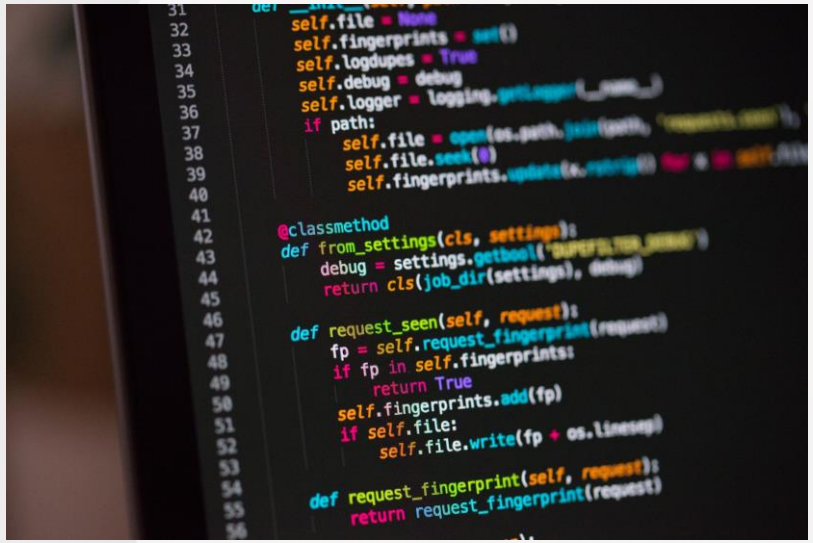
Level 5 Data Engineer
Module 2 Topic 5

NoSQL



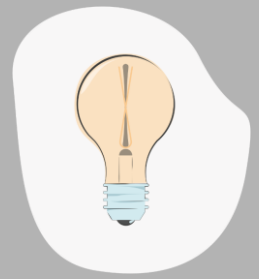
L5 Data Engineer Higher Apprenticeship
Module 2 / 12 (“**Databases and Data Lakes**”)
Topic 5 / 5

Progress check-in

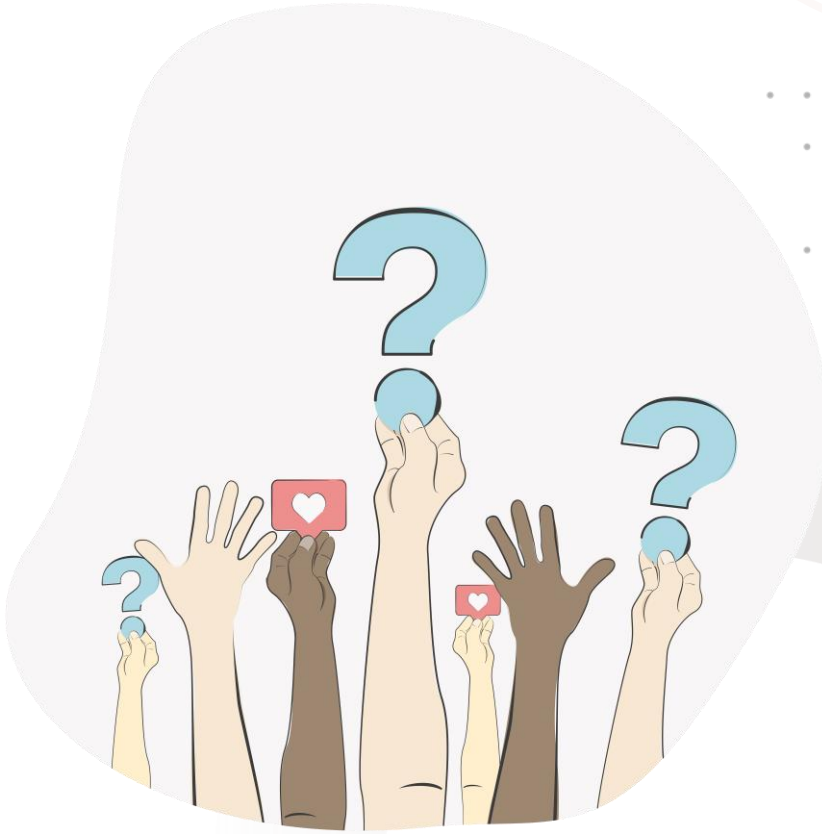
How are thing’s going on the programme?

What exercises/assignments have you done so far?

How are you feeling about working with different data formats?



Submit your responses to the chat!

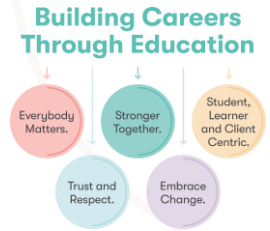


Session aim and objectives

Completion of this topic supports the following outcomes:

- Explain differences between SQL and NoSQL
- Evaluate examples of NoSQL technologies and what business requirements they answer
- Set the scene for data pipelines (future modules)
- Practice writing NoSQL queries with MongoDB

