Set Yourself Free With Cosmos DB

Roland Guijt
CONSULTANT | TRAINER | AUTHOR | MVP

@rolandguijt rolandguijt.com

The Plan

- Cosmos DB overview
- The case for No SQL
- NoSQL Data modelling
- Querying
- Features
- The .NET API

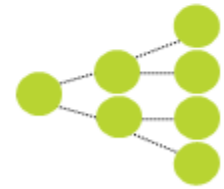
Azure Cosmos DB

... more coming soon

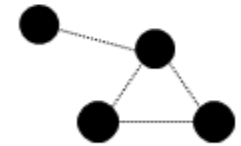
Key-Value

Column-family

Documents



Graph



Global distribution | Elastic scale out | Guaranteed low latency | Five consistency models | Comprehensive SLAs

A globally-distributed, multi-model database service

	.NET		JS		
DocumentDB API	✓	✓	✓	✓	
MongoDB API	✓	✓	✓		
Graph API	✓	✓	✓		✓
Table API	✓				

Why Cosmos DB?

~~"Use a relational database for all applications"~~

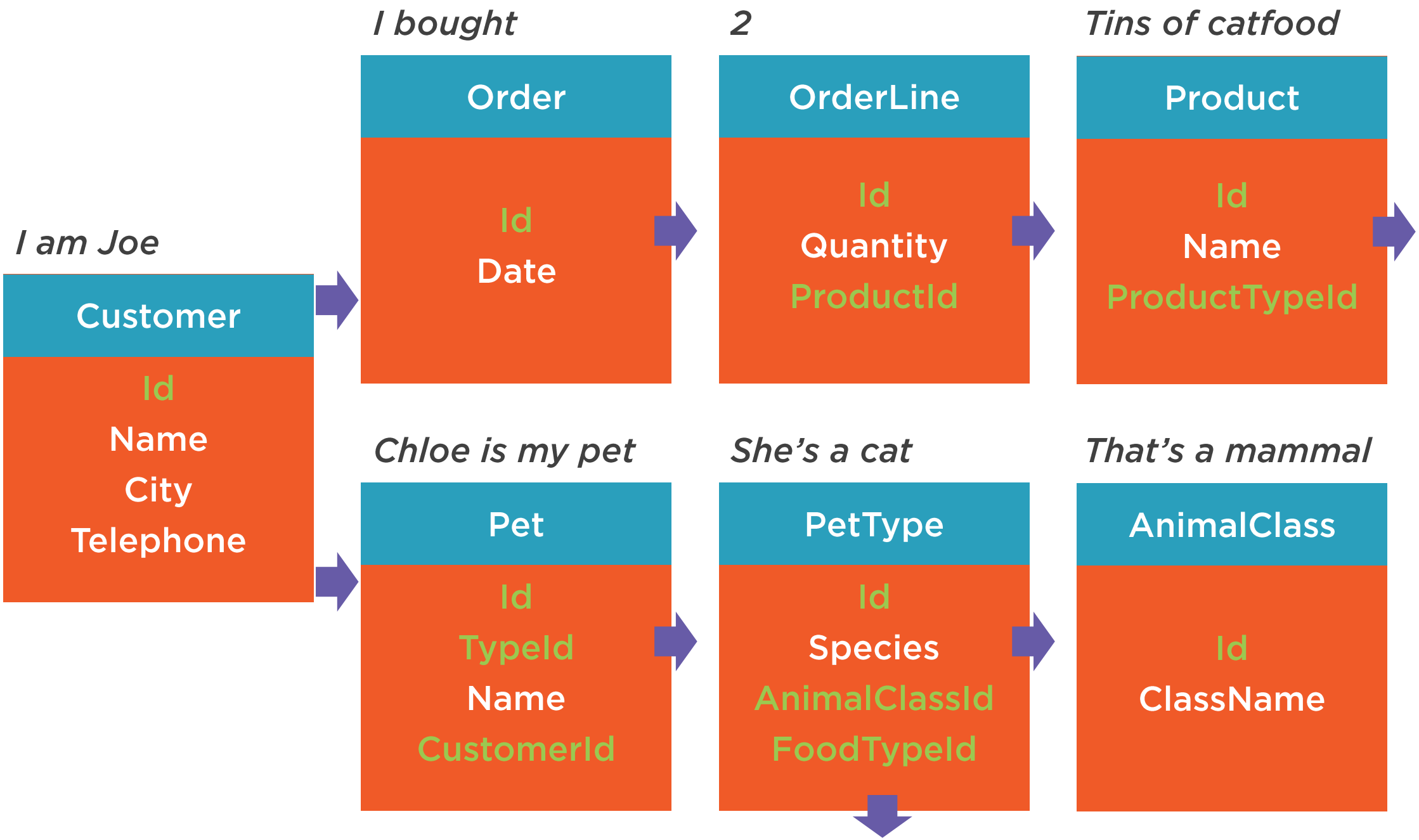
"Consider the type of database for every application you're writing"

