

OLTP Workload
Acceleration with Memory
Optimized Tables on
PostgreSQL

Who Are We...
Introduction

Motivation



Databases are Based on legacy architecture



PostgreSQL



Microsoft SQL Server

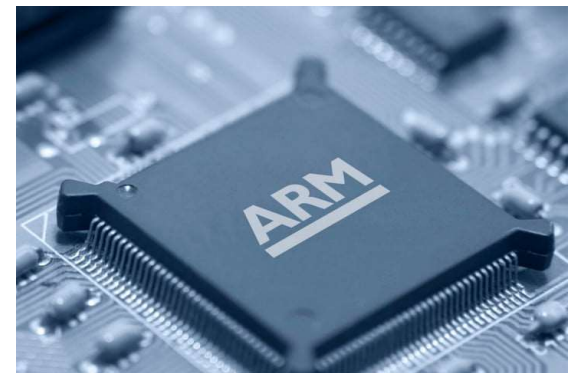


DRAM capacity goes up
while its price is going down



Multi socket servers with
many cores

Efficient ARM processors



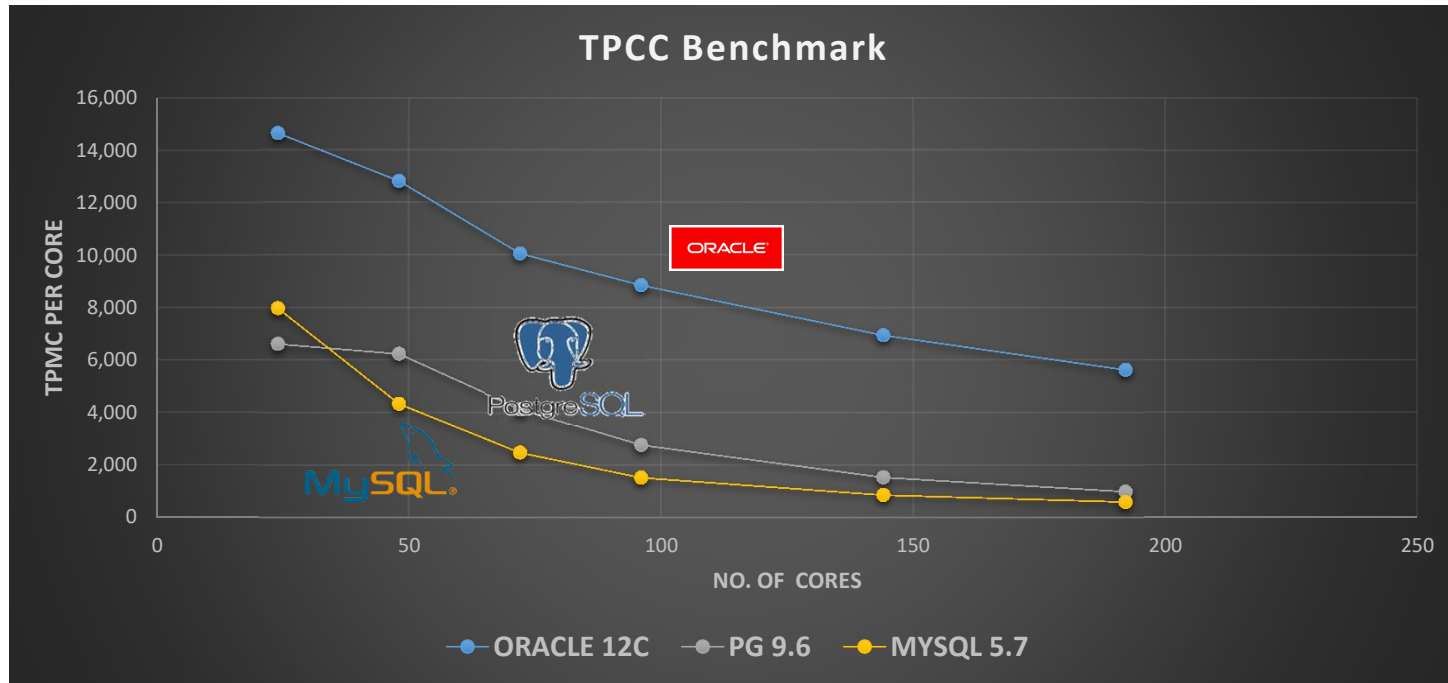
We purchased 8 sockets machine with 192 physical cores
(~\$120K)