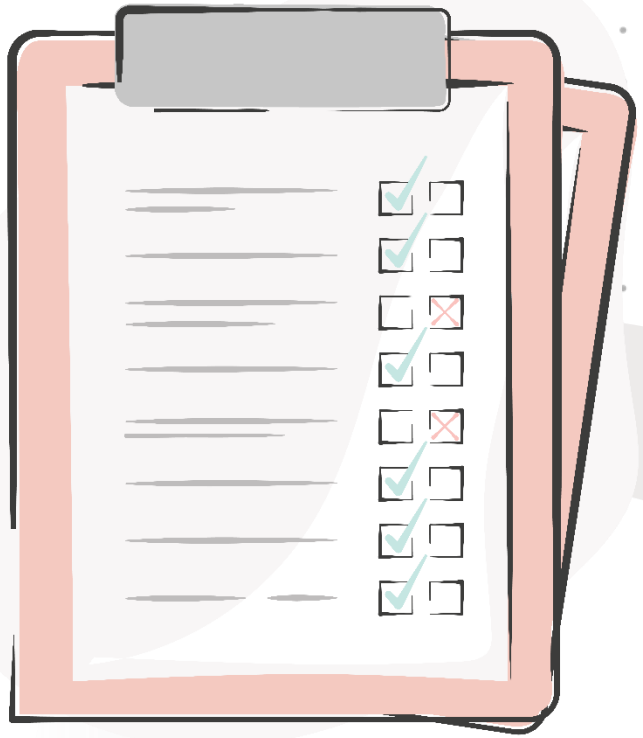
BPP



Webinar Agenda

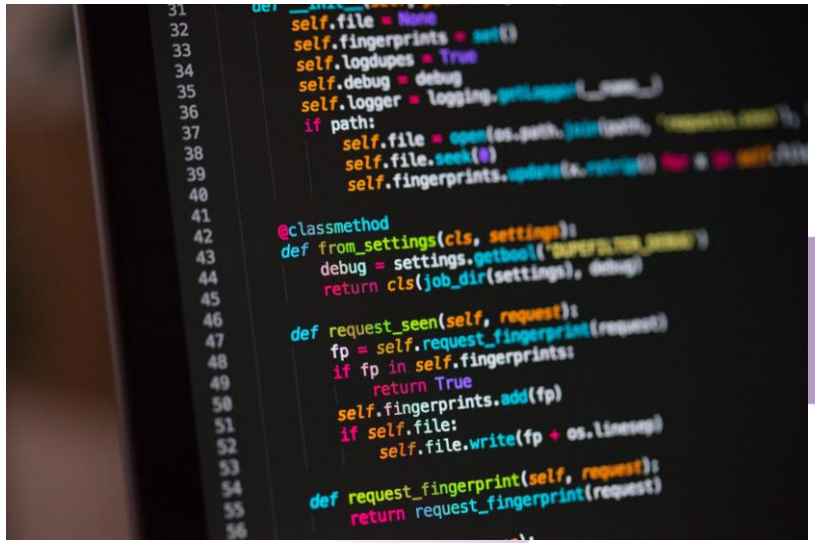
This webinar will include the following:

- JSON and API Recap
- Introduction to SQL
- NoSQL databases
- NoSQL solutions
- Types and applications of NoSQL databases



