

# H-Store:

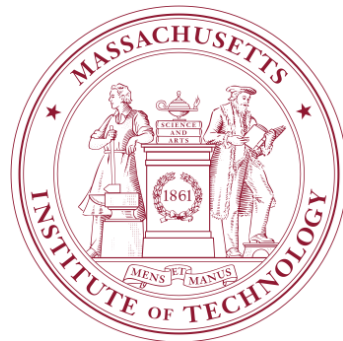
## A Specialized Architecture for High-throughput OLTP Applications

Evan Jones (MIT)

Andrew Pavlo (Brown)

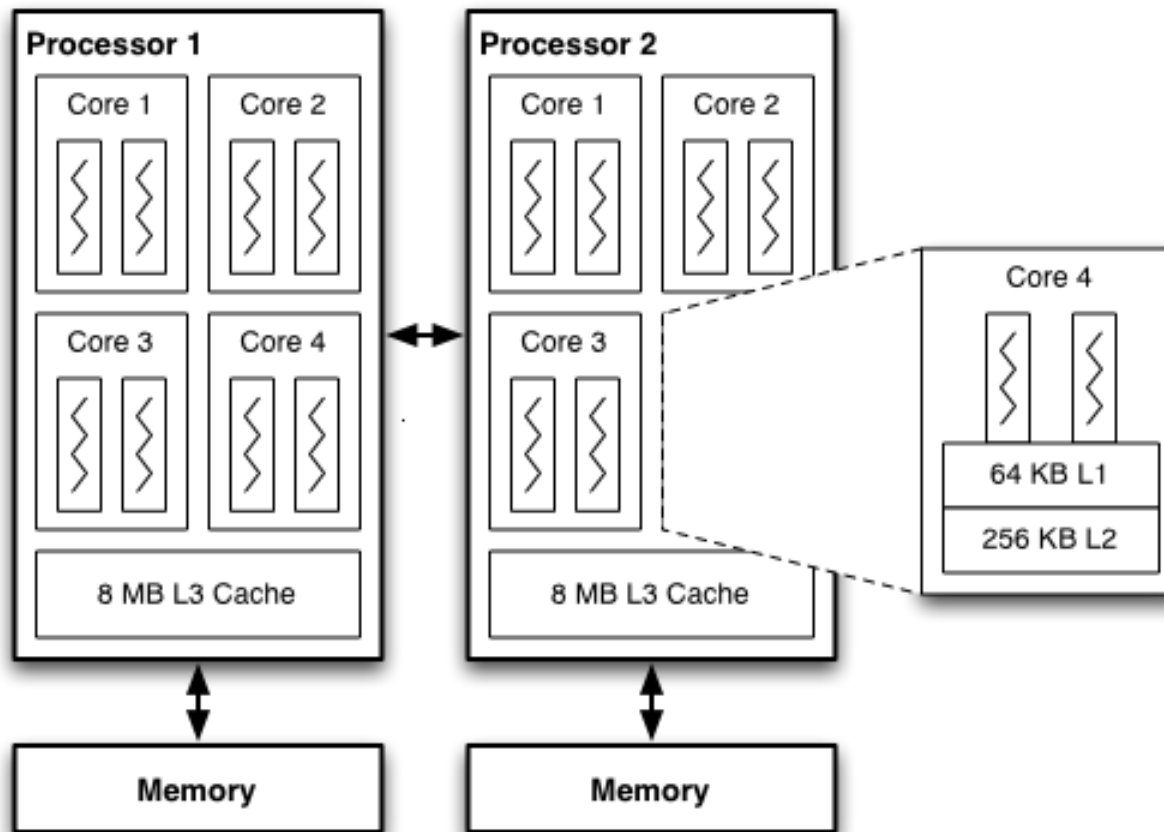
13<sup>th</sup> Intl. Workshop on High Performance Transaction Systems

