新增的功能有：

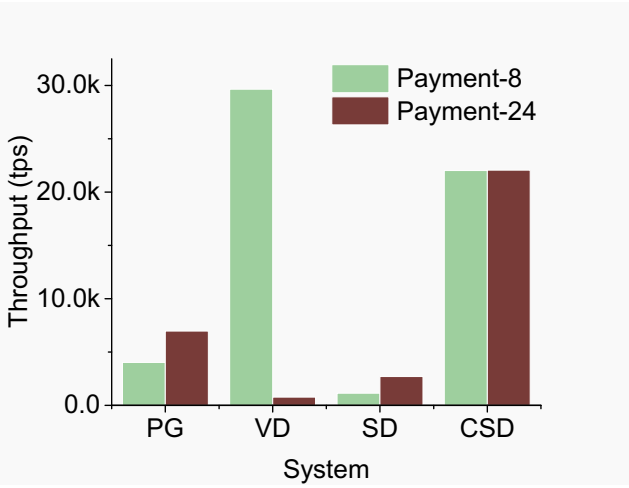
- SNAPSHOT ISOLATION 隔离级别
- 表锁
- 基于布隆过滤器的连接
- 日志同步优化

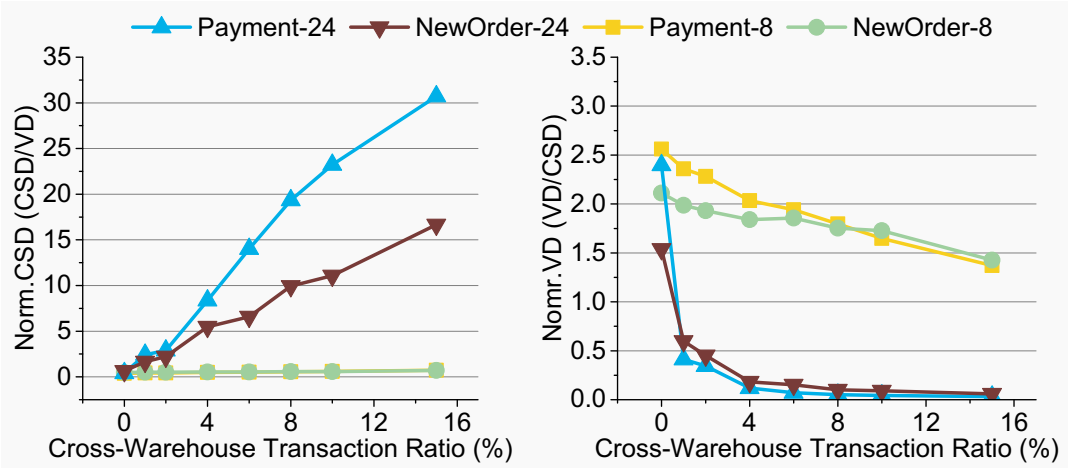
TPC-C, Smallbank, TATP Benchmarks





- Better than PG
- Better than VoltDB under complex workload
- With significant advantage when data cannot be naturally partitioned