BPP

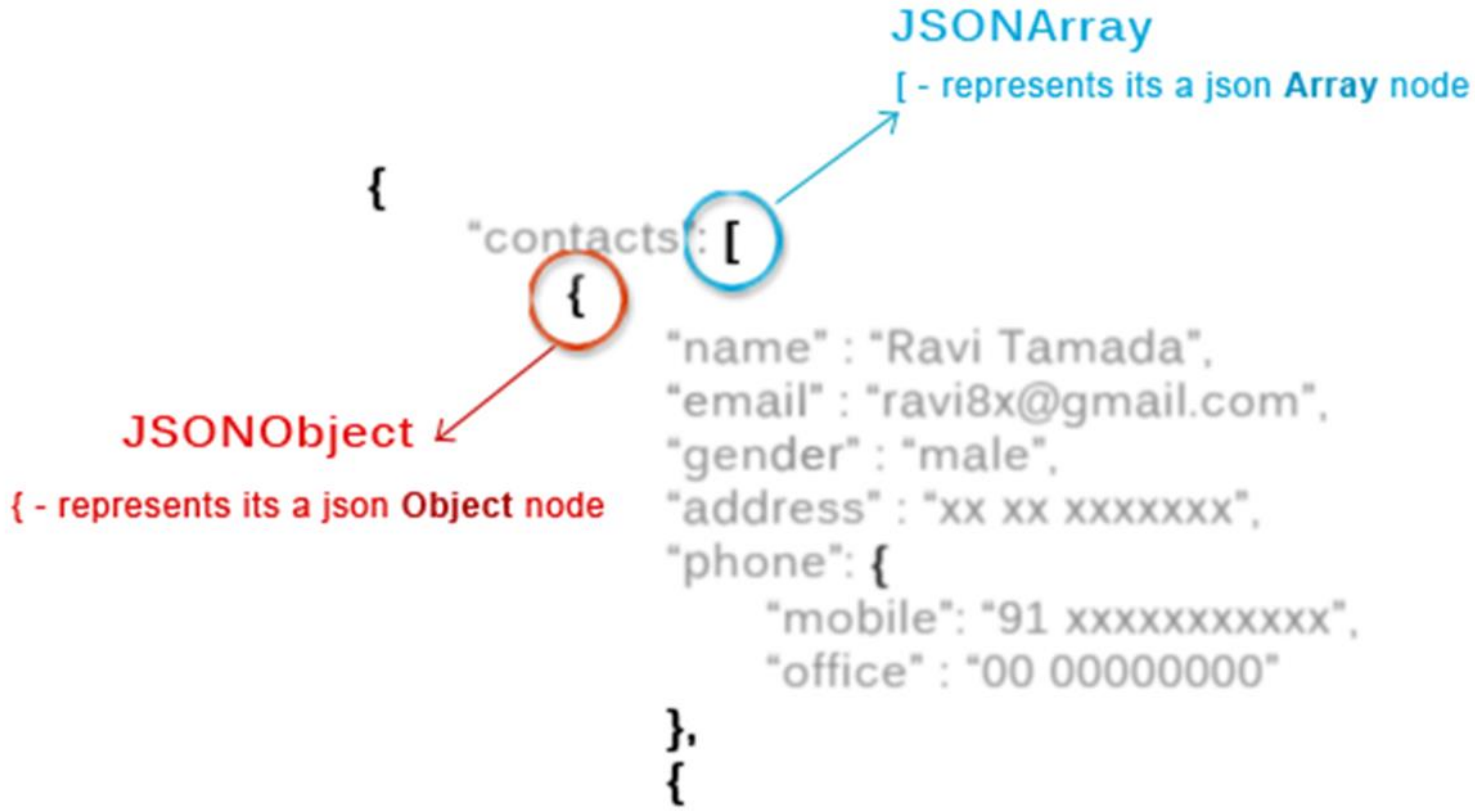
JSON and API Recap



BPP

Recap

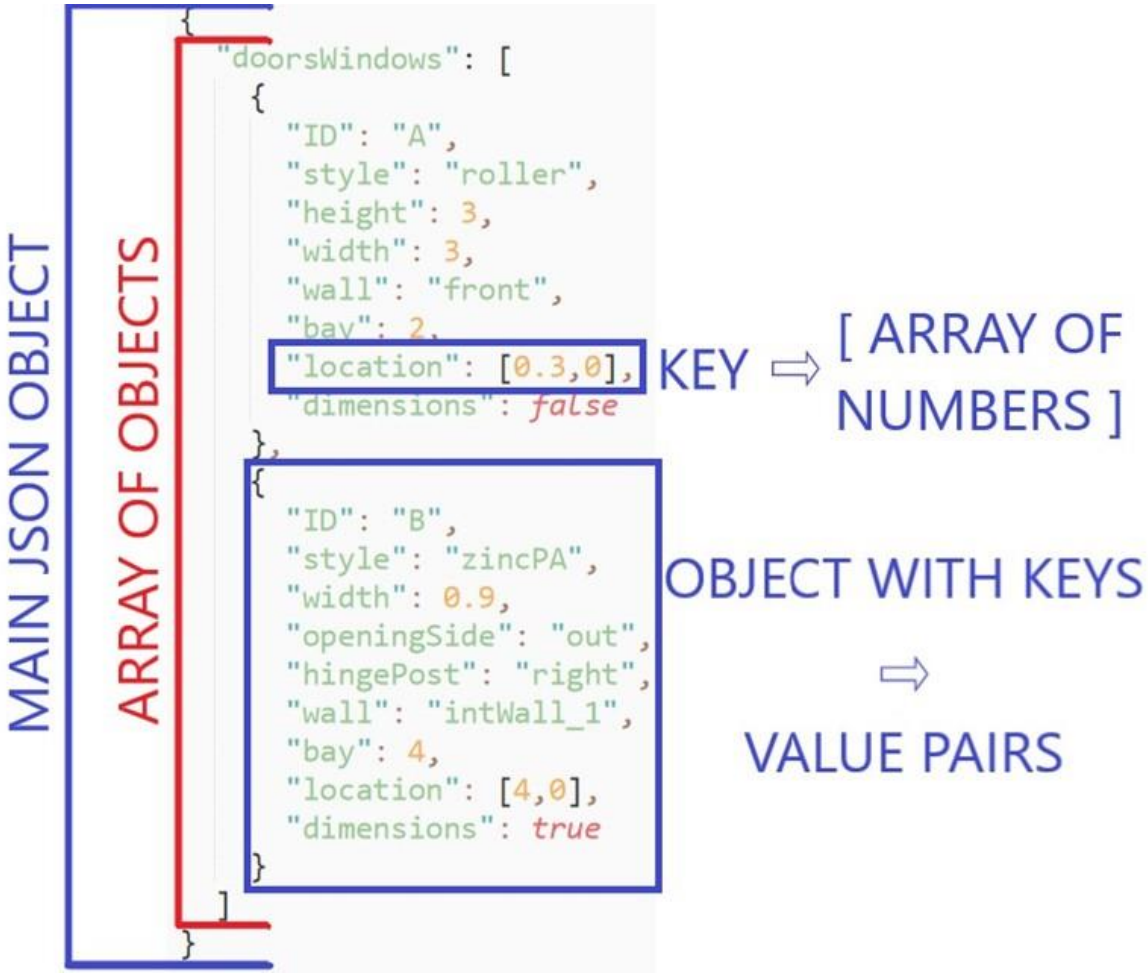
JSON



A diagram illustrating the difference between a square bracket and a curly brace in JSON