October 26, 2009



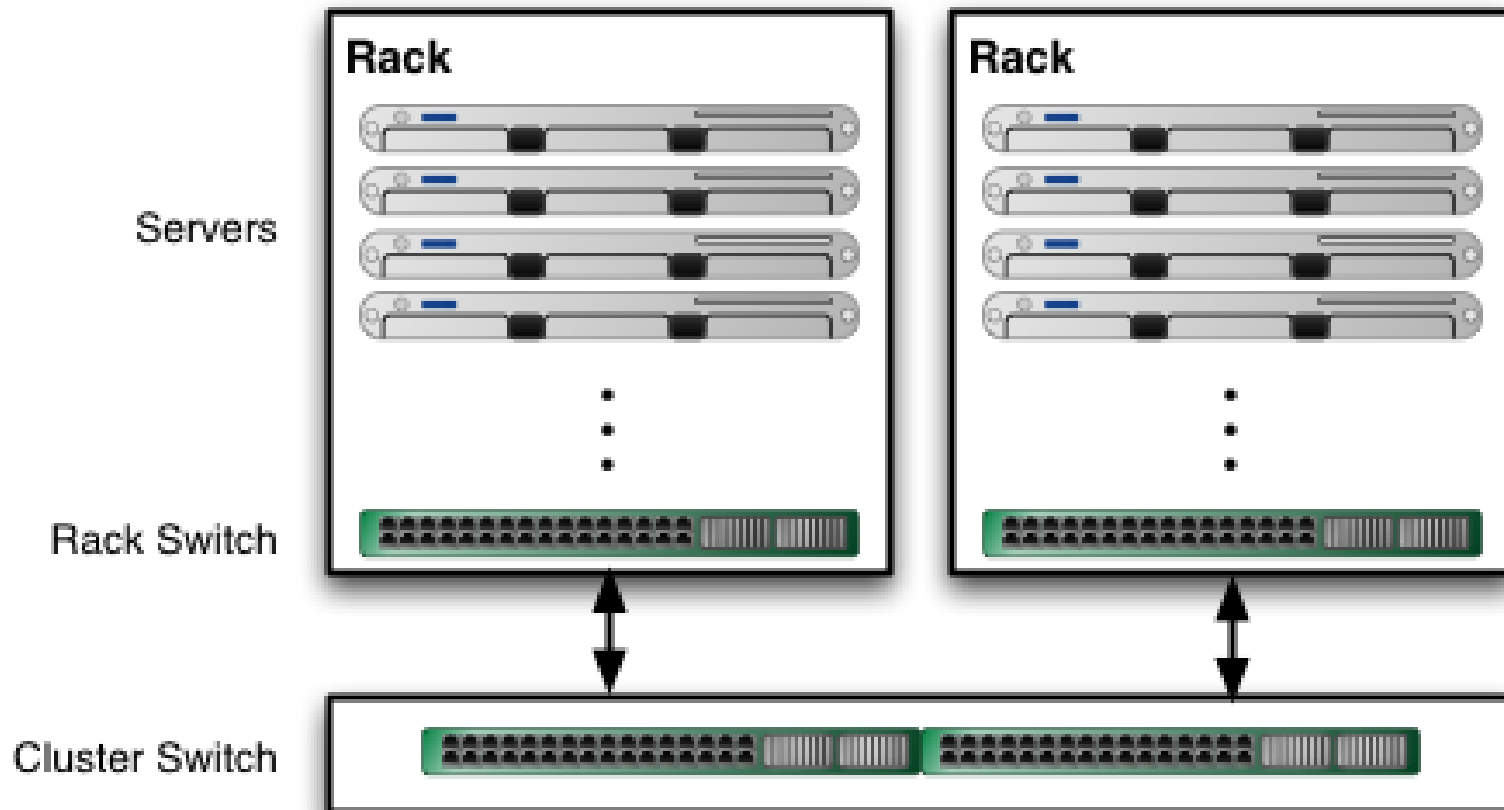
BROWN

# Intel Xeon E5540 (Nehalem/Core i7)



Source: Intel 64 and IA-32 Architectures Optimization Reference Manual

# Distributed Clusters



# Scaling OLTP on Multi-Core?

*Use a distributed shared-nothing design*

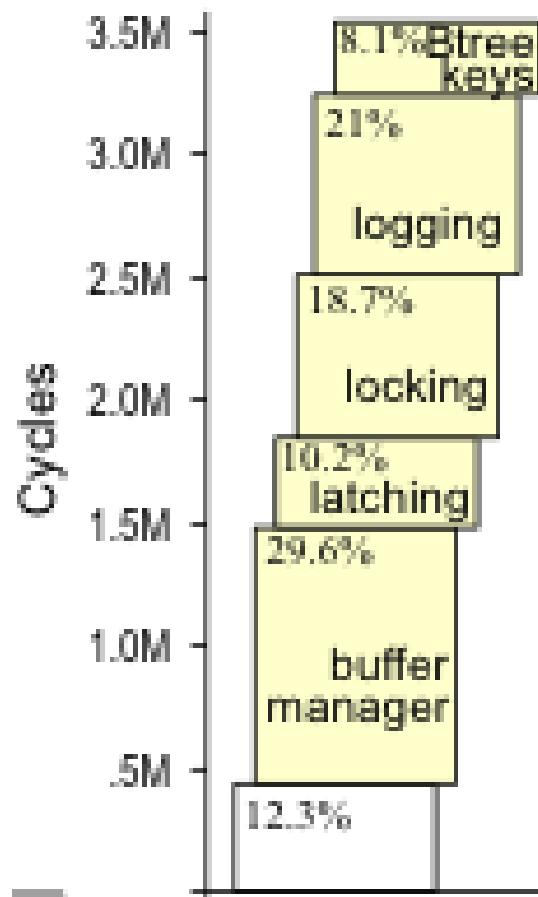


