

# Relational vs. Non-Relational

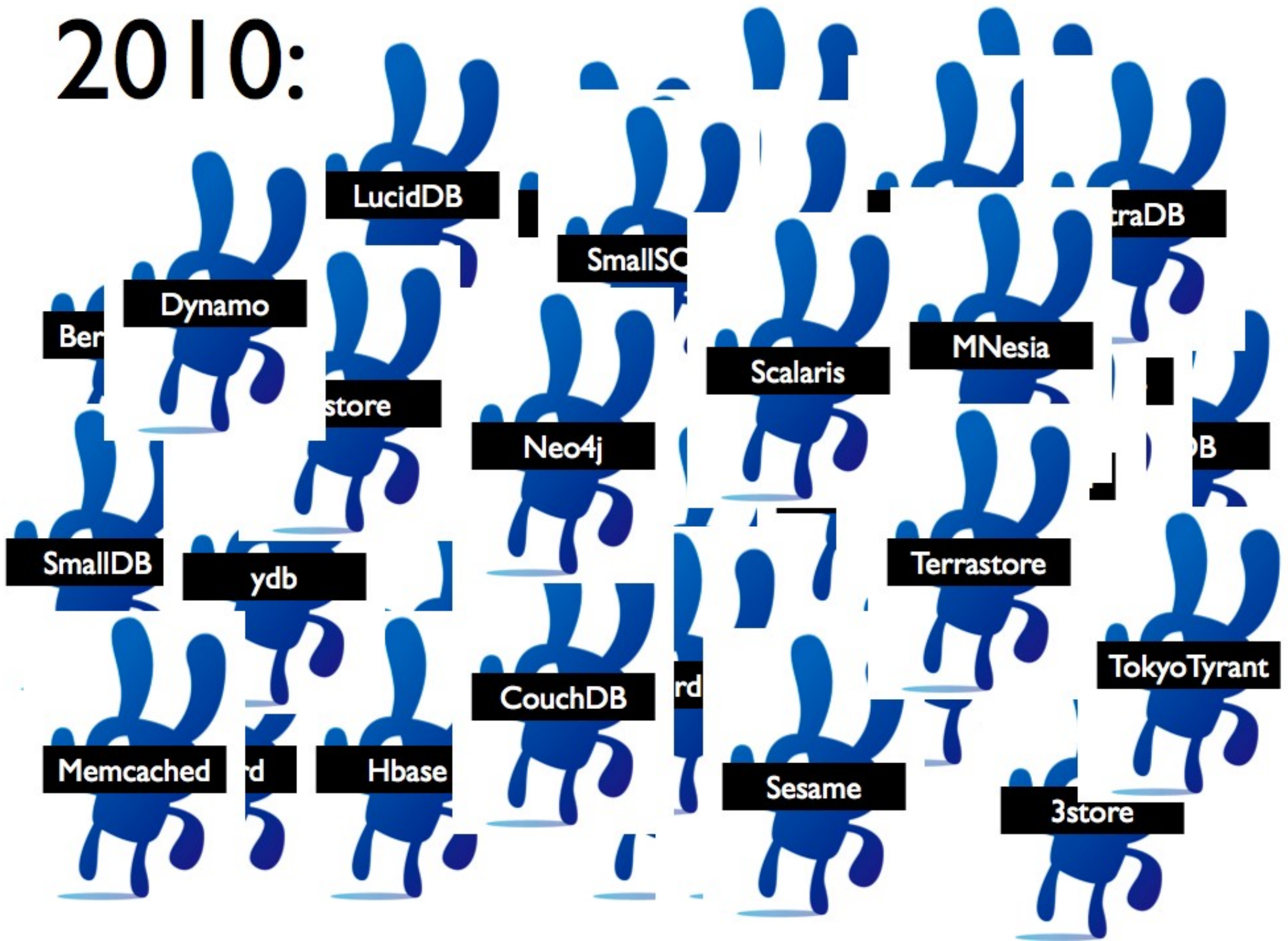
# 2003:

- MySQL
- PostgreSQL
- FireBird
- BerkeleyDB
- Derby
- HSQLDB
- SQLite

# 2003:

- **MySQL**
- **PostgreSQL**
- **FireBird**
- BerkeleyDB
- **Derby**
- **HSQLDB**
- **SQLite**

