EVERYTHING I KNOW ABOUT *FAST* DATABASES I LEARNED AT THE DOG TRACK

@andy_pavlo

BROWN
Database Group

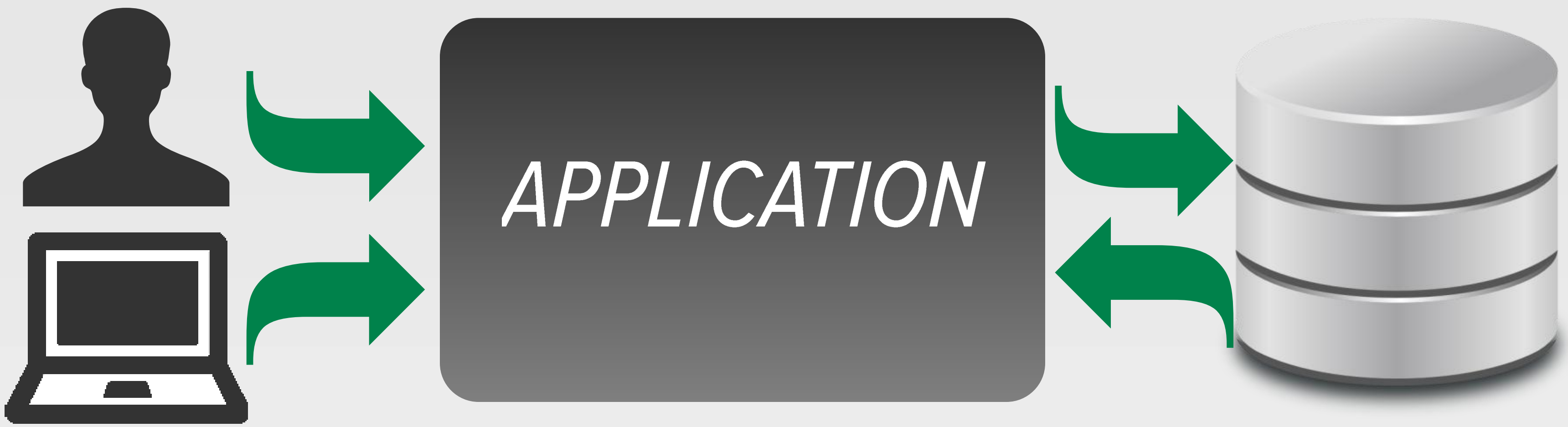
TRADITIONAL

NOSQL

NEWSQL

VOTER BENCHMARK

Japanese "American Idol"

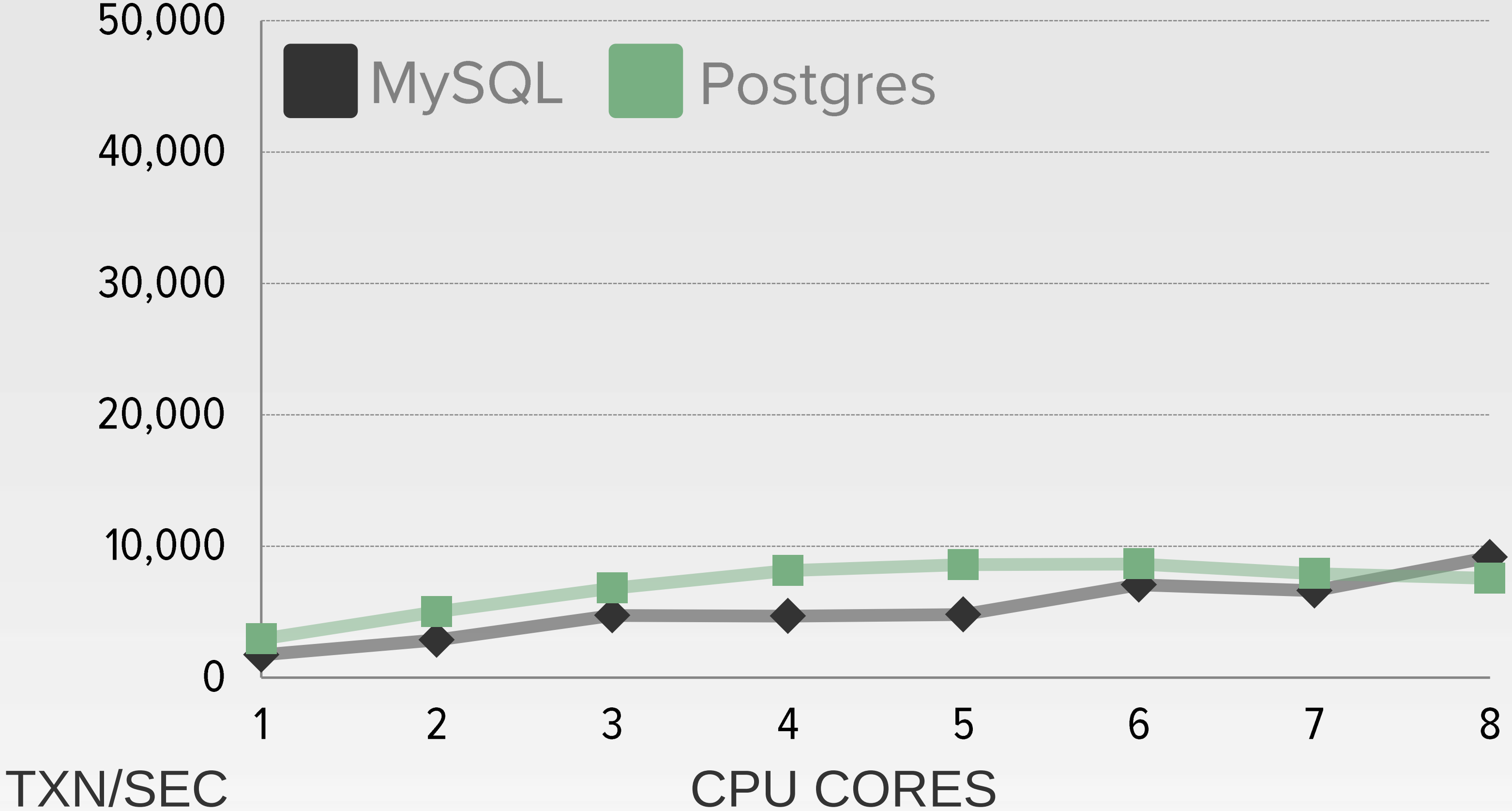


TRANSACTION

1. Check whether user has already voted.
2. Insert new vote entry.
3. Update vote count for contestant.

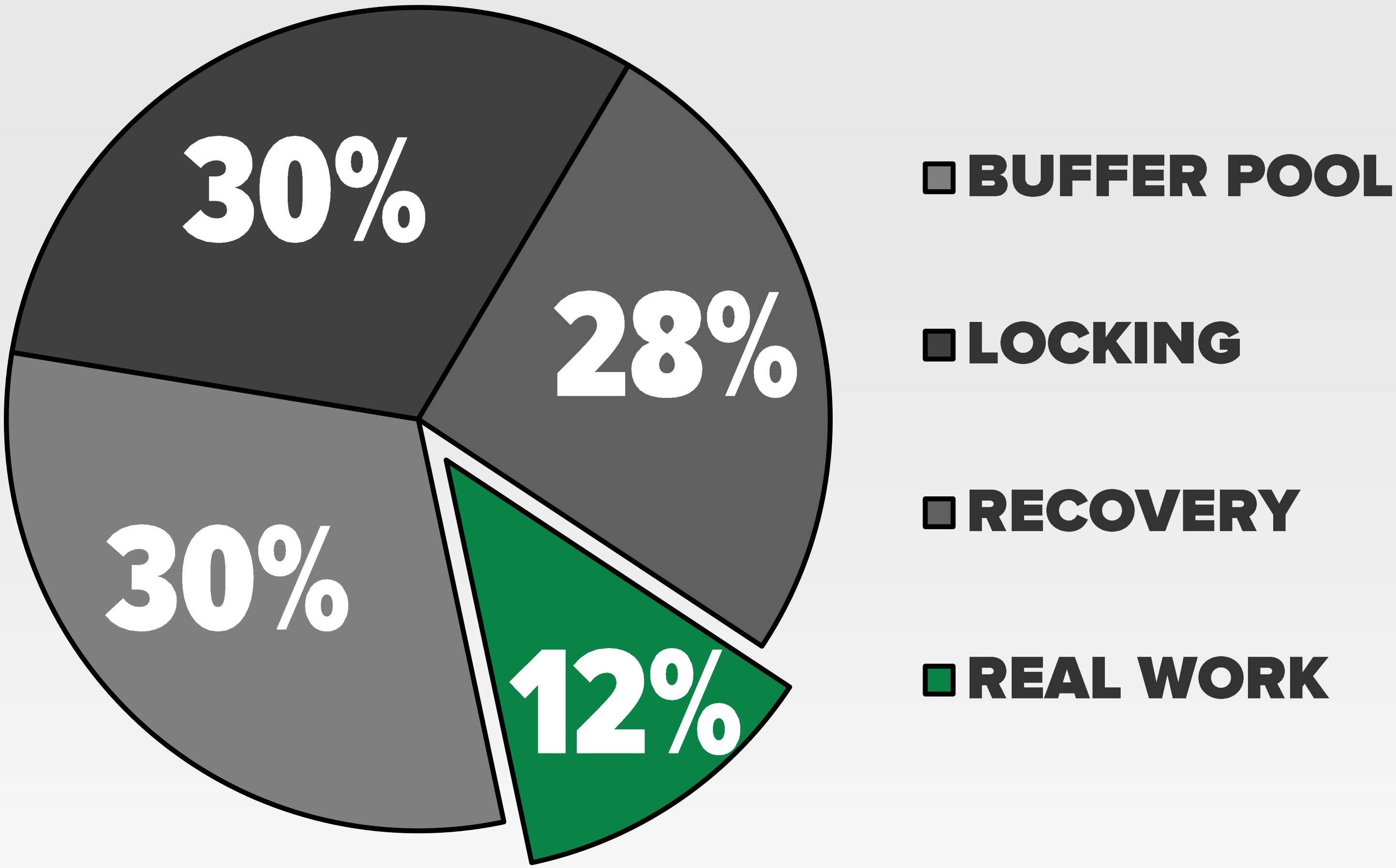
VOTER BENCHMARK

Japanese "American Idol"



TRADITIONAL DBMS

Measured CPU Cycles

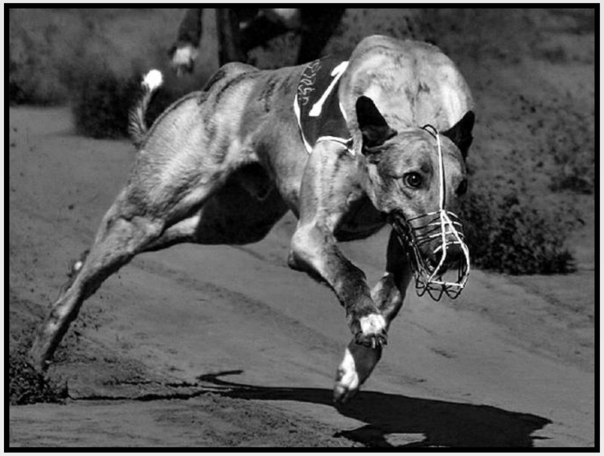


OLTP THROUGH THE LOOKING GLASS,
AND WHAT WE FOUND THERE
SIGMOD, pp. 981-992, 2008.


SCALABILITY
HIGH
(Many Nodes)
LOW
(One Node)
NoSQL
NewSQL
TRADITIONAL
DBMS
WEAK
(None/Limited)
GUARANTEES
STRONG
(ACID)

CAN YOU *SCALE UP* WITHOUT *GIVING UP* TRANSACTIONS?


DOG RACING



Fast

Repetitive

Small

Optimization

USE A LIGHTWEIGHT SYSTEM ***DESIGNED*** FOR OLTP TRANSACTIONS.

H-STORE: A HIGH-PERFORMANCE, DISTRIBUTED MAIN MEMORY TRANSACTION PROCESSING SYSTEM
Proc. VLDB Endow., vol. 1, iss. 2, pp. 1496-1499, 2008.

