Improving Database Performance by Removing the Database

Simon Munro

EMC® Consulting

@simonmunro

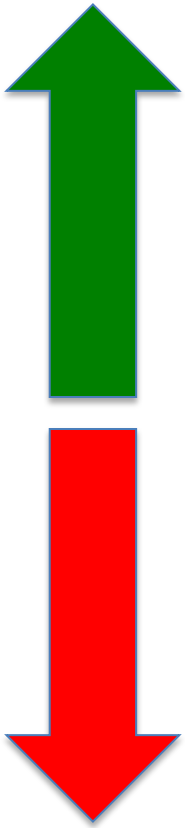
This is not about NoSQL

It is about why NoSQL won't die

For NoSQL, watch the video from SQLBits Goes West

@simonmunro

Isn't it a great time to be a data dude?



Data Growth

Everything Else

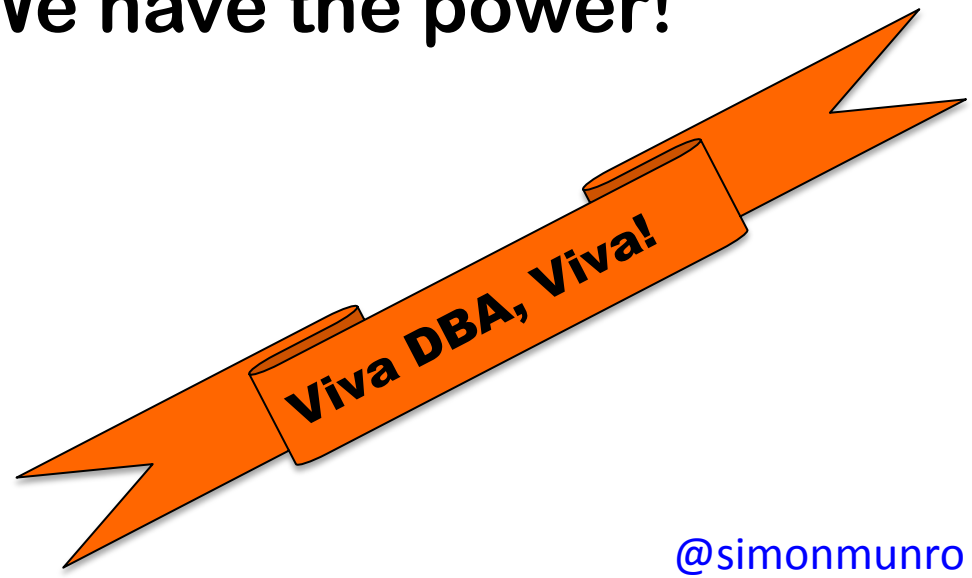
@simonmunro

Information is power

Information is on the increase

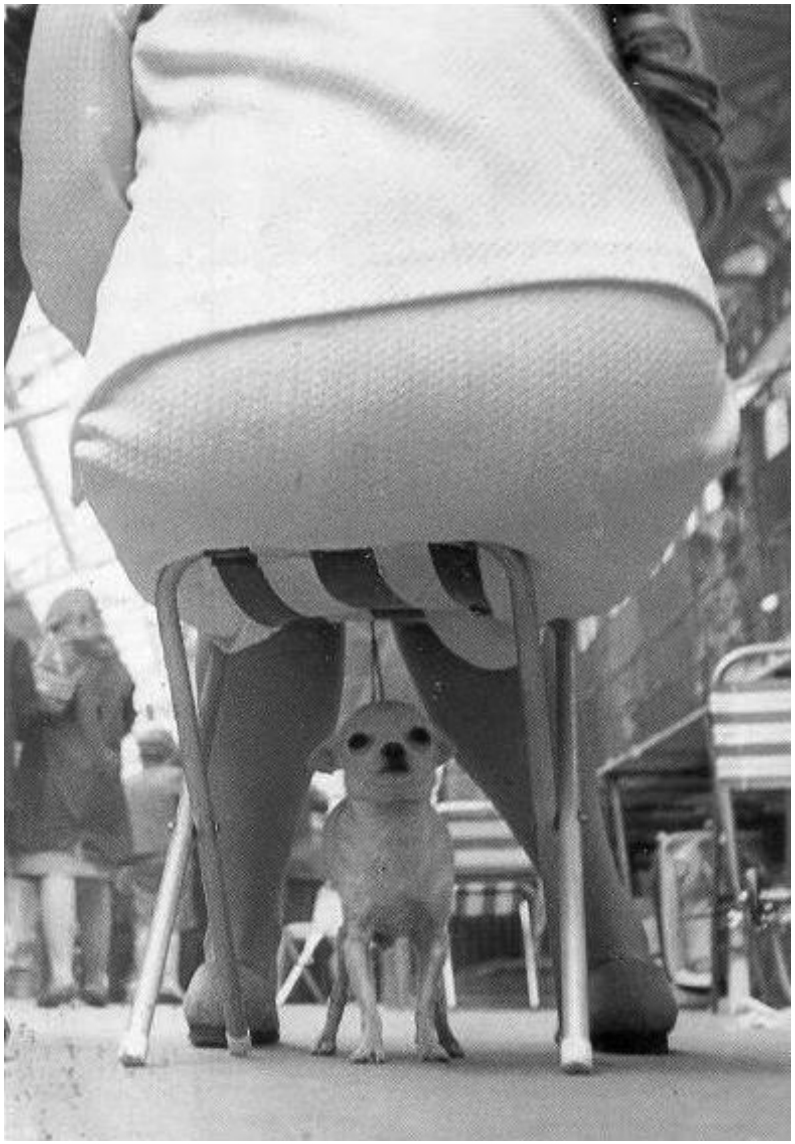
We administer the information

We have the power!



@simonmunro

So why are we so stressed?



@simonmunro

We need to handle increasing demands

Data Volumes

Transactional Volumes

User Volumes

Time to Market

Responsiveness

With better levels of service

(Yeah right… and with the same fake smiles)

- Availability
- Reliability
- Security
- Functionality
- Performance

Rising Costs

Licensing Costs

Specialised Hardware

Operational Costs

Vendor Lock-in

@simonmunro

It all takes more effort

Design Effort

Operational Effort

Collaboration Effort

Procurement

Setup

Migration

Decommissioning

But we have to lower costs

Recession Impact

Focus on Efficiency

Staff cuts

Marginal business cases

Infrequent Upgrades

What can change?