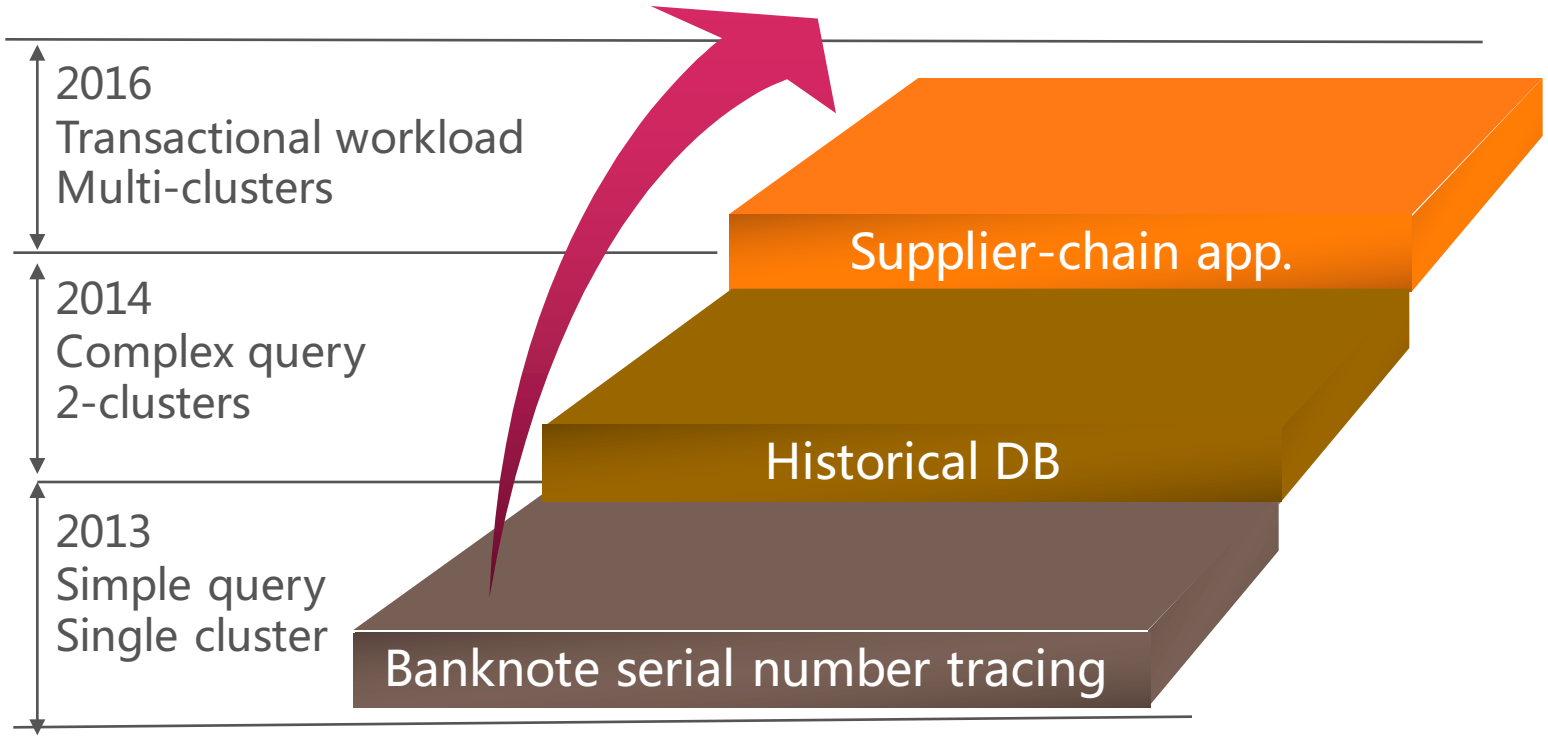




Application

- One of the largest banks in China
  - To use local/open-source DBMS for transactional applications (to replace DB2)
  - Three stages: historical DB => Hybrid DB => transactional DB
  - 2013 – present (in stage 2)
  - Code name: CBase within the bank

Stage 1 => Stage 2



Banknote serial number tracing

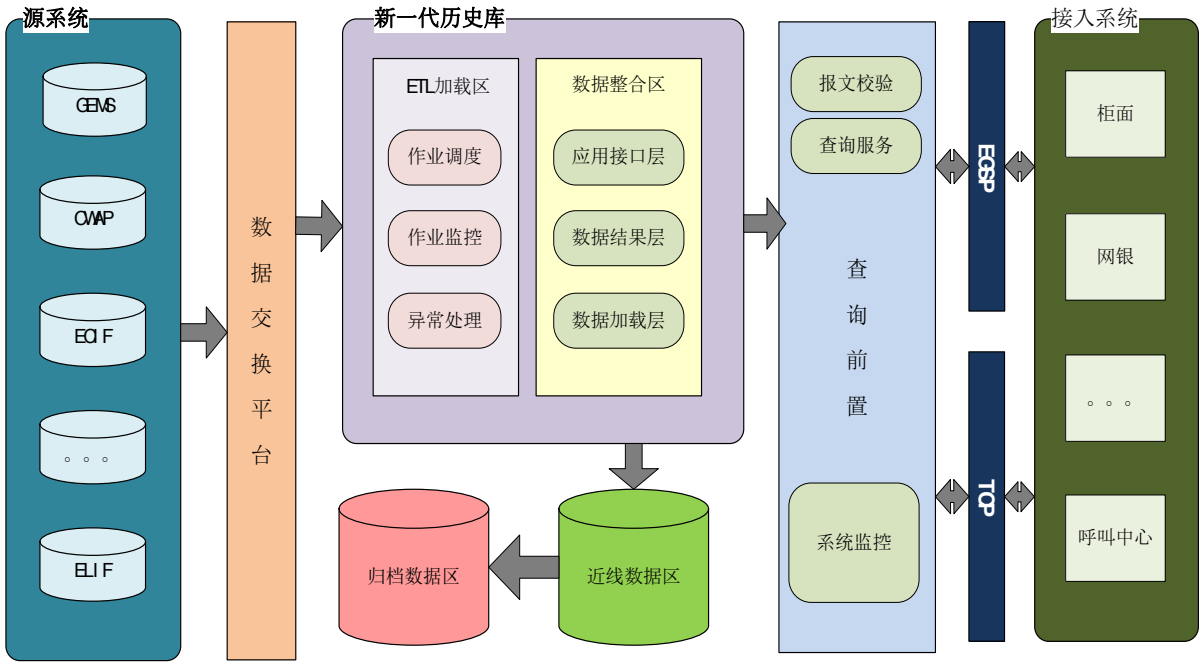
- All records of banknote serial numbers are stored for 90 days in the CBase
- More than 10TB data totally
- For counterfeit money detection etc.

