BUSINESS ACADEMY
AARHUS

Databases & XML (11) 02.05.2016

Time	Agenda
8.30	Polyglot persistence
8.45	Quiz
9.00	Choosing your database
9.20	Mongo vs SQL
9.35	API exercise
10.00	Pause
10.30	Schemaless
10.35	CAP and BASE
10.50	Groups for interdisciplinary project
11.00	Mongo exercises
12.00	Lunch
12.30	About the exam
12.45	Test exam
13.45	Experiences with the exam

Agenda

- Internet Week Denmark Festival – May 9th – 13th in Aarhus
- Open Data – brug offentlige data til innovation – May 12th

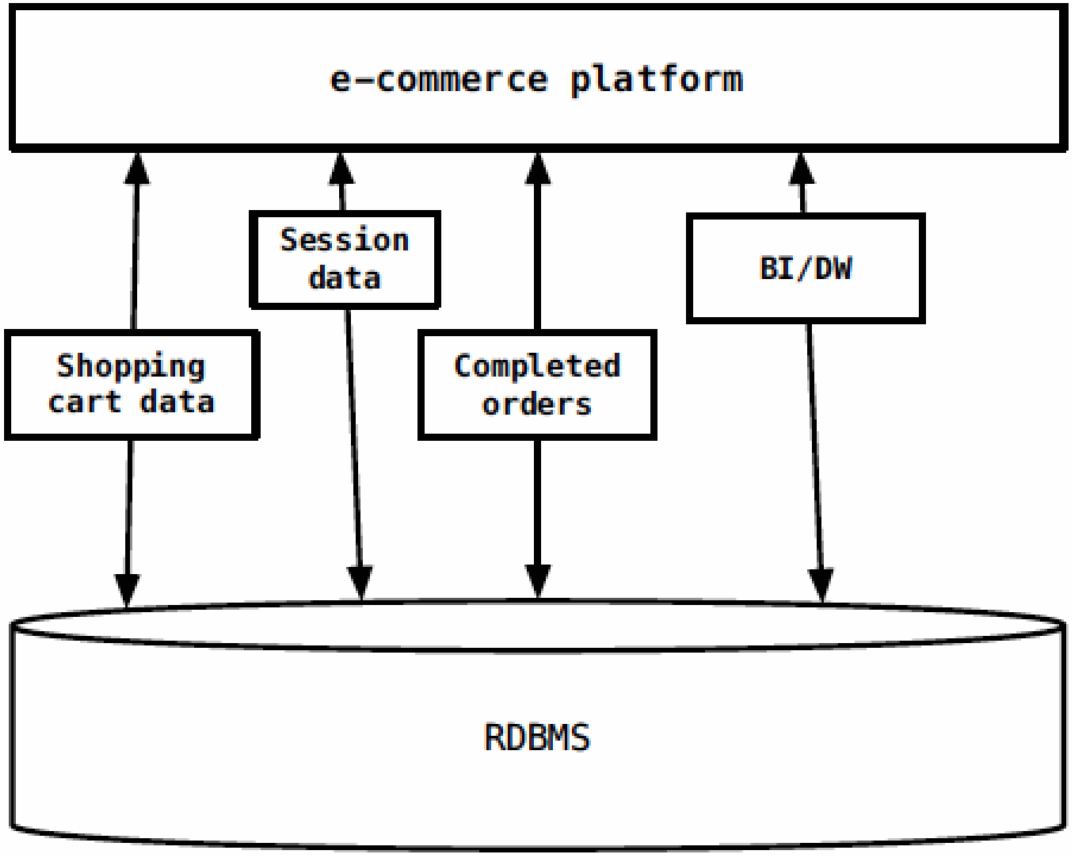


Figure 13.1 *Use of RDBMS for every aspect of storage for the application*

Polyglot Persistence (RDBMS)