Networking Topologies?

Hardware Infrastructure?

Storage?

Database Platforms?

Application Architectures?

Demands?

Requirements?

Do we want it to change?

It seems to work

It is familiar

We have the skills

Risk is low

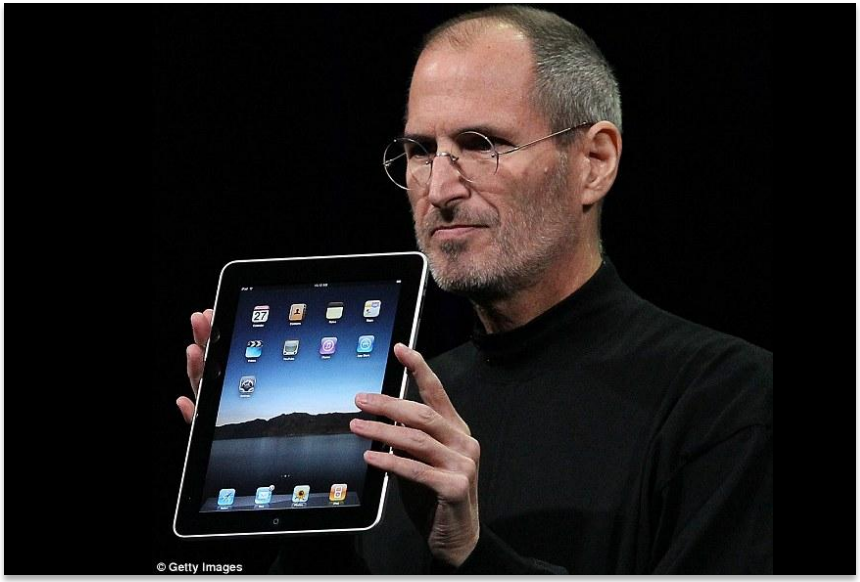
We have the infrastructure

Does it really work?

Designed for Purpose

@simonmunro

Misused Designs



SQL as a Good Design

De-facto data storage mechanism

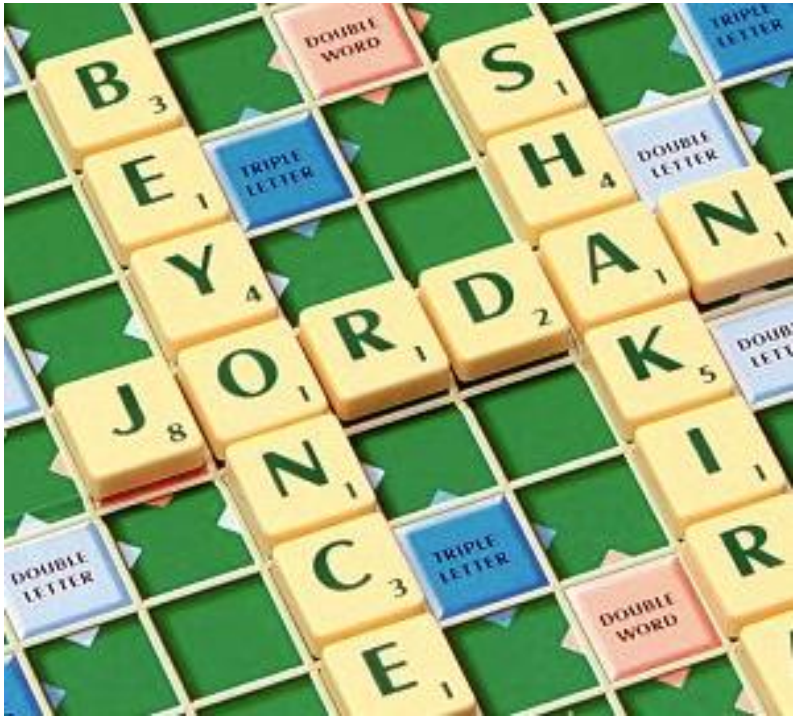
(Fairly) well understood patterns

- Database structures
- Querying

Close to classic relational model

Optimised embedded languages

SQL as Misused Design



API is far to open

‘Post relational’ datatypes

Suboptimal for some domains

Works for me

Why?

SQL is simply fantastic!

That is what is was designed to do

It's always been that way

Have you looked at alternatives?

Have you checked your assumptions?

SQL Patterns have become data management patterns

Backup

Query

Security

ACID

Data models

Application development

THIS is how we work with data!

@simonmunro

Scalability

It is hard

and expensive

and unfamiliar

and those are just the people problems!

