Drinking from a Fire Hydrant

How to process and analyze 1M messages/second

Mark Tsimelzon
President & CTO
Coral8, Inc.
www.coral8.com
mark@coral8.com

Question:

What do these applications have in common?

Financial Services

- Algorithmic Trading
- Arbitrage
- Risk Management

Network Management

- Server/App Monitoring
- Security Monitoring
- SLA management

Manufacturing

- Process Monitoring
- Yield Management
- Exception Monitoring

Sensor Networks

- RFID applications
- Location-based applications
- Power Grid monitoring

Web Applications

- Real-time personalization
- Real-time ad targeting
- Real-time ad management

Military / Homeland Security

- Battlefield monitoring
- WMD sensor monitoring
- Intelligence gathering

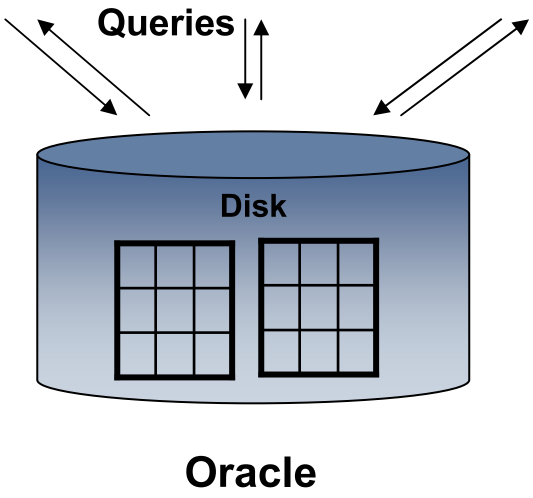
Answer:

An ability to:

- …Analyze (filter, aggregate, correlate, etc.)
- …Large volumes of data (1K-1M messages/sec)
- …With low latency (milliseconds/seconds)

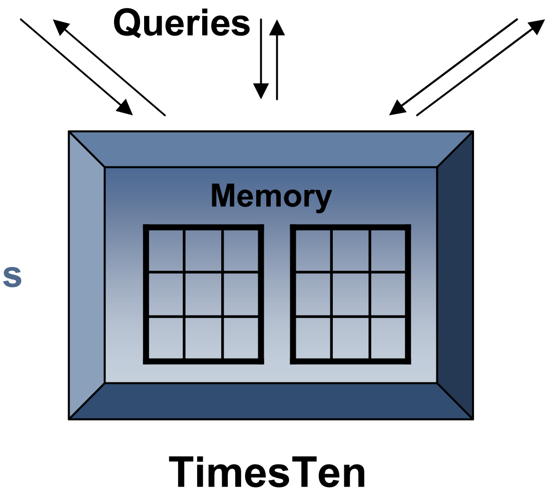
Solution, 1st attempt:

- Use good old database (Oracle, etc.)
- Good:
  - Familiar (everybody knows it)
  - Standard Relational Model
  - Standard Query Language for complex analysis
- Bad:
  - Performance is orders of magnitude below the requirements
  - Reason: DBs are disk-centric. Everything must be stored on disk. Disks are slow



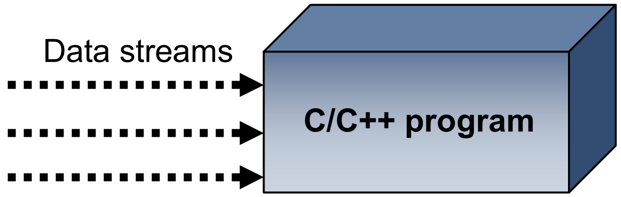
Solution, 2nd attempt:

- Use **in-memory** database (TimesTen, etc.)
- Good:
  - All the advantages of a database
  - Not disk-based, so should be no performance problems
  - Sounds like an ideal solution except…
- Bad:
  - Performance is still nowhere near sufficient. Two reasons:
  - If one wants to check something 10 times/sec, one needs to issue a query 10 times/sec, and the queries run independently
  - Not clustering-friendly



Solution, 3rd attempt:

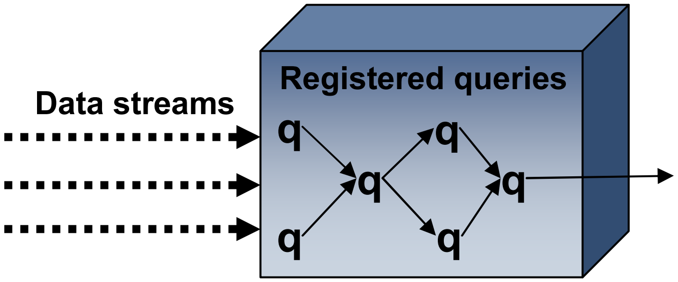
- Just write a whole damn thing in C/C++
- Good:
    - Can be fast if you know what you are doing
- Bad:
    - Few people know what they are doing
    - These people are expensive
    - Writing in C/C++ is slow
    - Changing & managing C/C++ code is tough
- Thought:
    - What if we could combine the performance of C/C++ code with the ease of use of a database?