DISK ORIENTED

MAIN MEMORY STORAGE

CONCURRENT EXECUTION

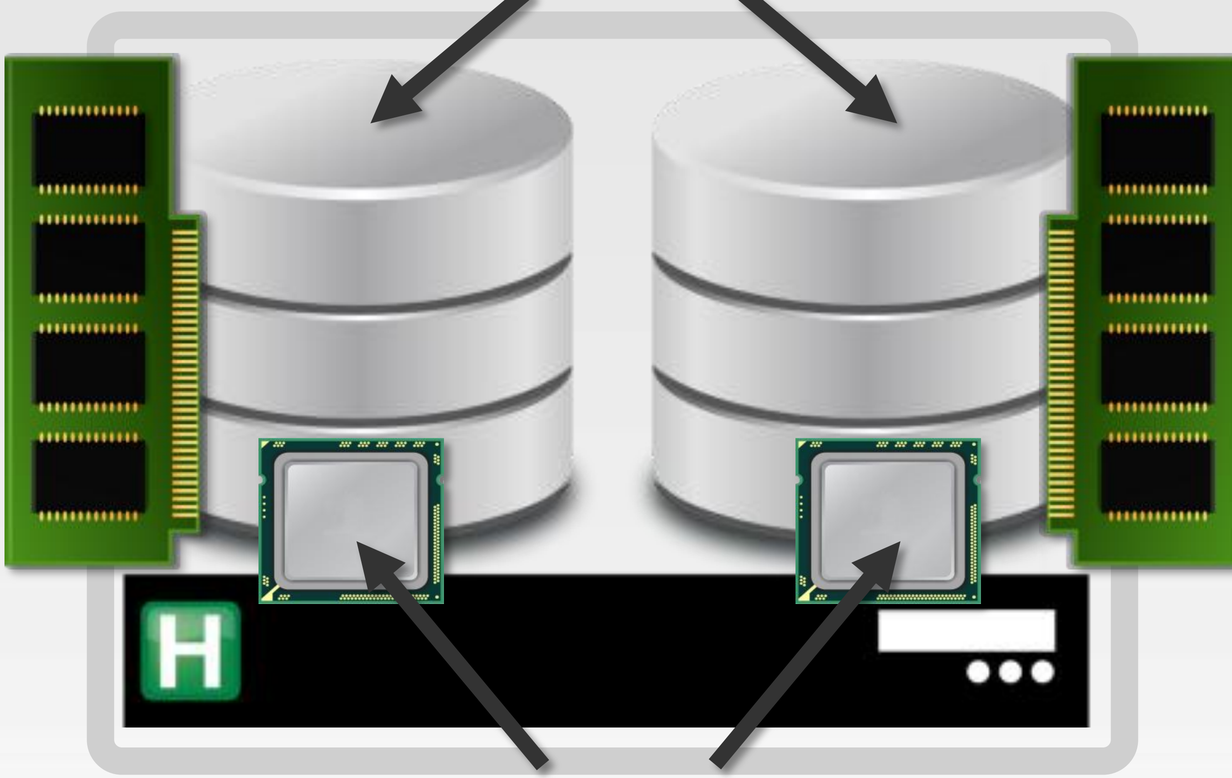
SERIAL EXECUTION

HEAVYWEIGHT RECOVERY

COMPACT LOGGING

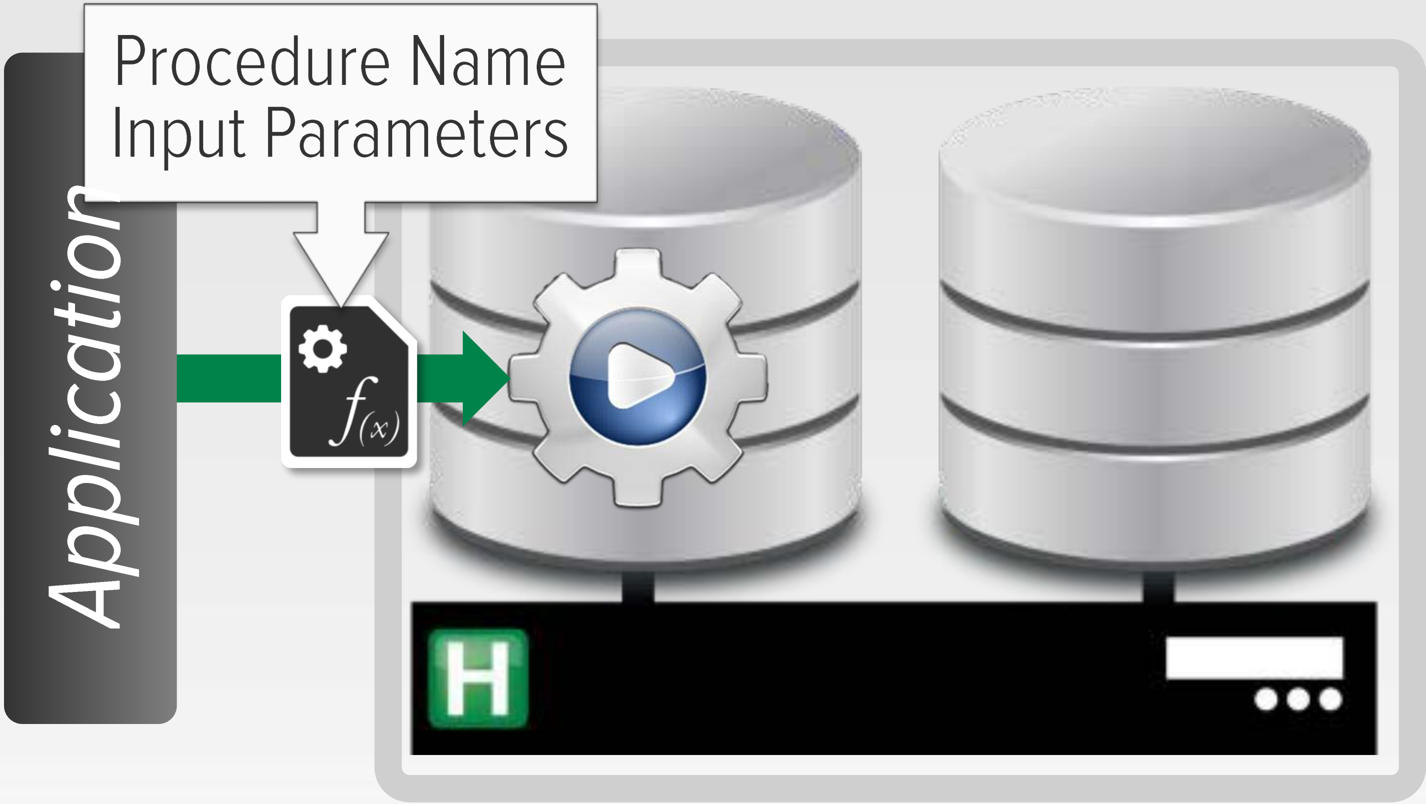
H-Store

PARTITIONS

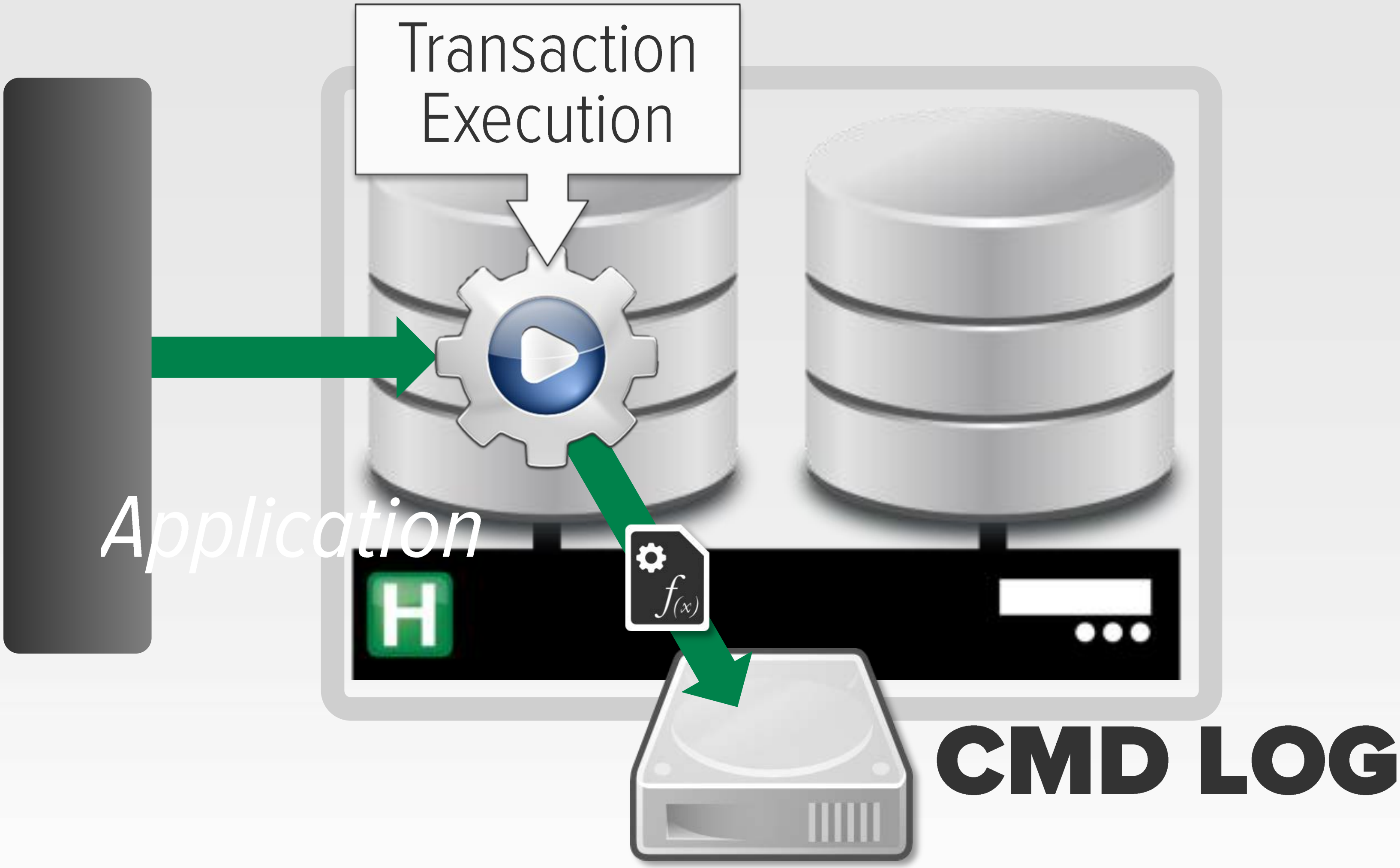


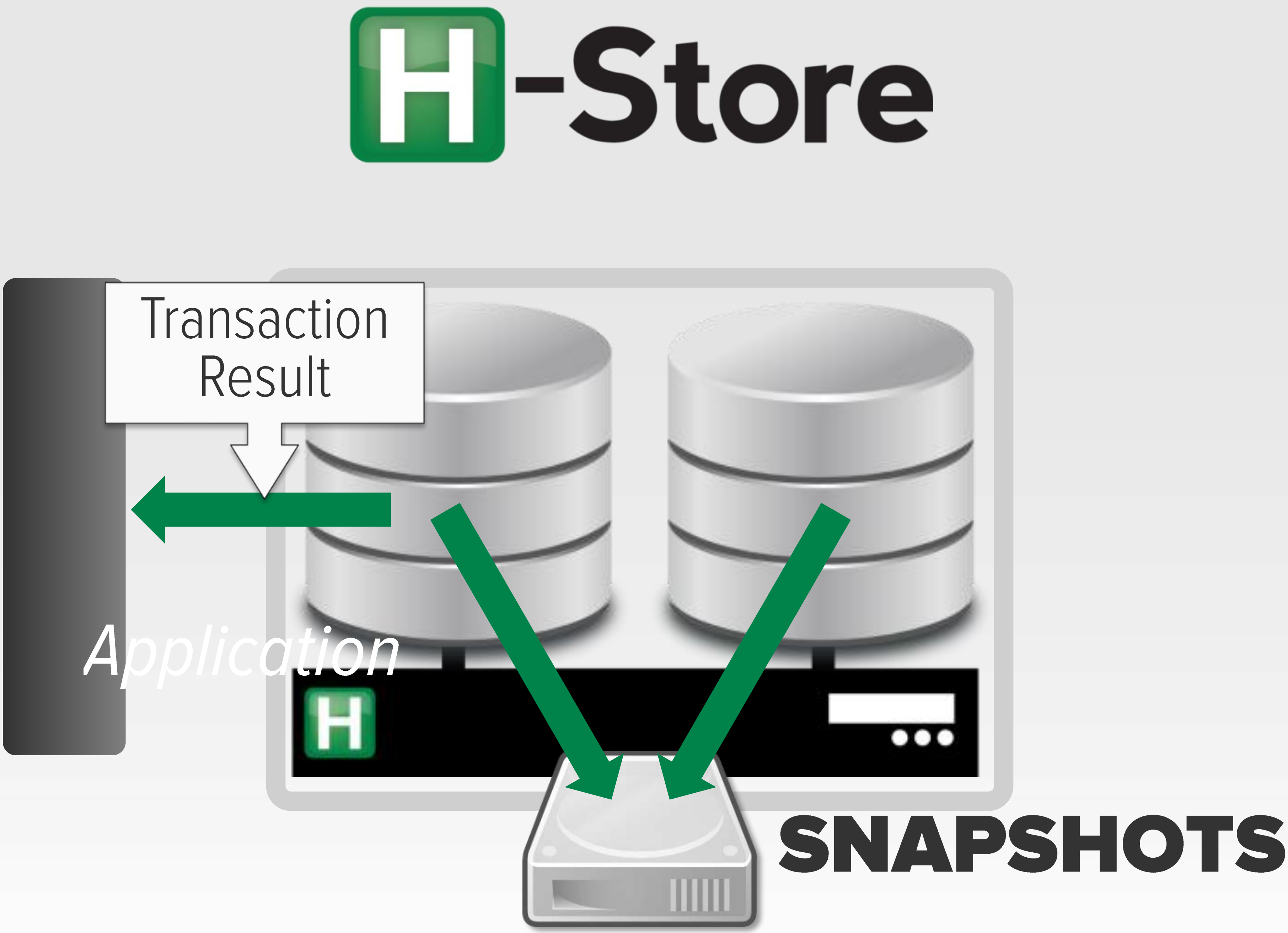
SINGLE-THREADED EXECUTION ENGINES

H-Store



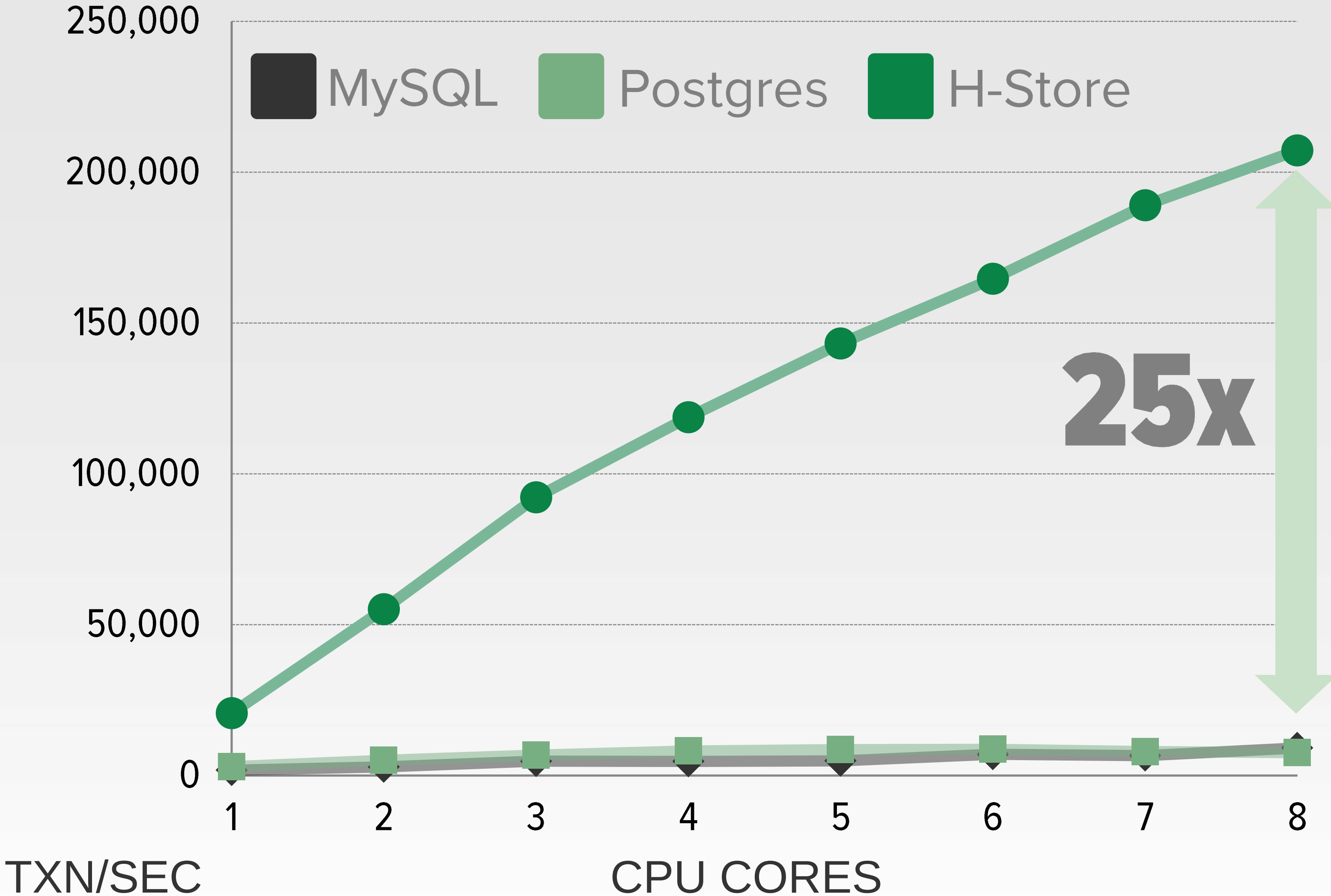
H-Store




H-Store
Transaction
Result
Application
SNAPSHOTS

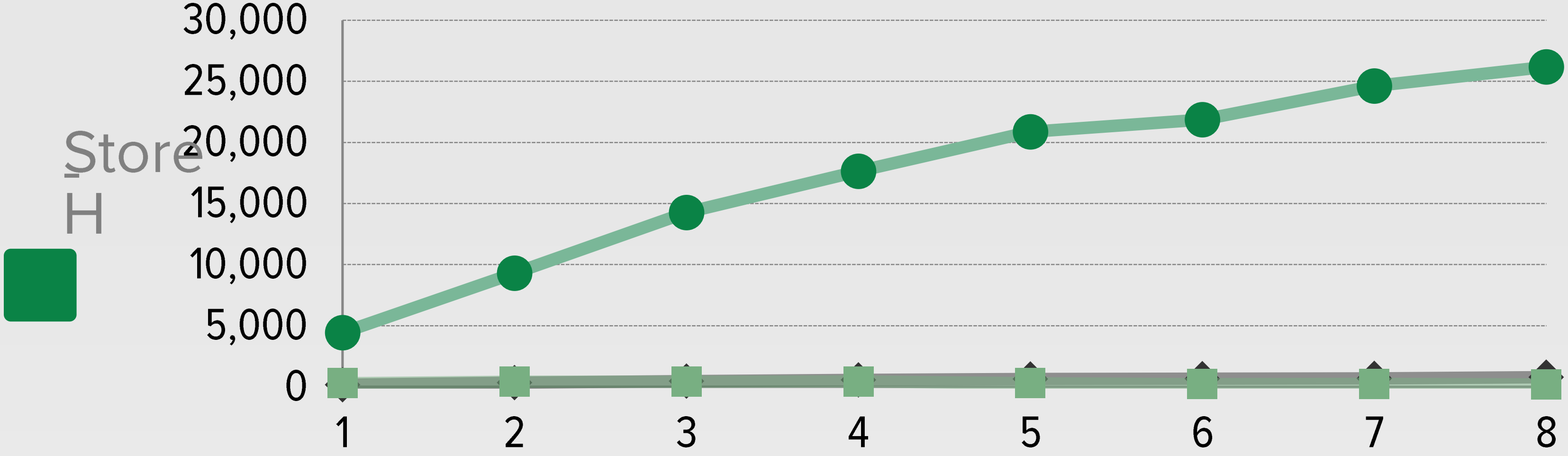
VOTER BENCHMARK

Japanese "American Idol"