Recap

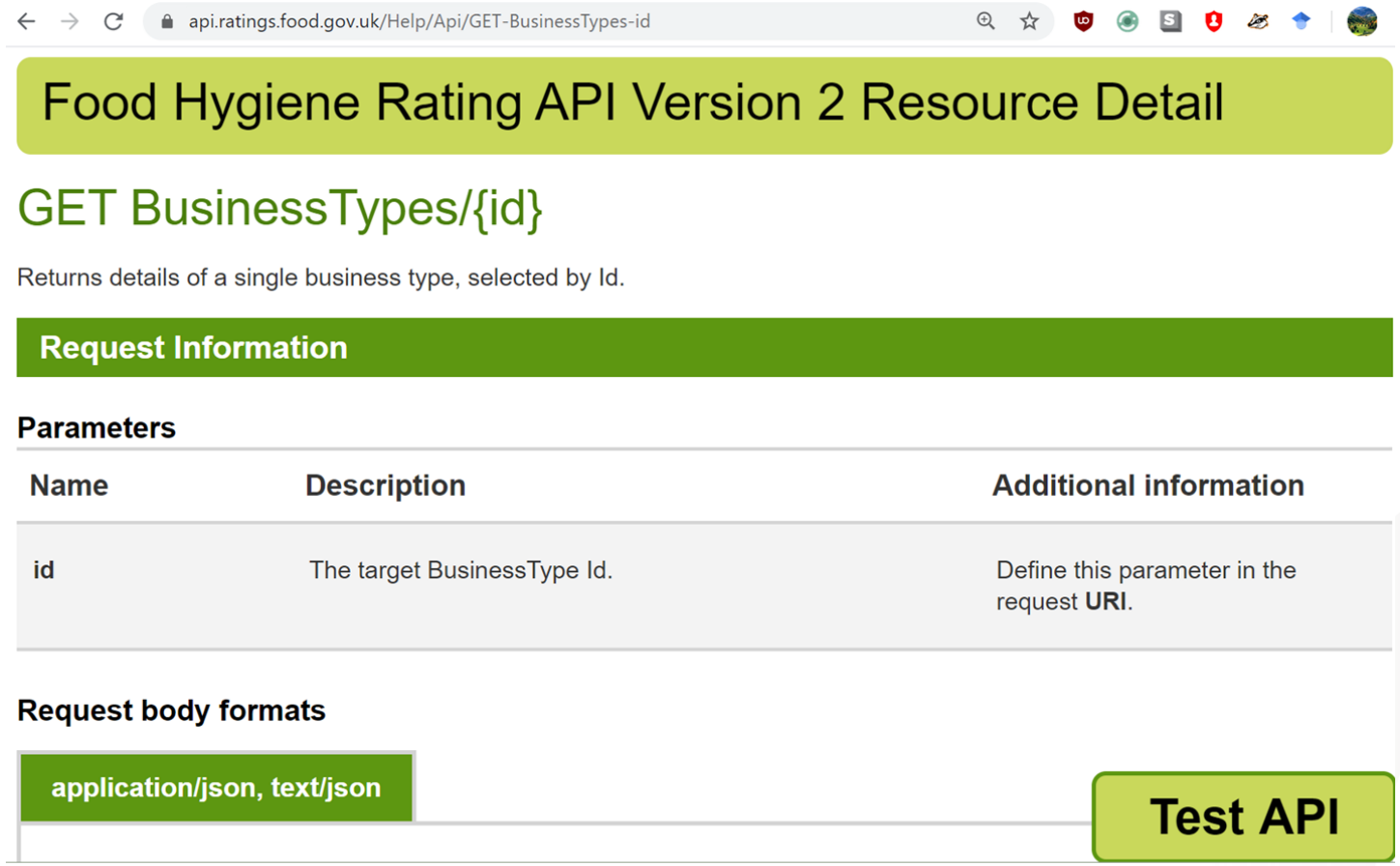
More JSON



A diagram illustrating the array of objects in a JSON file

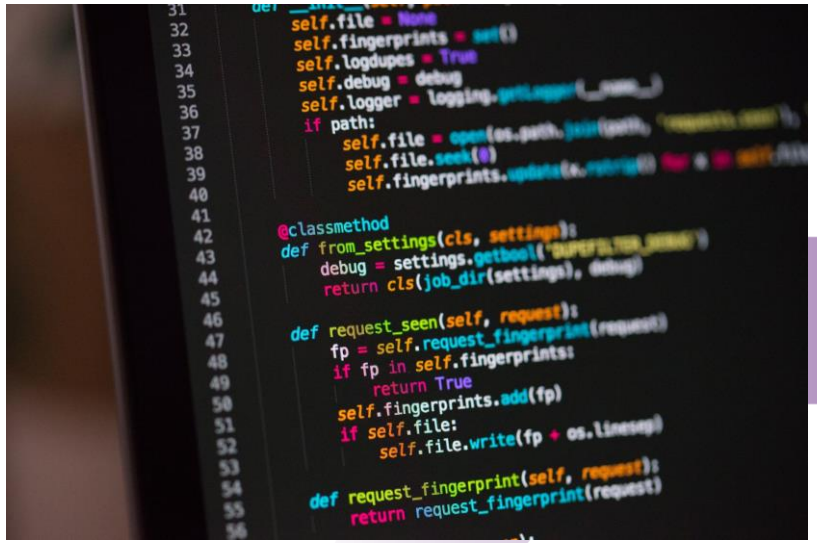
Recap

API



A diagram illustrating API documentation website

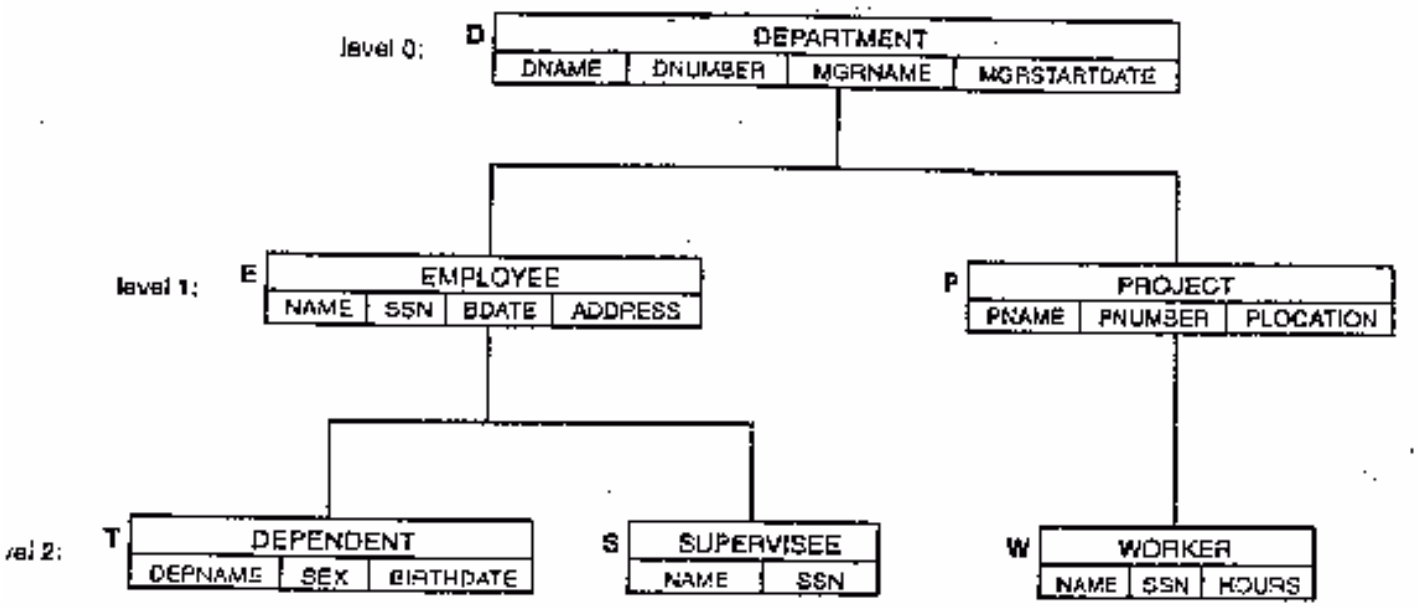