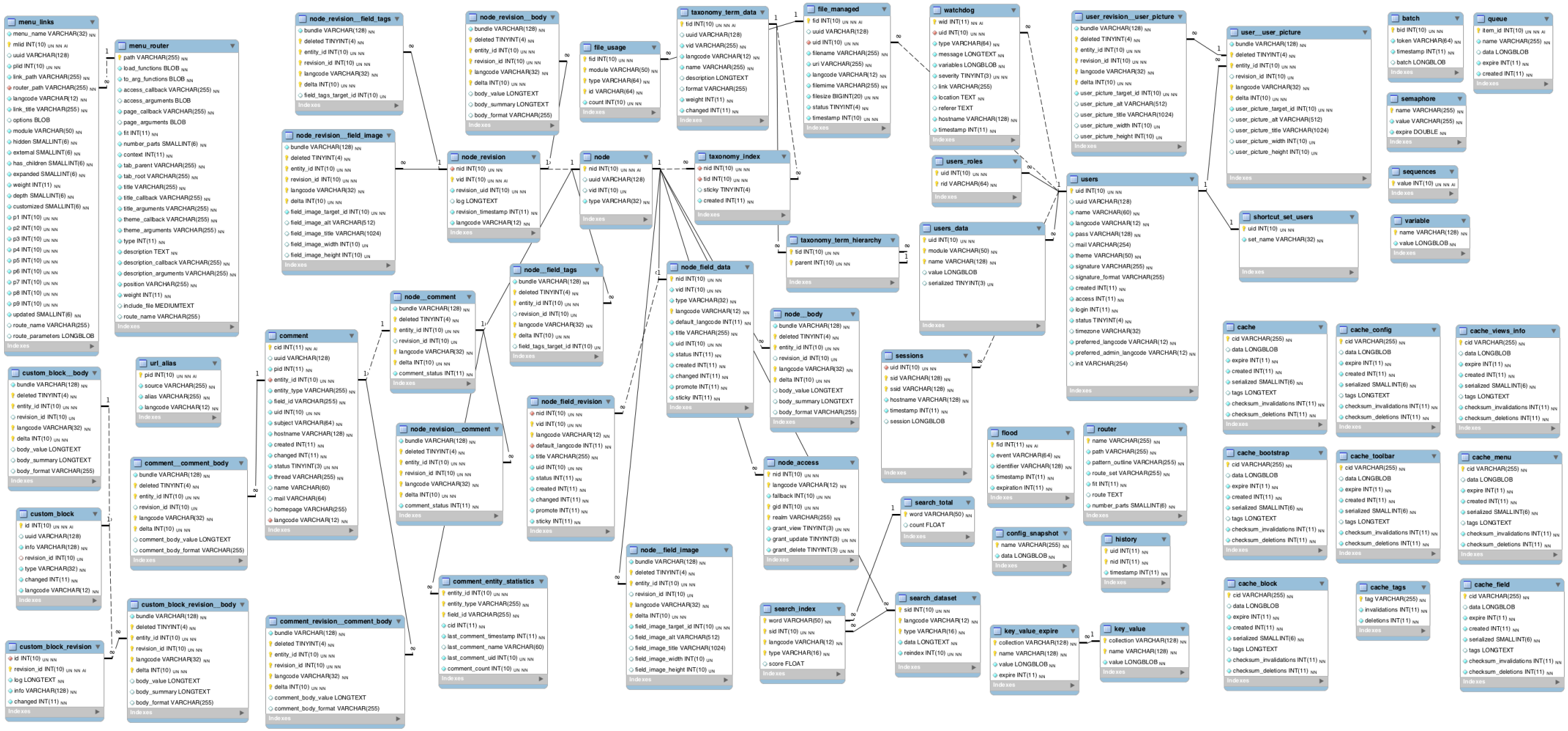
The Case For No-SQL

Normalize!

Do not duplicate!

No deviation from the scheme!


I am Joe
Customer
Id
Name
City
Telephone
I bought
Order
Id
Date
2
OrderLine
Id
Quantity
ProductId
Tins of catfood
Product
Id
Name
ProductTypeId
Chloe is my pet
Pet
Id
TypeId
Name
CustomerId
She's a cat
PetType
Id
Species
AnimalClassId
FoodTypeId
That's a mammal
AnimalClass
Id
ClassName


