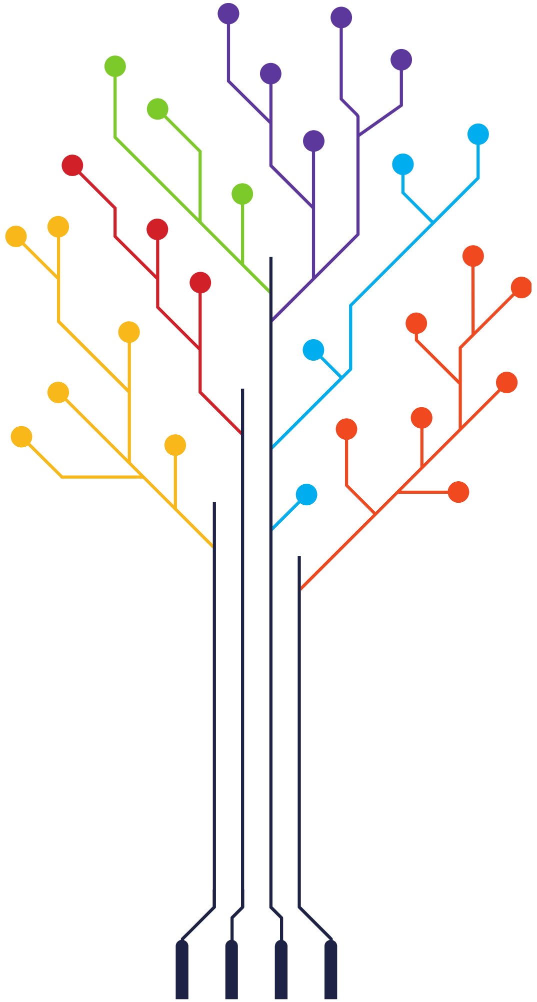

Designing a scalable data platform for high performance transaction processes

Coeo
Making SQL Sense

High performance transactional systems

- Revenue generating engines
- Database is in the critical path
- Lots of custom logic
- Data can never be lost

Problems people are having right now

Lots of custom code in the data tier → Database servers become CPU bound

Traditional database design doesn’t scale → Locking and deadlocking issues

Data is created too fast to update reporting databases → Conflicting workloads, out of date data or more locking

Coeo

Making SQL Sense

Platform Design goals

Understand workload profile

Read vs. write operations
Large vs. small queries
User vs. machine generated

Understand your platform

Physical vs. virtual vs. cloud
Storage performance
Software licensing model

Understand application architecture

Scale-out app tier vs. data-tier centric
Logic written in app or data languages
Expected pressure points during high volumes

Understand your expectations

Security and compliance
Availability
Analytics

Platform Design goals

Memory

Keep working data set in memory
Avoid large analytics queries
Consider in-memory features

Queries

Transactional systems favour loop joins
As well as indexed paths to the data
CPU work is the scalability killer

Storage

Avoid pagelatch contention
Create enough data files (25%+ of CPU cores)
Transaction log will always be a bottleneck
High write performance storage

Server architecture

Application network round trips
Individual CPU core performance
High availability features

In-memory OLTP

- **In-memory database engine within SQL Server**
  For high throughput and highly concurrent database applications

- **Application developers toolkit (not a /enable flag!)**

- **Memory optimised tables**

  Fully durable and ACID compliant
  No locks or latches
  Row versioning concurrency control
  Good for large insert operations (pagelatch)

  Accessible from regular T-SQL queries

- **Natively compiled stored procedures**

  T-SQL compiled once into native DLLs
  Access memory optimised tables
  Reduced T-SQL surface area
  Good for heavy CPU-bound calculations

  Executable by regular T-SQL queries