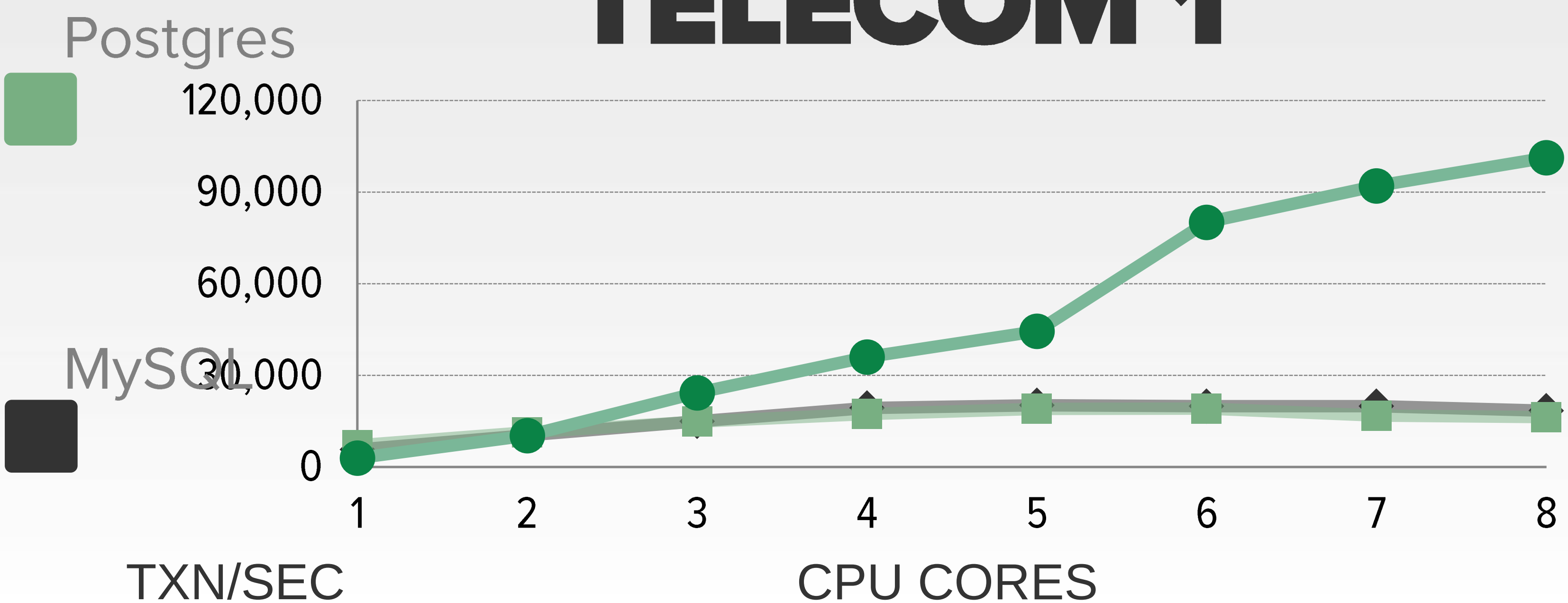


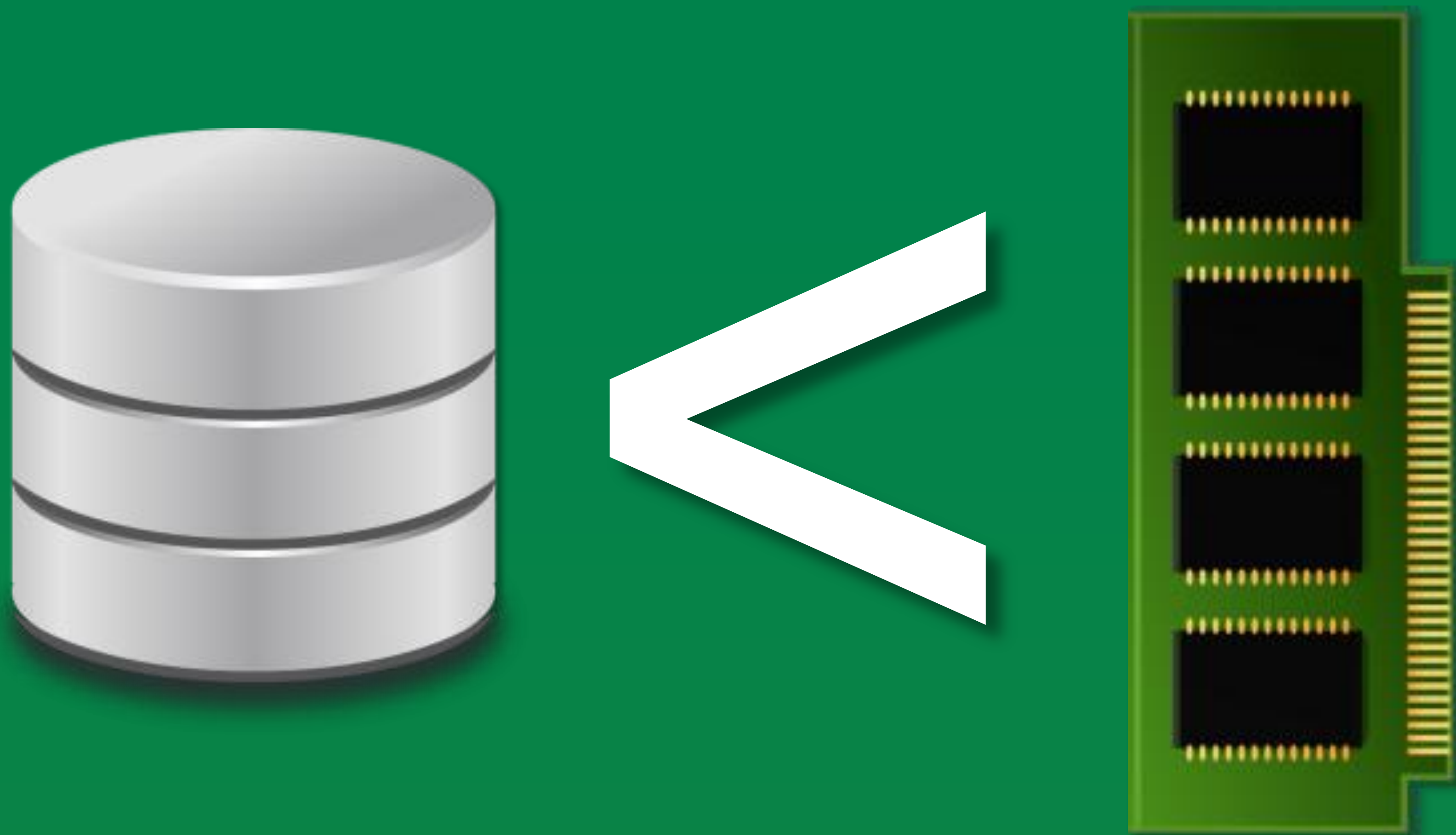
H-Store

TPC-C

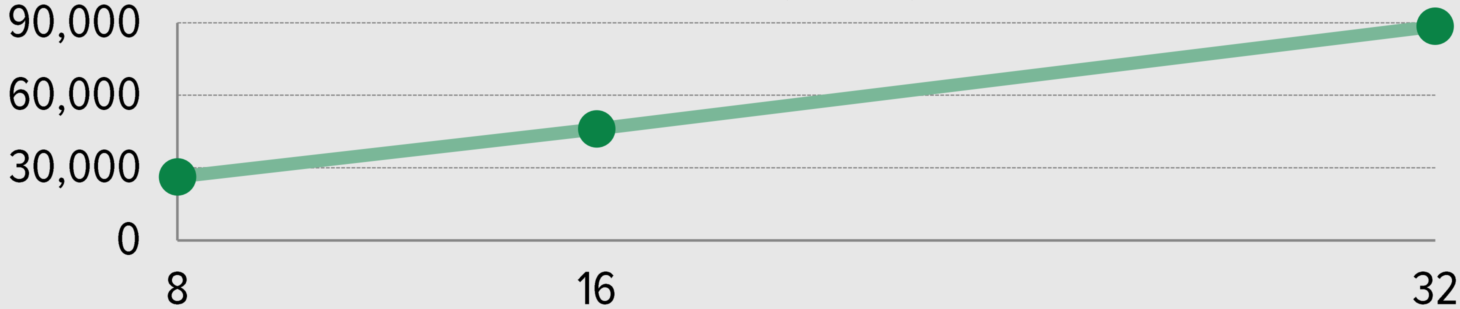


TELECOM 1

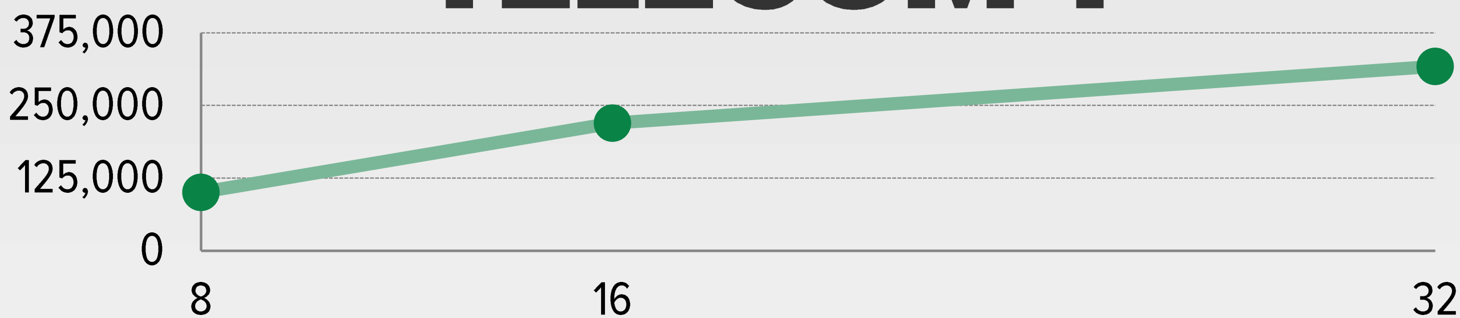




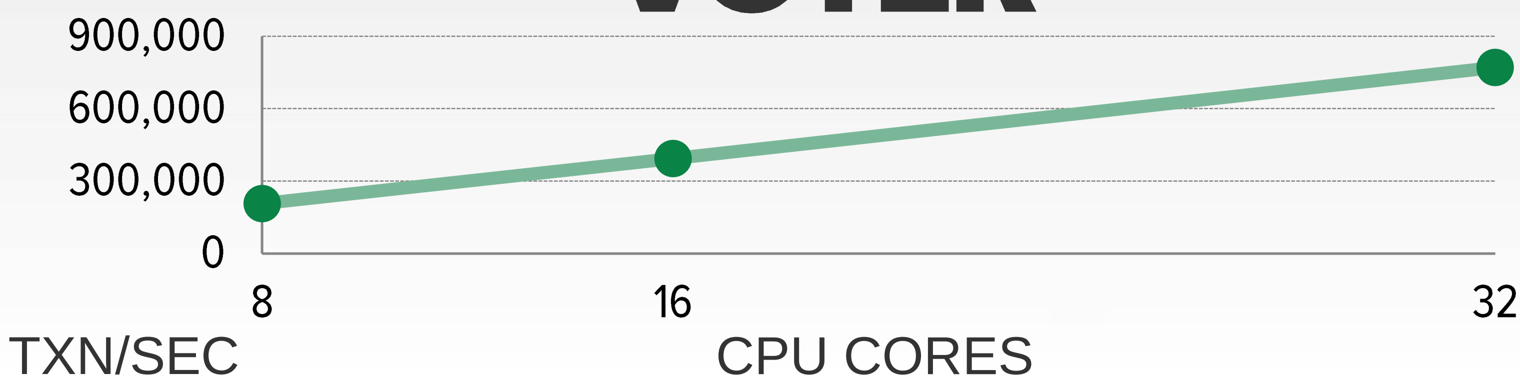
TPC-C



TELECOM 1



VOTER

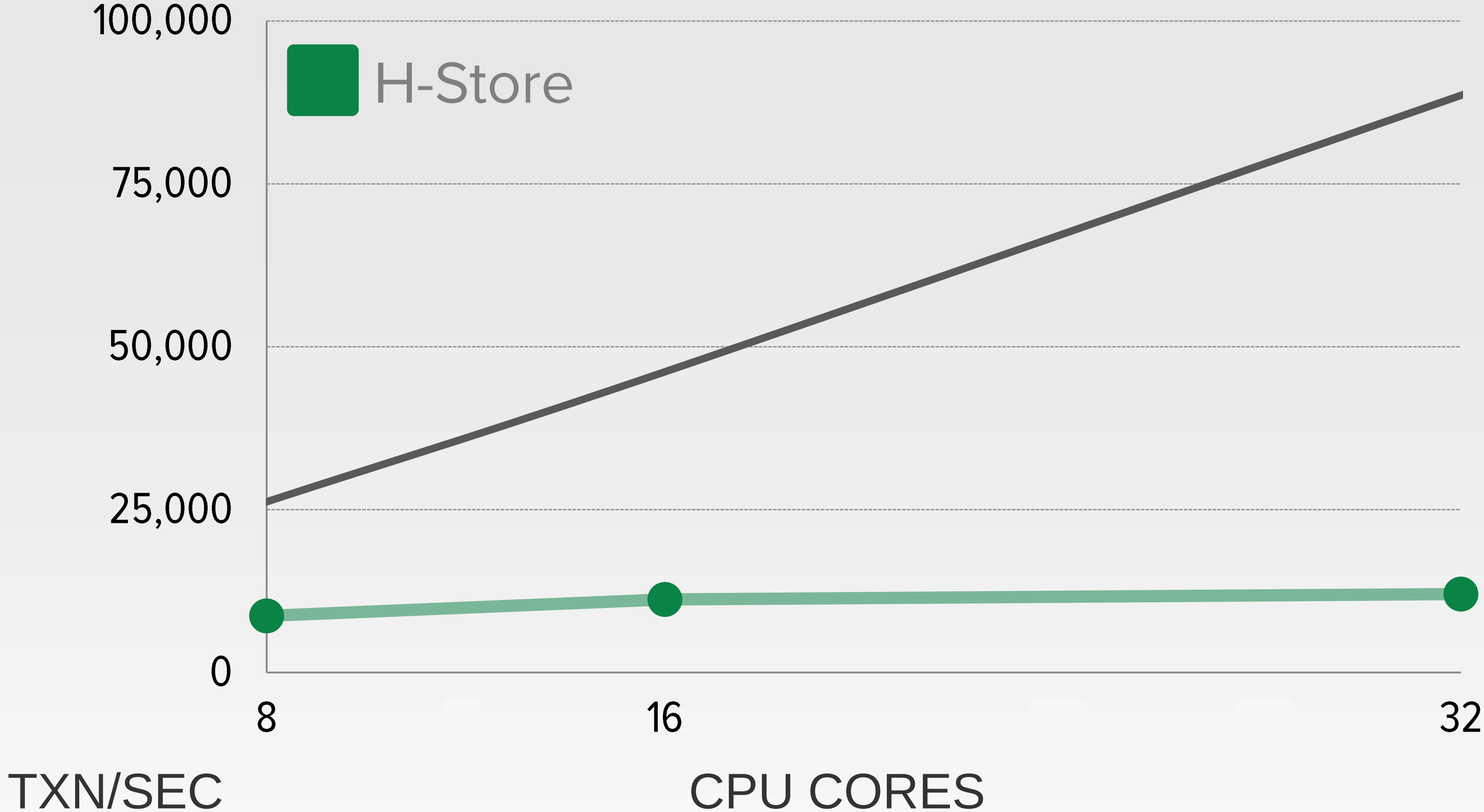


DISTRIBUTED TRANSACTIONS

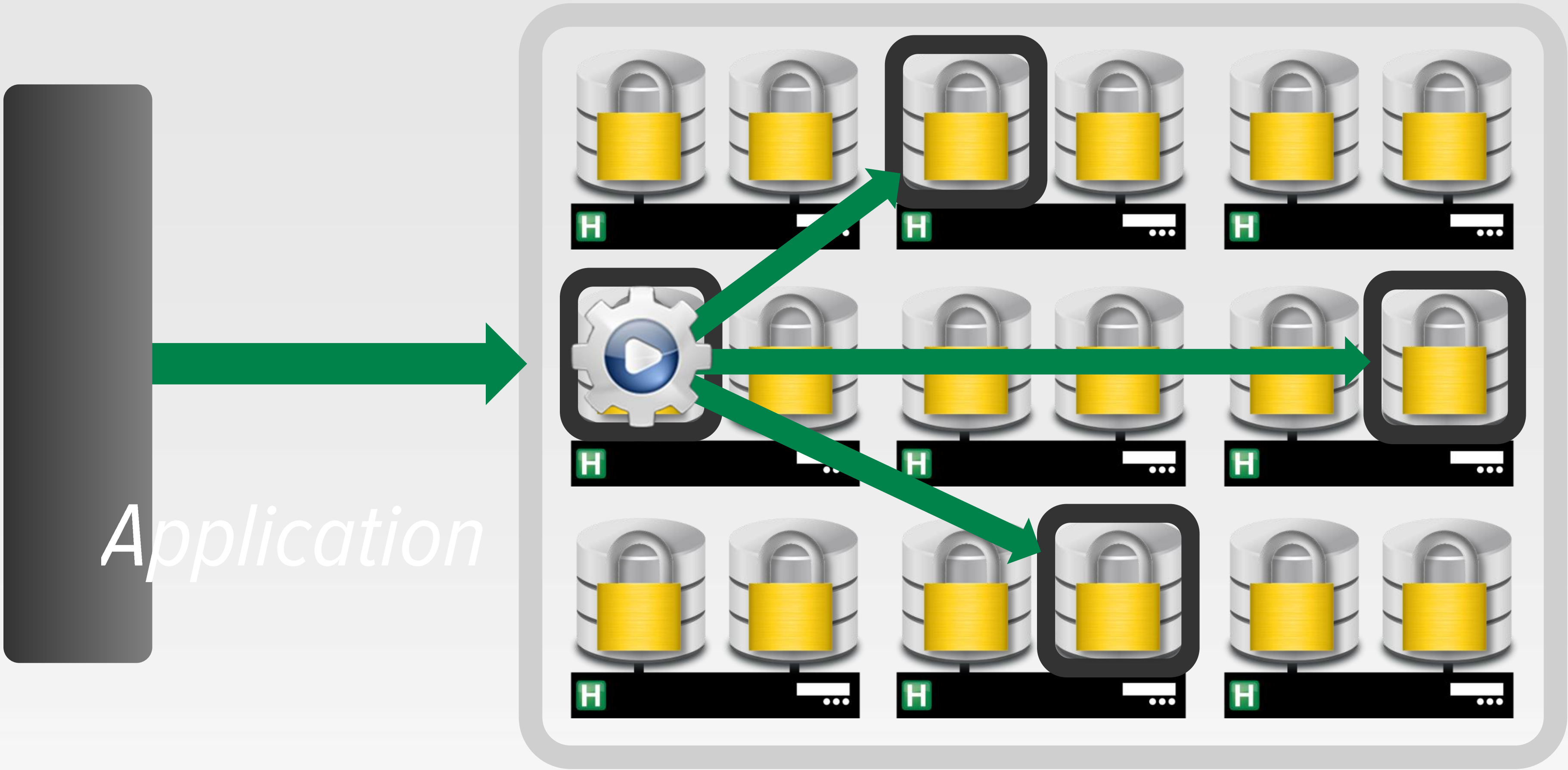
TPC-C BENCHMARK