menu_links
menu_name VARCHAR(32) NN
mlid INT(10) UN NN AI
uuid VARCHAR(128)
plid INT(10) UN NN
link_path VARCHAR(255) NN
router_path VARCHAR(255) NN
langcode VARCHAR(12) NN
link_title VARCHAR(255) NN
options BLOB
module VARCHAR(50) NN
hidden SMALLINT(6) NN
external SMALLINT(6) NN
has_children SMALLINT(6) NN
expanded SMALLINT(6) NN
weight INT(11) NN
depth SMALLINT(6) NN
customized SMALLINT(6) NN
p1 INT(10) UN NN
p2 INT(10) UN NN
p3 INT(10) UN NN
p4 INT(10) UN NN
p5 INT(10) UN NN
p6 INT(10) UN NN
p7 INT(10) UN NN
p8 INT(10) UN NN
p9 INT(10) UN NN
updated SMALLINT(6) NN
route_name VARCHAR(255)
route_parameters LONGBLOB
Indexes
menu_router
path VARCHAR(255) NN
load_functions BLOB NN
to_arg_functions BLOB NN
access_callback VARCHAR(255) NN
access_arguments BLOB
page_callback VARCHAR(255) NN
page_arguments BLOB
fit INT(11) NN
number_parts SMALLINT(6) NN
context INT(11) NN
tab_parent VARCHAR(255) NN
tab_root VARCHAR(255) NN
title VARCHAR(255) NN
title_callback VARCHAR(255) NN
title_arguments VARCHAR(255) NN
theme_callback VARCHAR(255) NN
theme_arguments VARCHAR(255) NN
type INT(11) NN
description TEXT NN
description_callback VARCHAR(255) NN
description_arguments VARCHAR(255) NN
position VARCHAR(255) NN
weight INT(11) NN
include_file MEDIUMTEXT
route_name VARCHAR(255)
Indexes
node_revision__field_tags
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN NN
langcode VARCHAR(32) NN
delta INT(10) UN NN
field_tags_target_id INT(10) UN
Indexes
node_revision__body
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN NN
langcode VARCHAR(32) NN
delta INT(10) UN NN
body_value LONGTEXT
body_summary LONGTEXT
body_format VARCHAR(255)
Indexes
file_usage
fid INT(10) UN NN
module VARCHAR(50) NN
type VARCHAR(64) NN
id VARCHAR(64) NN
count INT(10) UN NN
Indexes
taxonomy_term_data
tid INT(10) UN NN AI
uuid VARCHAR(128)
vid VARCHAR(255) NN
langcode VARCHAR(12) NN
name VARCHAR(255) NN
description LONGTEXT
format VARCHAR(255)
weight INT(11) NN
changed INT(11) NN
Indexes
file_managed
fid INT(10) UN NN AI
uuid VARCHAR(128)
uid INT(10) UN NN
filename VARCHAR(255) NN
uri VARCHAR(255) NN
langcode VARCHAR(12) NN
filemime VARCHAR(255) NN
filesize BIGINT(20) UN NN
status TINYINT(4) NN
timestamp INT(10) UN NN
Indexes
watchdog
wid INT(11) NN AI
uid INT(10) UN NN
type VARCHAR(64) NN
message LONGTEXT NN
variables LONGBLOB NN
severity TINYINT(3) UN NN
link VARCHAR(255)
location TEXT NN
referer TEXT
hostname VARCHAR(128) NN
timestamp INT(11) NN
Indexes
user_revision__user_picture
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN NN
langcode VARCHAR(32) NN
delta INT(10) UN NN
user_picture_target_id INT(10) UN NN
user_picture_alt VARCHAR(512)
user_picture_title VARCHAR(1024)
user_picture_width INT(10) UN
user_picture_height INT(10) UN
Indexes
user__user_picture
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN
langcode VARCHAR(32) NN
delta INT(10) UN NN
user_picture_target_id INT(10) UN NN
user_picture_alt VARCHAR(512)
user_picture_title VARCHAR(1024)
user_picture_width INT(10) UN
user_picture_height INT(10) UN
Indexes
batch
bid INT(10) UN NN
token VARCHAR(64) NN
timestamp INT(11) NN
batch LONGBLOB
Indexes
queue
item_id INT(10) UN NN AI
name VARCHAR(255) NN
data LONGBLOB
expire INT(11) NN
created INT(11) NN
Indexes
semaphore
name VARCHAR(255) NN
value VARCHAR(255) NN
expire DOUBLE NN
Indexes
node_revision__field_image
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN NN
langcode VARCHAR(32) NN
delta INT(10) UN NN
field_image_target_id INT(10) UN NN
field_image_alt VARCHAR(512)
field_image_title VARCHAR(1024)
field_image_width INT(10) UN
field_image_height INT(10) UN
Indexes
node_revision
nid INT(10) UN NN
vid INT(10) UN NN AI
revision_uid INT(10) UN NN
log LONGTEXT
revision_timestamp INT(11) NN
langcode VARCHAR(12) NN
Indexes
node
nid INT(10) UN NN AI
uuid VARCHAR(128)
vid INT(10) UN
type VARCHAR(32) NN
Indexes
taxonomy_index
nid INT(10) UN NN
tid INT(10) UN NN
sticky TINYINT(4)
created INT(11) NN
Indexes
users_roles
uid INT(10) UN NN
rid VARCHAR(64) NN
Indexes
users
uid INT(10) UN NN
uuid VARCHAR(128)
name VARCHAR(60) NN
langcode VARCHAR(12) NN
pass VARCHAR(128) NN
mail VARCHAR(254)
theme VARCHAR(50) NN
signature VARCHAR(255) NN
signature_format VARCHAR(255)
created INT(11) NN
access INT(11) NN
login INT(11) NN
status TINYINT(4) NN
timezone VARCHAR(32)
preferred_langcode VARCHAR(12) NN
preferred_admin_langcode VARCHAR(12) NN
init VARCHAR(254)
Indexes
sequences
value INT(10) UN NN AI
Indexes
shortcut_set_users
uid INT(10) UN NN
set_name VARCHAR(32) NN
Indexes
variable
name VARCHAR(128) NN
value LONGBLOB NN
Indexes
users_data
uid INT(10) UN NN
module VARCHAR(50) NN
name VARCHAR(128) NN
value LONGBLOB
serialized TINYINT(3) UN
Indexes
taxonomy_term_hierarchy
tid INT(10) UN NN
parent INT(10) UN NN
Indexes
node__field_tags
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN
langcode VARCHAR(32) NN
delta INT(10) UN NN
field_tags_target_id INT(10) UN
Indexes
node__field_data
nid INT(10) UN NN
vid INT(10) UN NN
type VARCHAR(32) NN
langcode VARCHAR(12) NN
default_langcode INT(11) NN
title VARCHAR(255) NN
uid INT(10) UN NN
status INT(11) NN
created INT(11) NN
changed INT(11) NN
promote INT(11) NN
sticky INT(11) NN
Indexes
node__comment
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN
langcode VARCHAR(32) NN
delta INT(10) UN NN
comment_status INT(11) NN
Indexes
comment
cid INT(11) NN AI
uuid VARCHAR(128)
pid INT(11) NN
entity_id INT(10) UN NN
entity_type VARCHAR(255) NN
field_id VARCHAR(255) NN
uid INT(10) UN NN
subject VARCHAR(64) NN
hostname VARCHAR(128) NN
created INT(11) NN
changed INT(11) NN
status TINYINT(3) UN NN
thread VARCHAR(255) NN
name VARCHAR(60)
mail VARCHAR(64)
homepage VARCHAR(255)
langcode VARCHAR(12) NN
Indexes
node__body
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN
langcode VARCHAR(32) NN
delta INT(10) UN NN
body_value LONGTEXT
body_summary LONGTEXT
body_format VARCHAR(255)
Indexes
sessions
uid INT(10) UN NN
sid VARCHAR(128) NN
ssid VARCHAR(128) NN
hostname VARCHAR(128) NN
timestamp INT(11) NN
session LONGBLOB
Indexes
cache
cid VARCHAR(255) NN
data LONGBLOB
expire INT(11) NN
created INT(11) NN
serialized SMALLINT(6) NN
tags LONGTEXT
checksum_invalidations INT(11) NN
checksum_deletions INT(11) NN
Indexes
cache_config
cid VARCHAR(255) NN
data LONGBLOB
expire INT(11) NN
created INT(11) NN
serialized SMALLINT(6) NN
tags LONGTEXT
checksum_invalidations INT(11) NN
checksum_deletions INT(11) NN
Indexes
cache_views_info
cid VARCHAR(255) NN
data LONGBLOB
expire INT(11) NN
created INT(11) NN
serialized SMALLINT(6) NN
tags LONGTEXT
checksum_invalidations INT(11) NN
checksum_deletions INT(11) NN
Indexes
custom_block__body
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN
langcode VARCHAR(32) NN
delta INT(10) UN NN
body_value LONGTEXT
body_summary LONGTEXT
body_format VARCHAR(255)
Indexes
url_alias
pid INT(10) UN NN AI
source VARCHAR(255) NN
alias VARCHAR(255) NN
langcode VARCHAR(12) NN
Indexes
node_revision__comment
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN NN
langcode VARCHAR(32) NN
delta INT(10) UN NN
comment_status INT(11) NN
Indexes
node_field_revision
nid INT(10) UN NN
vid INT(10) UN NN
langcode VARCHAR(12) NN
default_langcode INT(11) NN
title VARCHAR(255) NN
uid INT(10) UN NN
status INT(11) NN
created INT(11) NN
changed INT(11) NN
promote INT(11) NN
sticky INT(11) NN
Indexes
node_access
nid INT(10) UN NN
langcode VARCHAR(12) NN
fallback INT(10) UN NN
gid INT(10) UN NN
realm VARCHAR(255) NN
grant_view TINYINT(3) UN NN
grant_update TINYINT(3) UN NN
grant_delete TINYINT(3) UN NN
Indexes
flood
fid INT(11) NN AI
event VARCHAR(64) NN
identifier VARCHAR(128) NN
timestamp INT(11) NN
expiration INT(11) NN
Indexes
router
name VARCHAR(255) NN
path VARCHAR(255) NN
pattern_outline VARCHAR(255) NN
route_set VARCHAR(255) NN
fit INT(11) NN
route TEXT
number_parts SMALLINT(6) NN
Indexes
cache_bootstrap
cid VARCHAR(255) NN
data LONGBLOB
expire INT(11) NN
created INT(11) NN
serialized SMALLINT(6) NN
tags LONGTEXT
checksum_invalidations INT(11) NN
checksum_deletions INT(11) NN
Indexes
cache_toolbar
cid VARCHAR(255) NN
data LONGBLOB
expire INT(11) NN
created INT(11) NN
serialized SMALLINT(6) NN
tags LONGTEXT
checksum_invalidations INT(11) NN
checksum_deletions INT(11) NN
Indexes
cache_menu
cid VARCHAR(255) NN
data LONGBLOB
expire INT(11) NN
created INT(11) NN
serialized SMALLINT(6) NN
tags LONGTEXT
checksum_invalidations INT(11) NN
checksum_deletions INT(11) NN
Indexes
comment__comment_body
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN
langcode VARCHAR(32) NN
delta INT(10) UN NN
comment_body_value LONGTEXT
comment_body_format VARCHAR(255)
Indexes
search_total
word VARCHAR(50) NN
count FLOAT
Indexes
config_snapshot
name VARCHAR(255) NN
data LONGBLOB NN
Indexes
history
uid INT(11) NN
nid INT(11) NN
timestamp INT(11) NN
Indexes
custom_block
id INT(10) UN NN AI
uuid VARCHAR(128)
info VARCHAR(128) NN
revision_id INT(10) UN
type VARCHAR(32) NN
changed INT(11) NN
langcode VARCHAR(12) NN
Indexes
comment_entity_statistics
entity_id INT(10) UN NN
entity_type VARCHAR(255) NN
field_id VARCHAR(255) NN
cid INT(11) NN
last_comment_timestamp INT(11) NN
last_comment_name VARCHAR(60)
last_comment_uid INT(10) UN NN
comment_count INT(10) UN NN
Indexes
node__field_image
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN
langcode VARCHAR(32) NN
delta INT(10) UN NN
field_image_target_id INT(10) UN NN
field_image_alt VARCHAR(512)
field_image_title VARCHAR(1024)
field_image_width INT(10) UN
field_image_height INT(10) UN
Indexes
search_index
word VARCHAR(50) NN
sid INT(10) UN NN
langcode VARCHAR(12) NN
type VARCHAR(16) NN
score FLOAT
Indexes
search_dataset
sid INT(10) UN NN
langcode VARCHAR(12) NN
type VARCHAR(16) NN
data LONGTEXT NN
reindex INT(10) UN NN
Indexes
cache_block
cid VARCHAR(255) NN
data LONGBLOB
expire INT(11) NN
created INT(11) NN
serialized SMALLINT(6) NN
tags LONGTEXT
checksum_invalidations INT(11) NN
checksum_deletions INT(11) NN
Indexes
cache_tags
tag VARCHAR(255) NN
invalidations INT(11) NN
deletions INT(11) NN
Indexes
cache_field
cid VARCHAR(255) NN
data LONGBLOB
expire INT(11) NN
created INT(11) NN
serialized SMALLINT(6) NN
tags LONGTEXT
checksum_invalidations INT(11) NN
checksum_deletions INT(11) NN
Indexes
custom_block_revision__body
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN NN
langcode VARCHAR(32) NN
delta INT(10) UN NN
body_value LONGTEXT
body_summary LONGTEXT
body_format VARCHAR(255)
Indexes
key_value_expire
collection VARCHAR(128) NN
name VARCHAR(128) NN
value LONGBLOB NN
expire INT(11) NN
Indexes
key_value
collection VARCHAR(128) NN
name VARCHAR(128) NN
value LONGBLOB NN
Indexes
custom_block_revision
id INT(10) UN NN
revision_id INT(10) UN NN AI
log LONGTEXT NN
info VARCHAR(128) NN
changed INT(11) NN
Indexes
comment_revision__comment_body
bundle VARCHAR(128) NN
deleted TINYINT(4) NN
entity_id INT(10) UN NN
revision_id INT(10) UN NN
langcode VARCHAR(32) NN
delta INT(10) UN NN
comment_body_value LONGTEXT
comment_body_format VARCHAR(255)
Indexes

```
using (var db = new AdventureWorksContext())
{
    return db.Customer
        .Include(c => c.CustomerAddress)
            .ThenInclude(a => a.Address)
        .Include(c => c.SalesOrderHeader)
            .ThenInclude(h => h.SalesOrderDetail)
                .ThenInclude(d => d.Product)
                    .ThenInclude(p => p.ProductCategory)
        .Include(c => c.SalesOrderHeader)
            .ThenInclude(h => h.SalesOrderDetail)
                .ThenInclude(d => d.Product)
                    .ThenInclude(p => p.ProductModel)

        .Include(c => c.SalesOrderHeader)
            .ThenInclude(h => h.BillToAddress).ToList();
}
```