Status

- Went online in 2013-12
- Single cluster with 9 servers
- Features
  - Batch load with about 100GB data/day
  - Latency in several milliseconds over TBs of data
  - Linear scalability with respect to number of servers

Historical DB

- Move historical data to historical DB so that the online system is lightweighted
- 283 kinds of query services are supported by CBase

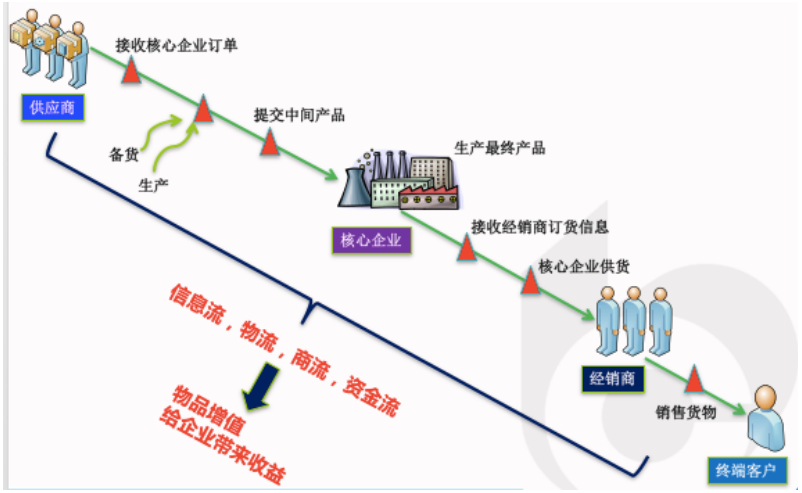


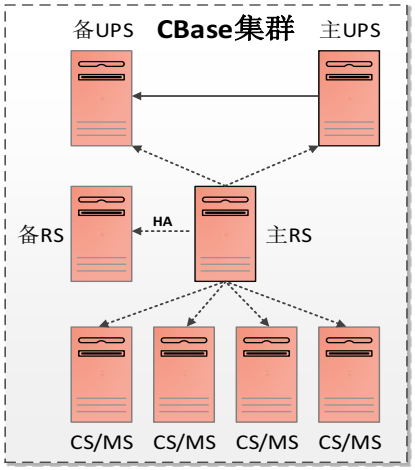
Status

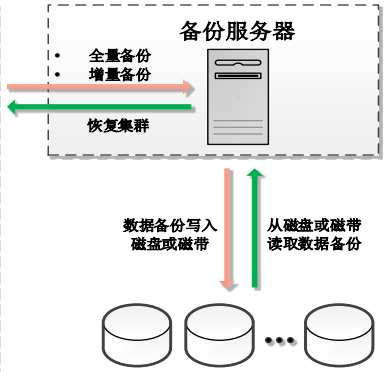
- Went online in 2014-09
- 2-clusters, with 21 servers each
- Features
  - About 400GB data/day
  - Latency in several milliseconds for most queries
  - Active/active high-availability
- Has become the single-point sign-in gateway of nearly all business logic

