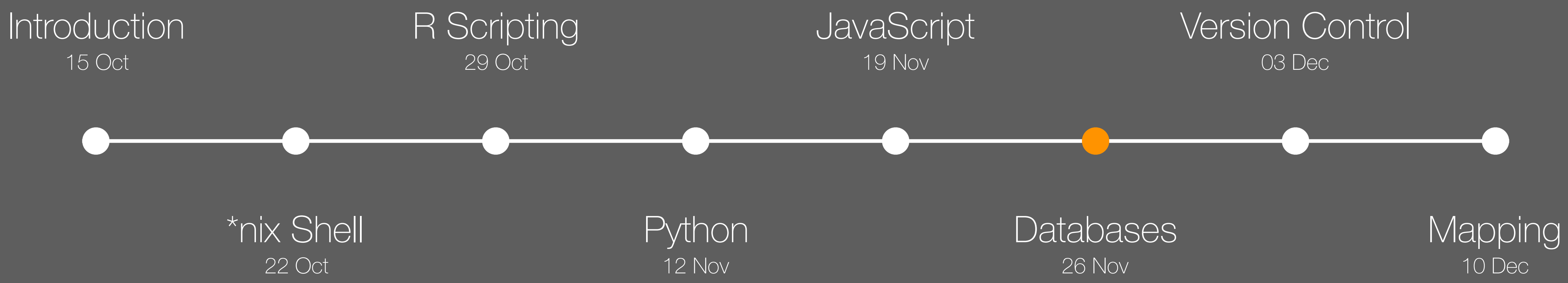
Databases

Bala
Dept. of Geography

Technical Tuesdays



Technical Tuesdays

Objectives

Introduction but **not a tutorial**

Tell people what is already there and **what is possible**

Give some **examples** for inspiration

Provide a **minimum viable environment** for
further learning and exploration

Quick Introduction to Databases

1 Context - Where does all of this come from?

2 Utility - For what these things are used?

3 Relevance - How can I use these for my purposes

4 Resources - Where can I learn more?

Layers of Abstraction

(Storage)

Disks

This the most basic form
of data storage. Just a bunch of
blank cells which can be either 1 or 0
HDD or SSD

Partitions

The first part of the disk has a broad
index on where sections of disk start
and end a.k.a partitions.
MBR or GPT

File System

This is the way files are stored on the
partitions. This is similar to partitions
in terms of recording start and end
but also has some hierarchy.
eg. FAT32, EXT4, APFS

Databases

This are one step higher where data
is stored as bunch of random files, but
the knows where the data is and can
retrieve it quickly with an index.
eg. Relational, NoSQL, Graph

Layers of Abstraction
(Storage)

Disks
This the most basic form
of data storage. Just a bunch of
blank cells which can be either 1 or 0
HDD or SSD

Partitions
The first part of the disk has a broad
index on where sections of disk start
and end a.k.a partitions.
MBR or GPT

File System
This is the way files are stored on the
partitions. This is similar to partitions
in terms of recording start and end
but also has some hierarchy.
eg. FAT32, EXT4, APFS

Databases
This are one step higher where data
is stored as bunch of random files, but
the knows where the data is and can
retrieve it quickly with an index.
eg. Relational, NoSQL, Graph

Layers of Abstraction

(Storage)

File System

Main purpose is storage and retrieval

Built for Flexibility, easy allocation of space

Hierarchical Structure

Tracks files with a **F**ile **A**llocation **T**able.

Databases

More specialised - Speed, Consistency, Reliability

Built to enforce Structure and Meaning

Can be Relational, Graph, Objects etc.

Tracks data using index.

History of Databases

Navigational DBMS '60s
Stores data in 1 parent, multiple child relations mapped to each other.
Strangely very similar to noSQL
CODASYL, IMS

History of Databases

Navigational DBMS '60s
Stores data in 1 parent, multiple child relations mapped to each other.
Strangely very similar to noSQL
CODASYL, IMS

Relational <'75
Stores data as related tables, based on common columns.
System R, INGRES

History of Databases

Navigational DBMS ‘60s

Stores data in 1 parent, multiple child relations mapped to each other.
Strangely very similar to noSQL
CODASYL, IMS

SQL dBs >'75

Uses Relation models, adds more functionality. Included ER model.
DB2, Oracle, Postgres

Relational <‘75

Stores data as related tables, based on common columns.
System R, INGRES

History of Databases

Navigational DBMS '60s
Stores data in 1 parent, multiple child relations mapped to each other.
Strangely very similar to noSQL
CODASYL, IMS