Scalability and NoSQL

Lack of SQL scalability out is the popular NoSQL argument

Actually, most NoSQL databases do not scale out either

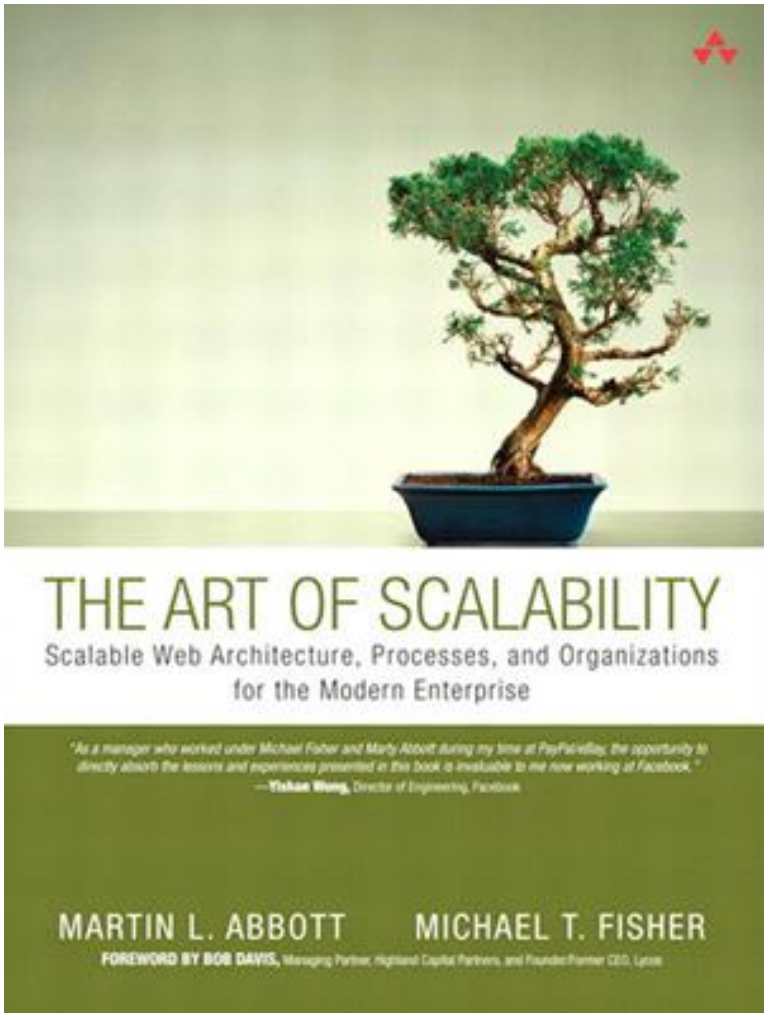
jamesgolick.com

CouchDB
Redis
Tokyo cabinet
MemcacheDB
Berkeley DB

Cassandra
Riak
Voldemort

@simonmunro

Scalability is more than an engineering problem



Operational processes

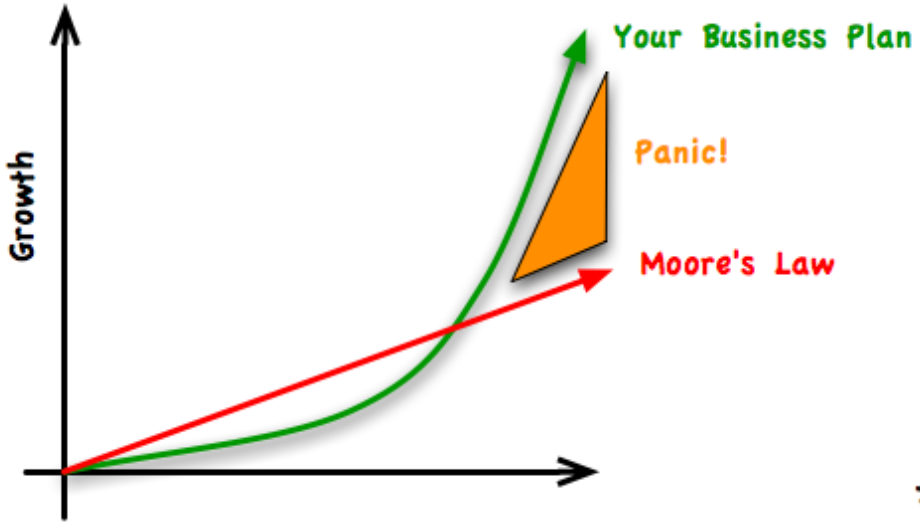
Maintenance

Skills

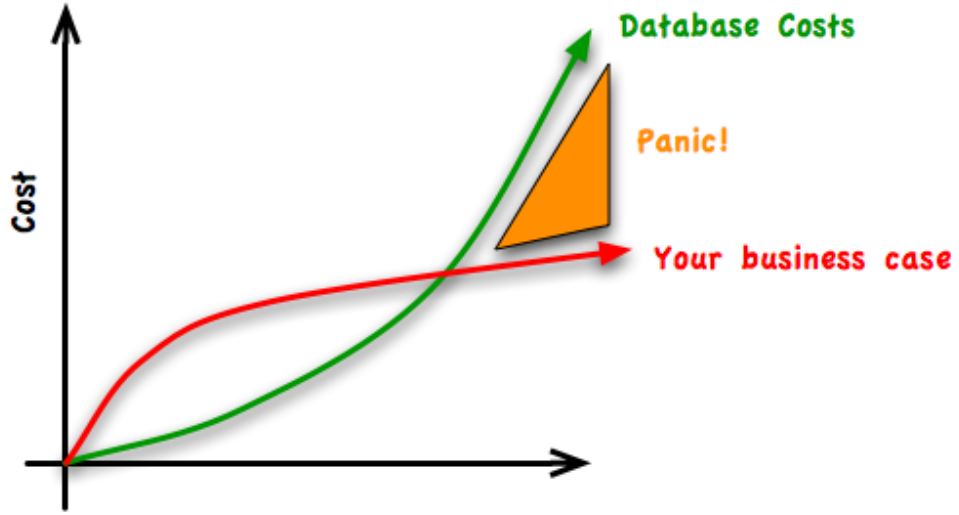
Partnerships

Legal

@simonmunro


Growth
Your Business Plan
Panic!
Moore's Law


Growth
Your Business Plan
Panic!
Your Operational Efficiency


Cost
Database Costs
Panic!
Your business case

@simonmunro

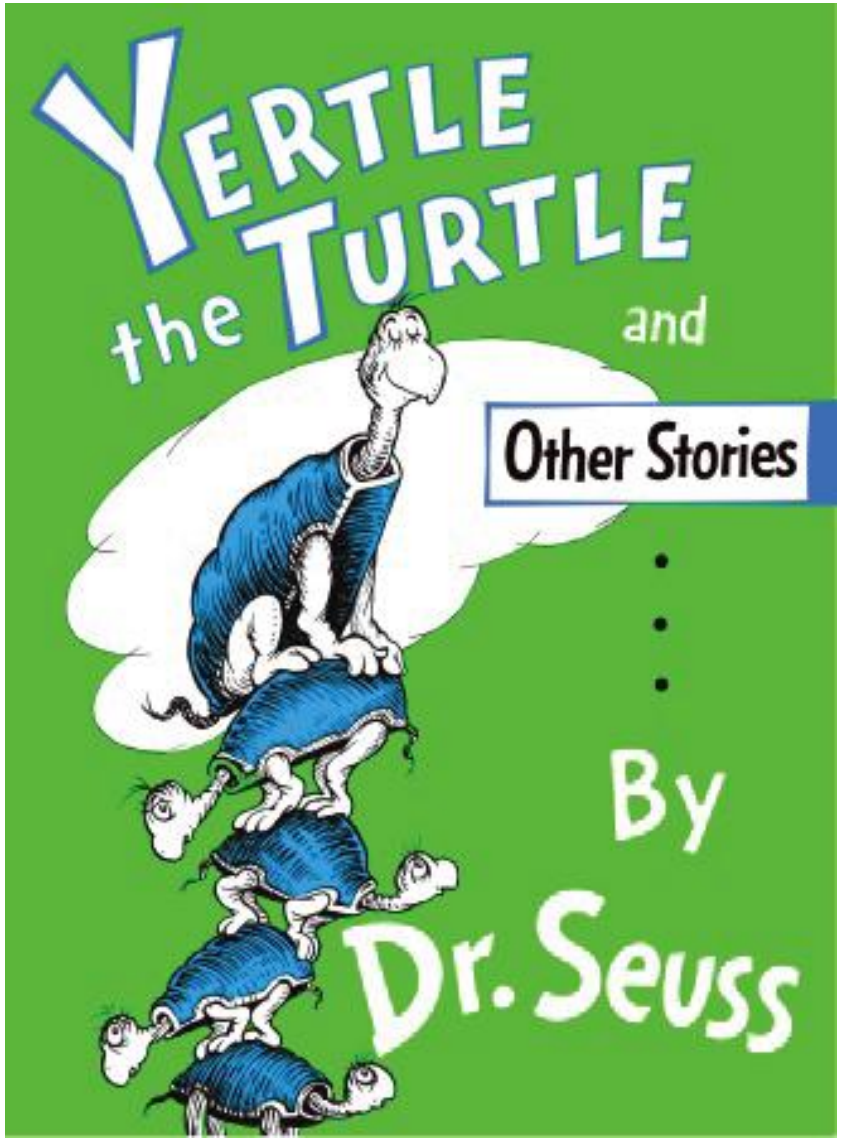
The core flaw in SQL oriented design

MSDN

http://technet.microsoft.com/en-us/library/cc966500.aspx

"The log records associated with the transaction (and all previous log records) must be written to **stable storage**. The transaction is not considered committed until the log records are correctly flushed to **stable media**. (Log is hardened.)"

Stack of Turtles



Processor

Memory

Disk Controller

Disk

@simonmunro

More turtles!

Processor

Memory

San Controller

Network

Switch

SAN Controller

SAN

RAID Controller

Disk

System Management

Firewall

Backup

Encryption

Fibre Network

Virtualisation

Monitoring

Is our performance bottleneck based on this?

@simonmunro

Storage Gets Better

Tape is dead

Disk is tape

Flash is disk

SSDs to the rescue!

Memory is cheap

What if we kept everything in memory?

Database Ivory Towers That Piss Me Off

@simonmunro

All data should be in the database

The Culture

Q: How long do you need to keep this data?
A: 5 years

Original atomic records need to be stored for future analysis

Keeping it in SQL is best so that it can be queried