Supplier-chain applications

- Hybrid workload
  - Complex transactions with tens or even hundreds of SQL operations
  - Complex analytical queries with many joins of large tables
  - High-availability requirements







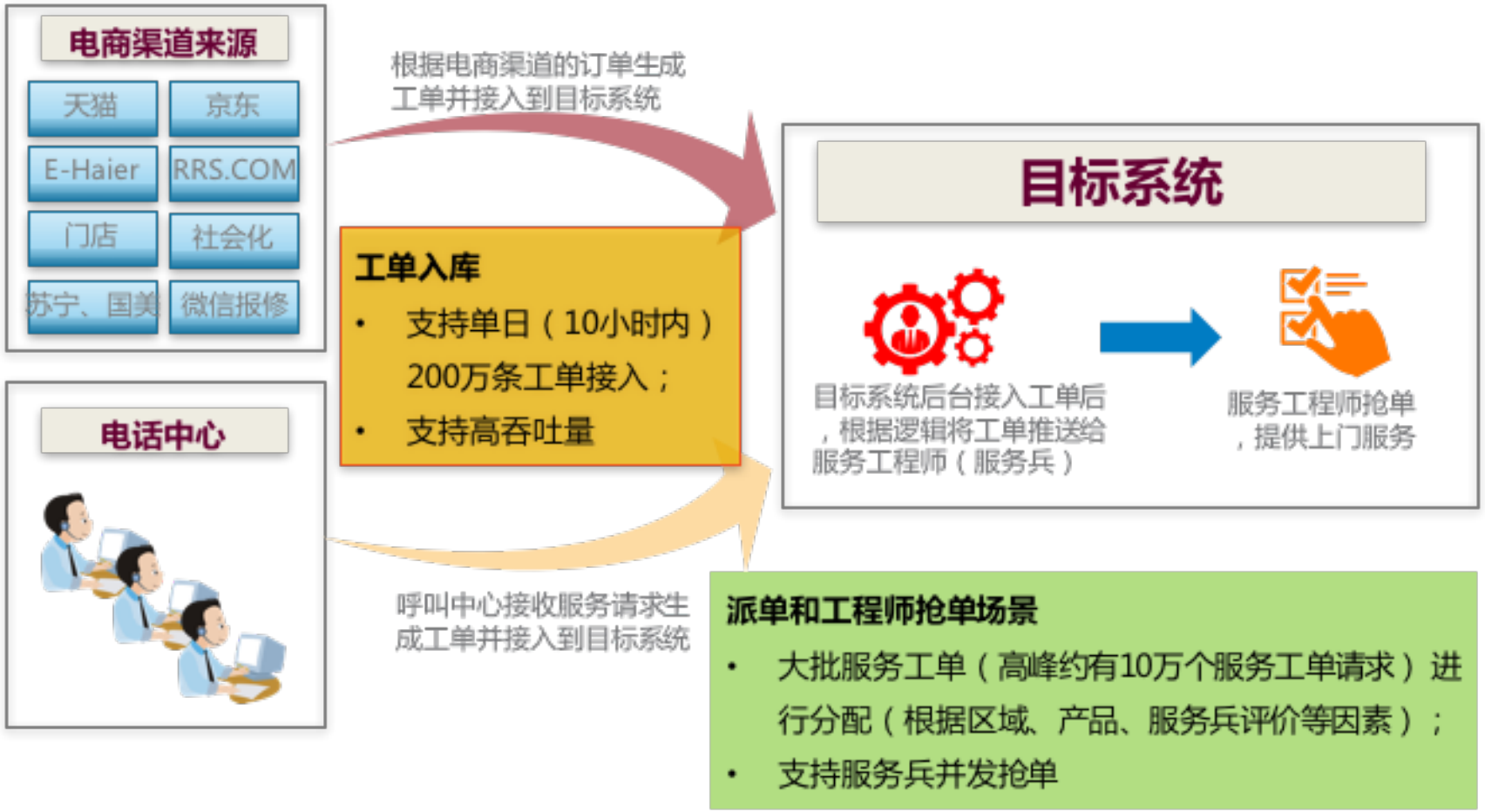
Status

- Went online in 2017-03
- Multi-clusters, each with 12 servers
- Features
  - Scale-out like most NoSQL systems
  - 5000tps for complex workload
  - 5ms latency for key-search
  - Complex queries are answered within 3s
  - Multi-clusters are synchronized with Paxos-like protocol to provide high-availability