SQL dBs >'75
Uses Relation models, adds more functionality. Included ER model.
DB2, Oracle, Postgres

Relational <'75
Stores data as related tables, based on common columns.
System R, INGRES

Desktop dBs 80s
Smaller footprint for desktop computers.
dBASE*

History of Databases

Navigational DBMS '60s
Stores data in 1 parent, multiple child relations mapped to each other.
Strangely very similar to noSQL
CODASYL, IMS

Relational <'75
Stores data as related tables, based on common columns.
System R, INGRES

SQL dBs >'75
Uses Relation models, adds more functionality. Included ER model.
DB2, Oracle, Postgres

Desktop dBs 80s
Smaller footprint for desktop computers.
dBASE*

Confused phase! '90s
Programming moved to Objects,
Effort to convert Objects to Relational
ORM

History of Databases

Navigational DBMS '60s
Stores data in 1 parent, multiple child relations mapped to each other.
Strangely very similar to noSQL
CODASYL, IMS

Relational <'75
Stores data as related tables, based on common columns.
System R, INGRES

SQL dBs >'75
Uses Relation models, adds more functionality. Included ER model.
DB2, Oracle, Postgres

Desktop dBs 80s
Smaller footprint for desktop computers.
dBASE*

Confused phase! '90s
Programming moved to Objects,
Effort to convert Objects to Relational
ORM

NoSQL '00s
Truly object oriented databases which are optimised to scale horizontally.
CouchDB, MongoDB, Cassandra etc.

History of Databases

Navigational DBMS '60s
Stores data in 1 parent, multiple child relations mapped to each other.
Strangely very similar to noSQL
CODASYL, IMS

Relational <'75
Stores data as related tables, based on common columns.
System R, INGRES

SQL dBs >'75
Uses Relation models, adds more functionality. Included ER model.
DB2, Oracle, Postgres

Desktop dBs 80s
Smaller footprint for desktop computers.
dBASE*

Confused phase! '90s
Programming moved to Objects,
Effort to convert Objects to Relational
ORM

NoSQL '00s
Truly object oriented databases which are optimised to scale horizontally.
CouchDB, MongoDB, Cassandra etc.

NewSQL '10s
Trying to achieve NoSQL's scaling with SQL's reliability.
Still very new!

History of Databases

Navigational DBMS '60s
Stores data in 1 parent, multiple child relations mapped to each other.
Strangely very similar to noSQL
CODASYL, IMS

Relational <'75
Stores data as related tables, based on common columns.
System R, INGRES

SQL dBs >'75
Uses Relation models, adds more functionality. Included ER model.
DB2, Oracle, Postgres

Desktop dBs 80s
Smaller footprint for desktop computers.
dBASE*

Confused phase! '90s
Programming moved to Objects,
Effort to convert Objects to Relational
ORM

NoSQL '00s
Truly object oriented databases which are optimised to scale horizontally.
CouchDB, MongoDB, Cassandra etc.

NewSQL '10s
Trying to achieve NoSQL's scaling with SQL's reliability.
Still very new!

Bunch of other specialised databases,

Spatial, Temporal, Parallel, In-Memory, Graph, Real-time, deductive, distributed etc.,

Quick Introduction to Databases

1 Context - Where does all of this come from?

2 Utility - For what these things are used?

3 Relevance - How can I use these for my purposes

4 Resources - Where can I learn more?

Use Cases

Navigational DBMS '60s
Stores data in 1 parent, multiple child relations mapped to each other.
Strangely very similar to noSQL
CODASYL, IMS

Relational <'75
Stores data as related tables, based on common columns.
System R, INGRES

SQL dBs >'75
Uses Relation models, adds more functionality. Included ER model.
DB2, Oracle, Postgres

