I Hate Your Database

Andrew Godwin
@andrewgodwin

flickr.com/96dpi
Hate? Databases?

Hate? Databases?
• Misuse
• Ignorance
• Lies
Different databases, different occasions
<table>
<thead>
<tr>
<th>Relational</th>
<th>PostgreSQL</th>
<th>MySQL</th>
<th>SQLite</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document</td>
<td>MongoDB</td>
<td>CouchDB</td>
<td></td>
</tr>
<tr>
<td>Key-value</td>
<td>Redis</td>
<td>Cassandra</td>
<td>Riak</td>
</tr>
</tbody>
</table>
Some Quick Theory
Atomicity
Consistency
Isolation
Durability
Consistency
Availability
Partition Tolerance
Relational Databases
Common Pitfalls

SELECT item1, item2, item3 FROM basket;

INSERT INTO groups (name, people) VALUES ('friends', 'aaron,betty,chris,damien')

SELECT file_bytes FROM pictures WHERE filename = 'foo.jpg'
names = set()
for book in Book.objects.filter(year=2012):
    names.add(book.author.name)
names = Author.objects.filter(books__year=2012).values_list('name', flat=True).distinct()
MySQL

No transactional DDL

Poor query optimiser

MyISAM: Full-table locking, no transactions

Oracle

Very fast for some operations
SQLite

Little integrity checking (slowly being fixed)

Impossible to do some table alterations

No concurrent access

Very low overhead

Very portable
PostgreSQL

Slow default configuration

Can be a little harder to learn / less familiar

Almost too many features

Incredibly reliable
Document Databases
db.insert(
    {
        'name': 'Sally',
        'tags': ['django', 'python', 'search'],
        'addresses': [
            {'type': 'jabber', 'jid': 'sally@eg.com'},
            {'type': 'phone', 'number': '011899981199'}
        ]
    }
)

db.find(
    {
        'tags': 'python',
        'addresses.type': 'jabber',
    }
)
Advantages

No fixed schema

Low barrier to entry

Closer to Python datatypes
Problems

Immature (but improving)

No transactions

No integrity checking
Key-value stores
db.set('foo', 'bar)
x = db.get('foo')

db.sadd('names', 'andrew')
db.sadd('names', 'brian')
y = db.scard('names')
Traits

Horizontal scaling (but with drawbacks)

Extremely fast

Can only fetch objects by key

Batch/map-reduce queries

Transactions not possible
Other database types
Spatial

Knowledge of projections useful

Spatial indexes really speed up some problems

Generally add-on to existing DB
Filesystems

Hierarchical key-value store

Allows multiple writers for appends

Supports very large files
Graph Databases

Allow efficient neighbour queries

Generally not much use for anything else
Round-Robin Database

Deliberately loses old data

Useful for logging or statistics
Final Thoughts
It’s unlikely your data all fits in one paradigm.
Just buying bigger servers goes a long way
If it sounds too good to be true, it probably is.