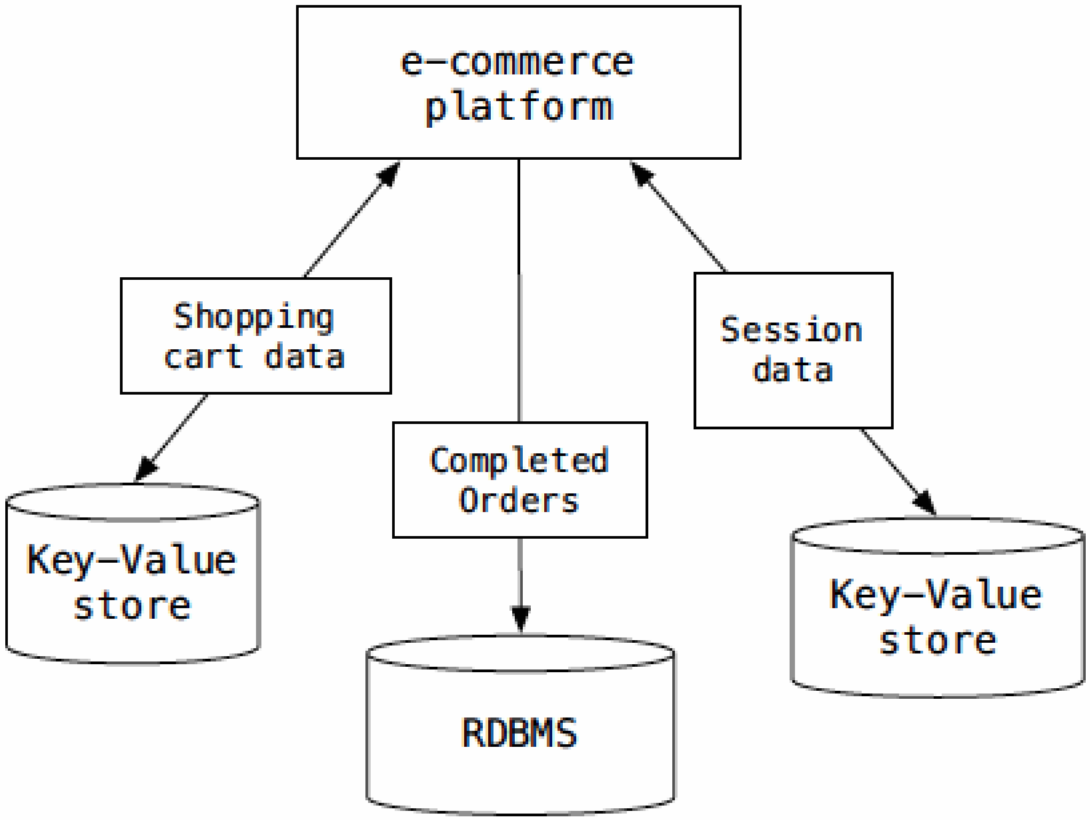


Figure 13.2 *Use of key-value stores to offload session and shopping cart data storage*

Polyglot Persistence (Polyglot)