# How to Make a Faster OLTP DBMS

- Main memory storage
  - *Replication for durability*
- Explicitly partition the data
- Specialized concurrency control
  - *Stored procedures only*
  - *Single partition: execute one transaction at time*
  - *Multiple partitions: supported but slow*



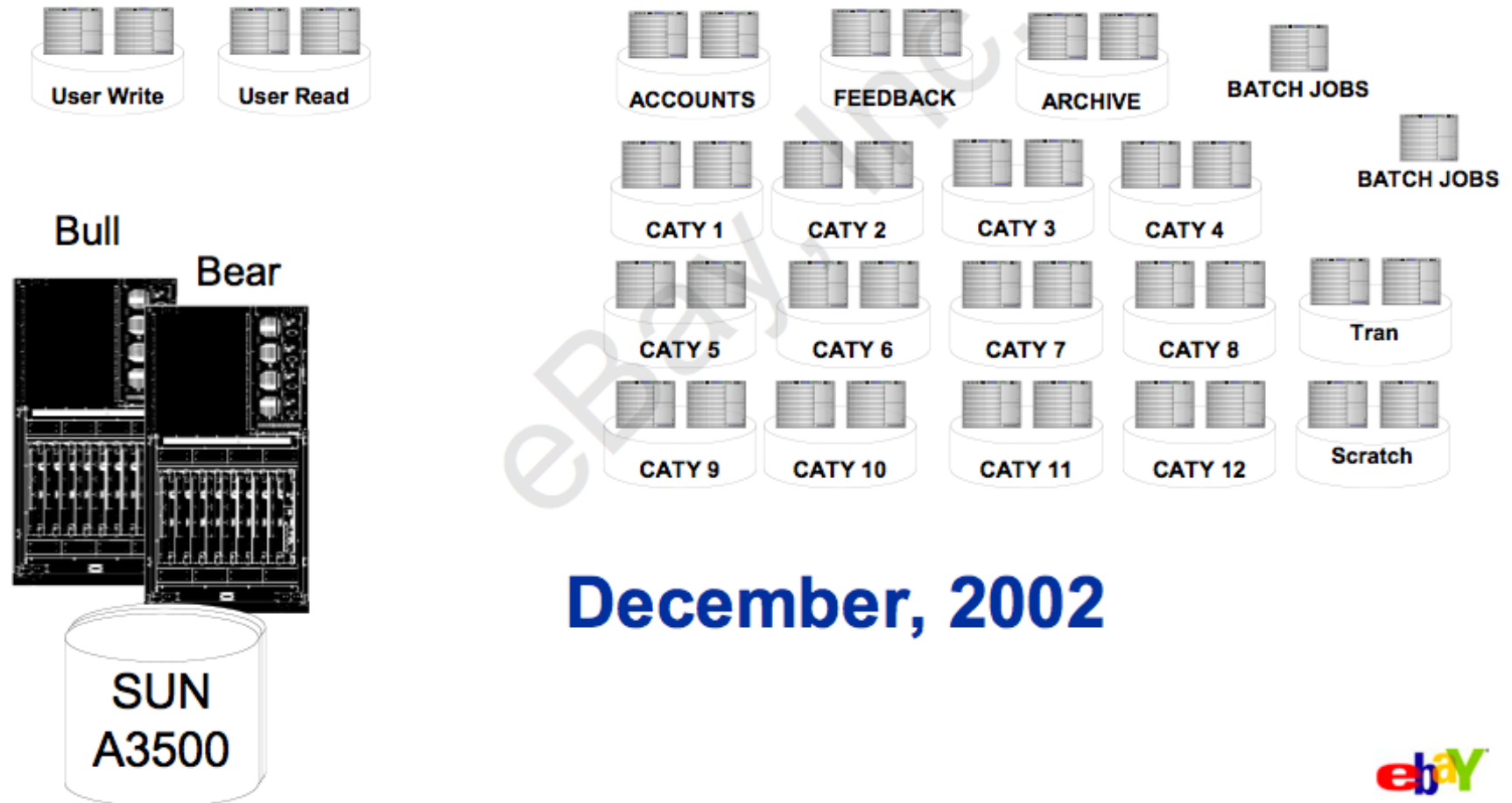
# OLTP: Where does the time go?