Desktop dBs 80s
Smaller footprint for desktop computers.
dBASE*

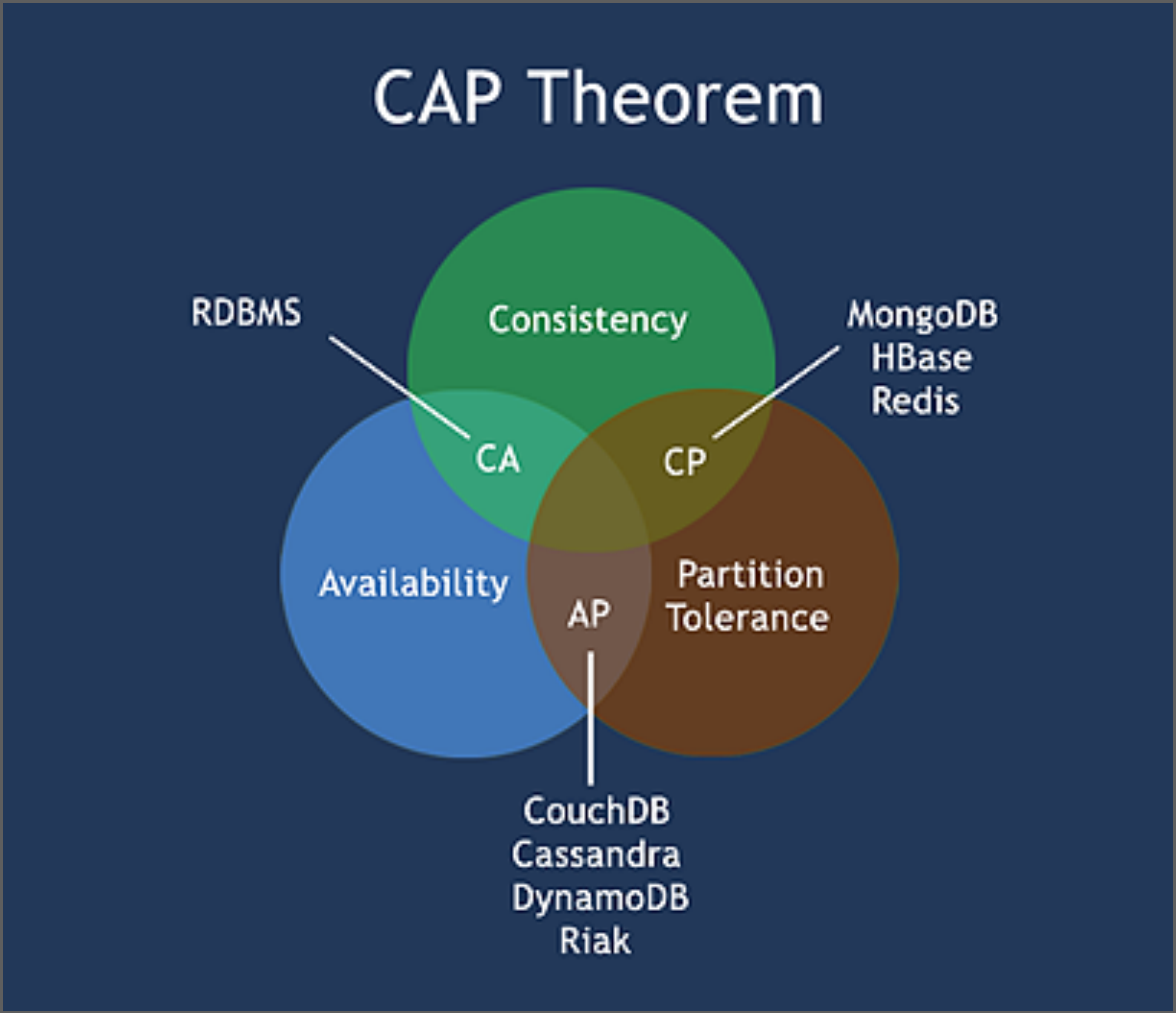
Confused phase! '90s
Programming moved to Objects,
Effort to convert Objects to Relational
ORM

NoSQL '00s
Truly object oriented databases which are optimised to scale horizontally.
CouchDB, MongoDB, Cassandra etc.

NewSQL '10s
Trying to achieve NoSQL's scaling with SQL's reliability.
Still very new!

Bunch of other specialised databases,

Spatial, Temporal, Parallel, In-Memory, **Graph**, Real-time, deductive, distributed etc.,



https://geeknarrator.com/2017/10/02/cap-theorem-consistency-availability-and-partition-tolerance/

DEMO

Summary - Relational

Relational databases are good for **Tabular data.**

Good for lots of records with same structure - **Vertical Scaling**

Selecting and filtering data

Grouping and ordering data.

Indices are very important for performance

Tables can be **linked** with each other

Data needs to be **normalised** for efficiency

Queries can be very complex and can give new information

*if you are in geography, just use Postgres
it is good with spatial data

DEMO

Summary - NoSQL

NoSQL databases are good for **Object like data.**

Good for fair number of records with complex structure - **Horizontal Scaling**

Creating database and collections

adding, updating and filtering data.

Mostly used with **Unstructured** data.