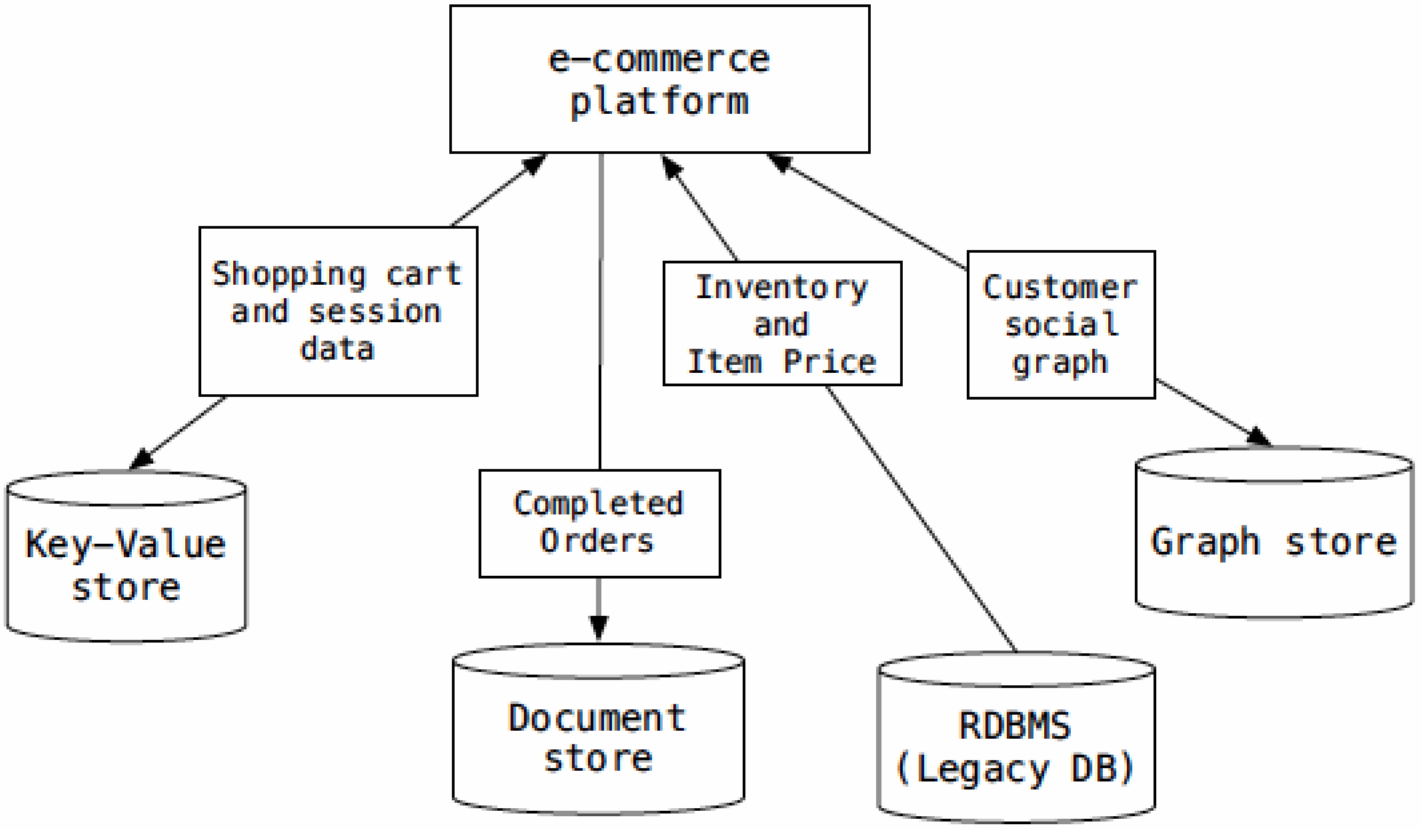


Figure 13.3 *Example implementation of polyglot persistence*

Polyglot Persistence (Polyglot)

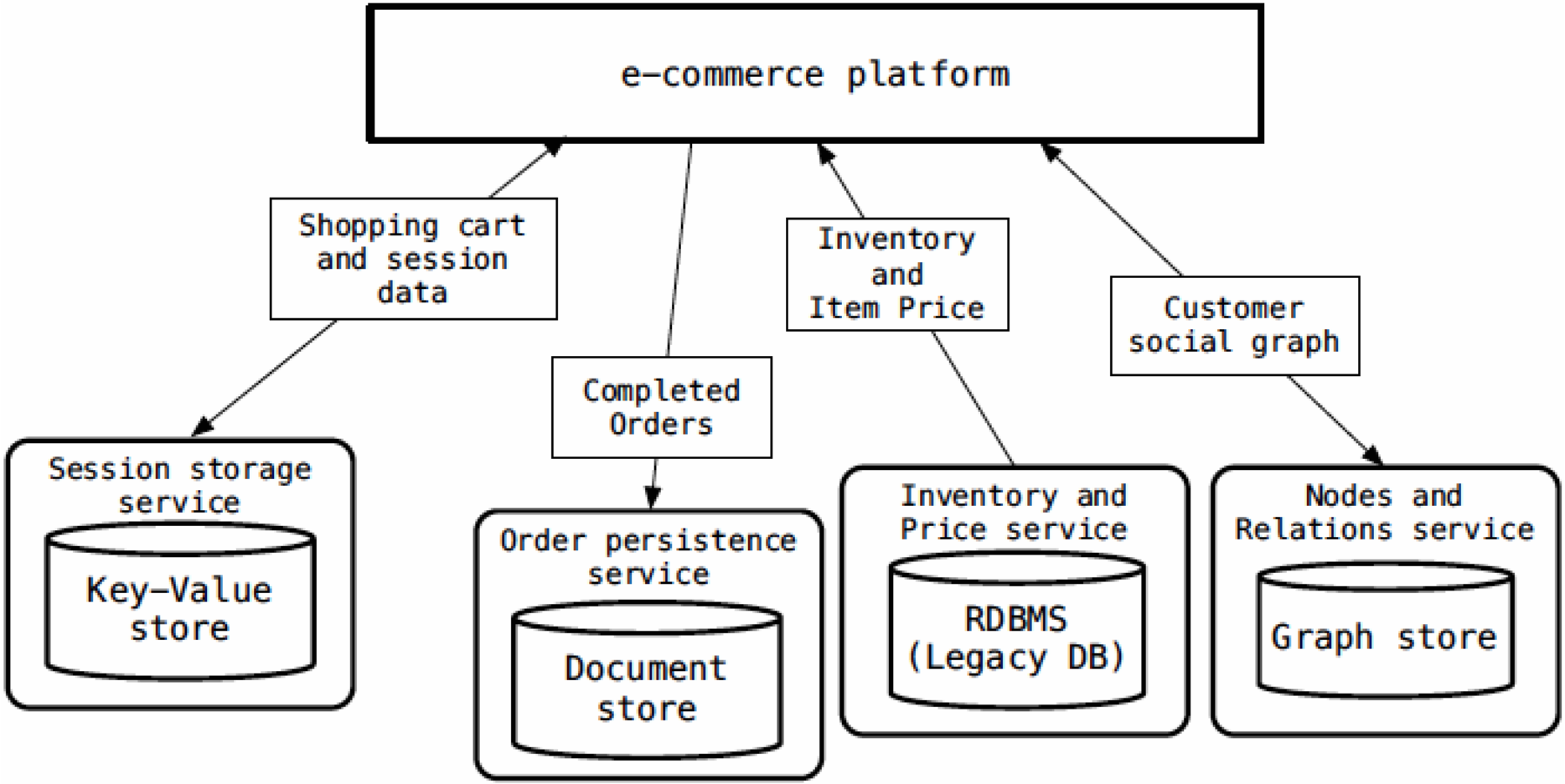


Figure 13.5 *Using services instead of talking to databases*

Polyglot Persistence (Polyglot)

13.8 Key Points

- Polyglot persistence is about using different data storage technologies to handle varying data storage needs.
- Polyglot persistence can apply across an enterprise or within a single application.
- Encapsulating data access into services reduces the impact of data storage choices on other parts of a system.
- Adding more data storage technologies increases complexity in programming and operations, so the advantages of a good data storage fit need to be weighed against this complexity.