All data should be in the database

The Reality

The business value of data is misunderstood

The cost of data retention is hidden

Users cannot query data

A lot of data is not in the database anyway

Single Version of the Truth

(One fact, one place, once)

The Culture

Master Data Management

Normalisation

Only editable in **this** database

Removal of duplicates

Integration is tedious

Single Version of the Truth

(One fact, one place, once)

The Reality

A complete myth

Data **always** lands up everywhere

System integration duplicates data

MDM effort is high

There are **a lot** of spreadsheets

Data is inherently temporal

All processing should be done in the batch run

The Culture

Single database

SQL Database as a source for all data

Aggregation load is to high
for transactional systems

It is optimal and fair for
all departments

All processing should be done in the batch run

The Reality

Batches always tend toward using up all available time

Batch code is the worst

Batch failures are a source of Panic, risk and stress

Operational costs for batch runs are high

Queryability

The Culture

All fields are queryable

User demand... apparently

This is what the relational database is for

Queryability

The Reality

Performance issues limit searches to key fields

Other structures are created to make querying easier

Rows, columns and tables are an abstraction anyway

Required by Auditors

The Culture

"We need an audit trail"

"Audit requires that we…"

Required by Auditors

The Reality

Real life auditors are seldom making requests

What are auditors doing specifying IT architectures anyway?

Auditors are change averse

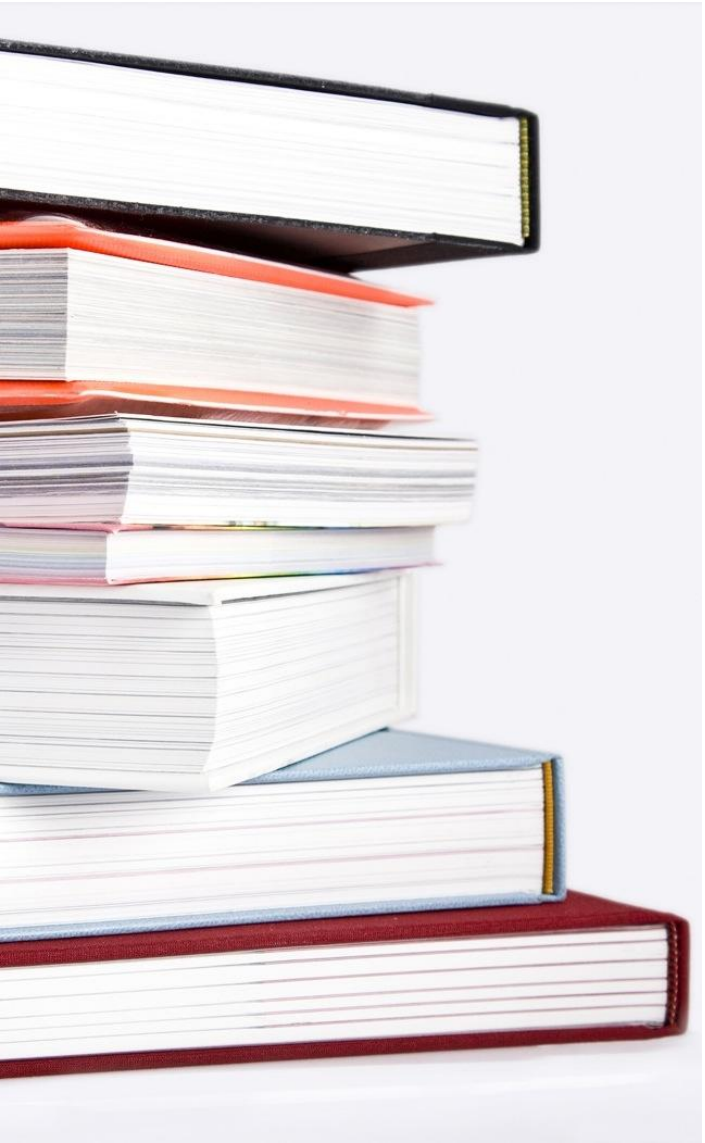
Required by Regulations

The Culture

"We can't do that because of regulations"