Source: S. Harizopoulos, D. J. Abadi, S. Madden, M. Stonebraker, "OLTP Under the Looking Glass", SIGMOD 2008.



# Users Rely on Partitioning



**December, 2002**

Source: R. Shoup, D. Pritchett, "The eBay Architecture," SD Forum, Nov. 2006.



© 2006 eBay Inc.



BROWN

# What about multi-core?

- **Traditional approach:**
  - *One database process*
  - *Thread per connection*
  - *Shared-memory, locks and latches*
- **H-Store approach:**
  - *Thread per partition*
  - *Distributed transactions*





# Example Microbenchmark

- One table per client

```
Table(id INTEGER, counter INTEGER)
```

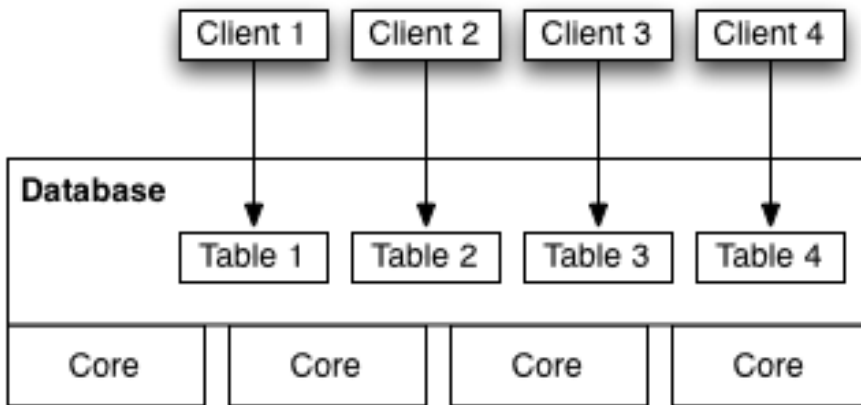
- Each client executes the following query:

```
UPDATE Table  
  SET counter = counter + 1  
  WHERE id = 0;
```