Remarks

Performance

Complexity of queries

Applications tend to display data from multiple tables in one go

The rule against duplication seems old fashioned

New features, refactoring anyone?

Item	CPU	Memory	Storage
Lenovo Thinkpad X1 Carbon	Core i7 3.3ghz	8 GB	256 GB SSD

Item	Author	Pages	Language
Harry Potter and the Sorcerer's Stone	J.K. Rowling	309	English
Game of Thrones: A Song of Ice and Fire	George R.R. Martin	864	English

Amazon has thousands of product types
That's a lot of variety

Class

Attribute

Object

ObjectAttribute

Use the right tool
for the right job

The Document DB API

Documents with hierarchy

Denormalized, embedded data

Data duplication ok

No schema, NoSQL

```
{
  "ItemType": "Book",
  "Title": "Harry Potter and the Sorcerer's Stone",
  "Author": "J.K. Rowling",
  "Pages": "864",
  "Languages": [
    "English", "Spanish", "Portuguese",
    "Russian", "French"
  ]
}
```

SQL vs Cosmos DocumentDB: Let’s race!

Race Data Structure: 847 Items

```
[
  {
    "id": "29485",
    ..
    "CustomerAddress": [
      {
        ..
      }
    ],
    "SalesOrderHeader": [
      {
        ..
        "SalesOrderDetail": [
          {
            ..
            "Product": {
                        ..
```

@rolandguijt

Was This a Fair Fight?