- Intel(R) Xeon(R) CPU E7-8890 v4 @ 2.20GHz (24 cores)
- 8 CPUs (sockets)
- 1 TB RAM
- 1200 SSD 200GB, SAS 12Gb/s
- Ubuntu 16.04.2 LTS

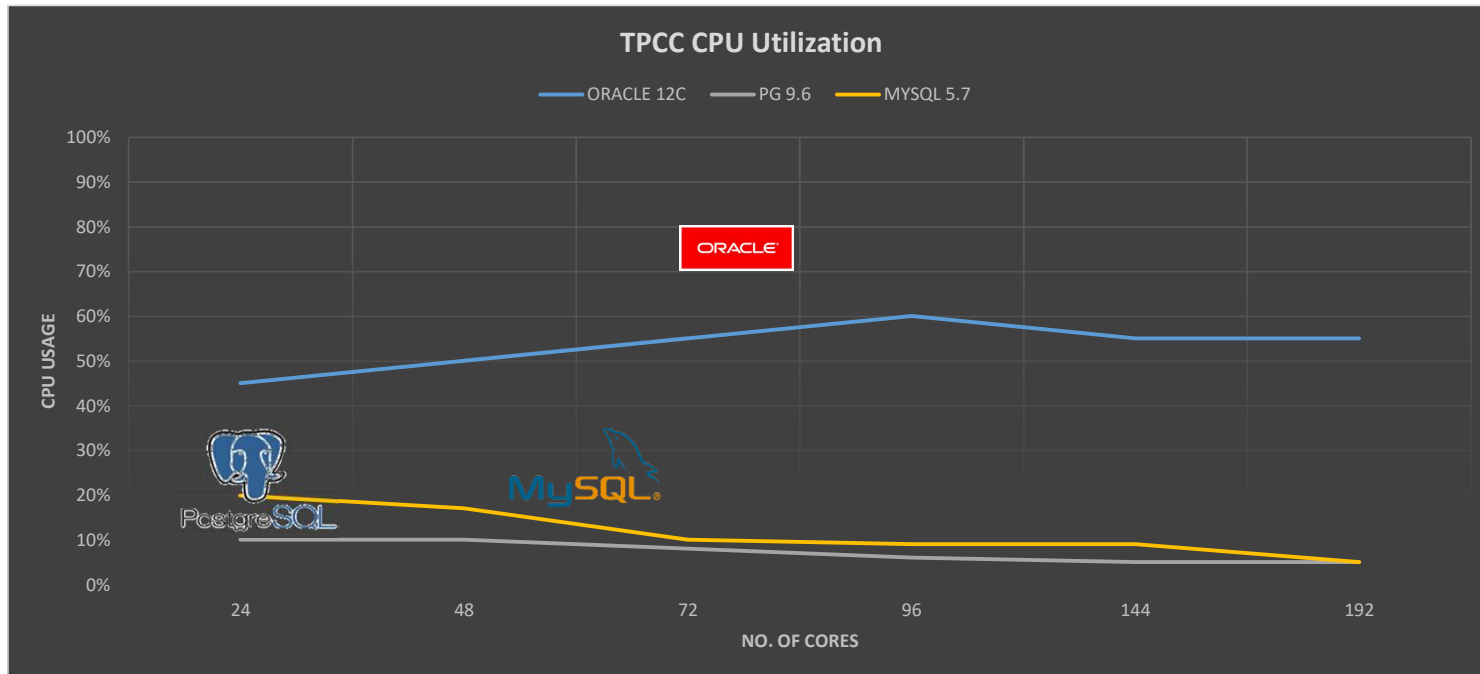
TPCC Benchmark

TPMCS Per Core



- 512 warehouses
- asynchronous commit
- huge buffer pools

TPCC Benchmark CPU Utilization



- 512 warehouses
- asynchronous commit
- huge buffer pools

- Poor resource utilization
- Poor Scalability