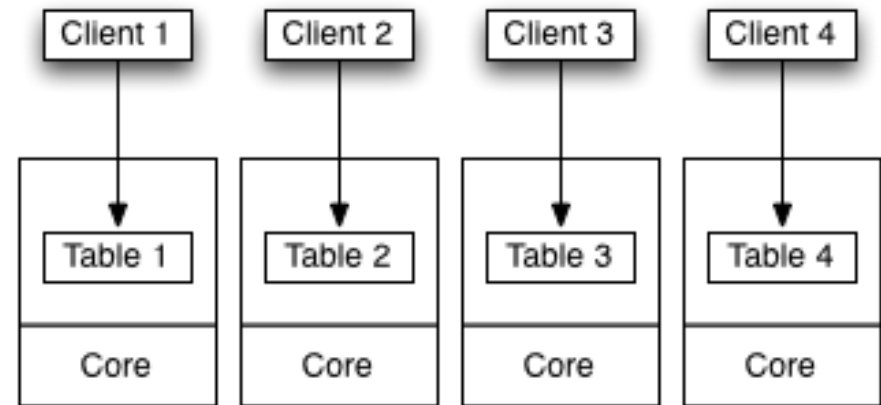
- Add clients to find maximum throughput
- Data on RAM disk



# Experimental Configuration



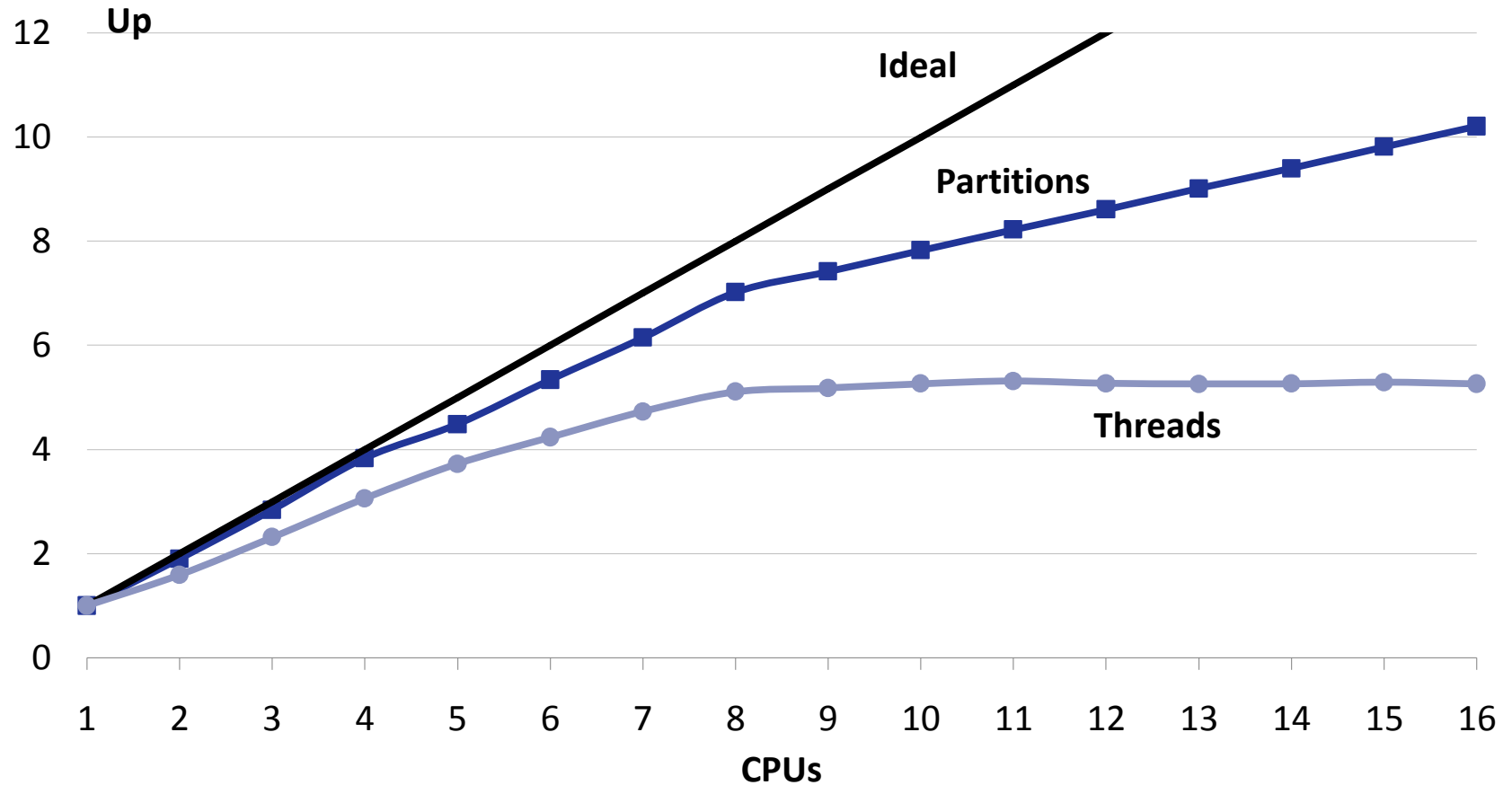
**Threads**



**Partitions**

# Partitions versus Threads

Relative Speed



# Scalability Analysis

Partitions scale better than threads.

- **Threads: contention for shared resources [1]**
- **Partitions: memory bottleneck causes sublinear scaling**