"We do it like this for regulatory reasons"

Required by Regulations



The Reality

Regulations are difficult to understand

Regulations are full of legalese
and contradictions

Most regulatory requirements are
based on myths

Consistency

The Culture

Every database operation needs to be within a transactional context

We have to ensure that in the event of a failure that data is correct

Data has to be in a consistent state before it can be used

All clients executing a simultaneous query should get the same result

Consistency

The Reality

Consistency is impossible in a Distributed environment

The Internet is a distributed environment

Given the choice, business would probably spend their money elsewhere

Even the most consistent data may not reflect reality

valid for

★CONSISTENCY★

win every argument

Issued against:

- ☐ NoSQL
- ☐ Eventually Consistent
- ☐ High Performance
- ☐ New Ideas
- ☐ Annoying People
- ☐ _____

PROSQL TRUMP CARD

valid for

★ ACID ★

win every argument

Issued against:

- ☐ Financial Transactions
- ☐ BASE
- ☐ Web Traffic
- ☐ New Ideas
- ☐ Annoying People
- ☐ _____

PROSQL TRUMP CARD

valid for

★ BIG VENDOR ★

win every argument

Issued against:

- ☐ Open Source
- ☐ NoSQL
- ☐ Unrecognised Names
- ☐ New Ideas
- ☐ Annoying People
- ☐ _____

PROSQL TRUMP CARD

valid for

★QUERYING★

win every argument

Issued against:

- ☐ Key-Value Stores
- ☐ XML
- ☐ Developers
- ☐ New Ideas
- ☐ Annoying People
- ☐ _____

PROSQL TRUMP CARD

valid for

★ AUDITABILITY ★

win every argument

Issued against:

- ☐ NoSQL
- ☐ Developers
- ☐ High Performance
- ☐ New Ideas
- ☐ Annoying People
- ☐ _____

PROSQL TRUMP CARD

valid for

★REGULATIONS★

win every argument

Issued against:

- ☐ Websites
- ☐ Distributed Processing
- ☐ Cloud Computing
- ☐ New Ideas
- ☐ Annoying People
- ☐ _____

PROSQL TRUMP CARD

sqlbits

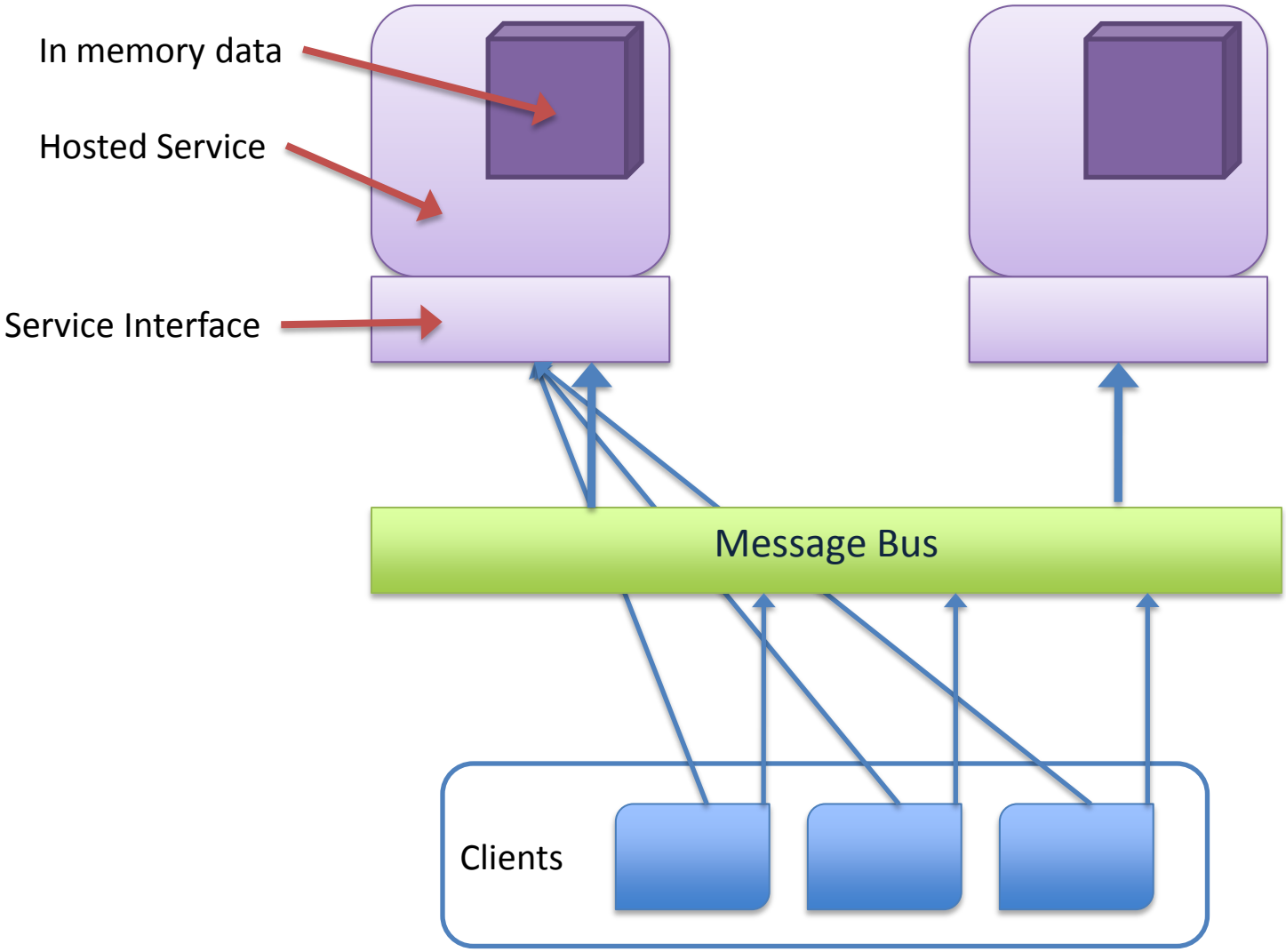
Sponsored by:

EMC® Consulting

Design by: CORE MATTERS people • performance • productivity

@simonmunro

Can we just store data in memory?



