



Databases & ORMs

Database Management Sys. (DBMS)

- How to define records (Data Definition)
- Creating/Updating/Deleting Records
- Querying/Retrieval of Records
- Administering the database system

Progression of Databases

- Navigational (< 1970s)
- Relational (> 1970s)
- NoSQL (> 2000s)



Postgres History

- 1970s (UC Berkley): INteractive Graphics REtrieval System
- 1980s: Postgres (“Ingres++”)
- 1995: POSTQUEL and Postgres95
- 1996: open source community adopts it
- Since: stability, testing, documentation, new features
- PostgreSQL

ACID

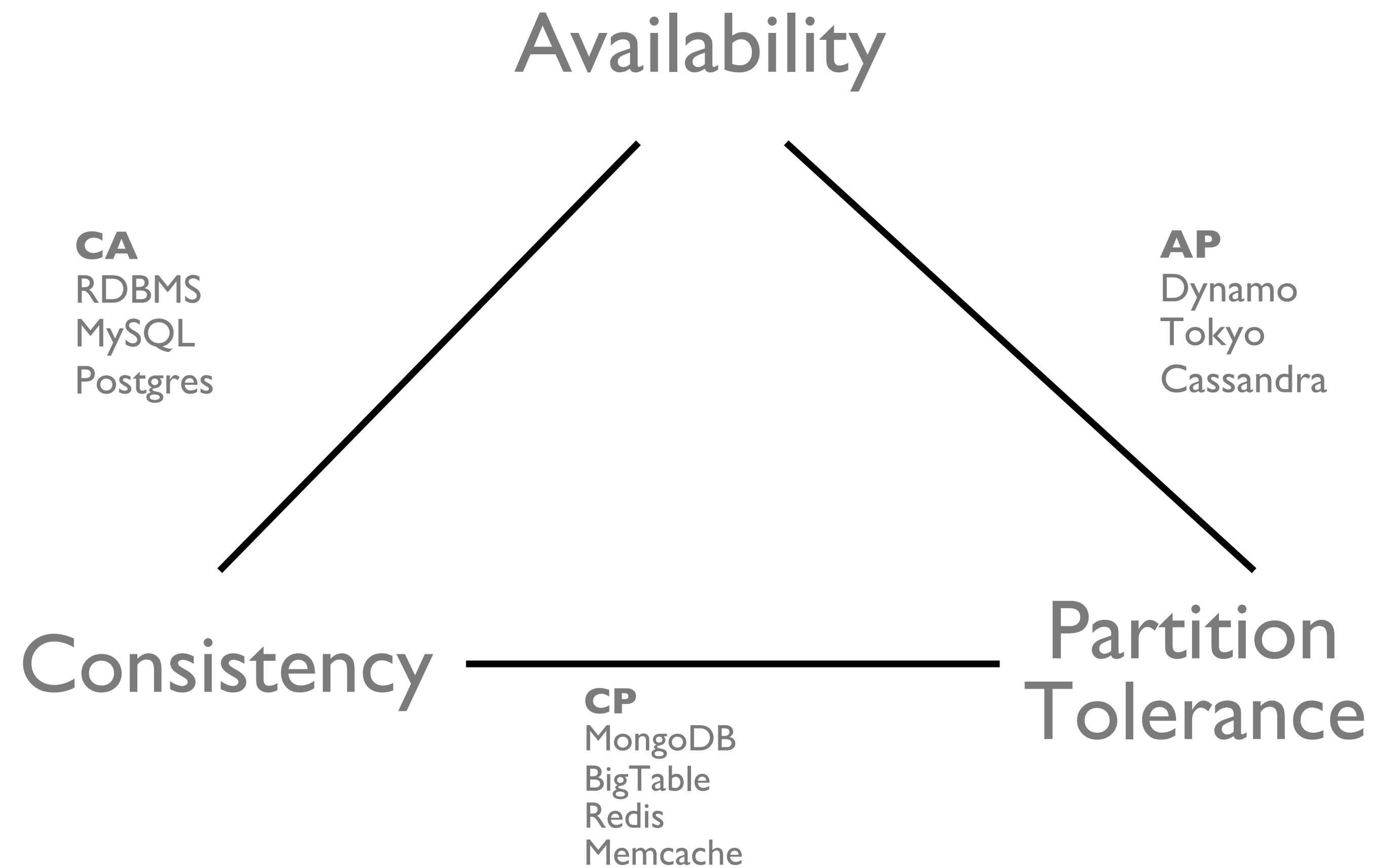
- **All about transactions**
- **Atomicity**
- **Consistency**
- **Isolation**
- **Durability**