***H-Store: Not just for distributed shared-nothing clusters***

[1] R. Johnson et al., "Shore-MT: A Scalable Storage Manager for the Multicore Era," EDBT 2009.



# Multi-core Design Problem

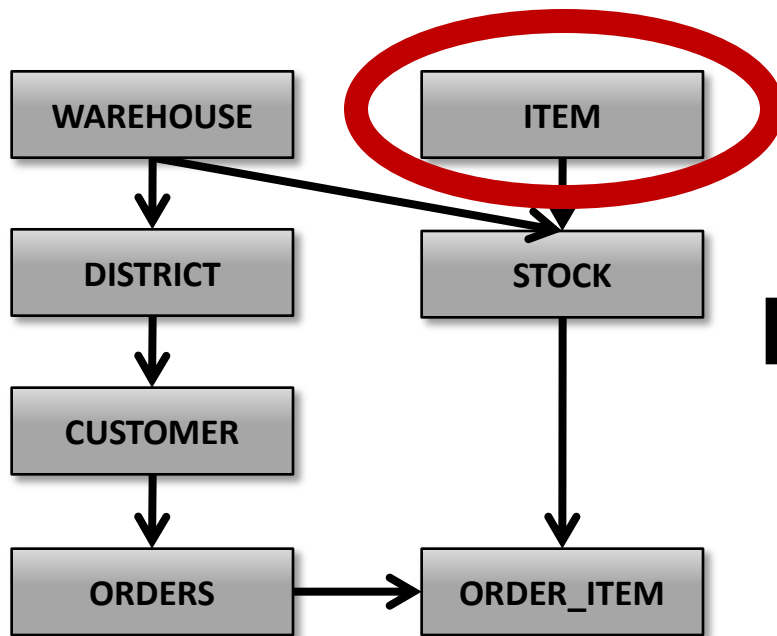
- How to automatically create a data placement scheme to improve multi-core throughput?
- Data Partitioning:
  - *Maximize the number of single-partition transactions.*
- Data Placement:
  - *Maximize the number of single-node transactions.*