8 Cores per Node

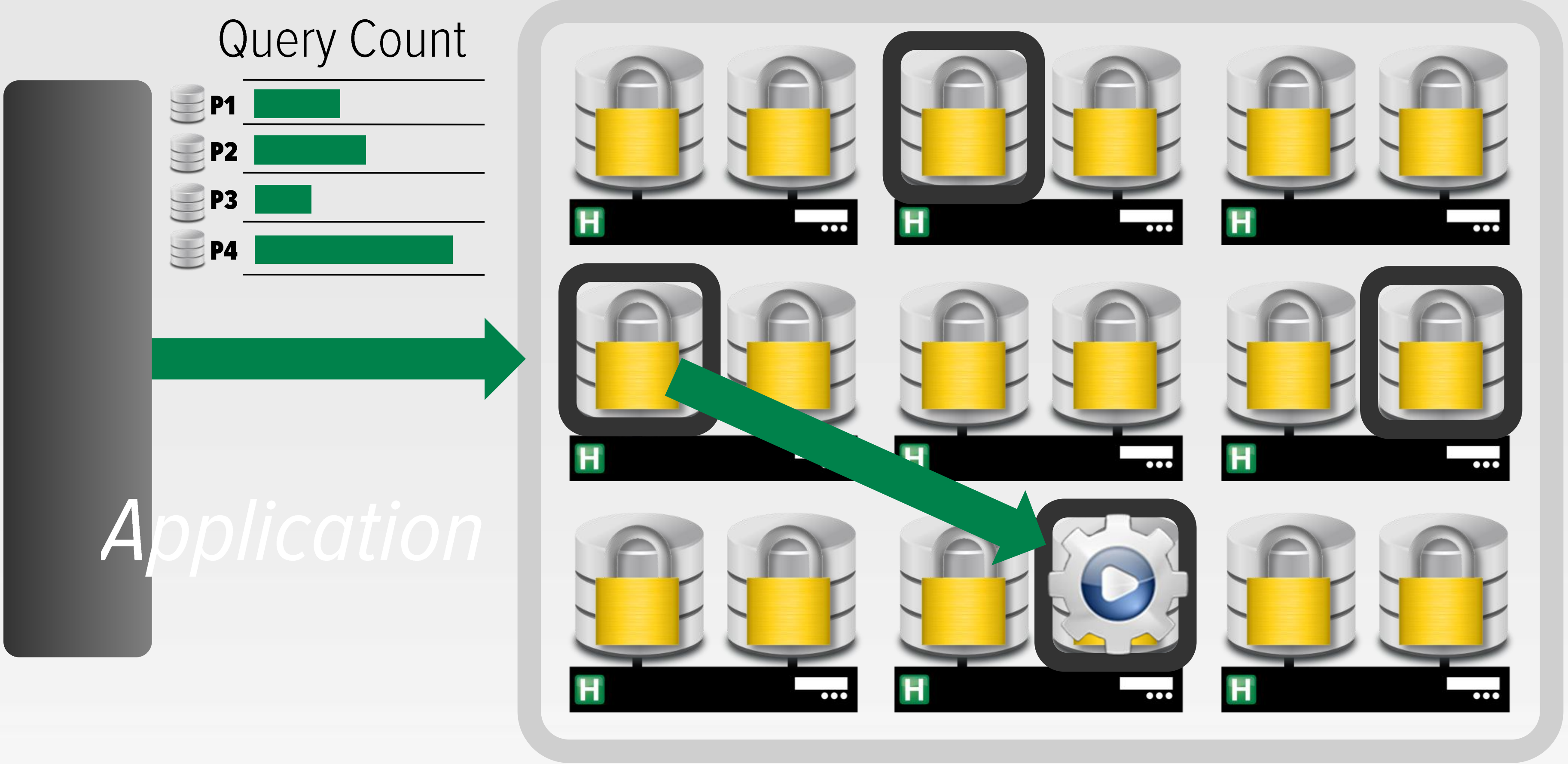
10% Distributed Transactions



DISTRIBUTED TRANSACTION



DISTRIBUTED TRANSACTION




NO
MARKINGS
AT
NIGHT TRIALS


Everyone

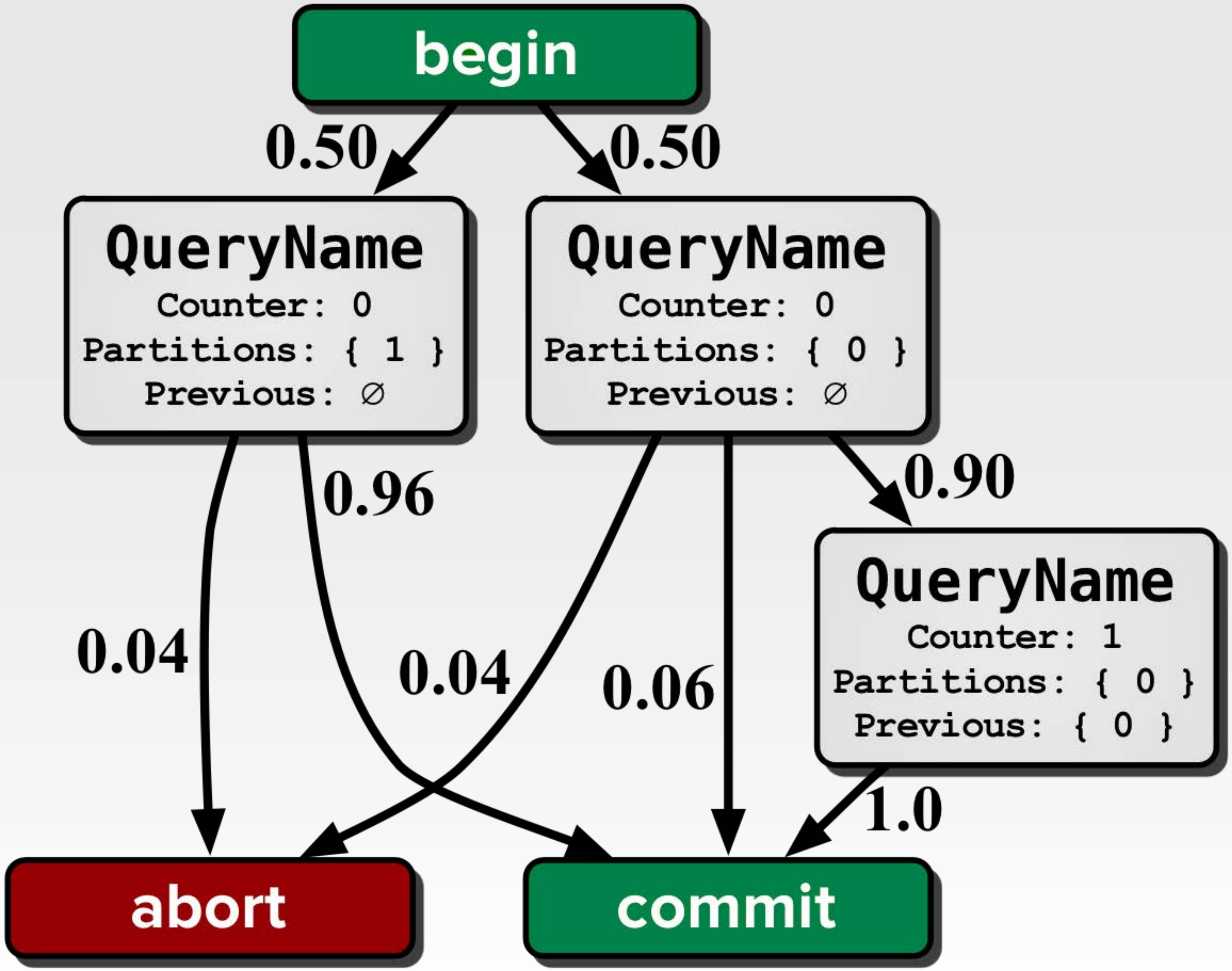

KNOW WHAT
THINGS WILL DO
BEFORE THEY START RUNNING

Optimization

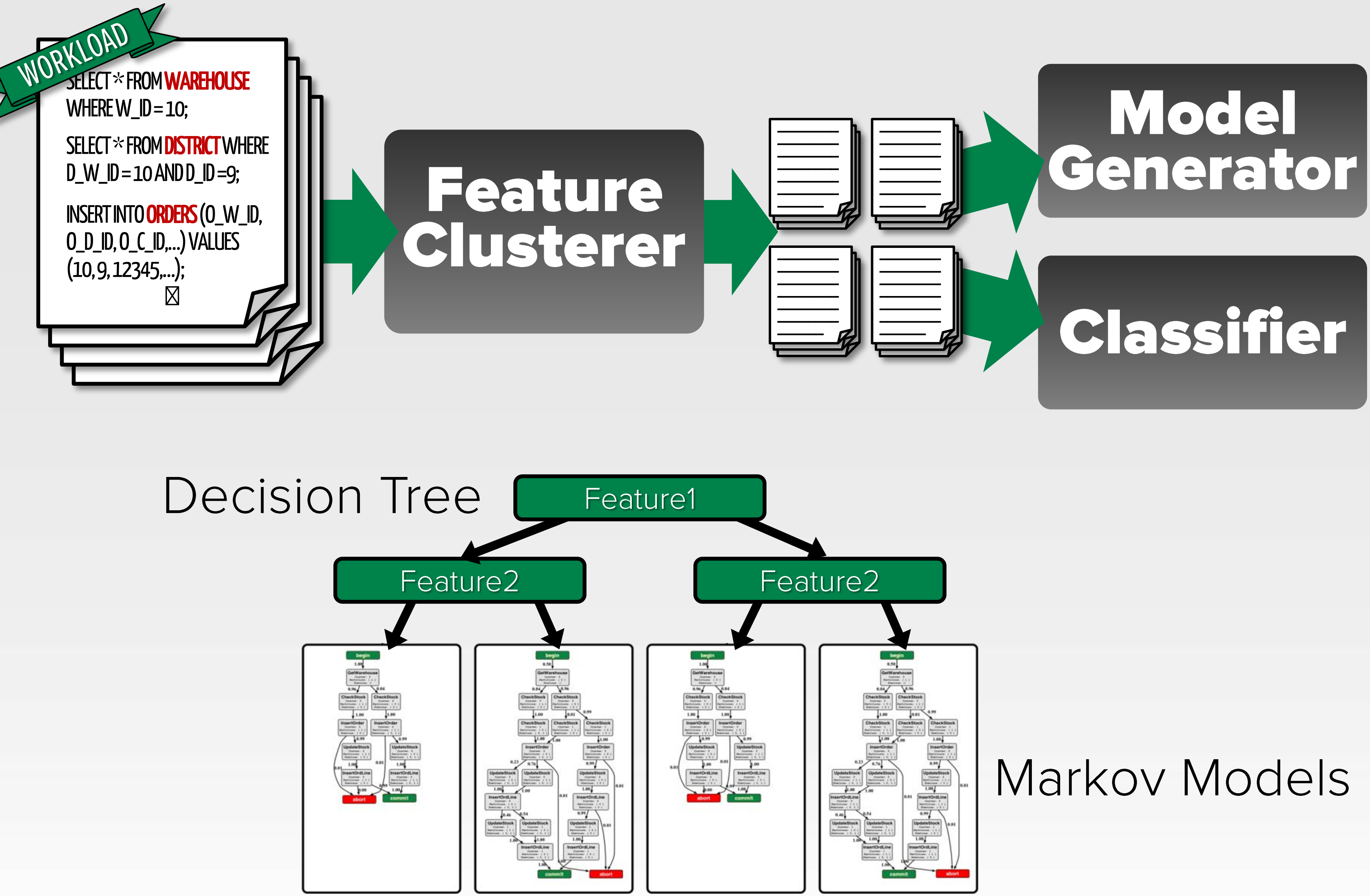
USE MODELS TO PREDICT TRANSACTION BEHAVIOR *BEFORE* EXECUTION.