CAP Theorem

- **Consistency:** all clients see the same data
- **Availability:** each client can always read or write
- **Partition Tolerance:** system works even if network splits
- **"Pick 2"**
 - (Reality is more complex...)

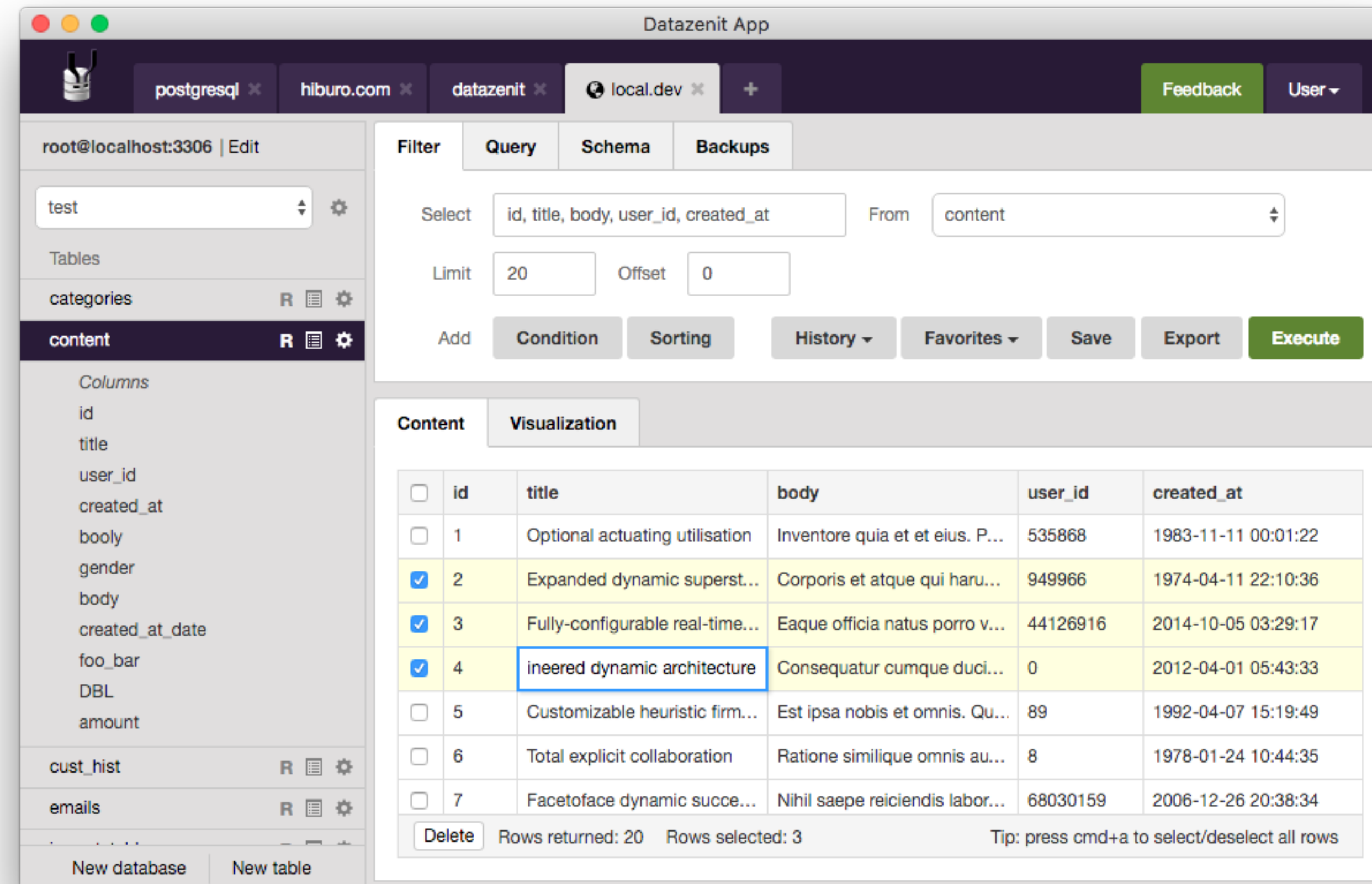


CAP, metaphorically speaking



Tooling

Datazenit



Tooling

Postico

Reporting reporting forex Connected. PostgreSQL 9.4.5

SQL Query	currency	base	rate	date	source_id
currencies	AED	USD	3.67291	2014-07-25	1
daily_proceeds	AFN	USD	56.485726	2014-07-25	1
forex	ALL	USD	103.5838	2014-07-25	1
forex_sources	AMD	USD	410.086	2014-07-25	1
fs_daily_proceeds	ANG	USD	1.787	2014-07-25	1
fs_daily_sales	AOA	USD	96.952626	2014-07-25	1
fs_orders	ARS	USD	8.169642	2014-07-25	1
fs_proceeds_weekly	AUD	USD	1.062844	2014-07-25	1
fs_weekly_sales	AWG	USD	1.79	2014-07-25	1
itc_daily_proceeds	AZN	USD	0.784067	2014-07-25	1
itc_daily_sales	BAM	USD	1.453024	2014-07-25	1
itc_proceeds_weekly	BBD	USD	2	2014-07-25	1
itc_reports	BDT	USD	77.61971	2014-07-25	1
itc_reports_daily	BGN	USD	1.452543	2014-07-25	1
itc_reports_monthly					
itc_reports_weekly					

Search Content Structure + Filter Page 1 of 342

currency: MULTIPLE
base: USD
rate: MULTIPLE
date: 2014-07-25
source_id: forex_source:
id: 1
name: Open Exchange R...

Sequelize

- **Sequelize is an Object-Relational Mapper (ORM)**
- **Access SQL databases from Node.js**