What are the patterns?

We already have them

Mainframes, Swift, Reuters, Trading

Knowledge is hidden or rusty

Diverted to SOA for a while

Being revived by the cloud

Already being used without DBA knowing

The Influence of the Cloud

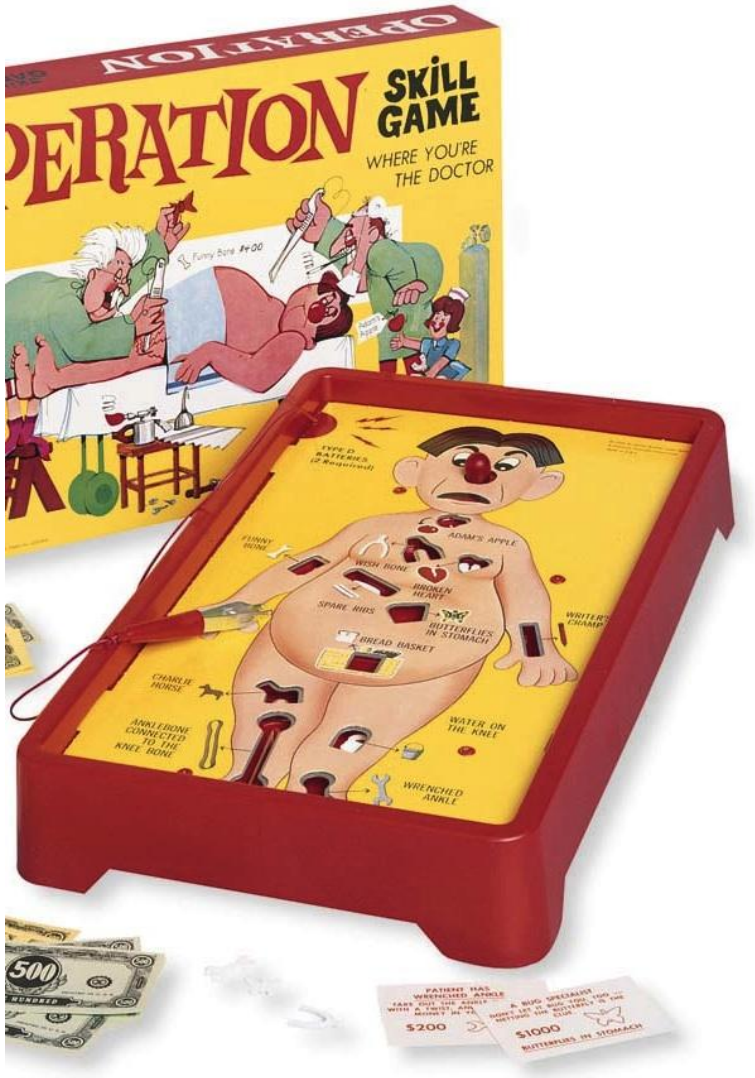
Someone else can deal with the stack of turtles

Cannot make assumptions about availability of a specific process

No control over underlying hardware

Applications are failure aware

SQL Removal Techniques



Change your approach to dealing with data

Change your application architectures

Change how business treats data

Work with other disciplines (e.g. developers and compliance)