13.8 Key Points

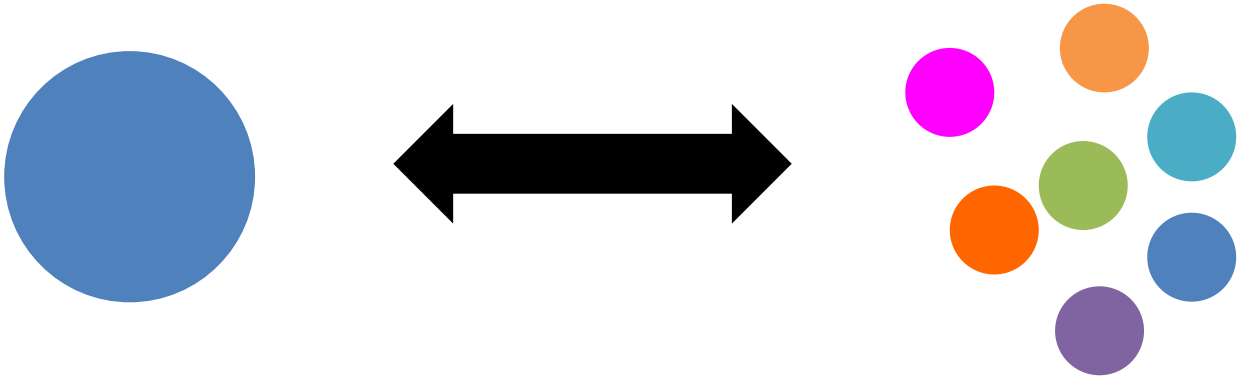
- Polyglot persistence is about using different data storage technologies to handle varying data storage needs.
- Polyglot persistence can apply across an enterprise or within a single application.
- Encapsulating data access into services reduces the impact of data storage choices on other parts of a system.
- Adding more data storage technologies increases complexity in programming and operations, so the advantages of a good data storage fit need to be weighed against this complexity.

13.8 Key Points

- Polyglot persistence is about using different data storage technologies to handle varying data storage needs.
- Polyglot persistence can apply across an enterprise or within a single application.
- Encapsulating data access into services reduces the impact of data storage choices on other parts of a system.
- Adding more data storage technologies increases complexity in programming and operations, so the advantages of a good data storage fit need to be weighed against this complexity.

13.8 Key Points

- Polyglot persistence is about using different data storage technologies to handle varying data storage needs.
- Polyglot persistence can apply across an enterprise or within a single application.
- Encapsulating data access into services reduces the impact of data storage choices on other parts of a system.
- Adding more data storage technologies increases complexity in programming and operations, so the advantages of a good data storage fit need to be weighed against this complexity.



http://martinfowler.com/bliki/PolyglotPersistence.html

http://memeagora.blogspot.dk/2006/12/polyglot-programming.html

http://www.techrepublic.com/article/developers-are-calling-it-quits-on-polyglot-programming/

Polyglot and Full-Stack

BUSINESS ACADEMY
AARHUS

Quiz time!

- Programmer Productivity
- Data-Access Performance
- Sticking with the Default

Choosing your database

- Look at what your software will need to do.
- Run through the current features and see if and how the data usage fits.
- As you do this, you may begin to see that a particular data model seems like a good fit.
- That closeness of fit suggests that using that model will lead to easier programming.

Polyglot persistence is about using multiple data storing solutions.

It may be that you'll see different data storage models fit different aspects of your data. Using multiple databases is inherently more complex than using a single store, but the advantages of a good fit in each case may be better overall.

- An aggregate-oriented database may be very fast for reading or retrieving aggregates compared to a relational database where data is spread over many tables.

- Test your potential databases' performance in the scenarios that matter to you. Reasoning about how a database may perform can help you build a short list, but the only way you can assess performance properly is to build something, run it, and measure it.

- You can't build your actual system, so you need to build a representative subset. Choose scenarios that are the most common, the most performance-dependent, and those that don't seem to fit you database model well. The latter may alert you to any risks outside your main use cases.

- There are many cases where you're better off sticking with the default option of a relational database:
  - They are well known; you can easily find people with experience of using them
  - They are mature, so you are less likely to run into the rough edges of new technology
  - There are lots of tools that are built on relational technology that you can take advantage of
  - You don't have to deal with the political issues of making an unusual choice – picking a new technology will always introduce a risk of problems should things run into difficulties

- To choose a NoSQL database you need to show a real advantage over relational databases for your situation.
- There are many cases where it is advantageous to use NoSQL – but not "all" or even "most".