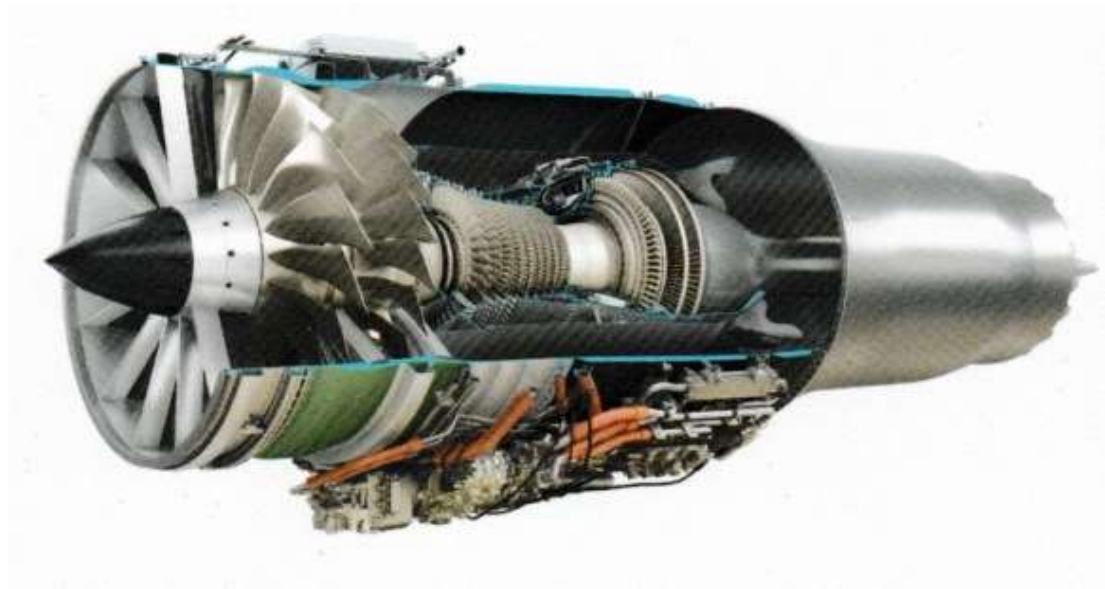
- How to build cost-effective OLTP engine with many cores?
- How to build effective OLTP engine on many cheaper cores (ARM64)?
- Can we address the problem while improving a popular legacy DBMS?



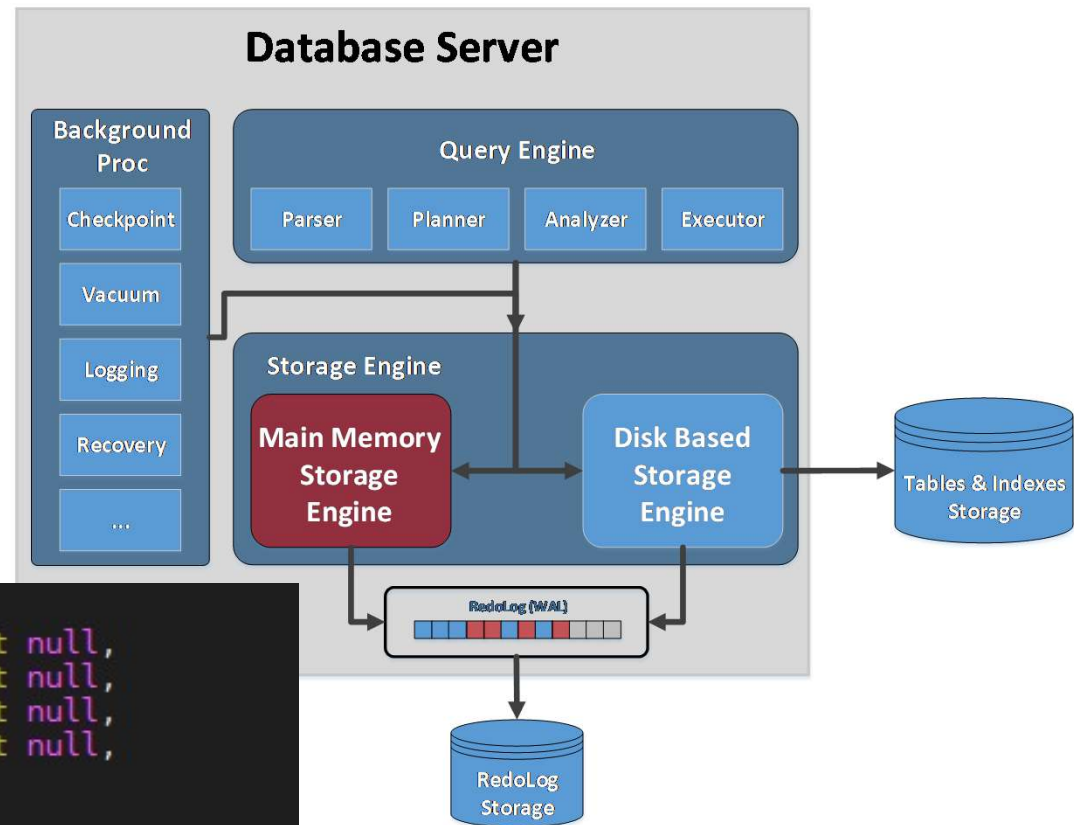
Research



- Shared Everything
- Optimistic
Concurrency Control
- Lock-free indexes
- Customized Memory
Management
(allocation, NUMA
awareness, caching)