Need a fresh idea…

Fresh Idea (the big one)

- DB was designed for **static data** and **dynamic queries**
- It is hard to design a system for **dynamic data** AND **dynamic queries**
- But perhaps we can design a system for **dynamic data** and **static queries?**



"DB upside down"

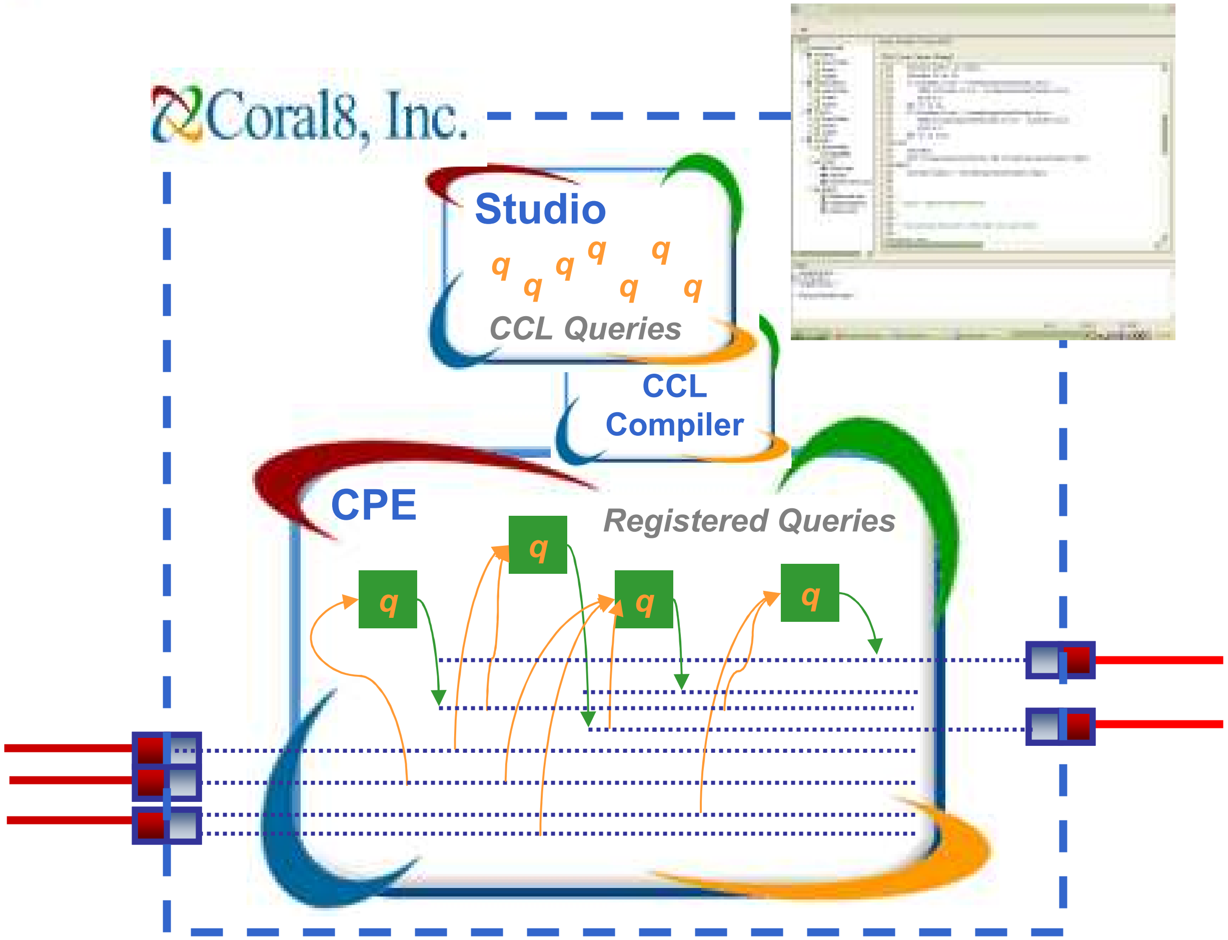
Academic Projects

- STREAM (Stanford: http://www-db.stanford.edu/stream/)
  - Extend SQL to support a registered query processing model and time/window extensions
- Aurora (Brandeis, Brown, MIT: http://www.cs.brown.edu/research/aurora/)
  - Workflow-like GUI to specify processing
- Telegraph (UC Berkeley: http://telegraph.cs.berkeley.edu/y)
  - All over the map…

Coral8's IDP System At Work

Live Input Data
Streams
Feeds
Messages
RFID Readers
Data Sniffers

Static Input Data
Database
Warehouse
DataMarts



Output Data
Dashboards
Databases
Mobile Devices
Email
Excel
Analytics
Monitors

Demo

- DEMO

Some Open Problems

- Clustering:
  - For performance
  - For HA
  - Automatic query parallelization
- Quality-of-Service
  - Guarantees
  - Priorities
  - Load Shedding
- Integration
  - Incoming data stream
  - Alerts, Actions, Visualization
  - Tight integration with DBs, OLAP, etc.
- Integration with Web Services stack
  - XML messages
  - SOAP
  - XQuery