Introduction to SQL



BPP

Setting the scene

Going back to the beginning…



An early data centre

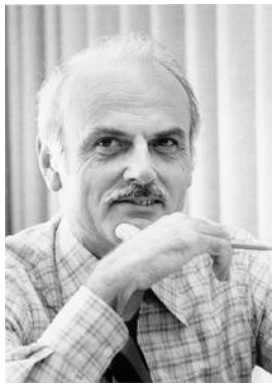
SQL Databases

Going back to the beginning…

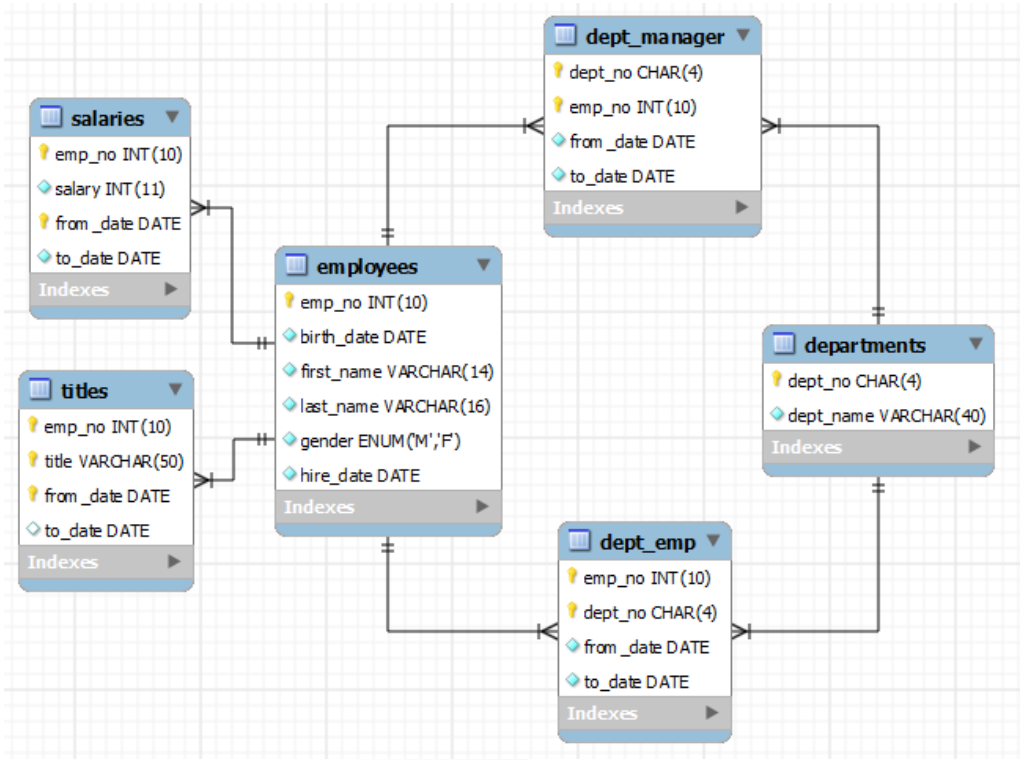
- Really better called “Relational Databases”