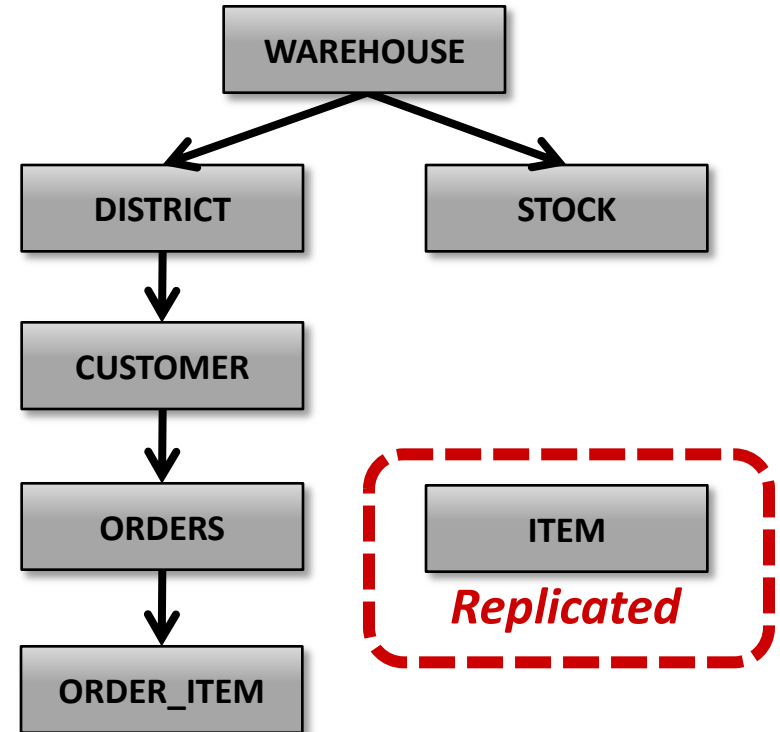
# Database Partitioning

- Select partitioning keys and construct schema tree.