Seamless integration of our
Memory-Optimized
Storage Engine in
PostgreSQL using FDW



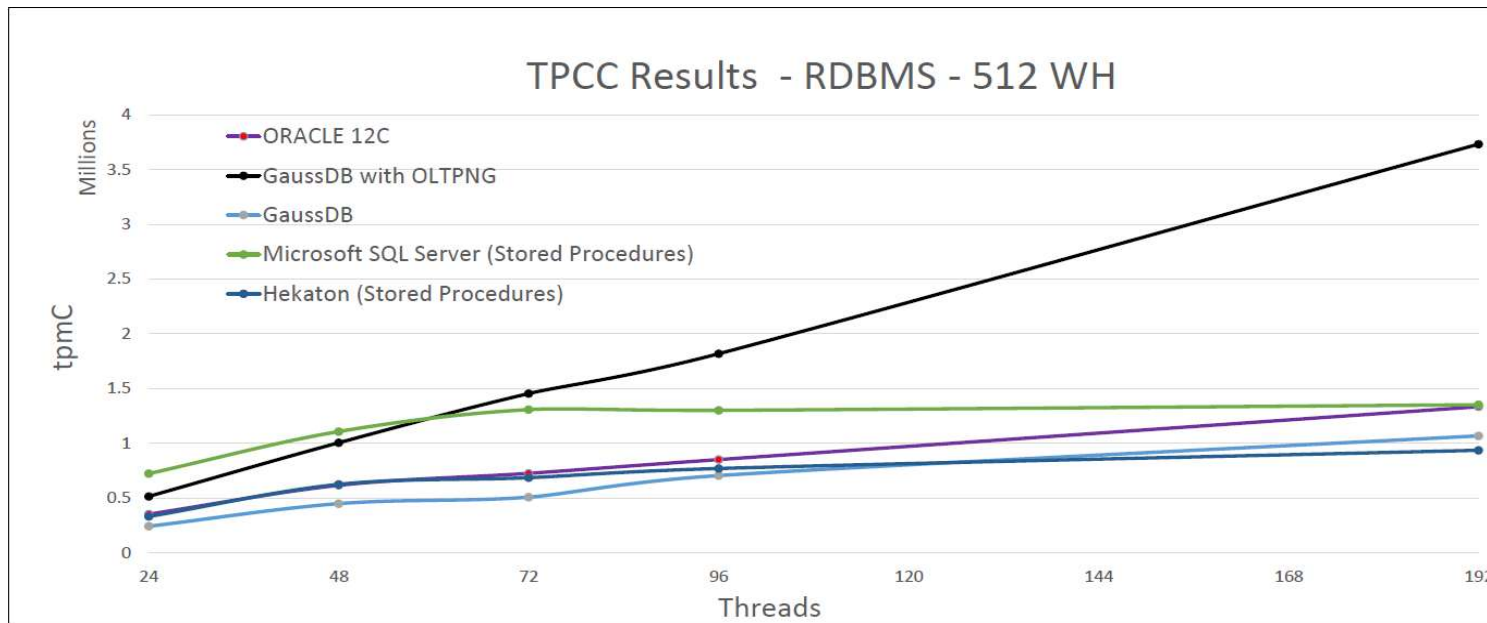
```
create table bmsql_order (  
  o_w_id integer not null,  
  o_d_id integer not null,  
  o_id integer not null,  
  o_c_id integer not null,  
  o_carrier_id integer,  
  o_ol_cnt integer,  
  o_all_local integer,  
  o_entry_d timestamp,  
  primary key (o_w_id, o_d_id, o_id)  
)
```

```
MEMORY OPTIMIZED;
```