ON PREDICTIVE MODELING FOR OPTIMIZING TRANSACTION EXECUTION IN PARALLEL OLTP SYSTEMS
Proc. VLDB Endow., Vol 5, Iss. 2, pp. 85-96, 2011

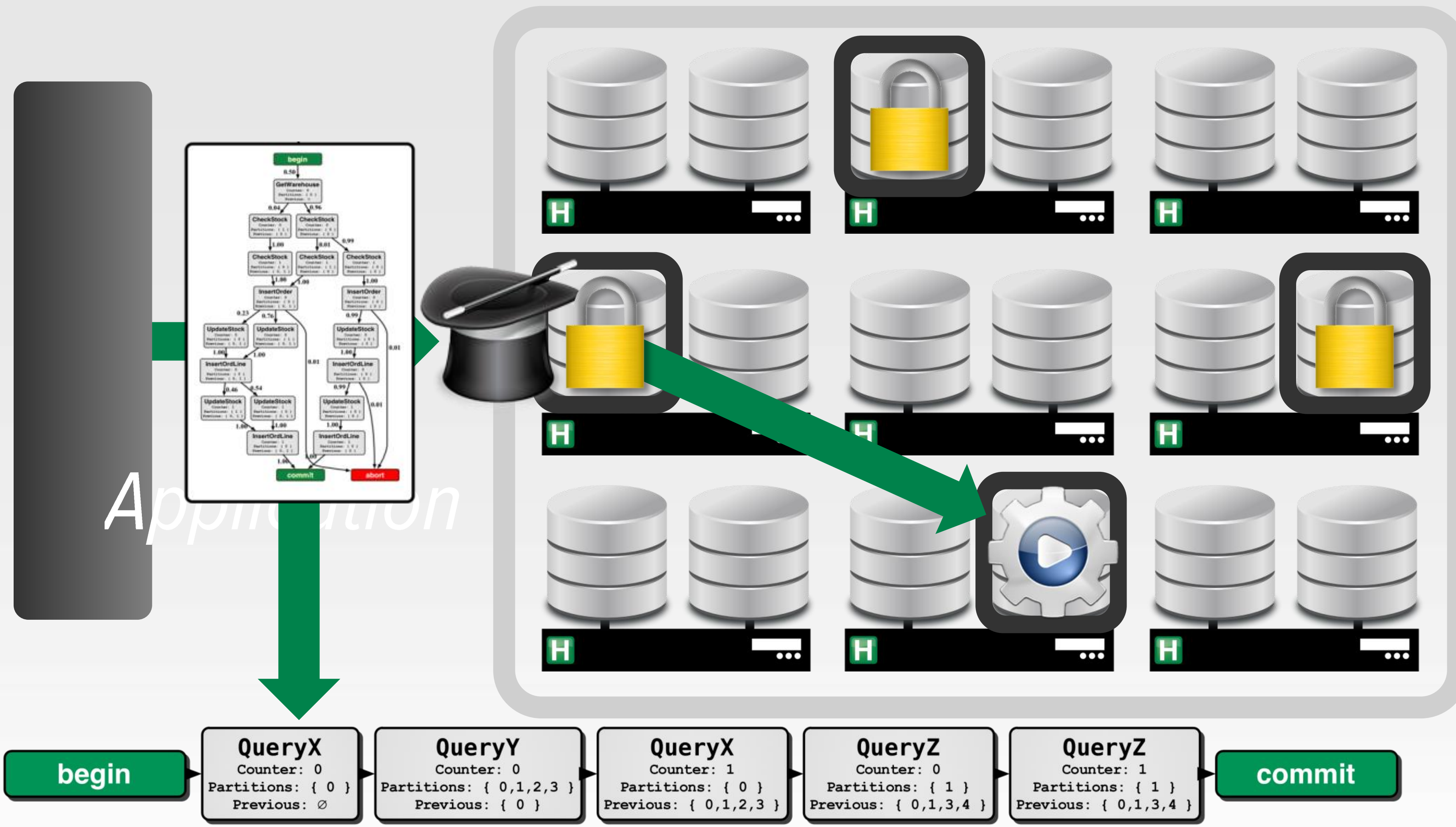
Houdini



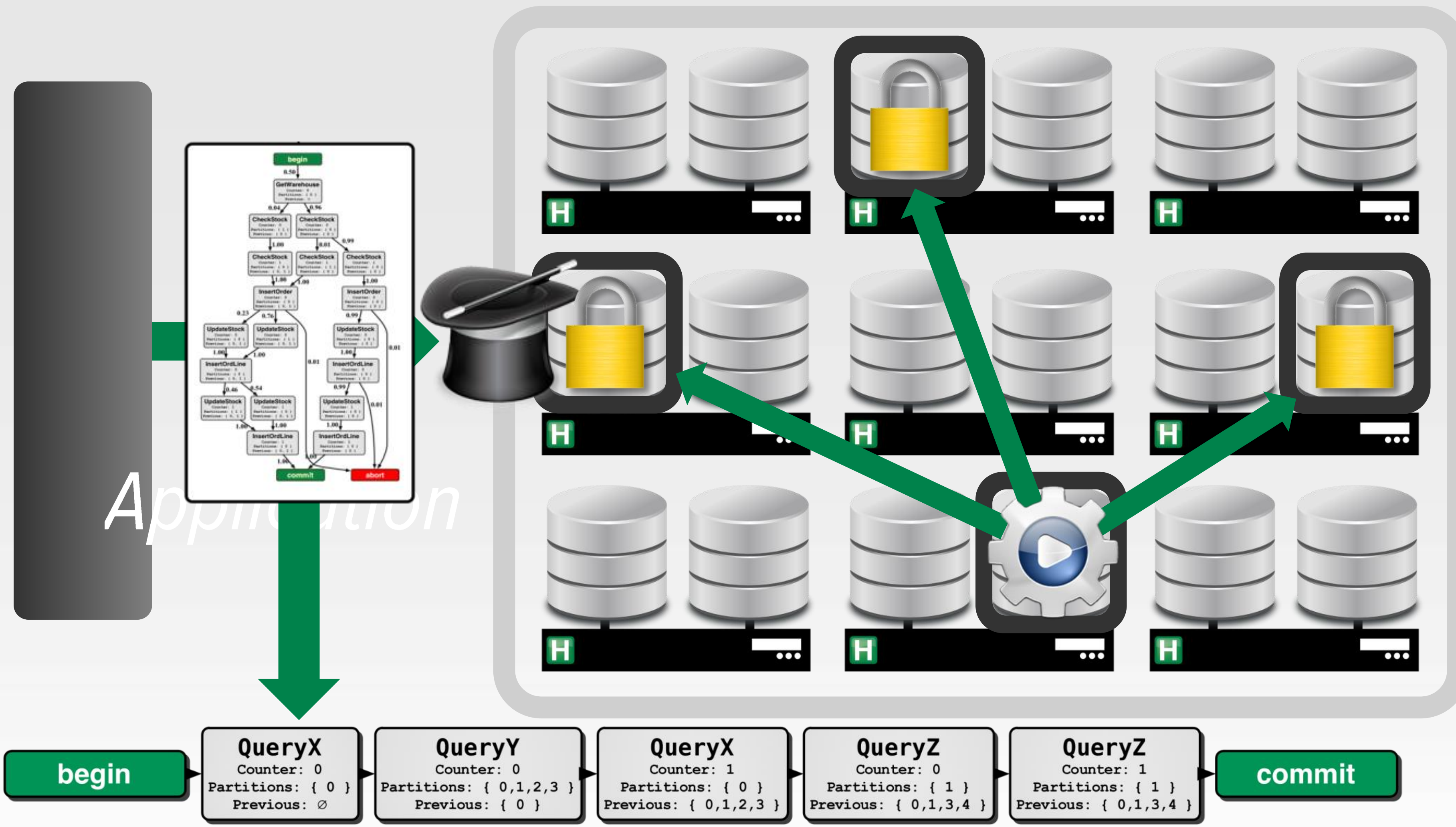
begin
0.50
0.50
QueryName
Counter: 0
Partitions: { 1 }
Previous: ∅
QueryName
Counter: 0
Partitions: { 0 }
Previous: ∅
0.96
0.90
QueryName
Counter: 1
Partitions: { 0 }
Previous: { 0 }
0.04
0.04
0.06
1.0
abort
commit


WORKLOAD
SELECT * FROM WAREHOUSE
WHERE W_ID = 10;
SELECT * FROM DISTRICT WHERE
D_W_ID = 10 AND D_ID = 9;
INSERT INTO ORDERS (O_W_ID,
O_D_ID, O_C_ID,...) VALUES
(10, 9, 12345,...);
⋮
Feature Clusterer
Model Generator
Classifier
Decision Tree
Feature1
Feature2
Feature2
Markov Models