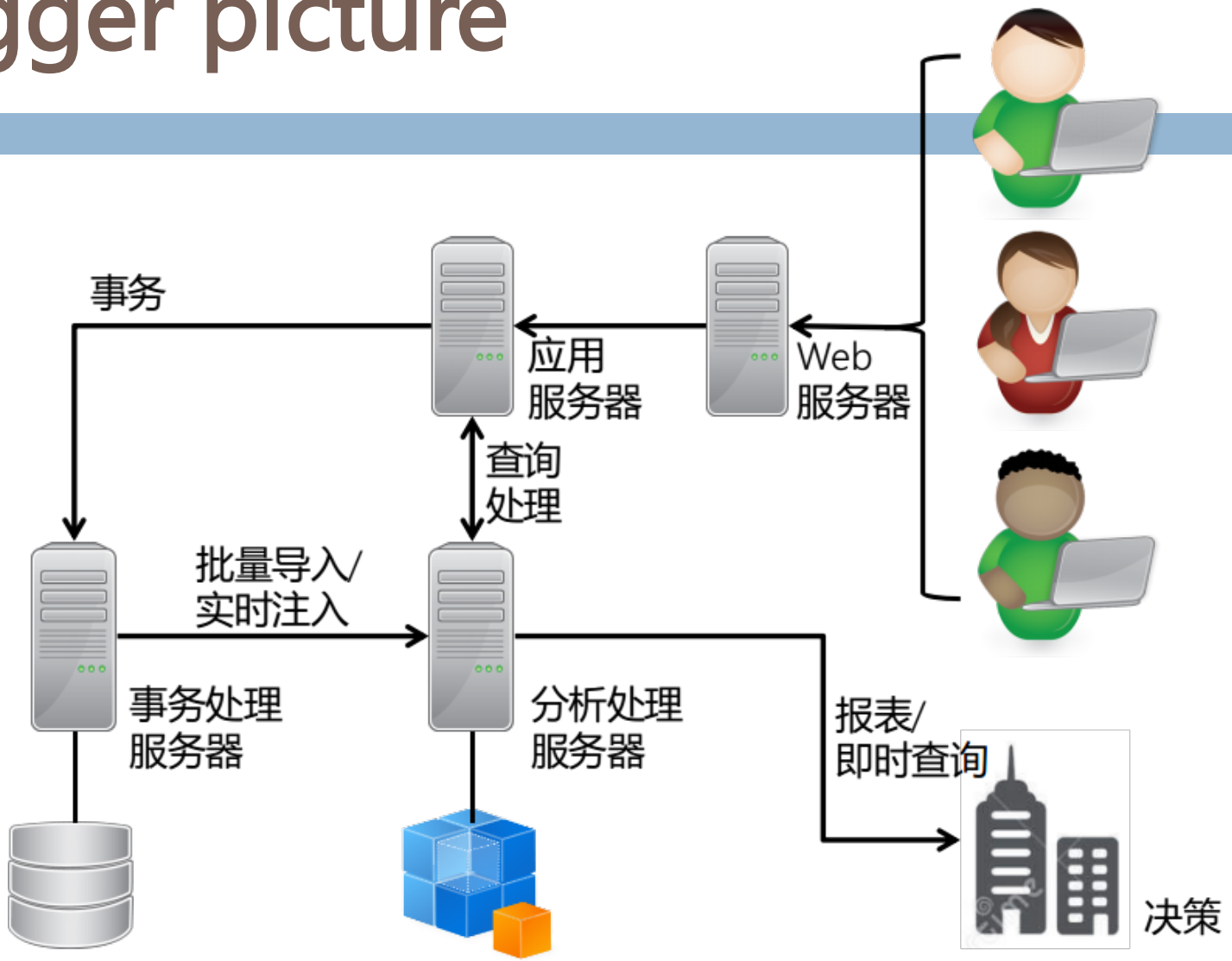
More applications this year

- Network Alliance (网联) e-payment clearinghouse
  - 2017-03 to 2017-04
- Loaning
  - 2017-09 to 2017-12