- Fully integrated Memory Optimized Storage engine with PostgreSQL
- Very high performance & cost effectiveness
- Full SQL support including stored procedures
- Cross storage engine (MM and disk) transactions
- Cluster mode (PGXC) with Memory Optimized Tables
- Full ACID with all isolation level
- Unique/non-unique Secondary indexes
- Replication & HA

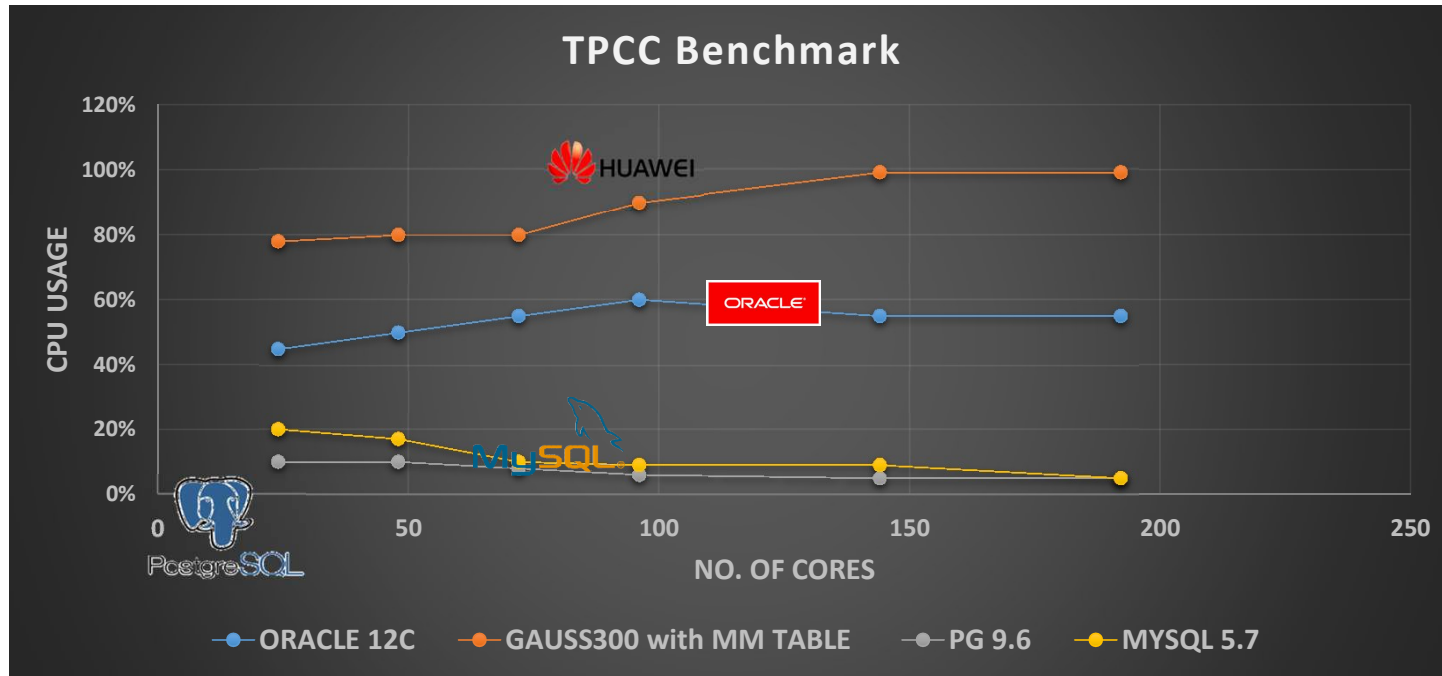


TPCC Benchmark



- 512 warehouses
- asynchronous commit
- huge buffer pools

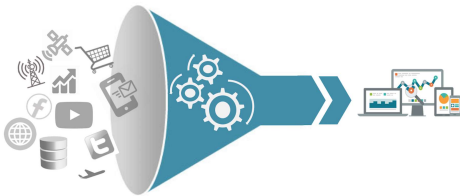
TPCC Benchmark CPU Utilization



- 512 warehouses
- asynchronous commit
- huge buffer pools



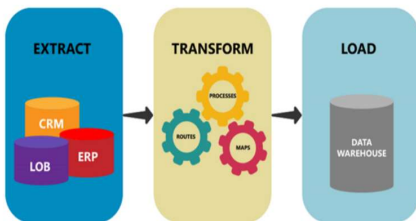
High throughput and low latency transaction Processing



Data ingestion, including IoT



Network management caching and session state



ETL and Batch analytical processing