- Key construct is the “Relation”, a.k.a. the table

  - Table is similar to a Sheet in an Excel file
  - Rows represent records or entries/observations
  - Columns represent attributes/features or variable names



Edgar Frank "Ted" Codd, inventor of the relational model for database management



An example of an SQL database

Recap

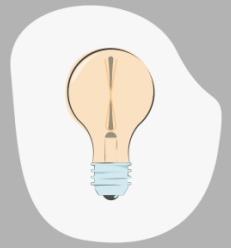
What is a standard SQL again…?

SQL:

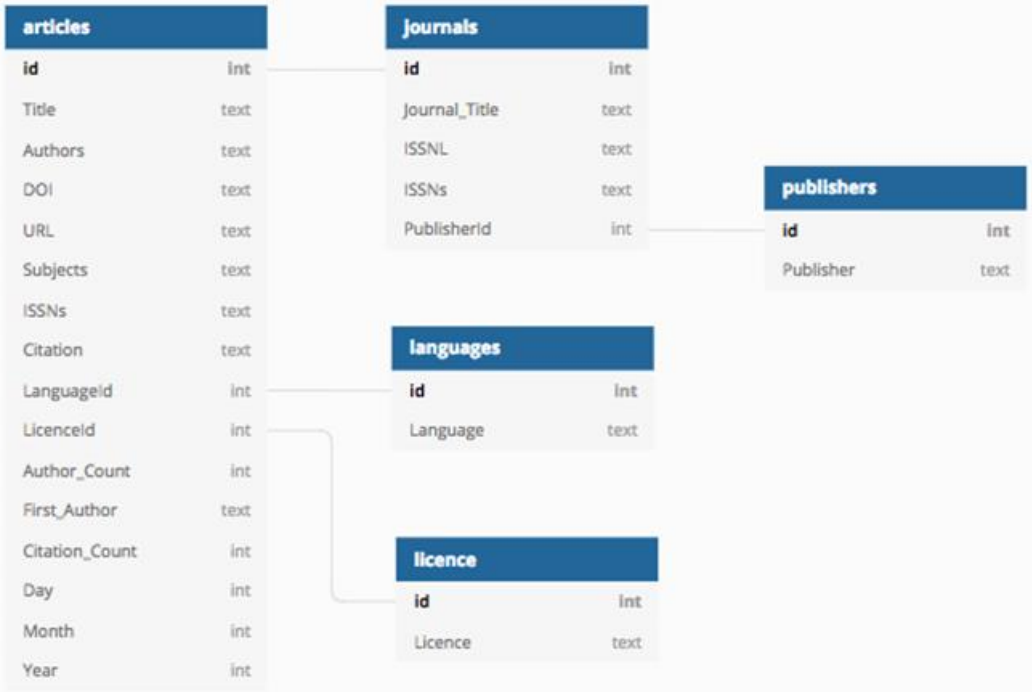
- Relational databases
- Used to create structured data schemas
- Relationships between the different entries are created with foreign keys

Spreadsheet methaphor for SQL:

- Spreadsheet is a database
- Each sheet represents a table



A table is represented by a two-dimensional table: each column represents a data item, each row represents an observation.