Sticking with the Default

Break – 5 minutes

BUSINESS ACADEMY
AARHUS

RDBMS vs NoSQL

RDBMS

- Structured and organized data
- Structured query language (SQL)
- Data and its relationships are stored in separate tables.
- Data Manipulation Language, Data Definition Language
- Tight Consistency
- BASE Transaction

NoSQL

- Stands for Not Only SQL
- No declarative query language
- No predefined schema
- Key-Value pair storage, Column Store, Document Store, Graph databases
- Eventual consistency rather ACID property
- Unstructured and unpredictable data
- CAP Theorem
- Prioritizes high performance, high availability and scalability

Vertical scaling

To scale vertically (or scale up) means to add resources within the same logical unit to increase capacity. For example to add CPUs to an existing server, increase memory in the system or expanding storage by adding hard drive.

Horizontal scaling

To scale horizontally (or scale out) means to add more nodes to a system, such as adding a new computer to a distributed software application. In NoSQL system, data store can be much faster as it takes advantage of "scaling out" which means to add more nodes to a system and distribute the load over those nodes.

Scalability

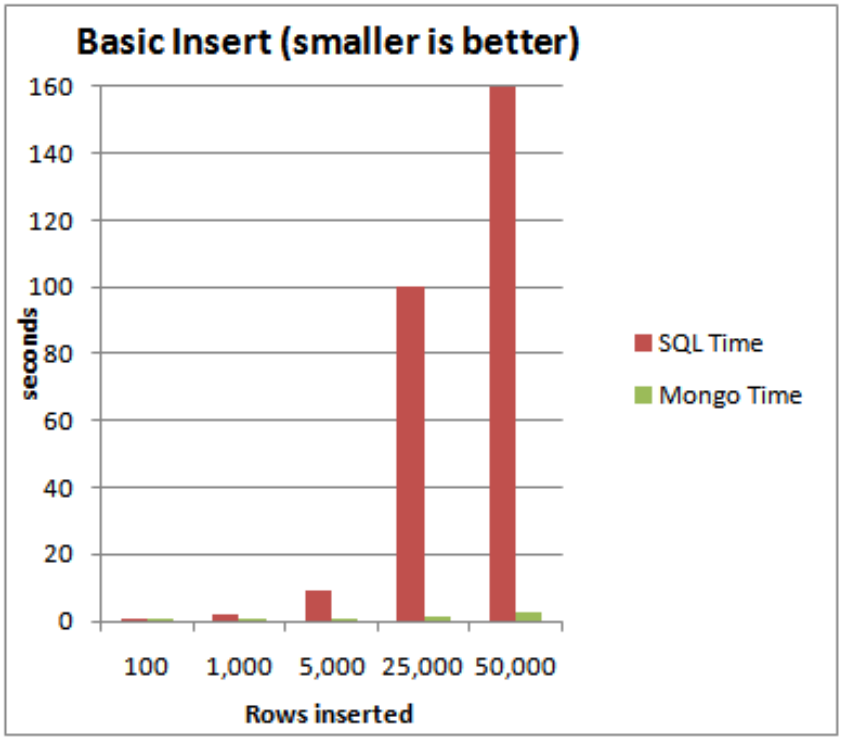
SQL Terms/Concepts	MongoDB Terms/Concepts
database	database
table	collection
row	*document* or *BSON* document
column	field
index	index
table joins	embedded documents and linking
primary key Specify any unique column or column combination as primary key.	primary key In MongoDB, the primary key is automatically set to the _id field.
aggregation (e.g. group by)	aggregation pipeline

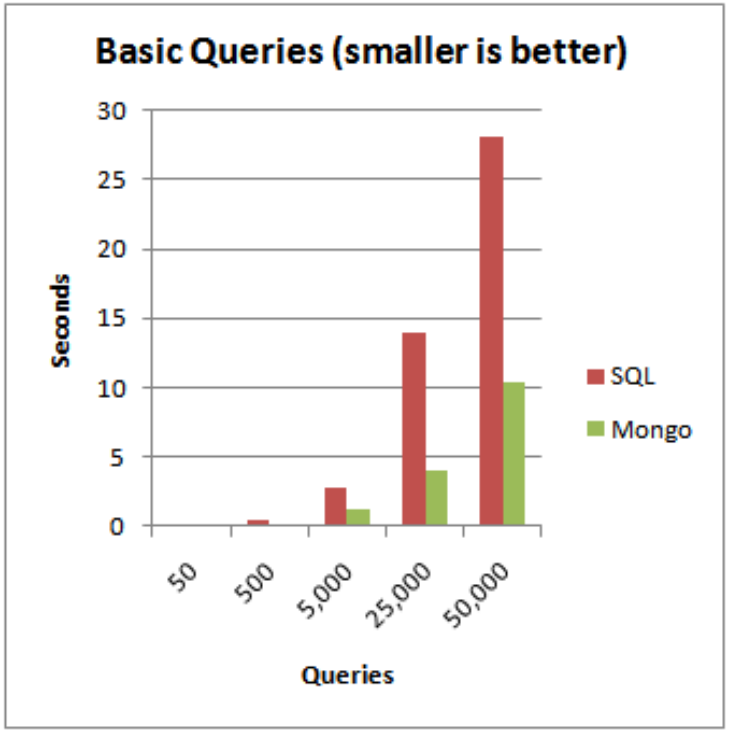
”What answers do I have?”