# 2010:



**“NoSQL  
movement”**

**All  
non-relational  
databases**

**Are *not***  
**the same**

## **Graph**

Neo4J

HyperGraphDB

Jena

## **Document**

CouchDB

BerkeleyDB-XML

Solr

## **Key-value**

Memcached

Tokyo Cabinet

db4o

RIAK

## **Distributed**

Cassandra

Hypertable

MySQL NDB

## **Hierarchical**

MongoDB

**All  
relational  
databases**

**Are *not***  
**the same**

## **Embedded**

SQLite

Firebird

SQLite

## **OLTP**

PostgreSQL

MySQL

Oracle

SQL Server

## **MPP**

TeraData

Greenplum

Aster

## **Streaming**

Streambase

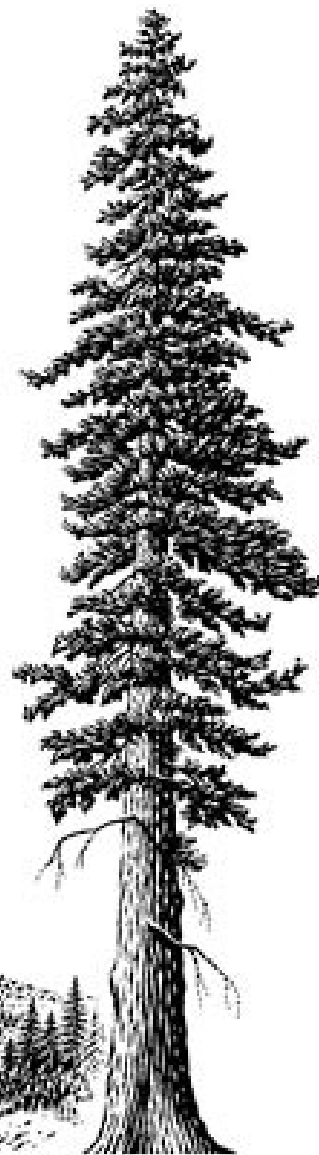
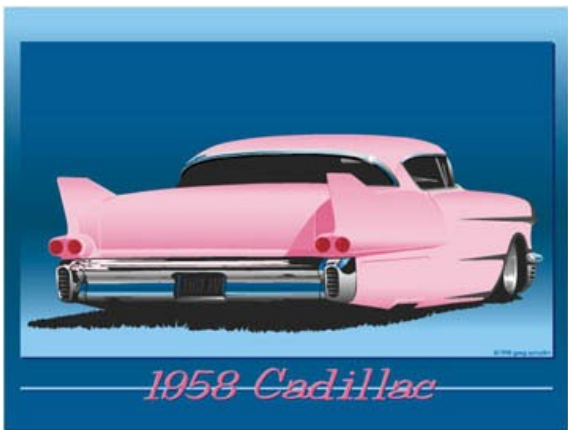
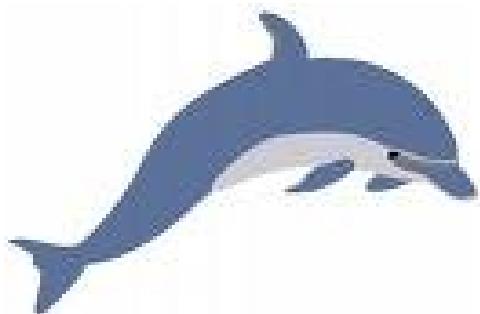
Truviso

## **C-Store**

LucidDB

MonetDB

# NoFins



# MYTHBUSTERS

**“No 900  
moment”**

A rectangular metal stamp, heavily rusted and tilted diagonally from the bottom-left towards the top-right. The word "RESISTED" is embossed in a bold, serif font across the center of the stamp. The stamp is positioned over the text "No 900" and "moment".

# Mythbust #2

**“revolutionary”**

**There**

**are**

**no**

**new**

**database**

**designs**

There are only new  
*implementations*  
*and*  
*combinations*

“A database storing application-friendly formatted objects, each containing collections of attributes which can be searched through a document ID, or the creation of ad-hoc indexes as needed by the application.”

**CouchDB, 2007**

**Pick, 1965**

# CouchDB, 2007

embeddable Pick

JSON storage

REST API

map/reduce

**“revolutionary”**

**“ evolutionary ”**

“renaissance  
of  
non-relational  
databases”

# Mythbust #3

“non-relational  
databases  
are toys”

# Google

## Bigtable

# Amazon

# Dynamo

FaceBook

Memcached

# US Veterans' Administration Pick, Caché

# Mythbust #4

**“Relational  
databases  
will become  
obsolete”**

*“Three decades past, the relational empire conquered the hierarchical hegemony. Today, an upstart challenges the relational empire's dominance ...”*

# XML Databases 2001

--Philip Wadler, Keynote  
VLDB, Rome, September 2001

Anyone remember  
XML databases?