Yes and no

The data is the same

Both are databases in Azure

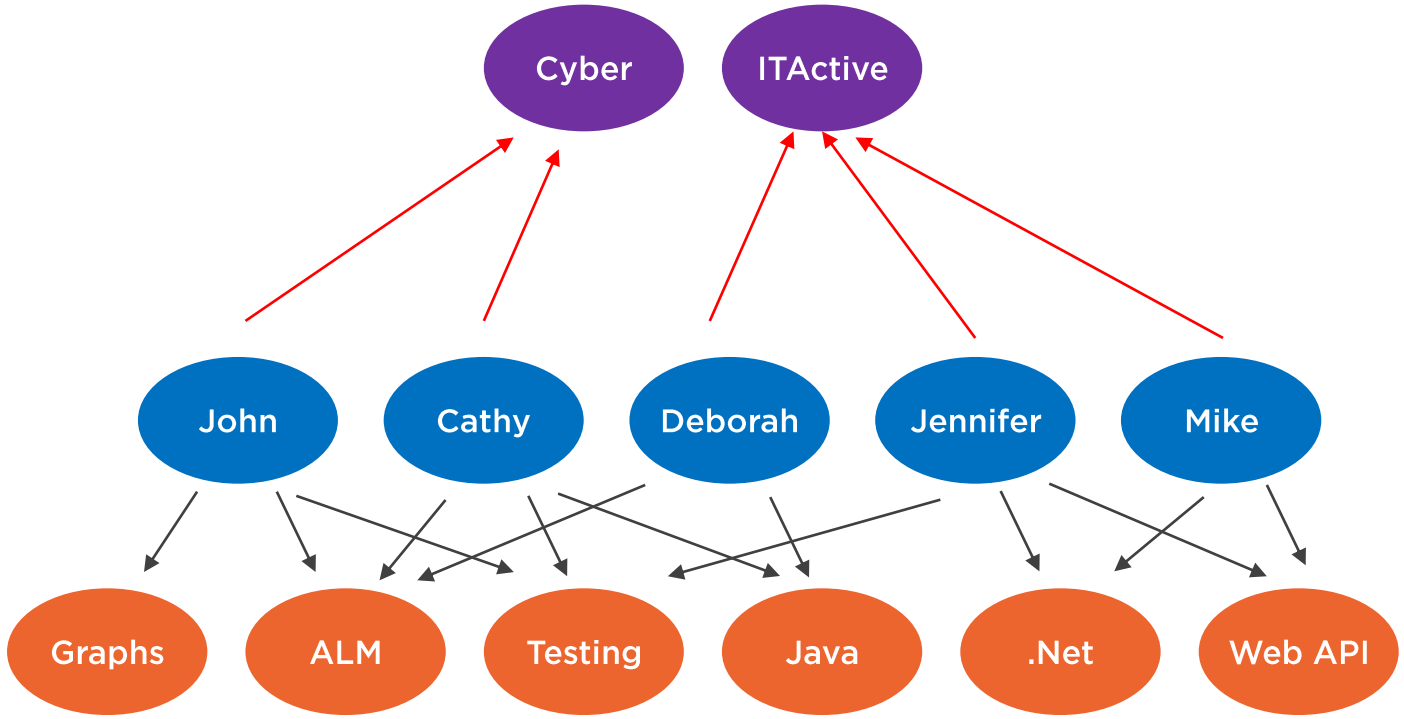
But they are different beasts

Generally: A document db is a lot faster when a lot of nesting is involved

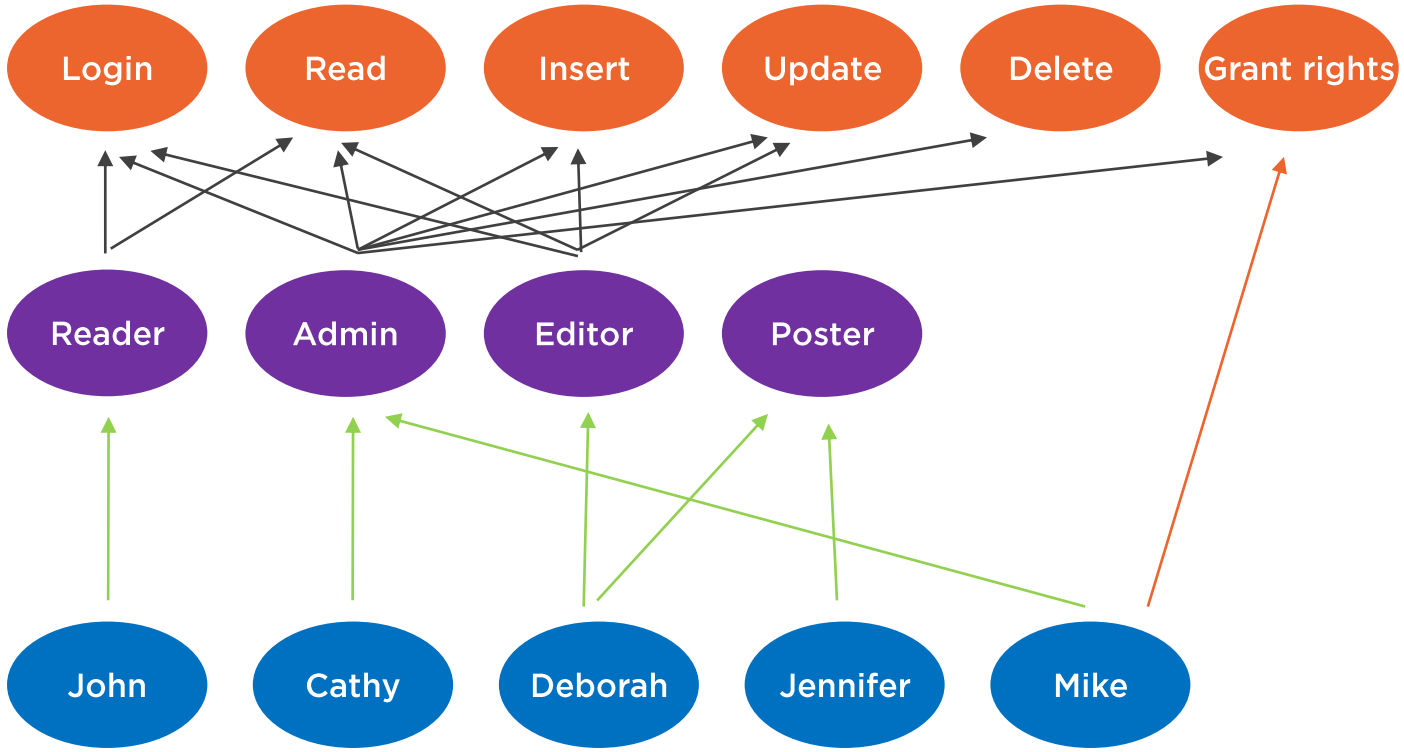
The more SQL joins, the faster a document db will be in comparison