- **Sequelize features:**
 - Schema modeling/validation
 - Data casting (convert SQL types to JS types)
 - Query building
 - Hooks (code that runs pre/post save/delete/update)
 - Class and instance methods of models
 - Getters, setters, and virtual fields



Tables

Models

+

=

+

Rows

Instances

Sequelize Basics

- Make a **Model** (interactive blueprint object)
- Extend the **Model** with **Hooks**, **Class & Instance Methods**, **Virtuals**, etc.
- Connect/sync the completed **Model** to an *actual* table in an *actual* SQL database
- Use the **Model** (Table) to create/find **Instances** (Table)
- Use the **Instances** to save/update/delete



Create a Model

```
var Sequelize = require('sequelize');  
var db = new Sequelize('postgres://localhost:5432/twitter');  
var User = db.define('User', {  
  name: Sequelize.String,  
  pictureUrl: Sequelize.String  
});
```



Sync Model to Table

```
User.sync().then(...);
```



Model & Instance Usage

```
var aPerson = User.build({  
  name: "David",  
  pictureUrl: "http://fillmurray.com/10/10"  
});
```

```
aPerson.save()  
  .then(...);
```

```
User.findAll()  
  .then(...);
```

Sequelize

- Lives inside Node.js process
- Knows how to communicate to various SQL DBMSs, including PostgreSQL

Wikistack

- Build a Wikipedia clone
- Walk you through installing and using sequelize
- Application of everything we've learned so far