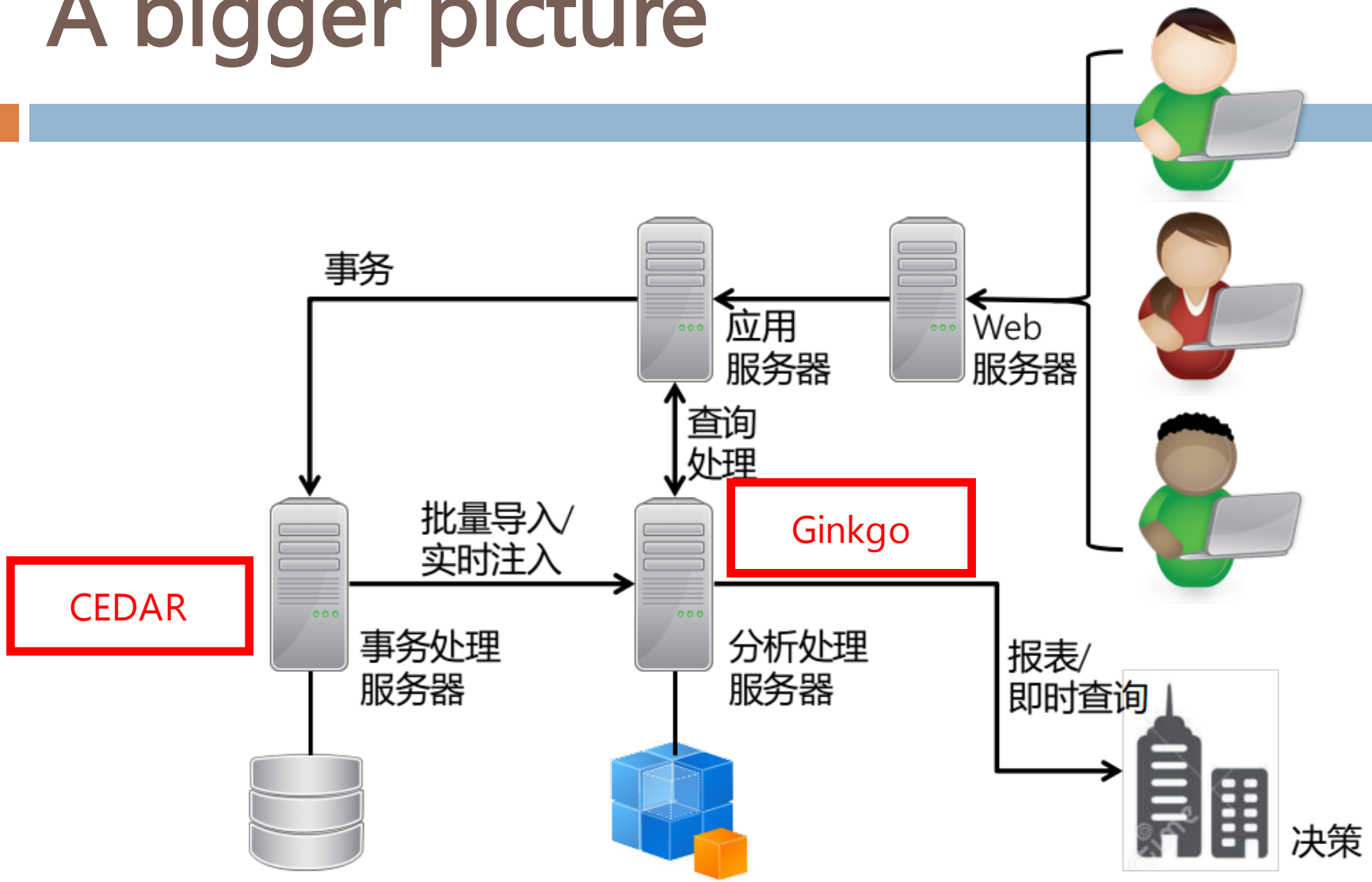
POC: O2O task assignment/taking



A bigger picture



A bigger picture



Summary

CEDAR: 雪松

- C: Cluster-oriented
- E: for Enterprise applications
- D: scalable Dbms
- AR: non-traditional ARchitecture