The Graph API

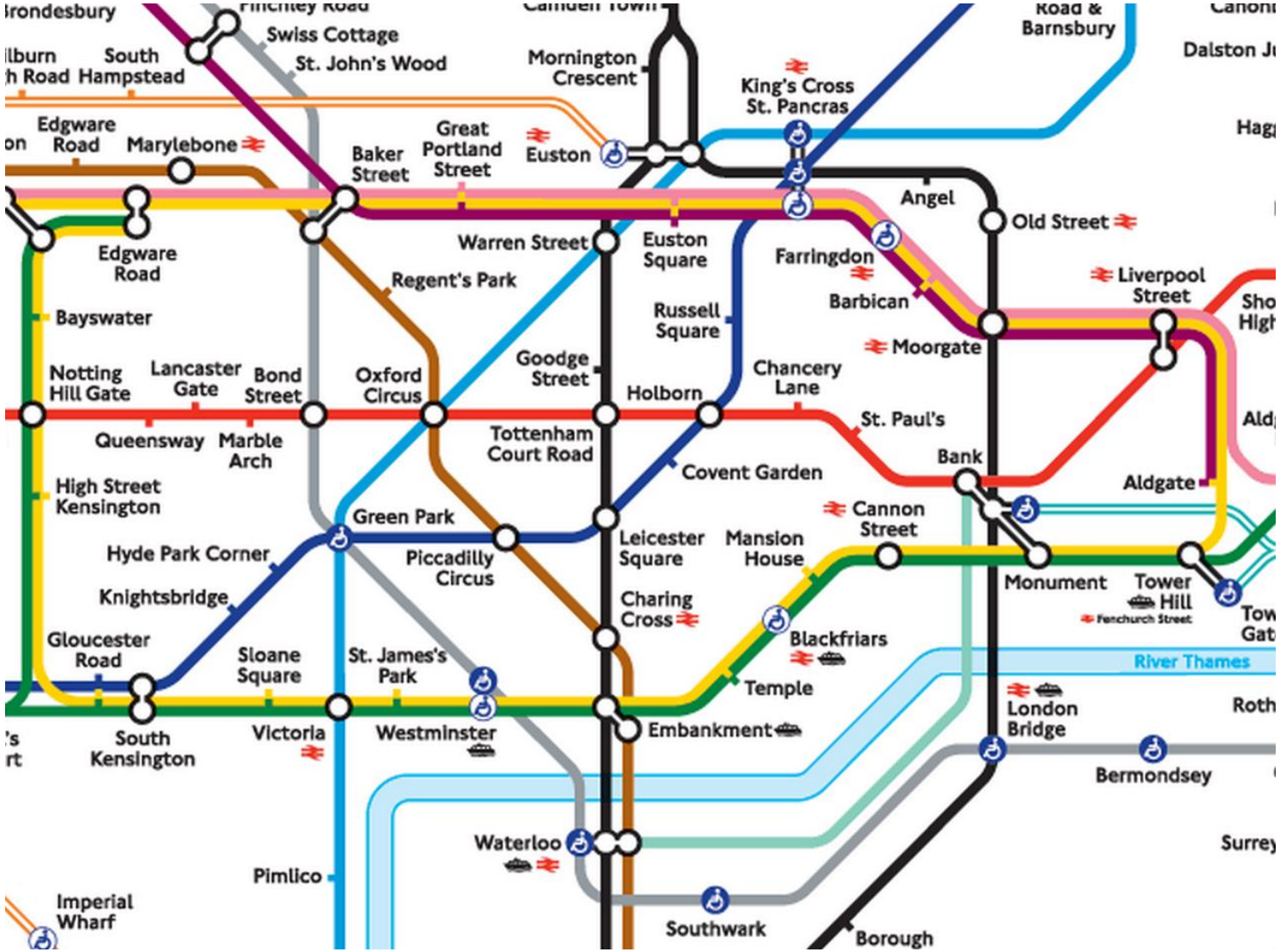
A Social Graph



Security


Login
Read
Insert
Update
Delete
Grant rights
Reader
Admin
Editor
Poster
John
Cathy
Deborah
Jennifer
Mike

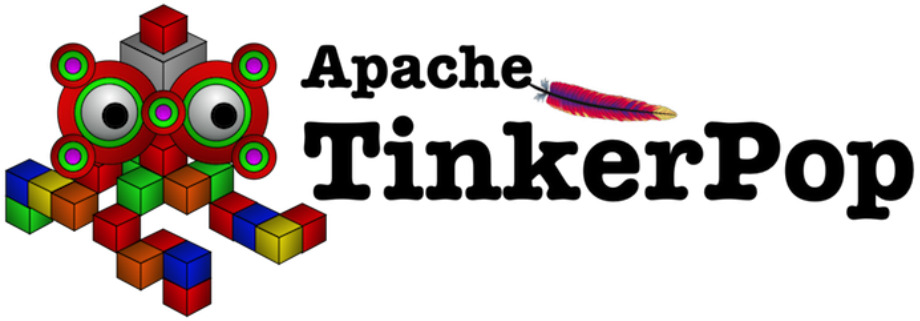
Logistics

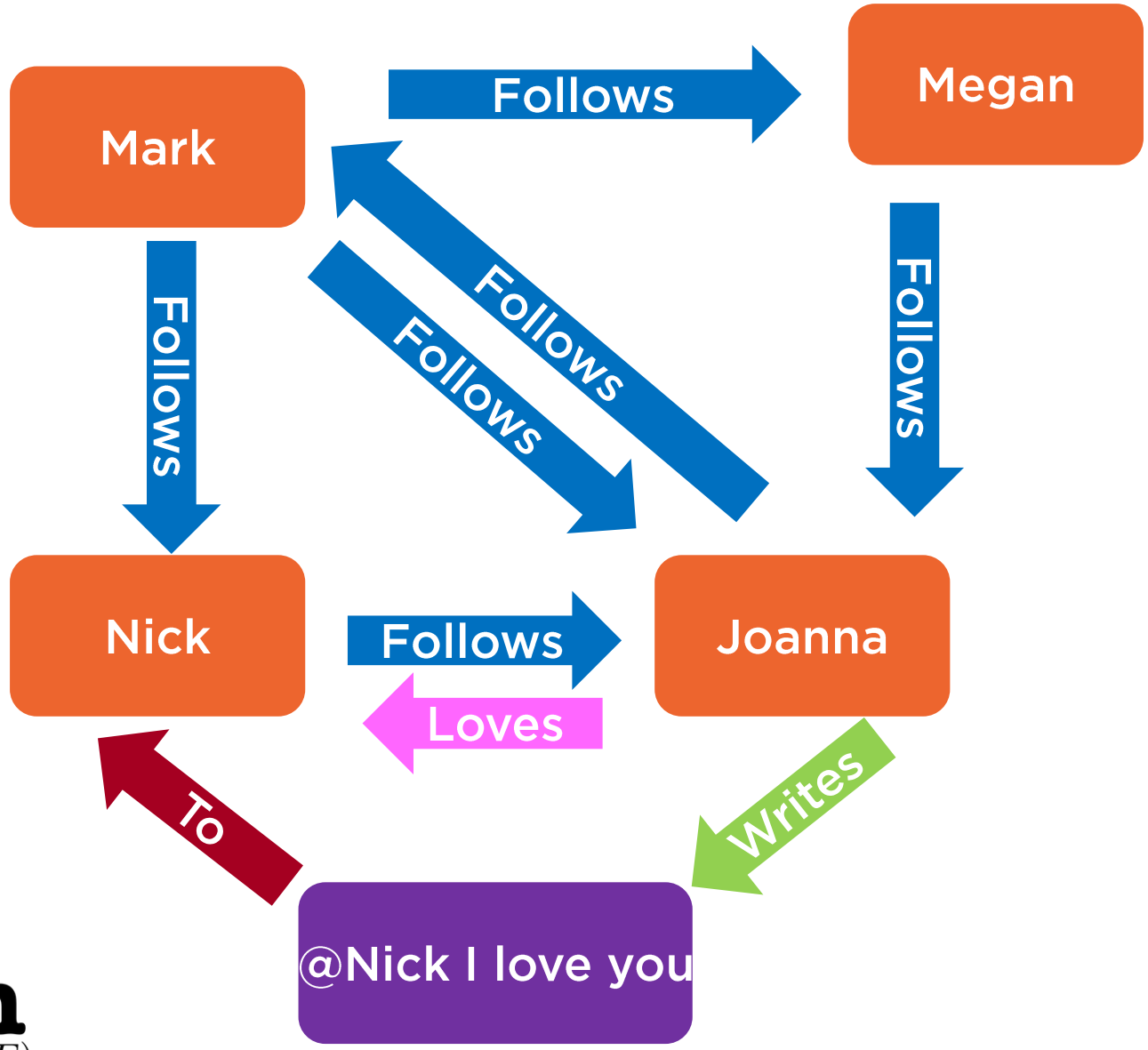

Swiss Cottage
St. John's Wood
South Hampstead
Mornington Crescent
King's Cross St. Pancras
Edgware Road
Marylebone
Great Portland Street
Euston
Baker Street
Angel
Old Street
Edgware Road
Warren Street
Euston Square
Farringdon
Regent's Park
Liverpool Street
Barbican
Bayswater
Russell Square
Moorgate
Goodge Street
Chancery Lane
Notting Hill Gate
Lancaster Gate
Bond Street
Oxford Circus
Holborn
St. Paul's
Queensway
Marble Arch
Tottenham Court Road
Bank
Covent Garden
High Street Kensington
Aldgate
Green Park
Cannon Street
Hyde Park Corner
Piccadilly Circus
Leicester Square
Mansion House
Monument
Tower Hill
Knightsbridge
Charing Cross
Fenchurch Street
Gloucester Road
Sloane Square
St. James's Park
Blackfriars
River Thames
Temple
London Bridge
South Kensington
Victoria
Westminster
Embankment
Bermondsey
Waterloo
Pimlico
Imperial Wharf
Southwark
Borough