No?

What happened?

established relational  
and non-relational  
databases  
hybridized XML

Oracle XML  
PostgreSQL XML2  
BerkeleyDB XML  
DB2

# Mythbust #5

**“Relational databases  
are for when you need  
ACID transactions.”**

# Transactions

≠

# Relational

# **Robust Transactions without Relationality:**

BerkeleyDB  
Amazon Dynamo

# **SQL Without Transactions:**

MySQL  
MyISAM  
MS Access

# Mythbust #6



**You**

**do**

**not**

**have**

**to choose**

**one  
database.**

Choose  
the database system  
which fits your  
*current*  
application goals.  
or ...

**Use more than one  
together**

**MySQL + Memcached**

**PostgreSQL +  
CouchDB**

**or ...**

**Use a Hybrid**

**MySQL NDB**

**PostgreSQL Hstore**

**HadoopDB**

But what about  
relational  
vs  
non-relational?

# Relational OLTP Databases\*

- Transactions: more mature support
  - including multi-statement
- Constraints: enforce data rules absolutely
- Consistency: enforce structure 100%
- Complex reporting: keep management happy!
- Vertical scaling (but not horizontal)

\* *mature ones, anyway*

# SQL vs. Not SQL

SQL promotes:

- portability
- managed changes over time
- multi-application access
- many mature tools

# SQL vs. Not SQL

But ...

*SQL is a full programming language,  
and you have to learn it to use it.*

# SQL vs. Not SQL

No-SQL allows:

- programmers as DBAs
- no impedance
- fast interfaces
- fast development

The main reason  
to use SQL-RDBMSs

# “Immortal Data”

*your data has  
a life  
independent  
of this specific  
application implementation*

**How *do* I  
choose?**

Define the problem  
you are trying  
to solve

- “I need a database for my blog”
- “I need to add 1000's of objects per second on a low-end device.”
- “I need my database to enforce business rules across several applications.”
- “I want my application to be location-sensitive.”
- “I need to cache data and access it 100K times per second.”
- “I need to produce summary reports across 2TB of data.”
- “I have a few hundred government documents I need to serve on the web and mine”
- “I need to know who-knows-who-knows-who.”
- “I need to data mine 1000's of 30K bug reports per minute.”

# Define the features *you actually need*

- many connections
- multi-server scalability
- complex query logic
- APIs
- redundancy
- data integrity
- schema/schemaless
- data mining

**f t the database  
to the task**

“I need a database  
for my blog”

# Use anything!

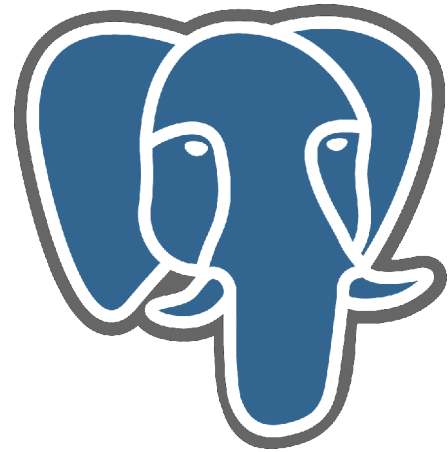
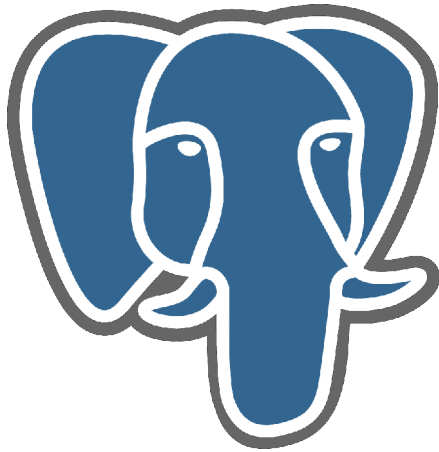
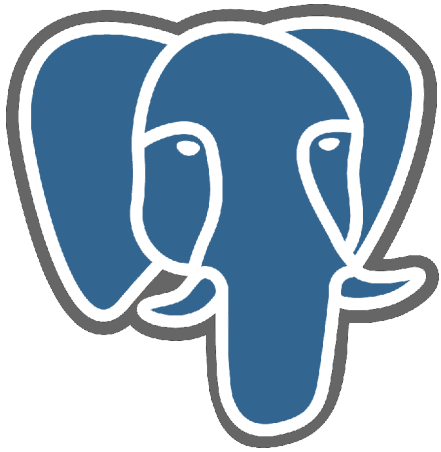
- MySQL
- PostgreSQL
- MongoDB
- SQLite
- CouchDB
- Flatfiles
- DBaseIII
- Something you wrote yourself

“I need my database  
to unify several  
applications and  
keep them consistent.”