”What questions do I have?”

Data modelling

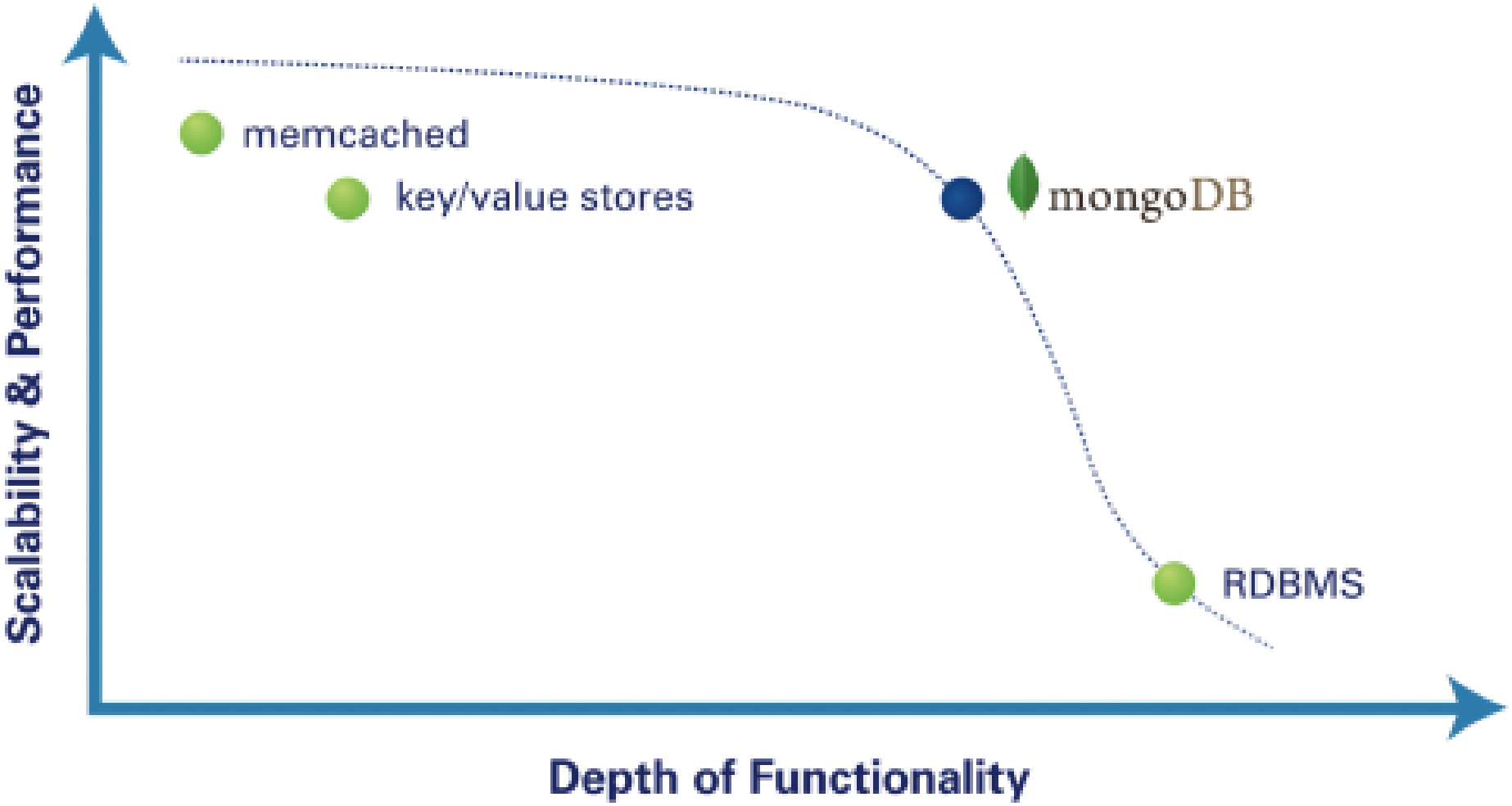
BUSINESS ACADEMY
AARHUS





http://blog.michaelckennedy.net/2010/04/29/mongodb-vs-sql-server-2008-performance-showdown/

MongoDB vs. SQL Server



Scalability/Functionality

APIs

So far we have looked a little bit into the APIs of:

- Youtube
- OMDB (movie database)

Look online and find an API that you are interested in.