Vertice = Node

Edge = Relation

Vertice = Node




Mark
Follows
Megan
Follows
Follows
Follows
Follows
Nick
Follows
Joanna
Loves
To
Writes
@Nick I love you

Gremlin

$G = (V, E)$

Graphs

- Easily extendable and expandable
- Traversal is fast!
- Friendly to the human brain
- Whiteboard compatible

No-SQL data modelling

Always Denormalize, Except When..

The data is boundless

The data changes frequently

Post document:

```
{
    "id": "1",
    "name": "What's new in the coolest Cloud",
    "summary": "A blog post by someone real famous",
    "recentComments": [
        {"id": 1, "author": "anon", "comment": "something useful, I'm sure"},
        {"id": 2, "author": "bob", "comment": "wisdom from the interwebs"},
        {"id": 3, "author": "jane", "comment": "....."}
    ]
}
```

Comment documents:

```
{
    "postId": "1"
    "comments": [
        {"id": 4, "author": "anon", "comment": "more goodness"},
        {"id": 5, "author": "bob", "comment": "tails from the field"},
        ...
        {"id": 99, "author": "angry", "comment": "blah angry blah angry"}
    ]
},
..
```

Person document:

```
{
    "id": "1",
    "firstName": "Thomas",
    "lastName": "Andersen",
    "holdings": [
        { "numberHeld":  100, "stockId": 1},
        { "numberHeld":  50, "stockId": 2}
    ]
}
```

Stock documents:

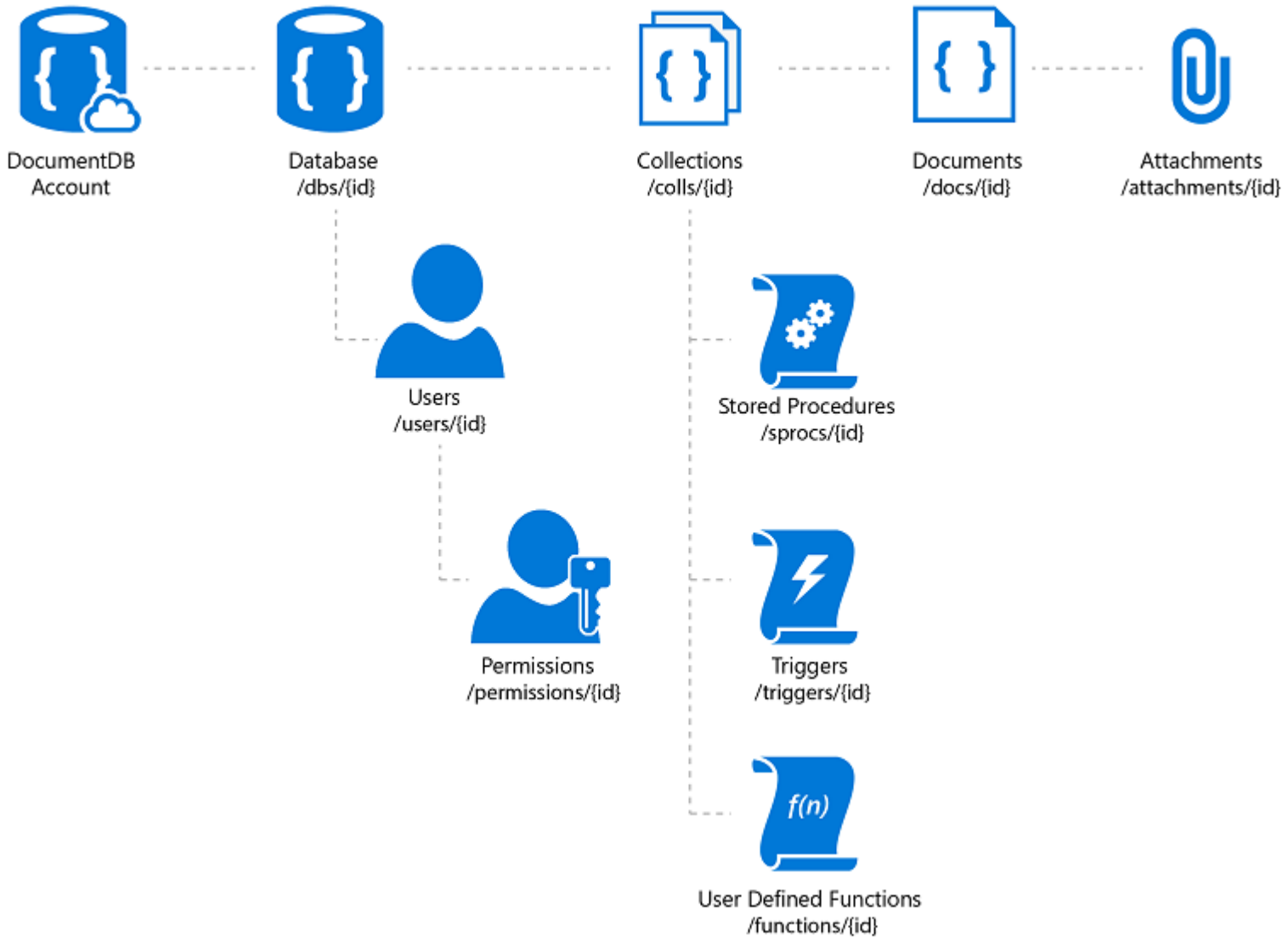
```
{
    "id": "1",
    "symbol": "zaza",
    "open": 1,
    "high": 2,
    "low": 0.5,
    "vol": 11970000,
    "mkt-cap": 42000000,
    "pe": 5.89
},
{
    "id": "2",
    "symbol": "xcxc",
    "open": 89,
    "high": 93.24,
    "low": 88.87,
    "vol": 2970200,
    "mkt-cap": 1005000,
    "pe": 75.82
}
```