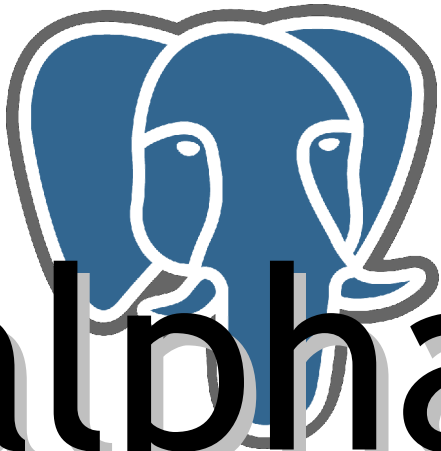
# PostgreSQL

“OLTP  
SQL-Relational Database”

*“It's not just a database: it's a development  
platform”*



**Postgres 9**



**alpha out now!**

**“I need my application  
to be location-aware.”**

# PostGIS

“Geographic Relational  
Database”

# PostGIS

- Queries across “contains” “near” “closest”
- Complex geometric map objects
  - polygons
  - lines (roads, etc)
- Make your own Google Maps!
  - Sunday, 4:30pm, Concourse A

“I need to store  
1000's of event objects  
per second on  
embedded hardware.”

db4object

“Embedded Key-Value  
Store”

# db4object

- German Train System
  - Insert 1000's of objects per second
  - Low-end embedded console computer
  - Simple access in native programming language (Java, .NET)
- 
- *compromise: embedded SQL database: SQLite*

“I need to access  
100K objects per  
second  
over 1000's of  
connections.”

# memcached

“Distributed In-Memory  
Key-Value Store”

# memcached

- Use: public website
- Used for caching 1000's of serialized objects per second
- Available for 100000's of requests per second across 1000's of connections
- Cache each object only once per site
- *Supplements* a relational database

*Alternatives: Redis, TokyoTyrant*

“I need to produce  
complex summary  
reports  
over 2TB of data.”

# LucidDB

“Relational  
Column-Store”

# LucidDB

- For reporting and analysis
- Large quantities of data (TB)
- Complex OLAP & analytics queries
- Business intelligence
- *compliments* a transactional database

“I have 100's of  
government documents  
I need to  
serve on the web  
and mine for data.”

# CouchDB

“Document Store”

# CouchDB

1. CividDB Project
2. Storing lots and lots of government documents
3. Don't know what's in them
4. Don't know how they are structured
5. Store them, figure out structure later by data mining.

*It's also good for web sites!*



# Neo4j

“Graph Database”

# Neo4j

- Social Network Website
- 6 degrees of separation
- “you may also like”
- type and degrees of relationship

“I get 1000's of  
30K bug reports  
per minute  
and I need  
to mine them for  
trends.”

# Hadoop

“Massively Parallel Data Mine”

# Hadoop

- Massive bug report data feed
- 1000's of bug reports per minute
- Each bug report 2-45K of data
- Need to extract trends and correlate inexact data
- Summarize in daily & weekly reports

# Conclusion

- Different database systems do better at different tasks.
  - every database feature is a tradeoff
  - no database can do all things well
- Relational vs. non-relational doesn't matter
  - pick the database(s) for the project or the task

# Questions?

- PostgreSQL Project
  - [www.postgresql.org](http://www.postgresql.org)
  - [josh@postgresql.org](mailto:josh@postgresql.org)
- PostgreSQL Experts
  - [www.pgexperts.com](http://www.pgexperts.com)
    - [www.pgexperts.com/documents.html](http://www.pgexperts.com/documents.html)
- Open Source Database Survey
  - Selena Deckleman
  - Open Source Database Survey:  
[www.ossdbsurvey.org](http://www.ossdbsurvey.org)
- PostgreSQL Booth trade show floor
- **PostgreSQL 9.0 BOF, Tonight 7pm**

Copyright 2010 Josh Berkus, distributable under the creative commons attribution license, except for 3rd-party images which are property of their respective owners.  
Special thanks to Selena Deckelman for the bunnies image and for the Open Source Database survey.

**PGX**  
POSTGRESQL  
EXPERTS, INC.