Summary

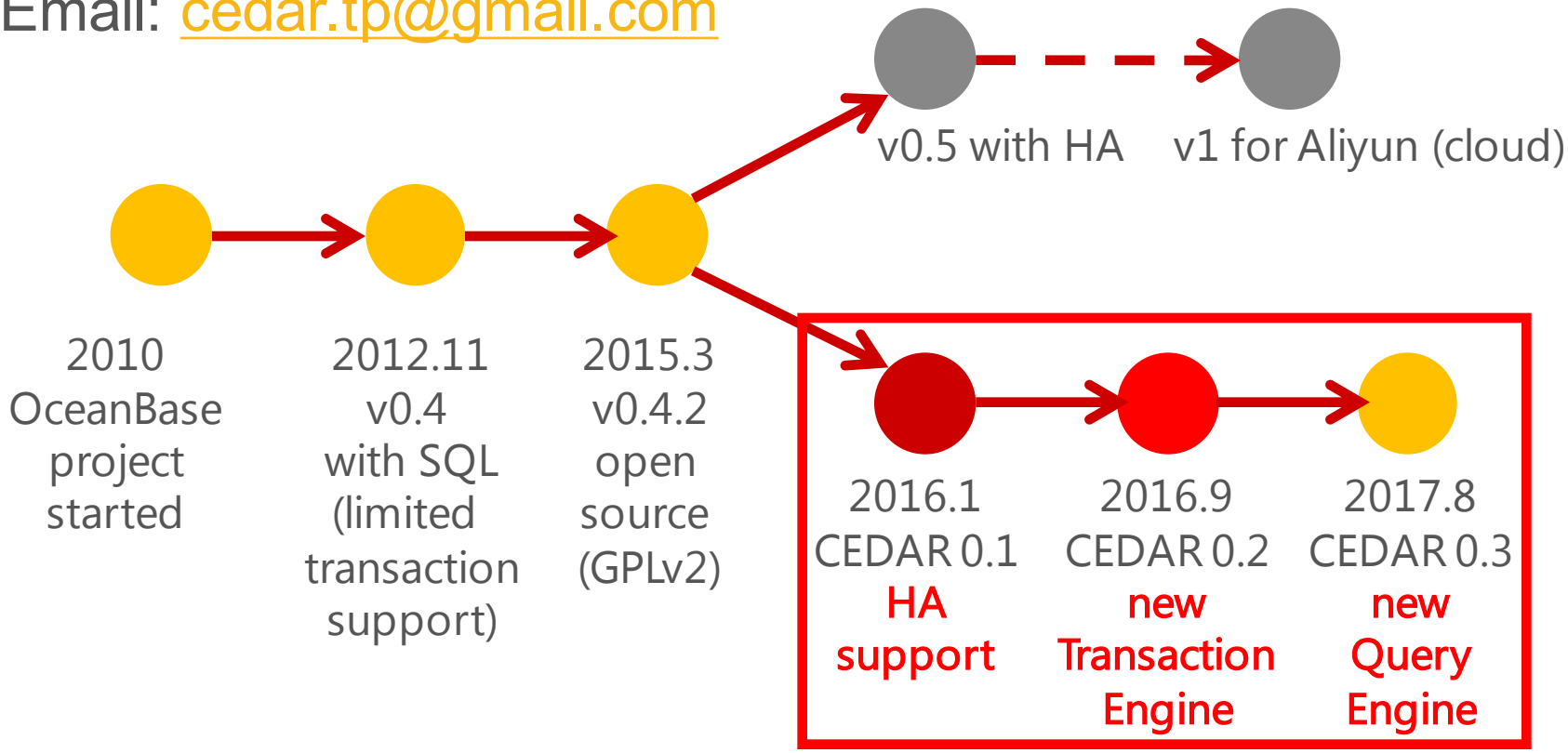
- SQL support: ODBC/API interfaces
- Transaction support: ad-hoc transactions and store-procedures
- Efficient query optimization/execution: various distributed join implementation + indexing schemes
- Deployment with High-Availability support
- Highly Scalable:
  - Read/write separation
  - Hot/cold separation
- Management/maintenance-friendly: Import/export toolkit and monitoring/diagnose toolkit

Summary

- Full-fledged DBMS
  - with SQL and Transaction Processing support
- Scable architecture
  - cluster-oriented: commodity PC server with large memory, SSD drive, and high-speed network
- Mission-critical-app.-oriented
  - apps in enterprises, banks, communications, etc.
- GPLv2

Thanks!

Homepage: https://github.com/daseECNU/CEDAR
Email: cedar.tp@gmail.com



Acknowledgement

華東師範大學 中國人民大學 印孚瑟斯

数据科学联合实验室

ECNU-RUC-Infosys Data Science Joint Lab

Thanks!