@simonmunro

Changing Approach to Data

Cache

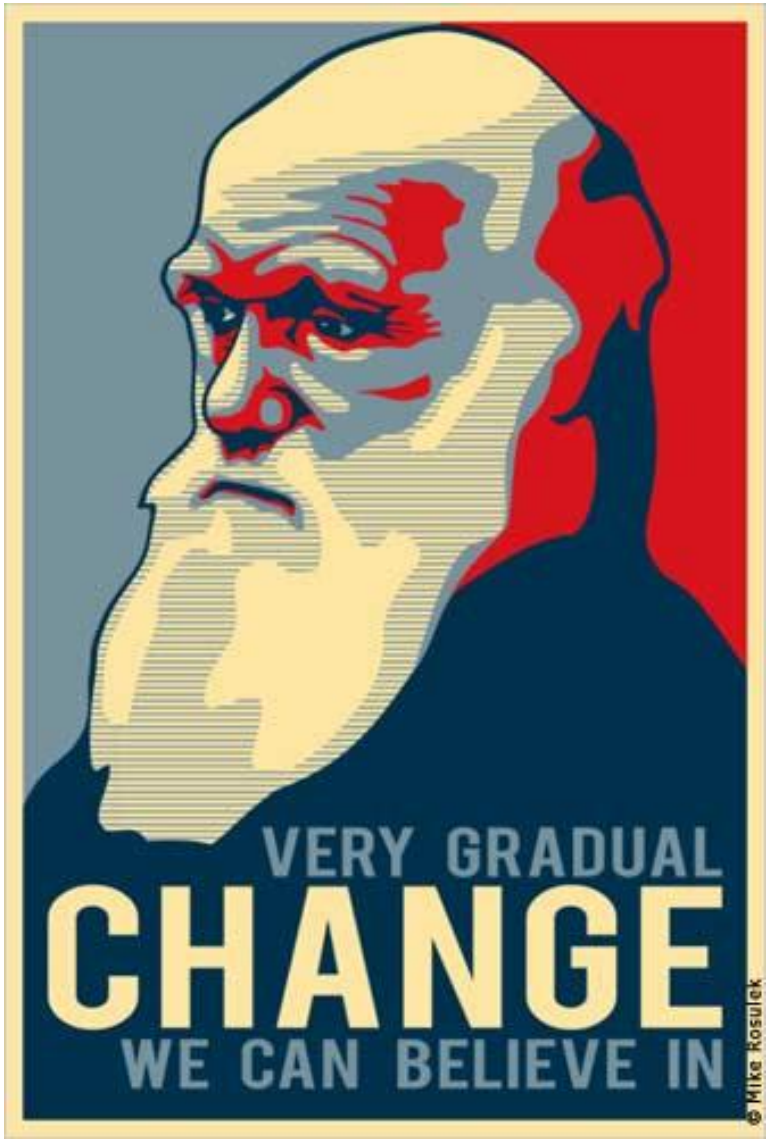
Read-only data stores

Specialised Data Stores

Main Memory Databases

Pre-emptive archiving

Changing Approach to Application Architectures



Message Orientation

Eventual Consistency

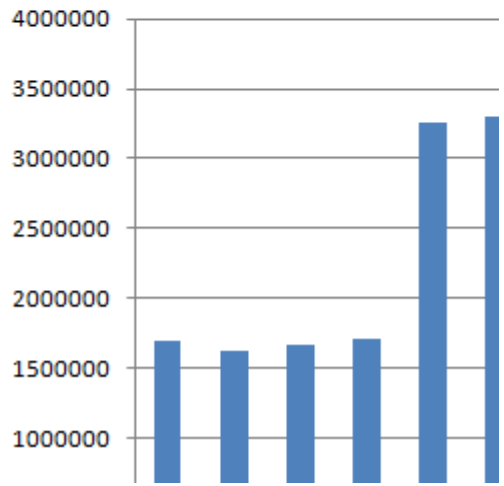
Store and process data as Close to source as possible

Design for service degradation

Apology based computing

An Example : Changing Batch Jobs

eriod	Commodity	Position
30 days	Coal	1699903
30 days	Gas	1624782
30 days	LNG	1669611
30 days	Oil	1705847
to 90 days	Coal	3252545
to 90 days	Gas	3300940

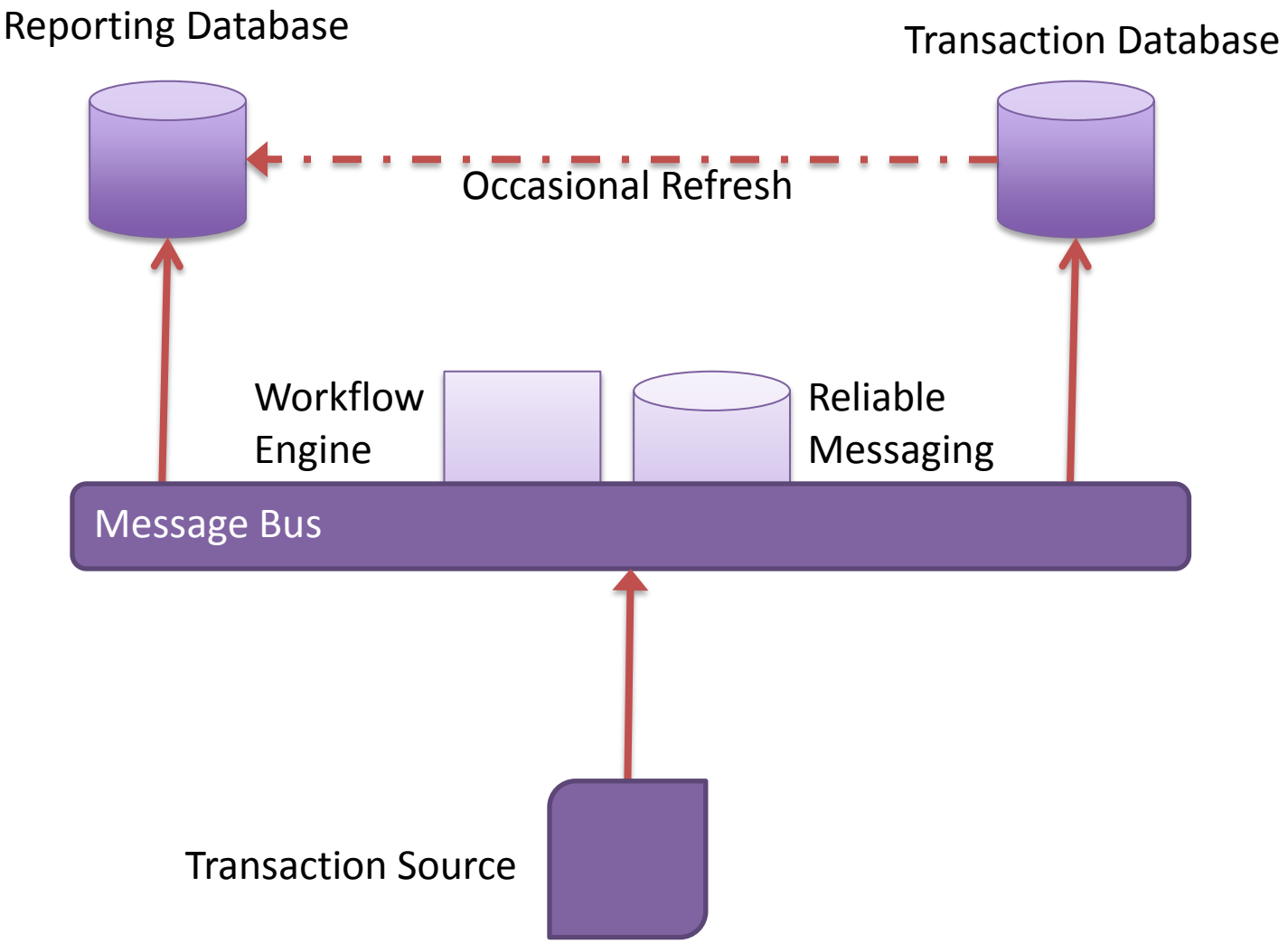


The Scenario:

Users require access to Summaries of a large transactional database

The transactional database is under load so summaries are run in the overnight batch job

The Result


Reporting Database
Transaction Database
Occasional Refresh
Workflow Engine
Reliable Messaging
Message Bus
Transaction Source