Features For All APIs

Azure CosmosDb

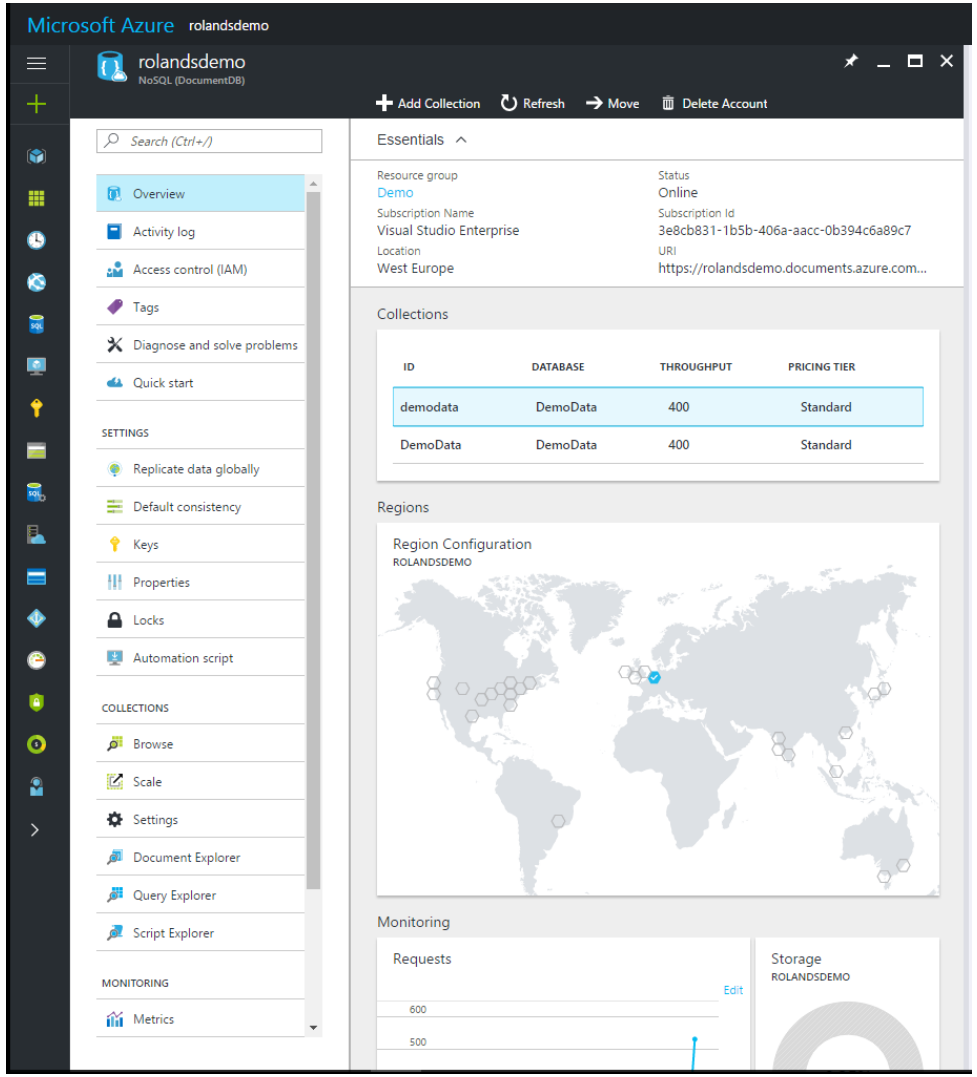
- Cloud based no-sql platform
- Elastically scalable throughput and storage
- Multi-region data replication
- Fully managed
- Backups
- Automatic indexing
- REST based with .Net, Java, Node, Python client libraries available
- SLA

Core Concepts



@rolandguijt

Portal Experience



@rolandguijt

Querying

Talking to CosmosDB From .NET

Indexes

Default Index Policy

```
{
  "indexingMode": "consistent",
  "automatic": true,
  "includedPaths": [
    {
      "path": "/*",
      "indexes": [
        {
          "kind": "Range",
          "dataType": "Number",
          "precision": -1
        },
        {
          "kind": "Hash",
          "dataType": "String",
          "precision": 3
        }]
    }]
}
```

Supports between and orderby

Supports equality

@rolandguijt

You can’t filter on a property when it doesn’t have at least a hash index

Indexing Options

- Use default policy
- Can override per document (RequestOptions)
- Set policy to manual. Switch on per document.
- For bulk import or write optimization
- Exclude document paths from indexing
- Only use range indexes where needed

Performance and Scaling

Data throughput is measured in Request Units: RUs.
1 RU represents the resources needed to read a 1KB document

Vertical Scaling

More RUs/second
More storage

Max. 10.000 RUs/second
Max. 10 GB

Horizontal Scaling

No limit on RUs/second and storage

Department = sales

Department = HRM

@rolandguijt

Global Scaling

Click on a location to add or remove regions from your DocumentDB account.

* Each region is billable based on the throughput and storage for the account. Learn more

WRITE REGION

West Europe

READ REGIONS

Canada East

West India

Australia East

Consistency Levels

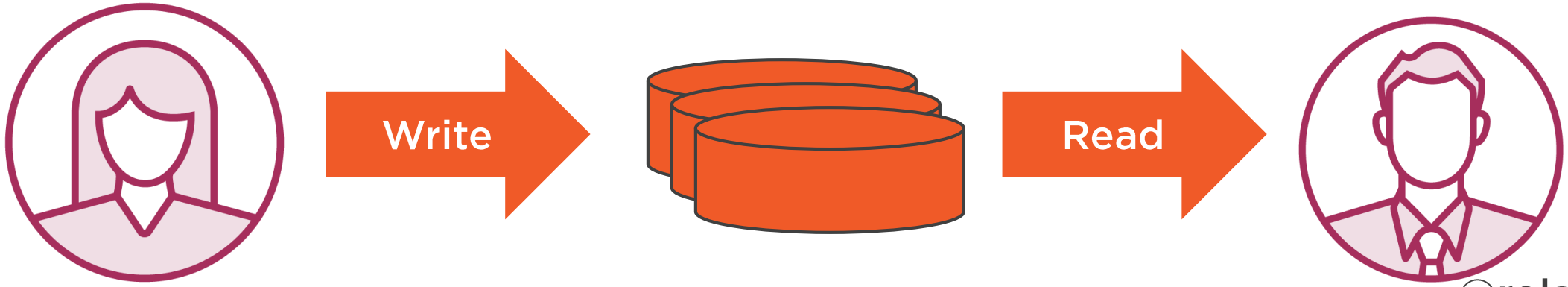
LEFT TO RIGHT → *Relaxed consistency => better performance and availability*

Always the latest version of a document
Slowest

Configurable slack (# of version or time) is allowed
Reads are faster

For "user" scenarios
Reads for data user owns are always consistent

We don't care about consistency just about performance



@rolandguijt

Pricing

$0,25 per GB/month

$0,06/month for 1 RU per second
400 ($24) minimum
(throughput)

RU/seconds are reserved

Getting Started

- Just create a database!
- Query playgrounds
- Data migration tool
- CosmosDB Emulator
- CosmosDB Studio