## TPC-C Schema

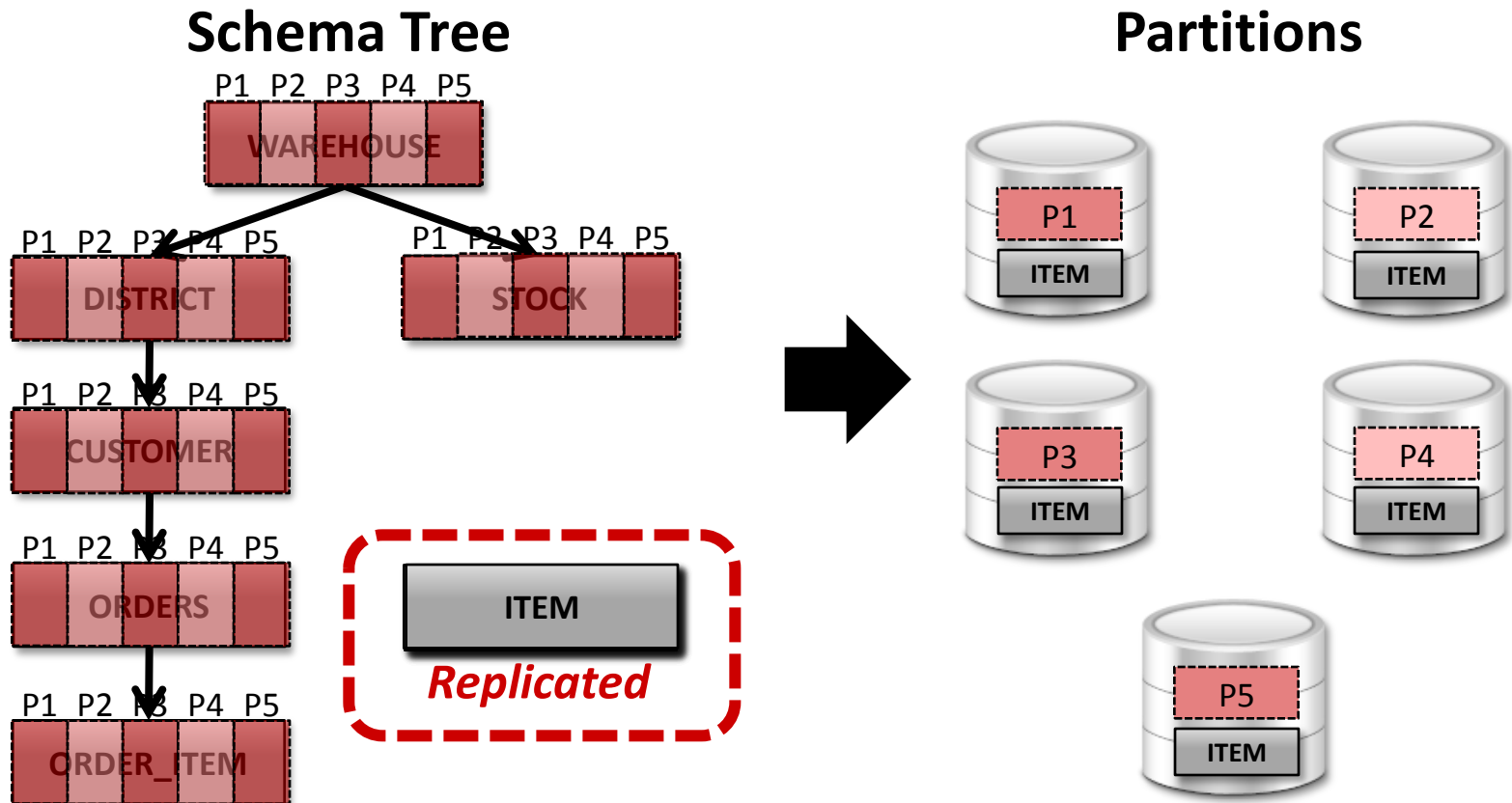


## Schema Tree



# Database Partitioning

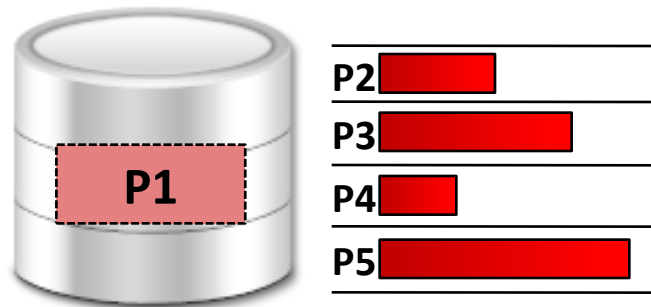
- Combine table fragments into partitions.



# Data Placement

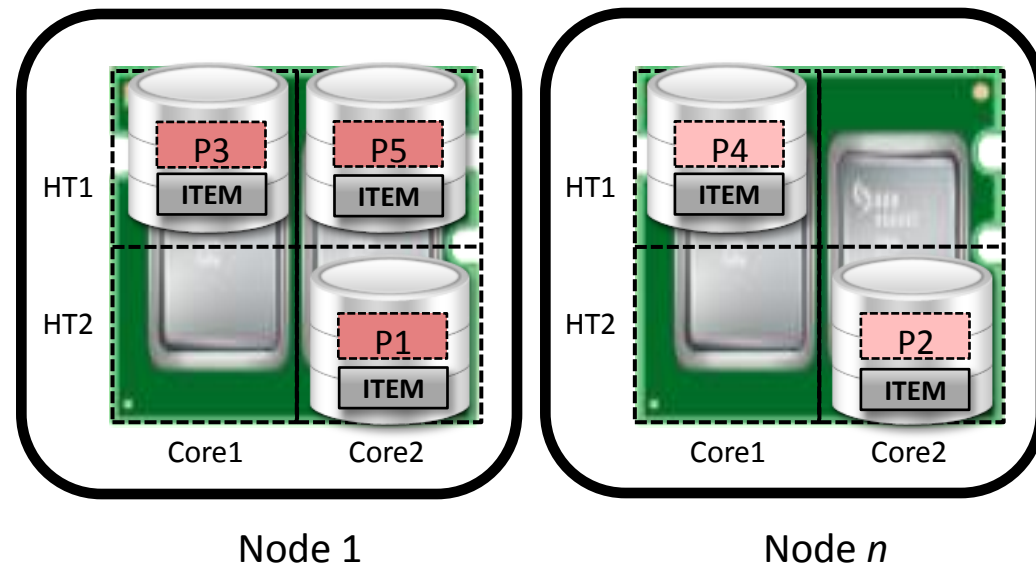
- Assign partitions to cores on each node.

## Partitions



## Partition Affinity

## Cluster Node





# H-Store's Future



- New Name. New Company.
- Six full-time developers.
- Open-source project (GPL)
- Beta by end of 2009
  - *Multiple deployments in financial service areas.*



# More Information

- H-Store Info + Papers:
  - <http://db.cs.yale.edu/hstore/>
- VoltDB Project Information:
  - <http://www.voltdb.org/>