Resistance to Change

Incumbent Investment

Risk of Failure

Fear of Failure

Jobs Protection

Egos

Vendor Interests

ORACLE®

Q3 2010
New software licences US$1.7B
Updates and product support US$3.2B

Microsoft®

2009 Annual Report
Revenue US$ 58B
Client (Windows) US$ 14.7B
Server and Tools US$ 14.1B

@simonmunro

It's Not Easy

High Engineering Cost

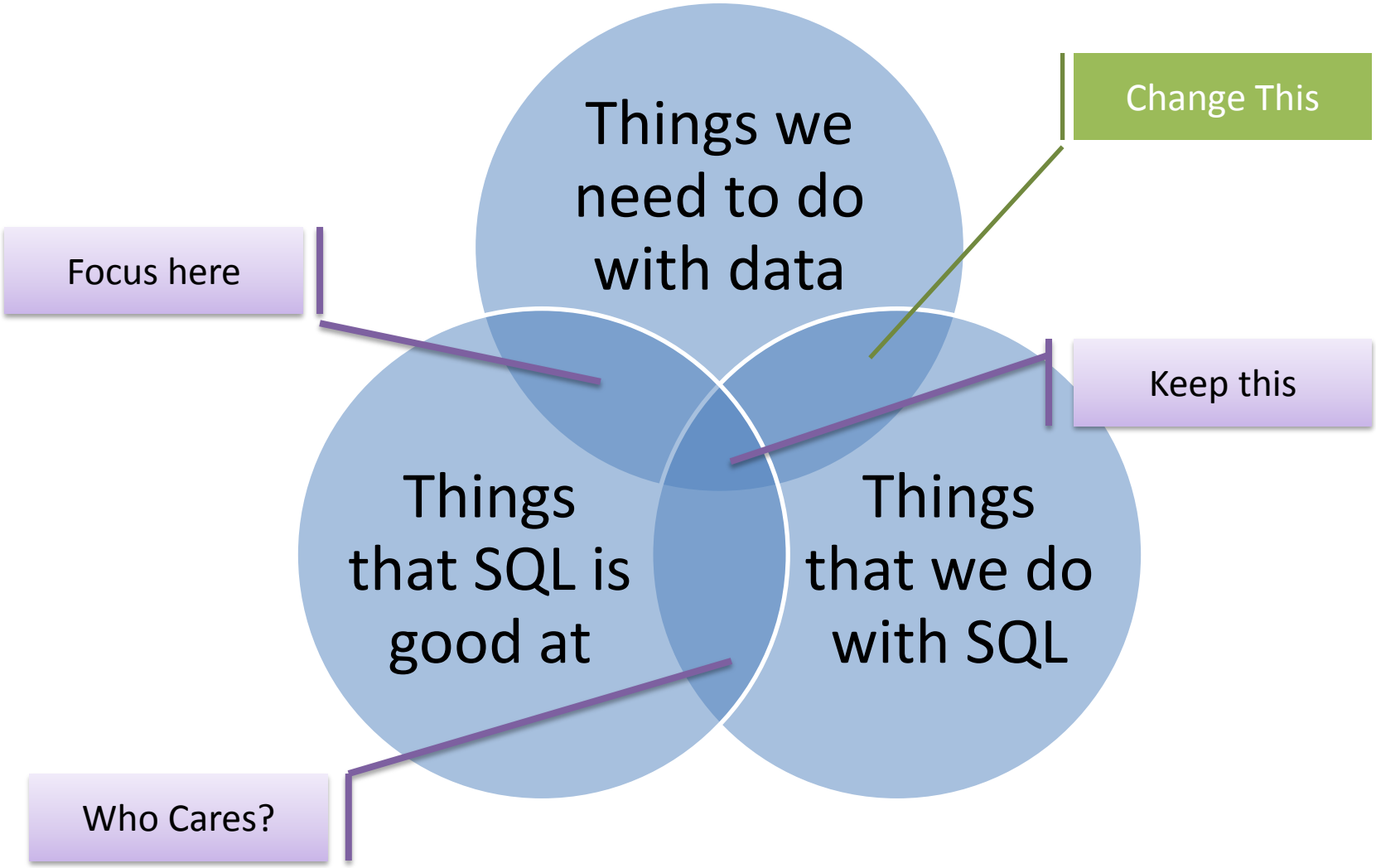
Lack of Design Patterns

Lack of Experience

Risk of Poor Implementation

Sponsor Support

No Big Vendor


Things we need to do with data
Things that SQL is good at
Things that we do with SQL
Change This
Focus here
Keep this
Who Cares?

The Data API is changing

Data demands are exploding

We are wasting effort, money and sanity

SQL doesn't do everything

Alternatives can make a big difference to solutions

Get involved in the debate

Close the gap to 'them'

Advance the state of the art

Be less restrictive

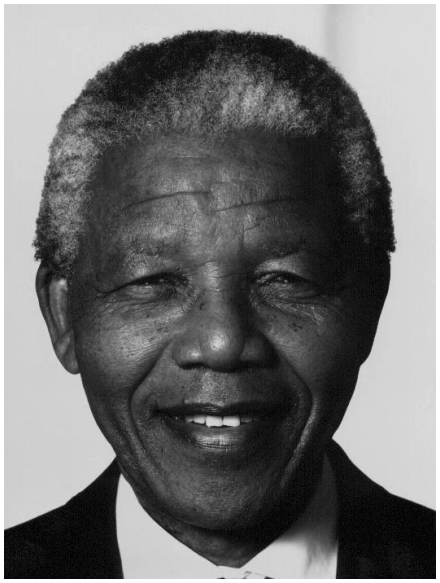
Embrace change

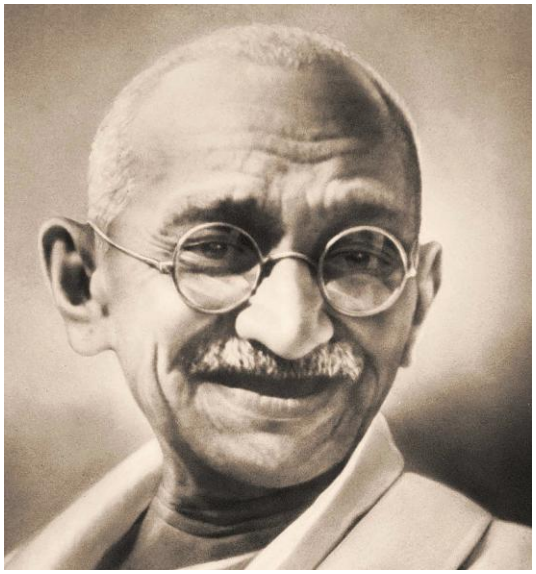
Understand the business needs

Its time for database professionals to take back the database and stop people pissing all over our domain

@simonmunro

What kind of trusted advisor are you?





@simonmunro

Fill in feedback

Visit the sponsors please

Look for the video uploads

Slides (and trump cards) on simonmunro.com and SQLBits soon

@simonmunro