DISTRIBUTED TRANSACTION



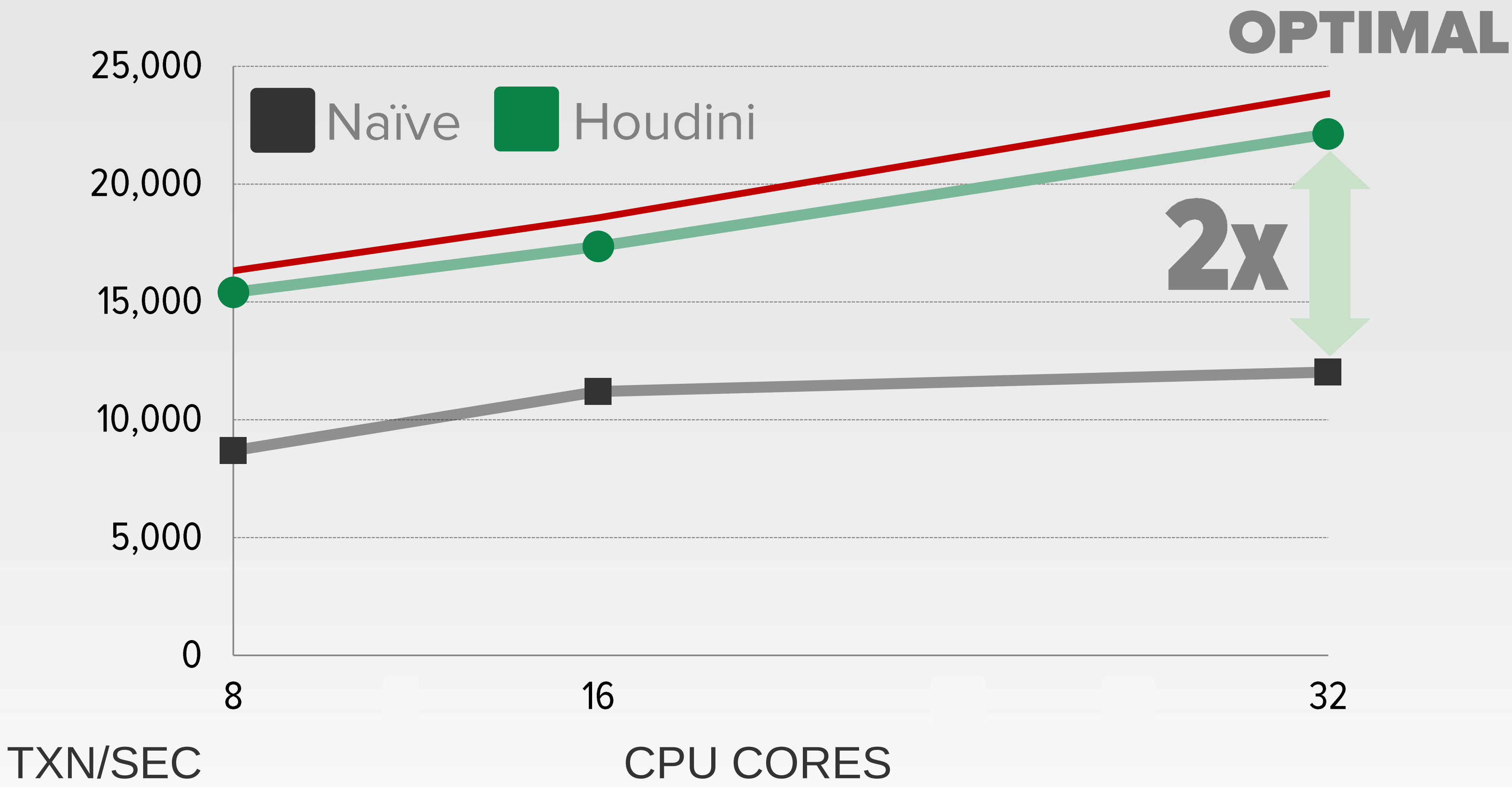
DISTRIBUTED TRANSACTION



Houdini

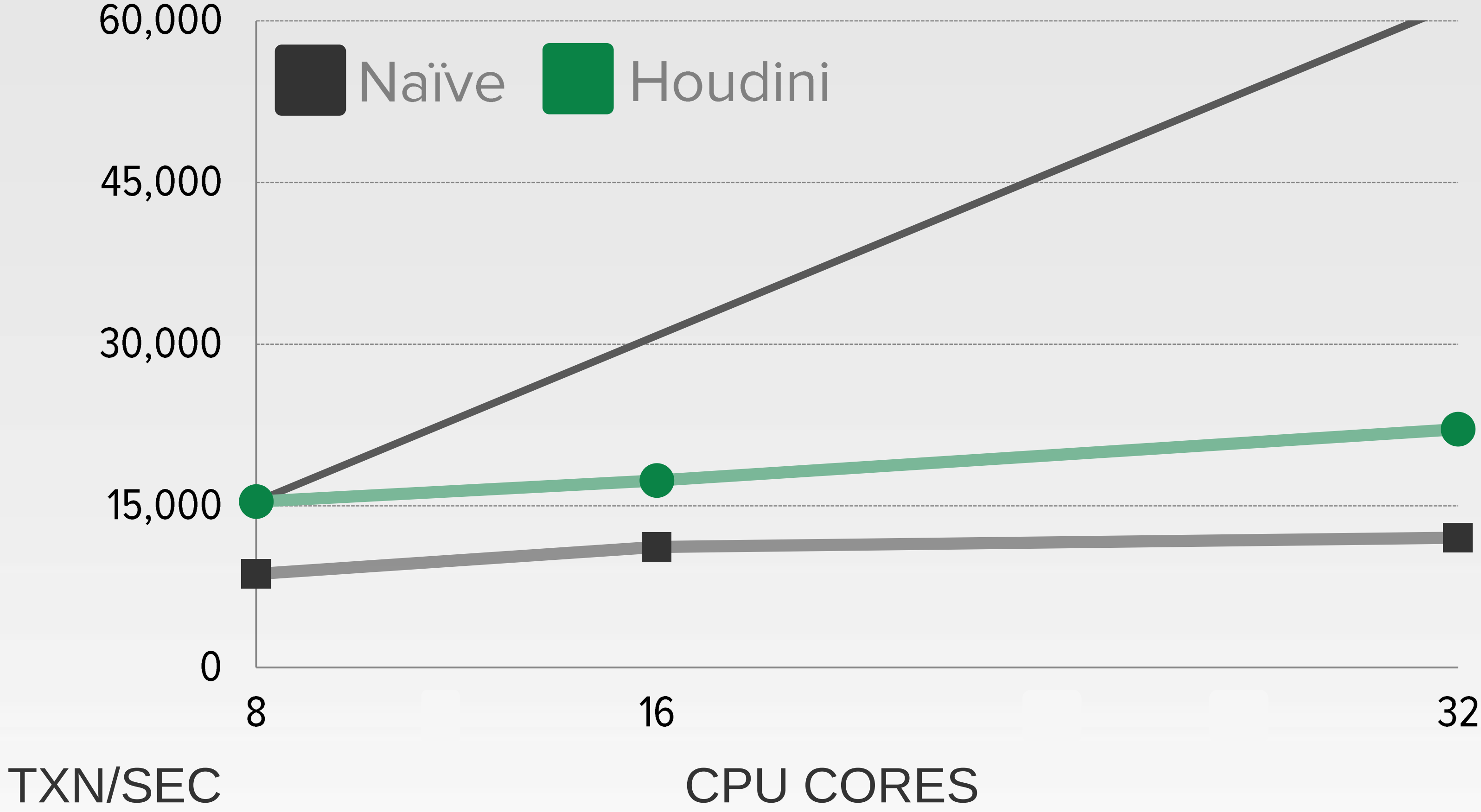
TPC-C BENCHMARK

8 Cores per Node

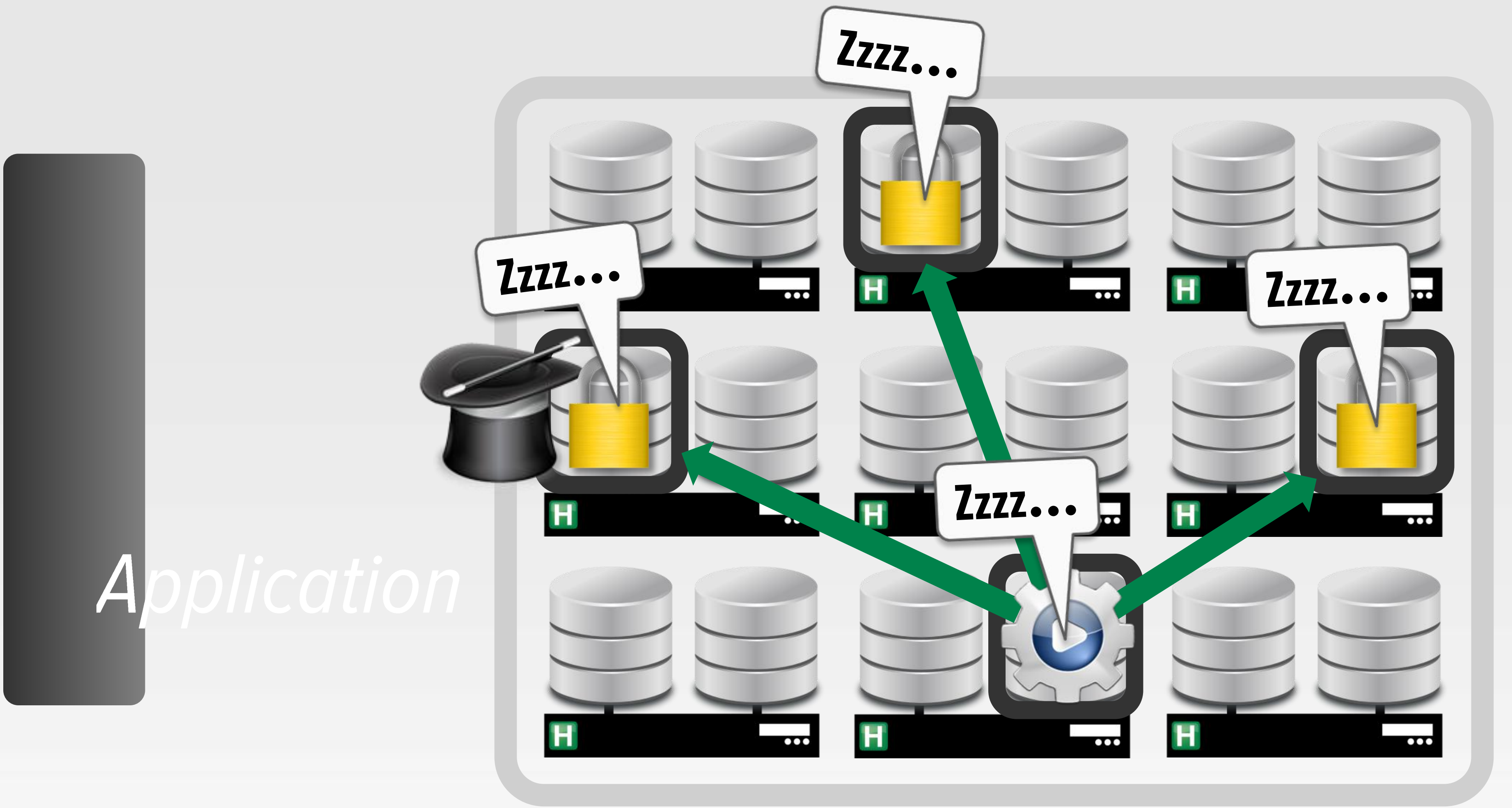


TPC-C BENCHMARK

8 Cores per Node



DISTRIBUTED TRANSACTION




DO SOMETHING
USEFUL
WHENEVER YOU ARE IDLE

Optimization

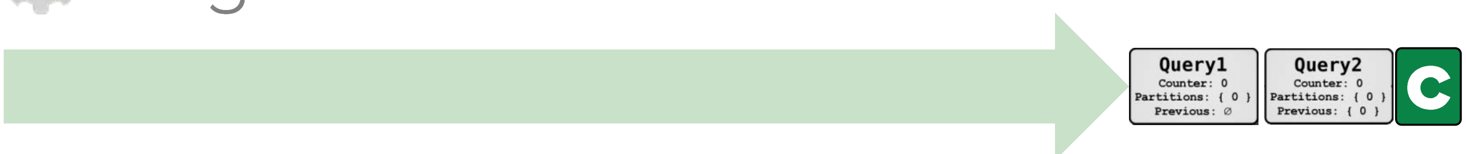
SPECULATIVELY EXECUTE TRANSACTIONS WHEN THE SYSTEM IS STALLED.

THE ART OF SPECULATIVE EXECUTION
In Preparation (April 2013).

