Socrates: The New SQL Server in the Cloud
Goals

- Highly available
- Elastic
  - Shared-disk
- High performance
  - Low log latency
- Large-scale databases
  - Eliminate $O(\text{database size})$ operations
- Backwards compatibility
Existing Solutions (HADR, shared-disk)

- Azure SQL DB
- Google Spanner
- Amazon Aurora
- Oracle

Figure 1: HADR Arch. (Replicated State Machines)
Architecture

- Compute Tier
  - Compute Nodes
- Logging Tier
  - XLOG Service
- Storage Tier
  - Page Servers
- Persistence Tier
  - XStore

Figure 2: Socrates Architecture
Compute Tier

- Stateless
- Single primary
  - Read + Write
- Multiple secondaries
  - Read
  - Can promote from failover
- Unaware of network separation
  - All IO is transparently virtualized
  - Unaware of other replicas
Compute Tier - Primary

- Pushdown
  - Backups
  - Log durability

- Resilient Buffer Pool Extension (RBPEX)
  - Cache spills into local SSD
  - Resilient
  - Only need to apply log updates after reboot

- getPage(pageId, LSN)
  - Page Server returns page at least as new as Log Sequence Number
  - Keep track of newest LSN per page when writing
  - Read back if evicted from cache
Compute Tier - Secondaries

- Log blocks are not persisted
- Only store hot data pages
- New log records (from XLOG) applied to cached pages
XLOG (Logging Tier)

- Primary to Landing Zone (LZ)
  - Synchronous
  - 3 replicas
  - Small storage
- Primary to XLOG
  - Asynchronous
- Pending Area to LogBroker
  - Only after in LZ can logs progress to replicas and archive
- Destaging
  - Written to XStore
  - Written to XLOG local cache
  - Removal from LZ
- Replicas + Pager Servers from LogBroker
  - Pulled from LogBroker following memory hierarchy

Figure 3: XLOG Service
Page Servers (Storage Tier)

- Stateless
- New log records (from XLOG) applied to a partition of the database
- All data stored in RBPEX cache
- Dense cache organization
  - Stride-preserving cache layout
  - Avoids read amplification
- Can asynchronously seed a new Page Server
XStore (Persistence Tier)

- **Durability**
  - Replicated to 6 nodes across 3 AZs

- **Snapshots**
  - Full copy sent per week
  - Daily delta
  - Log backup per 5 minutes

- **Supports Backup and Recovery**
  - Point-in-time restore (requires snapshot and small-ish delta log)
Performance - Throughput

- 5% less CPU utilization vs HADR
  - On CDB default mix
  - More time in Ios
- In Update-heavy CDB
  - Log becomes bottleneck
  - Still higher as backups to XStore are handled downstream

<table>
<thead>
<tr>
<th></th>
<th>CPU %</th>
<th>Write TPS</th>
<th>Read TPS</th>
<th>Total TPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>HADR</td>
<td>99.1</td>
<td>347</td>
<td>1055</td>
<td>1402</td>
</tr>
<tr>
<td>Socrates</td>
<td>96.4</td>
<td>330</td>
<td>1005</td>
<td>1335</td>
</tr>
</tbody>
</table>

Table 2: CDB Throughput: HADR vs. Socrates (1TB)

<table>
<thead>
<tr>
<th></th>
<th>SF</th>
<th>Log MB/s</th>
<th>CPU %</th>
</tr>
</thead>
<tbody>
<tr>
<td>HADR</td>
<td>30000</td>
<td>56.9</td>
<td>46.2</td>
</tr>
<tr>
<td>Socrates</td>
<td>30000</td>
<td>89.8</td>
<td>73.2</td>
</tr>
</tbody>
</table>

Table 5: CDB Log Throughput: HADR vs. Socrates
Performance - Caching

- HADR 100% hit-rate, but limits database size
- Still large cache hit rate despite
  - Cache only 15%, 1% database size

<table>
<thead>
<tr>
<th>Data Size</th>
<th>Scale Factor</th>
<th>Memory Size</th>
<th>RBPEX Size</th>
<th>Local cache hit %</th>
</tr>
</thead>
<tbody>
<tr>
<td>1TB</td>
<td>20000</td>
<td>56GB</td>
<td>168GB</td>
<td>52</td>
</tr>
</tbody>
</table>

Table 3: Socrates Cache Hit Rate (CDB)

<table>
<thead>
<tr>
<th>Data Size</th>
<th>Customers</th>
<th>Memory Size</th>
<th>RBPEX Size</th>
<th>Local cache hit %</th>
</tr>
</thead>
<tbody>
<tr>
<td>30TB</td>
<td>3.1M</td>
<td>88GB</td>
<td>320GB</td>
<td>32</td>
</tr>
</tbody>
</table>

Table 4: Socrates Cache Hit Rate (TPC-E)
Questions?