Output some interesting information from that API on some webpage (e.g. your videowatcher page).

Exercise

Break 30 minutes

BUSINESS ACADEMY
AARHUS

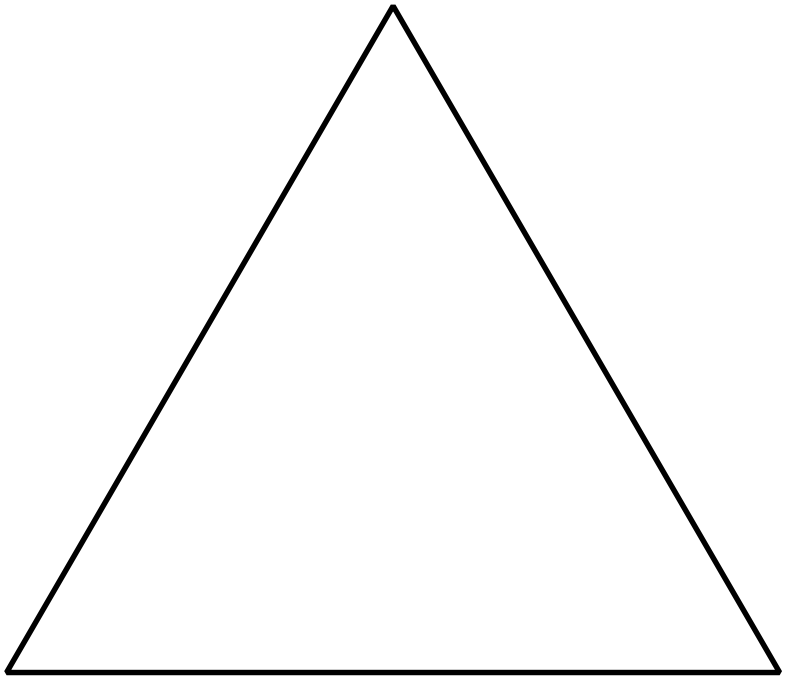
Spot the data types

```
{
    "product": "t-shirt",
    "price": 12.50,
    "popular": true,
    "stock": 300,
    "tags": ["organic", "printed"],
    "reviews": null,
    "variants":
        [
            {"variant": "Black Night", "image": "black.jpg"},
            {"variant": "Blue Morning", "image": "blue.jpg"}
        ]
}
```