Social networks, news portals etc.

DEMO

Summary - Neo4j

Neo4j is good for **Graphs / networks.**

I have **no idea** about who uses this

But it looks **super cool**

I have been looking to use it for **5 years**

If you can find a reason **please use it**

Quick Introduction to Databases

1 Context - Where does all of this come from?

2 Utility - For what these things are used?

3 Relevance - How can I use these for my purposes

4 Resources - Where can I learn more?

Should I Use Databases?

Of course You should!!

When should I Use a data base?

Meaning in the **structure** of the data

Data that need to be **stored** and **retrieved** frequently

Data that need to be **analysed**

Data that need to be used by **different people**

Data that need to be **secure**

Data that need to be **fast** and **reliable**

When should I **NOT** Use a data base?

Data is **hierarchical**

Data **doesn’t change** much

Data is used very **infrequently**

You have **media** - picture, videos etc.

You have a **small** dataset <10mb

You have a **large** dataset >10TB

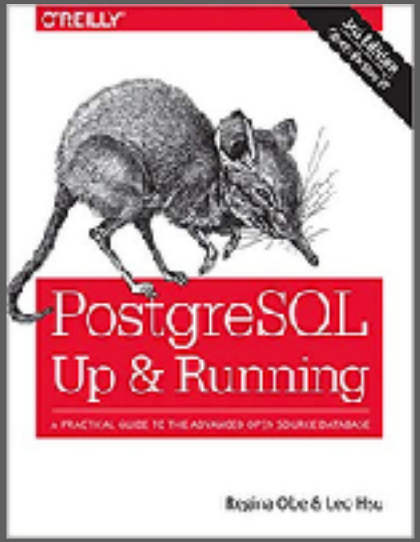
Quick Introduction to Databases

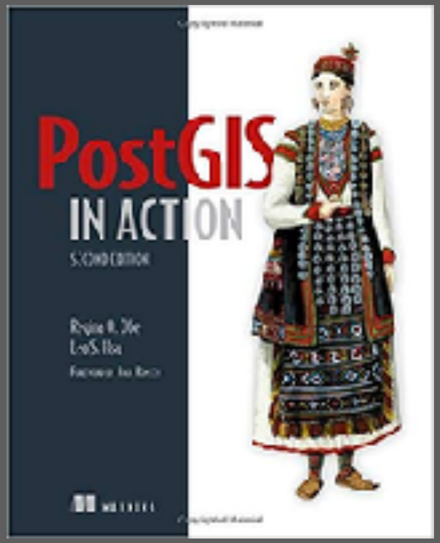
1 Context - Where does all of this come from?

2 Utility - For what these things are used?

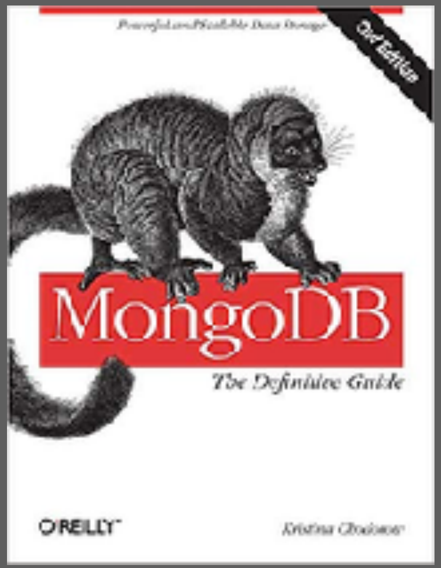
3 Relevance - How can I use these for my purposes

4 Resources - Where can I learn more?

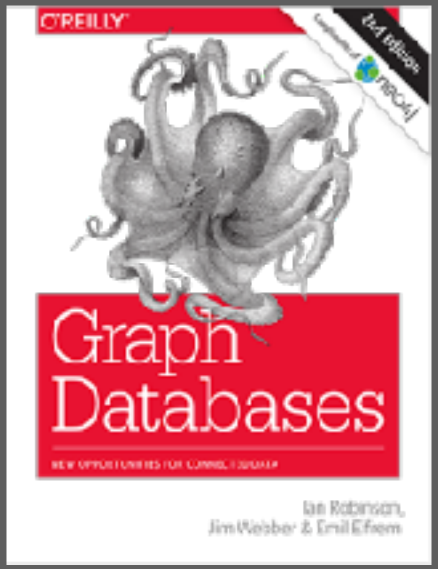




Official Postgres Documentation



Official Mongo Documentation



Free E-Books!

Questions