An example of standard SQL

Why is SQL not sufficient

Let’s understand more…

- Rigid Schema - Consider the following scenario…

The schema of MySQL

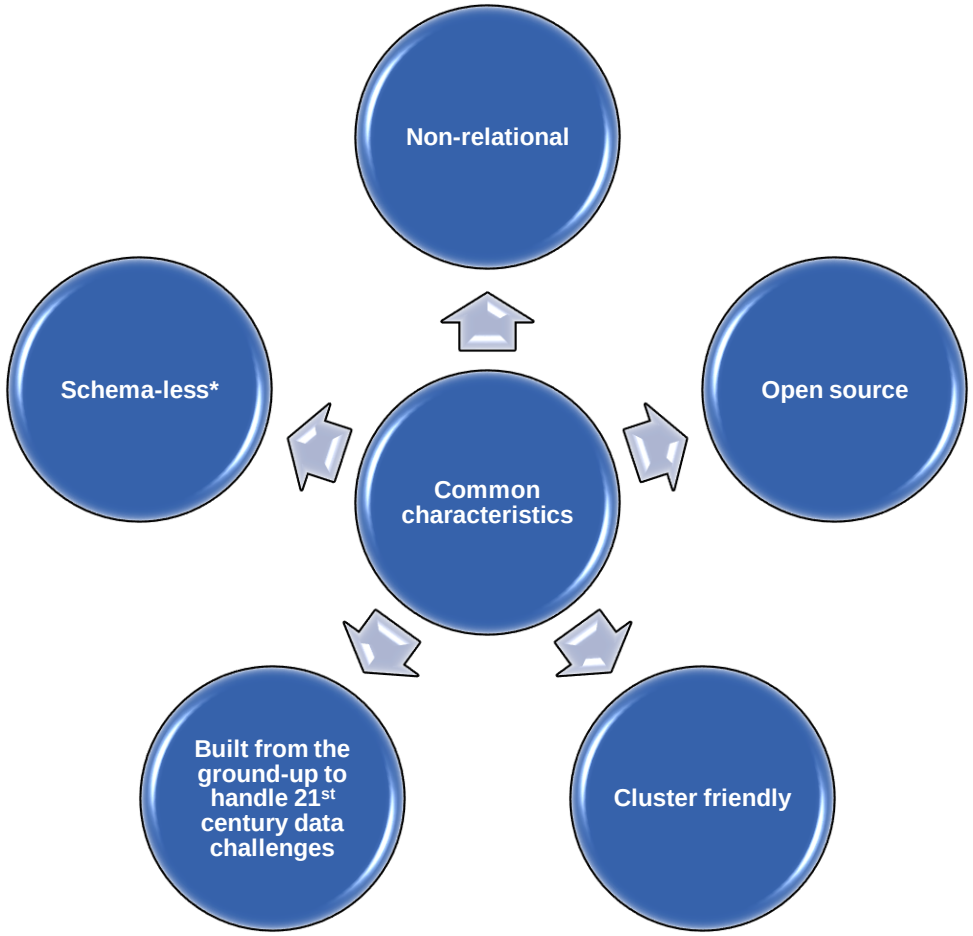
Scalability

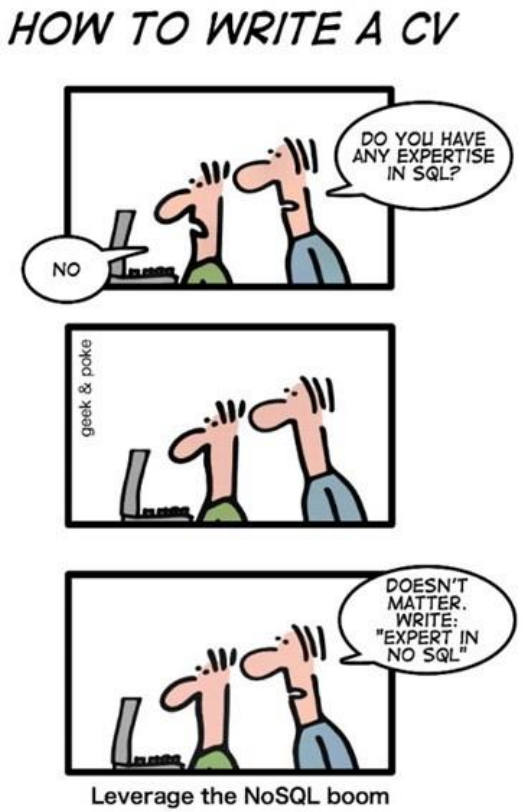
- SQL requires ‘Joins’
- Think ‘Big data’

Enter NoSQL

A really generic and unofficial definition

An ill-defined set of mostly open-source databases, mostly developed in the 21st century, and mostly not using SQL.