BUSINESS ACADEMY
AARHUS

CAP og BASE

Availability



Consistency

Partition Tolerance

CAP

Consistency

All clients see the same data after execution of operation

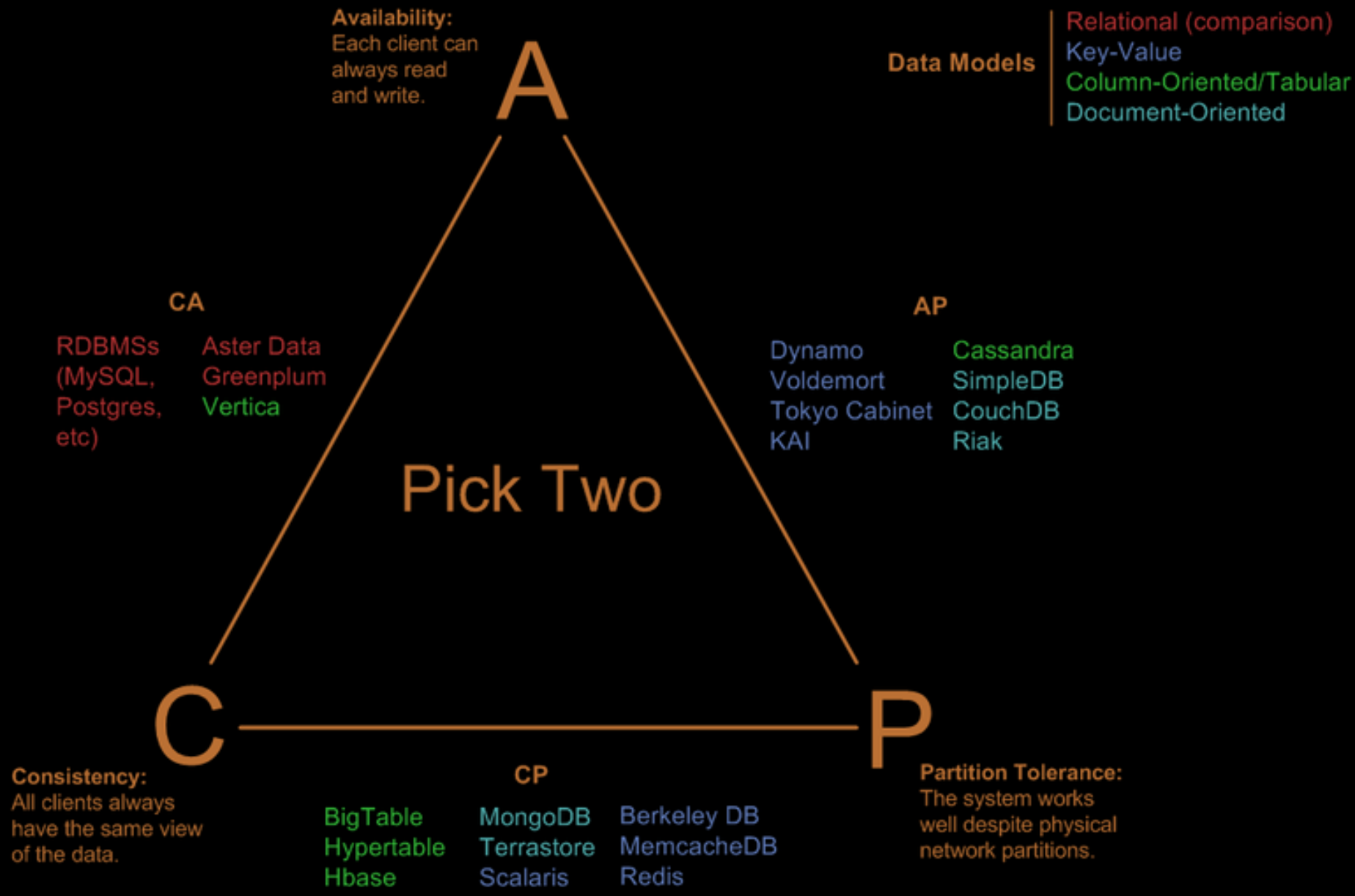
Availability

Always on – no downtime

Partition Tolerance

Continues to function in case of failure in a node

Visual Guide to NoSQL Systems



Basically **A**vailable

Indicates that the system does guarantee availability

Soft State

Indicates that the state of the system may change due to the eventual consistency

Eventual Consistency

Indicates that the state of the system will become consistent (if it has the opportunity – is not disturbed)

ACID	BASE
Atomic	Basically Available
Consistency	Soft State
Isolation	Eventual Consistency
Durable	

Eric Brewer's revision of the CAP theorem and BASE

(Why "2 of 3" is misleading)

Finally, all three properties are more continuous than binary. Availability is obviously continuous from 0 to 100 percent, but there are also many levels of consistency, and even partitions have nuances, including disagreement within the system about whether a partition exists.

http://www.infoq.com/articles/cap-twelve-years-later-how-the-rules-have-changed

Group 1:
Christian
Steffen BP
Troels
Andreas Bøsig

Group 2:
Jakob Bak
David
Alice
Frederik

Group 3:
Kaloyan
Simeon
Nikolay
Elias

Group 4:
Steffen P
Mikkel VBA
Nikolaj
Rostislav

Group 5:
Neli
Michelle
Mikkel Ottesen
Andreas WK

Group 6:
Martin
Jannie
Mikkel HK
Terkel

Group 7:
Martynas
Natalia
Adrian
Thomas

Group 8:
Mike
Mihail
Malik
Vilius

Groups for mandatory

Break 5 minutes

BUSINESS ACADEMY
AARHUS

- Pick two of your favourite series with at least 4 seasons on OMDB (e.g. Game of Thrones and Modern Family)
- Make a database in your preferred mongo tool called OMDB.
- Insert each season of your series into a collection called series.
- That means that you now should have at least 8 documents in your collection series.

Series database

- June 6th, 7th or 8th sometime between 8.30 and 16.00
- English or Danish

- 30 minutes preparation after receiving your question
  - Books, papers, computer, internet allowed
  - No communication with others
    - No calling a friend
    - No Facebook
    - No StackOverflow or other foras
- 20 minutes examination
- 5 minutes for deciding the grade and giving you the reason for it

About the exam

- Data Modeling
- SQL – stored procedures and views
- SQL – transactions and triggers
- JSON
- XML
- NoSQL

Subjects for the exam

STORED PROCEDURES AND VIEWS

```
CREATE PROCEDURE [dbo].[selectTagsFromBlogPost]
-- Add the parameters for the stored procedure here
@blogid int = 0
AS
BEGIN
-- SET NOCOUNT ON added to prevent extra result sets from
-- interfering with SELECT statements.
SET NOCOUNT ON;

    -- Insert statements for procedure here
SELECT T.TagName
FROM Tag as T
INNER JOIN TagBlogPost as TBP ON TBP.TagId = T.TagId
INNER JOIN BlogPost as BP ON BP.BlogPostId = TBP.BlogPostId
WHERE BP.BlogPostId = @blogid;
END
```

What does this code snippet do? How would you use this code?

Explain what conclusions you can draw on the data model behind this. Reflect upon what benefits one may get from placing this in a stored procedure – and are there any cons?