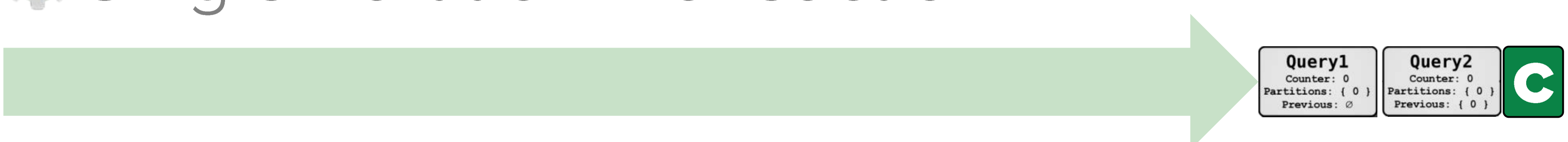
SERIALIZABLE SCHEDULE

Distributed Transaction

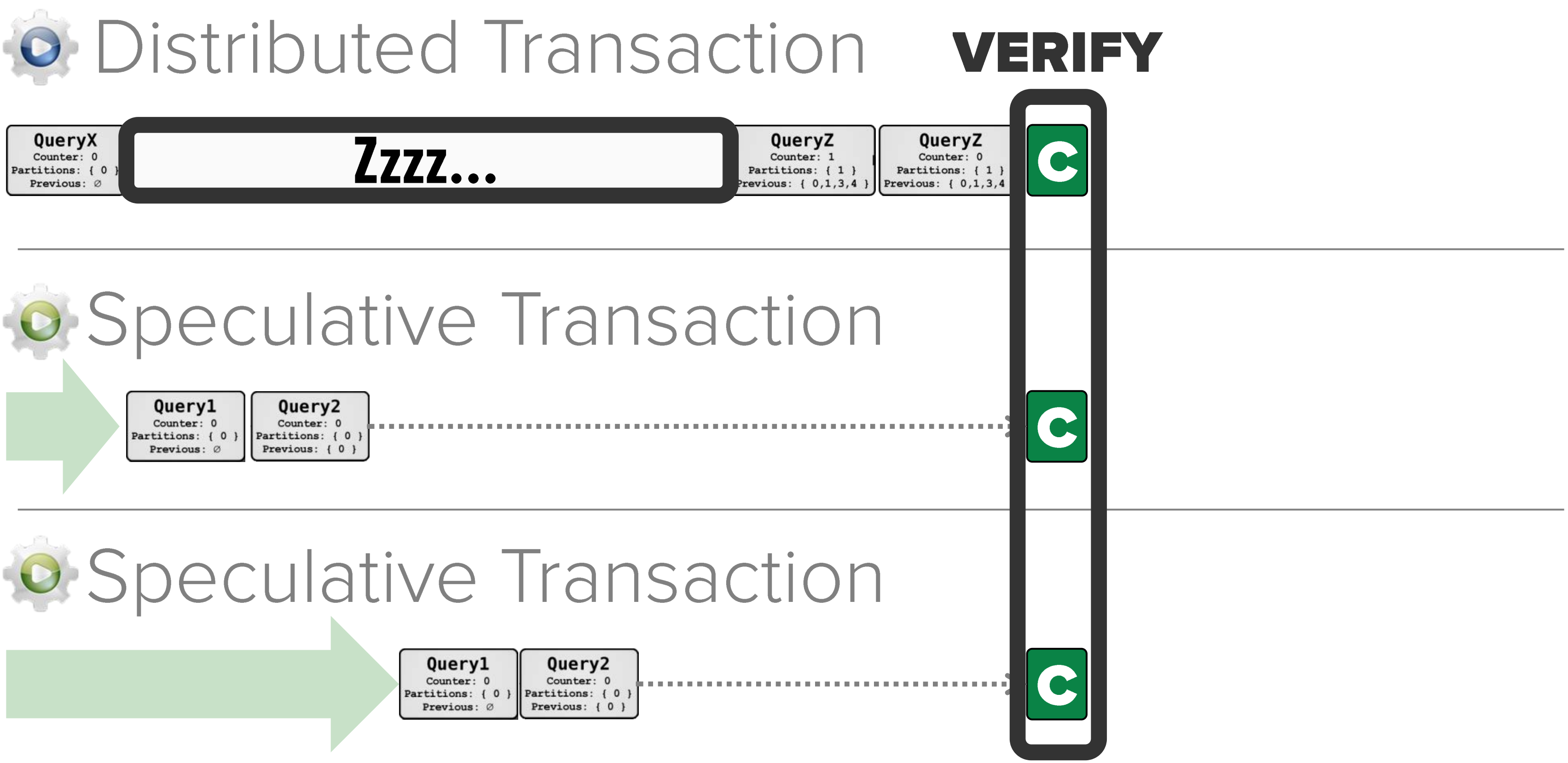
Single-Partition Transaction



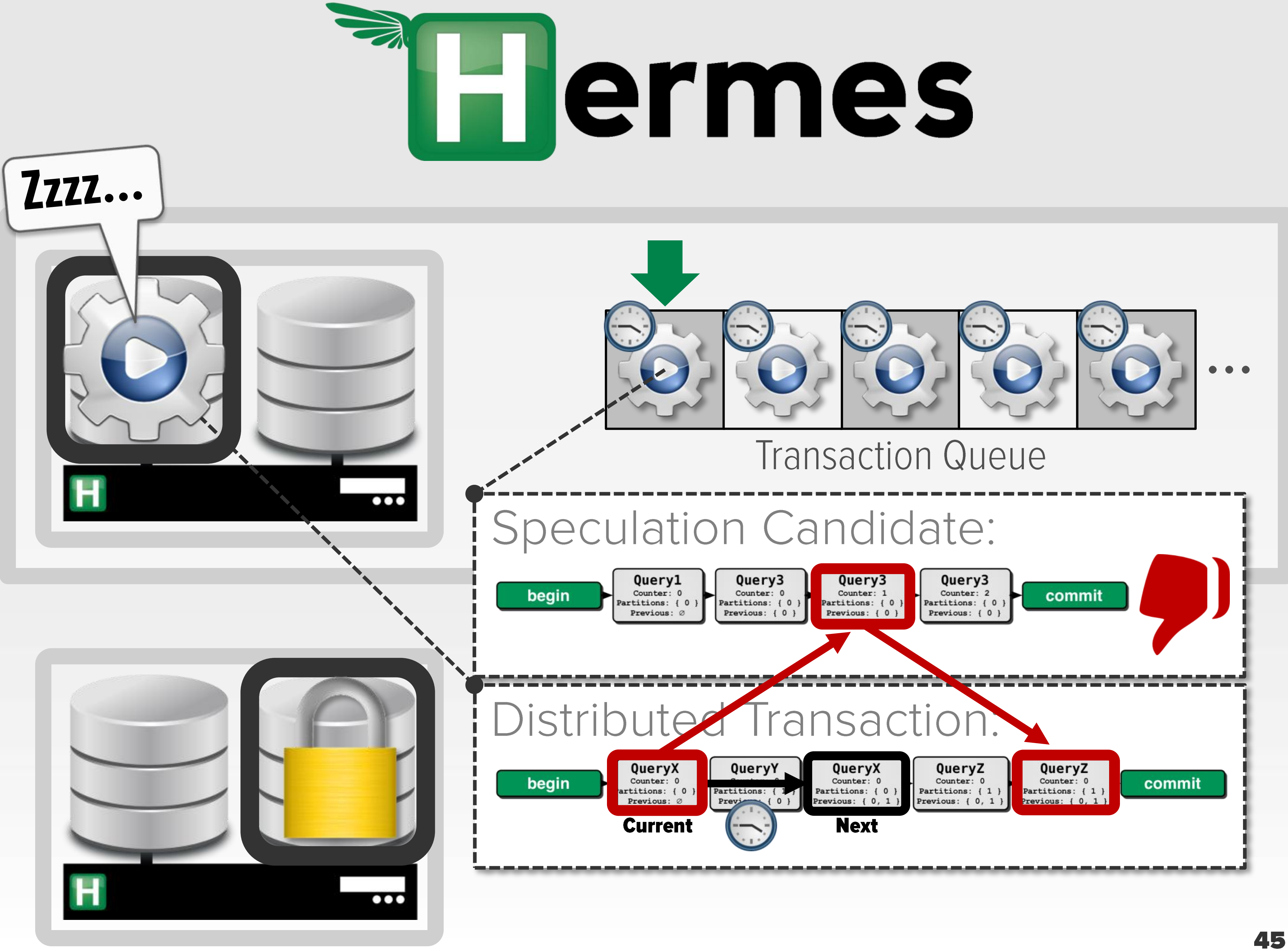
Single-Partition Transaction



SERIALIZABLE SCHEDULE

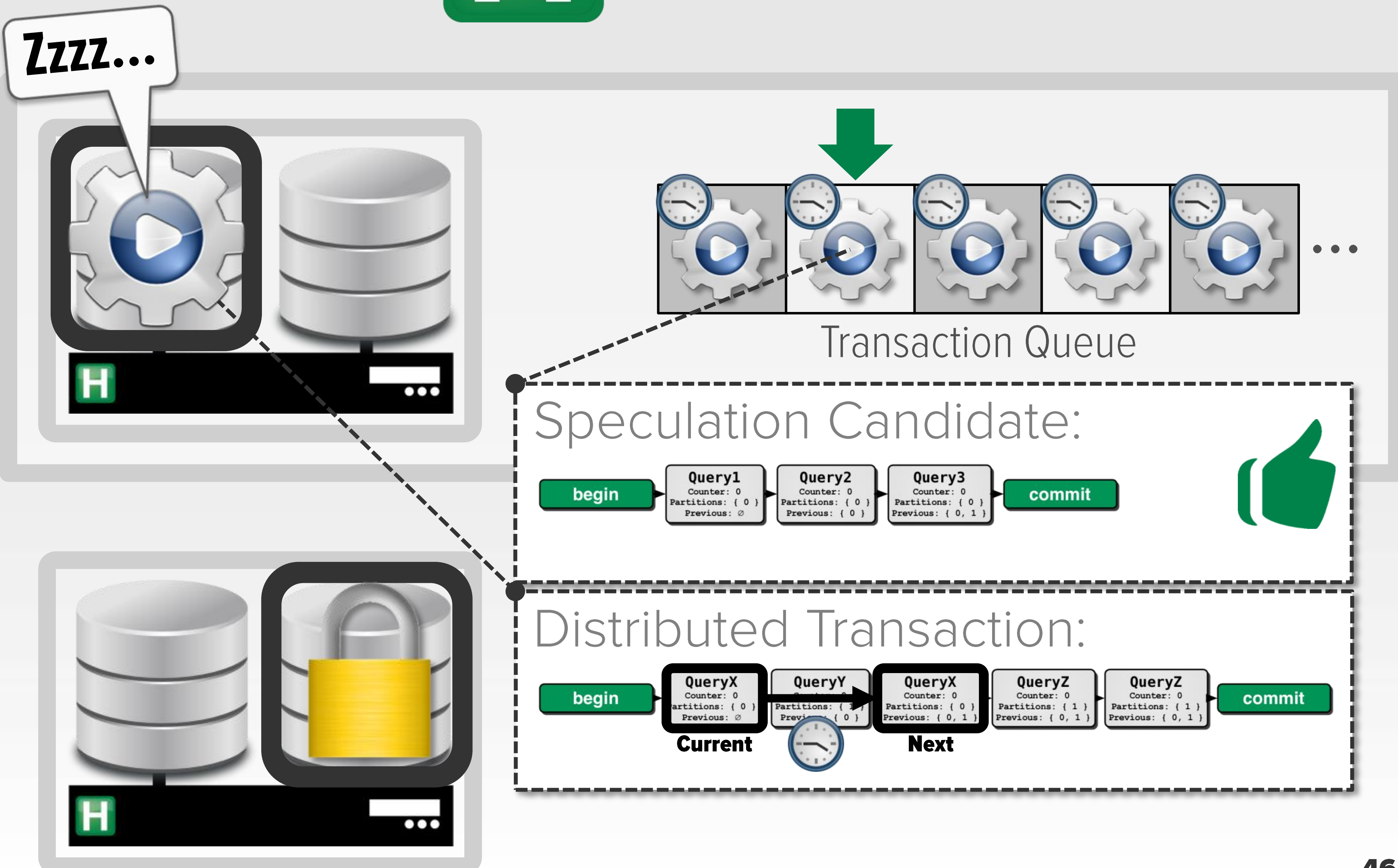


Hermes



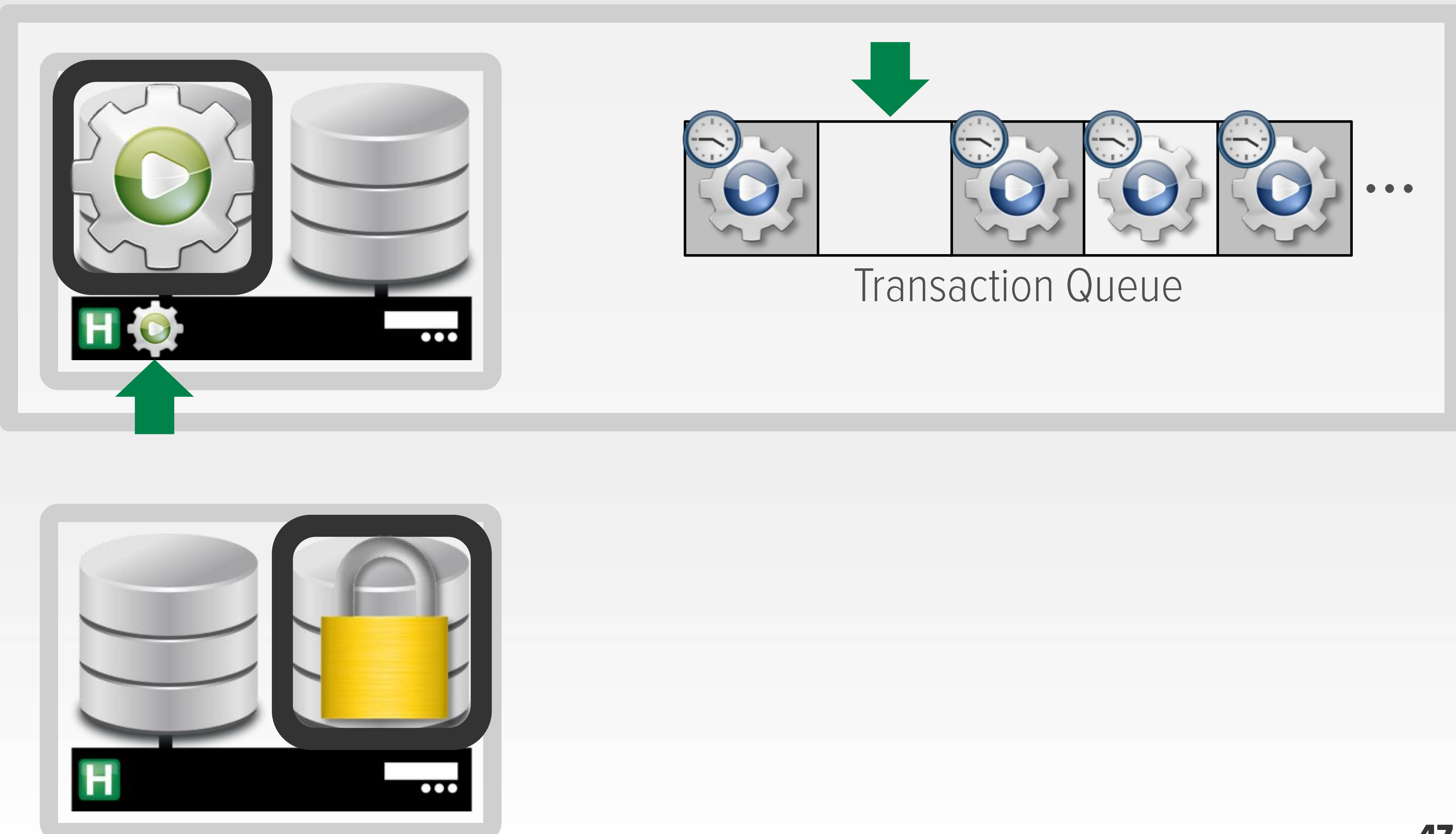
Hermes
Zzzz...
Transaction Queue
...
Speculation Candidate:
begin
Query1
Counter: 0
Partitions: { 0 }
Previous: ∅
Query3
Counter: 0
Partitions: { 0 }
Previous: { 0 }
Query3
Counter: 1
Partitions: { 0 }
Previous: { 0 }
Query3
Counter: 2
Partitions: { 0 }
Previous: { 0 }
commit
Distributed Transaction:
begin
QueryX
Counter: 0
artitions: { 0 }
Previous: ∅
QueryY
{ 0 }
QueryX
Counter: 0
Partitions: { 0 }
Previous: { 0, 1 }
QueryZ
Counter: 0
Partitions: { 1 }
Previous: { 0, 1 }
QueryZ
Counter: 0
Partitions: { 1 }
Previous: { 0, 1 }
commit
Current
Next

Hermes



Zzzz...
Transaction Queue
Speculation Candidate:
begin
Query1
Counter: 0
Partitions: { 0 }
Previous: ∅
Query2
Counter: 0
Partitions: { 0 }
Previous: { 0 }
Query3
Counter: 0
Partitions: { 0 }
Previous: { 0, 1 }
commit
Distributed Transaction:
begin
QueryX
Counter: 0
Partitions: { 0 }
Previous: ∅
QueryY
QueryX
Counter: 0
Partitions: { 0 }
Previous: { 0, 1 }
QueryZ
Counter: 0
Partitions: { 1 }
Previous: { 0, 1 }
QueryZ
Counter: 0
Partitions: { 1 }
Previous: { 0, 1 }
commit
Current
Next

Hermes



STORED PROCEDURE

VoteCount:

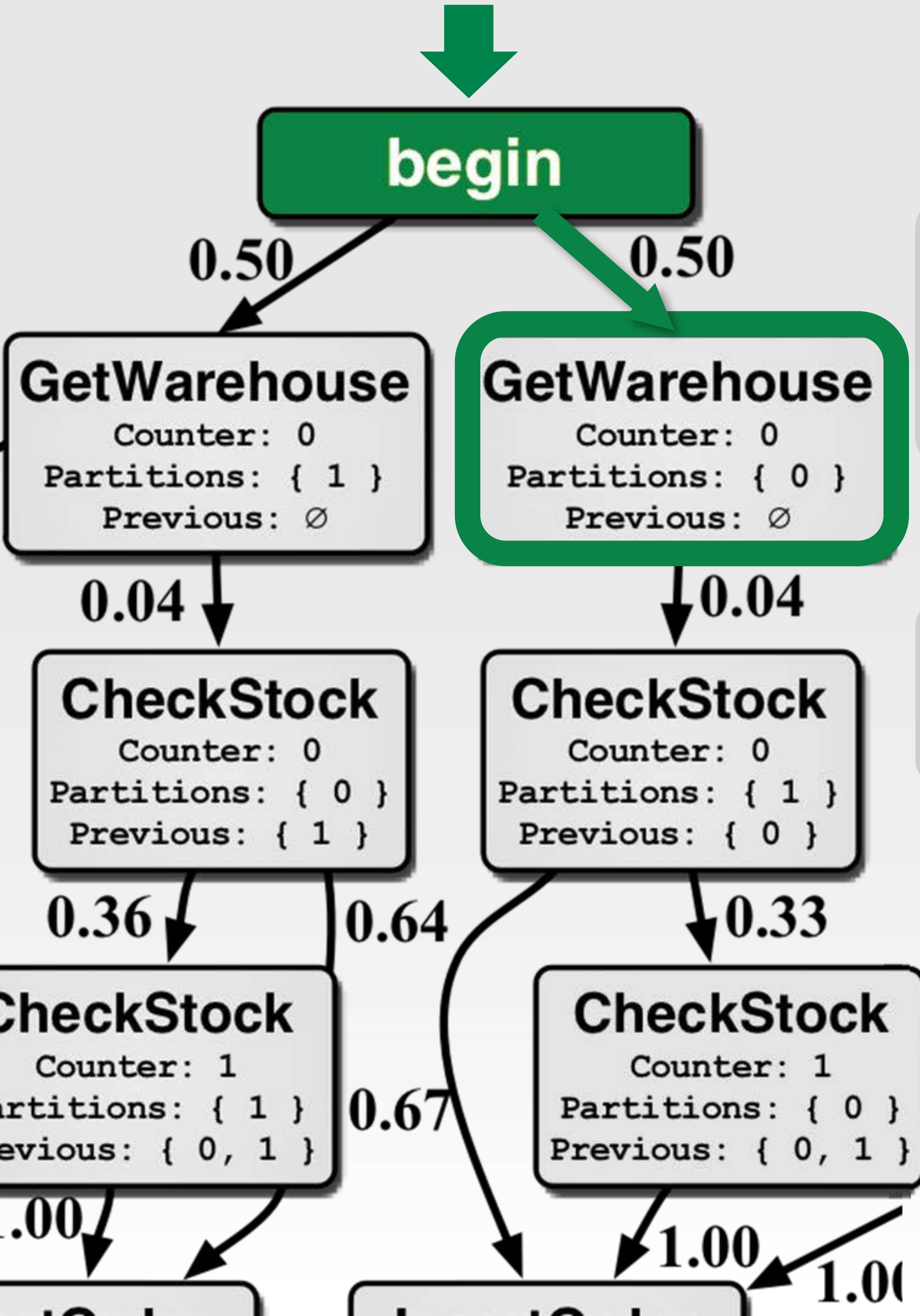
```
SELECT   COUNT(*)
  FROM   votes
 WHERE   phone_num = ?;
```

InsertVote:

```
INSERT INTO votes
  VALUES ( ?, ?, ? );
```

```
run ( phoneNum, contestantId, currentTime ) {
    result = execute ( VoteCount, phoneNum );
    if (result > MAX_VOTES ) {
        return ( ERROR );
    }
    execute ( InsertVote, phoneNum,
                          contestantId,
                          currentTime );
    return ( SUCCESS );
}
```

Hermes

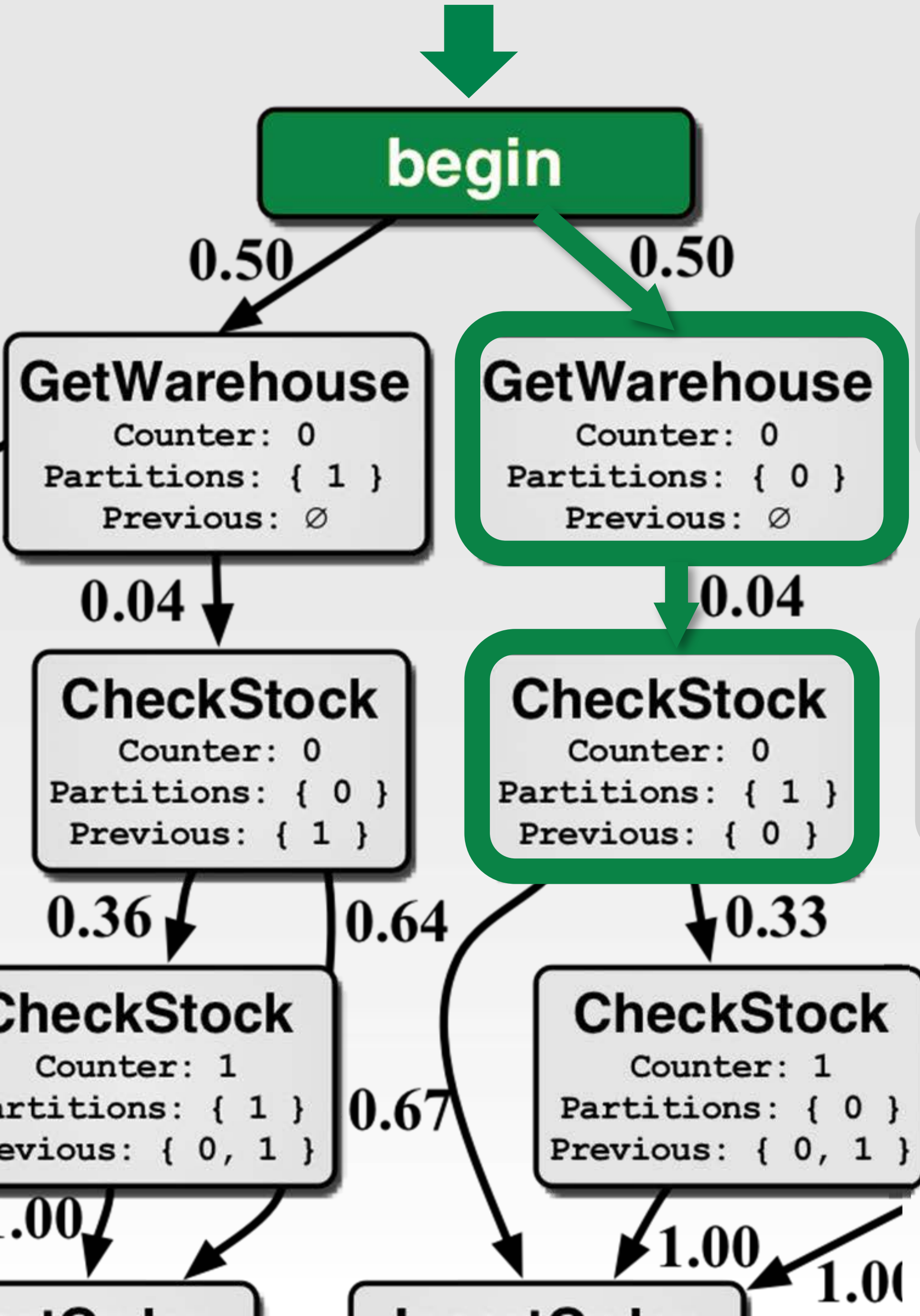


Input Parameters:

```
w_id    =0
i_w_ids =[1,0]
i_ids   =[1001,1002]
```

GetWarehouse:

```
SELECT * FROM WAREHOUSE
  WHERE W_ID = ?
```



Input Parameters:

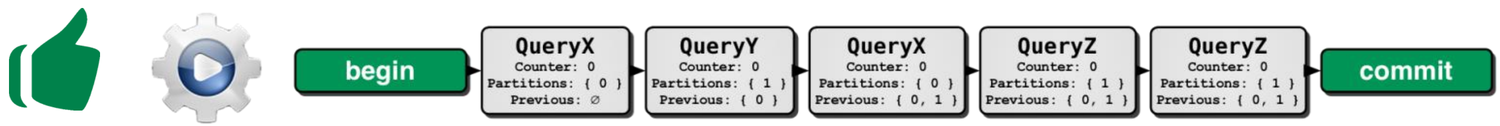
```
w_id    =0
i_w_ids    =[1,0]
i_ids    =[1001,1002]
```

CheckStock:

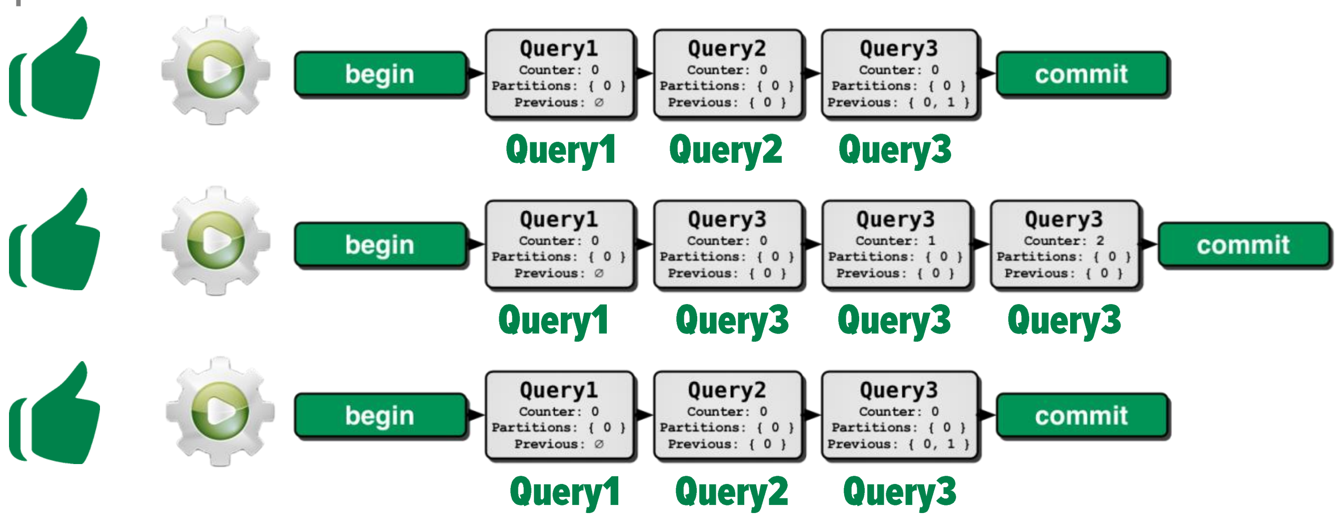
```
SELECT  S_QTY  FROM  STOCK
  WHERE  S_W_ID =   ?
    AND  S_I_ID =   ? ;
```

VERIFICATION

Distributed Transaction

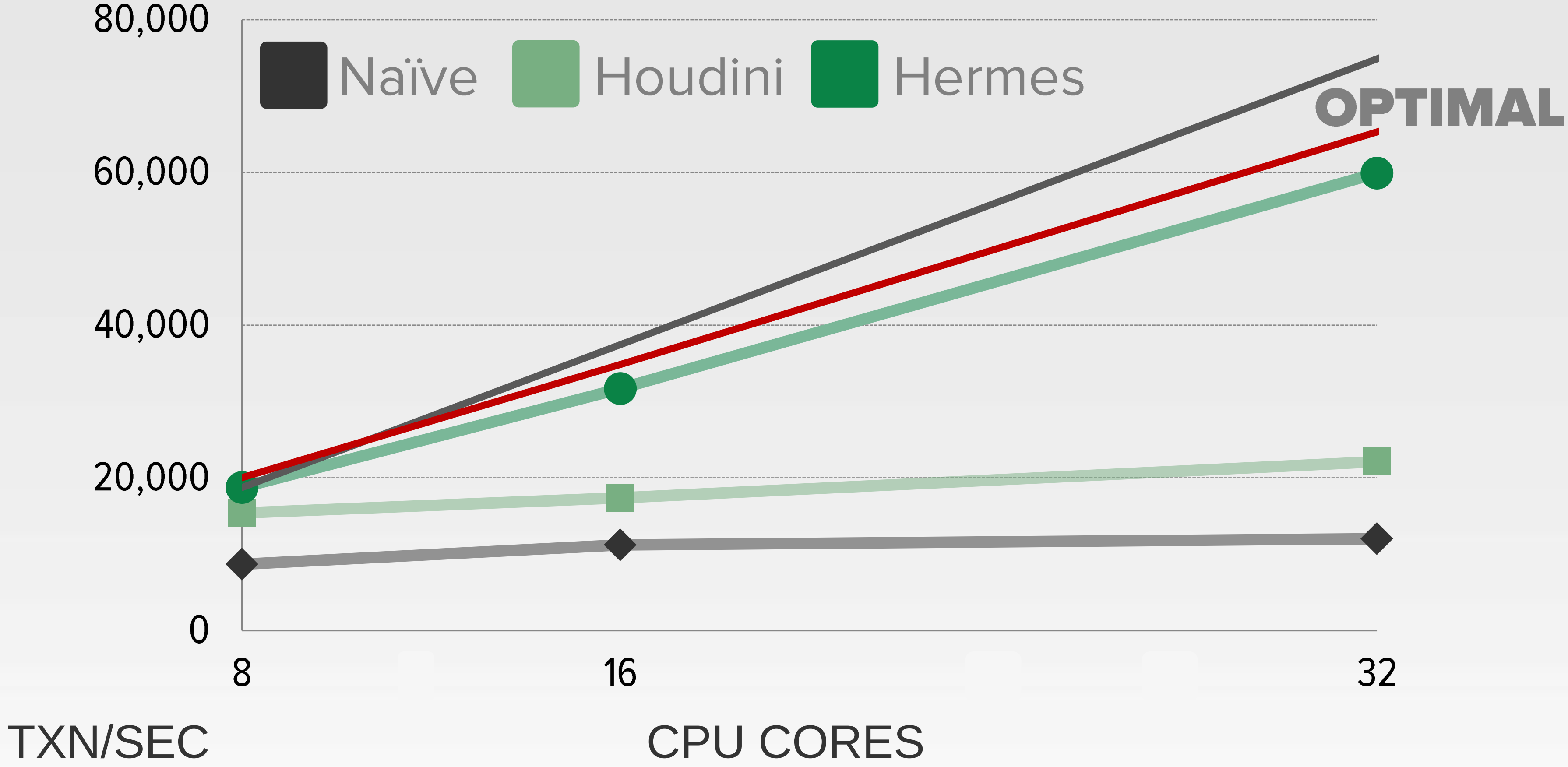


Speculative Transactions



TPC-C BENCHMARK

8 Cores per Node




2


THE MORE YOU KNOW,
THE *FASTER* YOU CAN GO.

REAL-WORLD IMPACT

- On-line Voting Applications.
- On-line Gaming & Gambling.
- Network Traffic Monitoring.
- High-Frequency Trading.

VOLTDB

- OPTIMIZE SYSTEM FOR TRANSACTIONS
- PREDICT TRANSACTION BEHAVIOR
- SAFELY INTERLEAVE TRANSACTIONS

FUTURE WORK

WORKLOAD EXPANSION

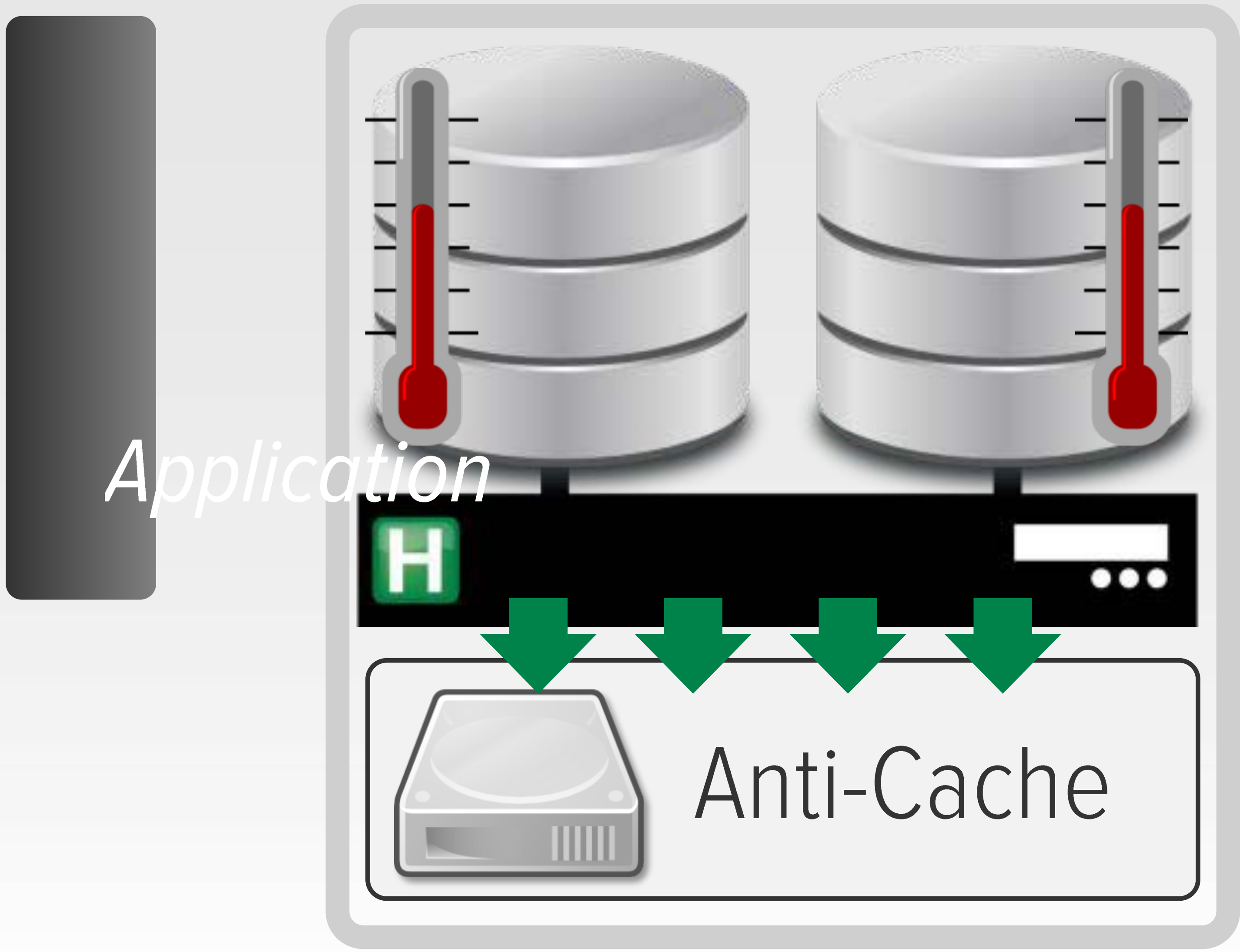
- Stream Processing.
- Non-Partitionable Workloads.
- Real-time Analytics.
  - ⮩ Hybrid Storage Models.
  - ⮩ Relaxed Consistency.

ISTC BIG DATA

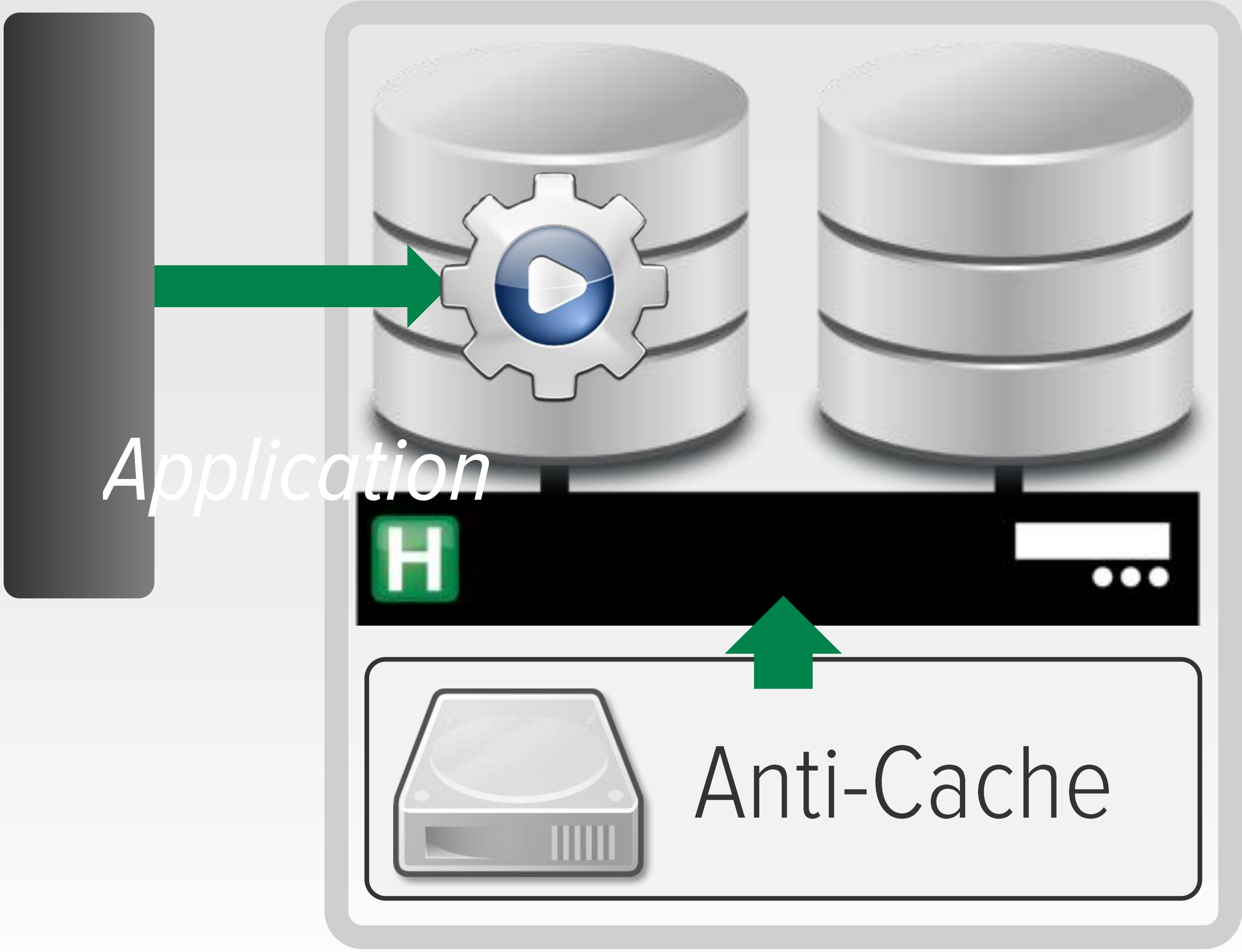
IMPROVING SCALABILITY

- Elastic Deployments.
- New Hardware:
  - Non-Volatile Memory.
  - Many-Core Architectures.
- Anti-Caching.

ANTI-CACHING



ANTI-CACHING



HOW TO *SCALE UP* WITHOUT *GIVING UP* TRANSACTIONS.


H-Store
hstore.cs.brown.edu
@andy_pavlo