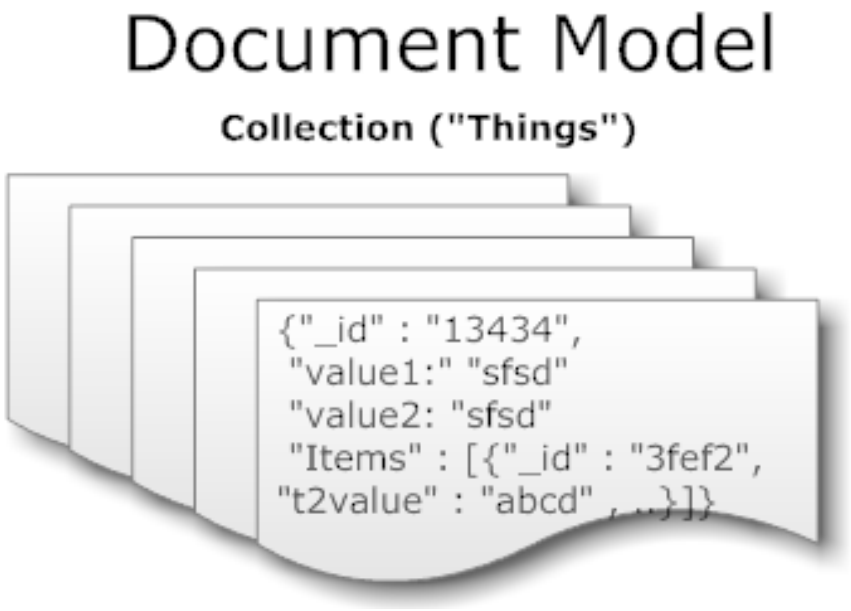


Discussion

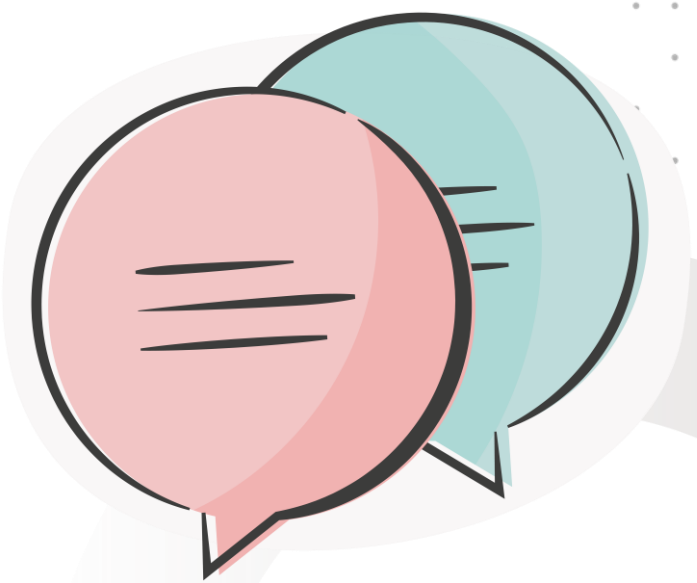
What is a document?

Not what you think

Word document <> NoSQL document

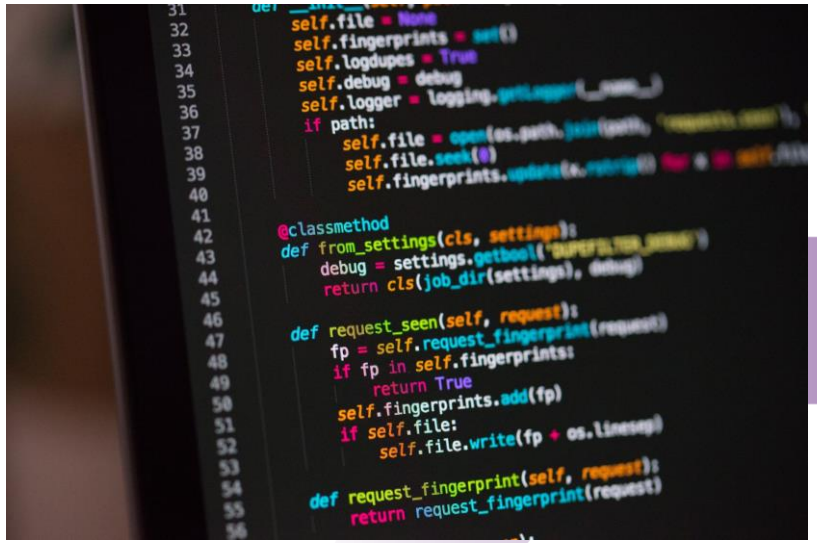


An example of a NoSQL document



Discussion

NoSQL databases



BPP

Example NoSQL document

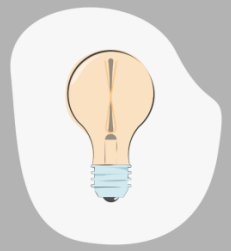
A MongoDB document…


```
{
"business_id": "rncjoVoEFUJGCUoC1JgnUA",
 "full_address": "8466 W Peoria Ave\nSte
6\nPeoria, AZ 85345",
"open": true,
"categories": ["Accountants", "Professional
Services", "Tax Services",],
 "city": "Peoria",
"review_count": 3,
"name": "Peoria Income Tax Service",
 "neighborhoods": [],
 "longitude": -112.241596,
 "state": "AZ",
"stars": 5.0,
"latitude": 33.581867000000003,
"type": "business"
}
```

Document-based databases

When should you use them?

- The **schema is not clear** in advance or changes quite often
- You want to **dump data** into your database **very fast**
- You often want to **search for exactly one record**
- You do not want to do **complicated queries** and merges



A table is represented by a two-dimensional table: each column represents a data item, each row represents an observation.

An example of standard SQL

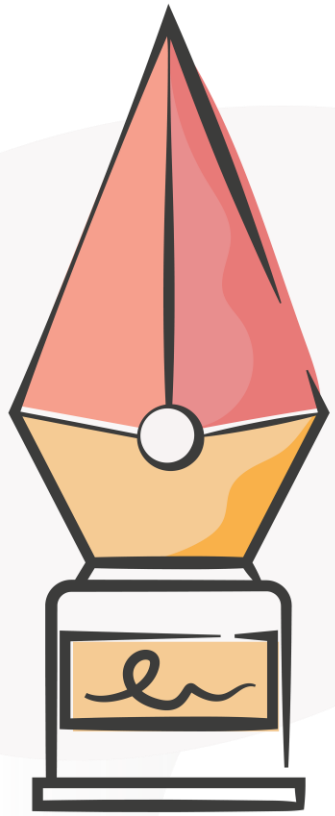
Terminology and concepts

SQL vs MongoDB

SQL Terms / Concepts	MongoDB Terms / Concepts
database	*database*
table	*collection*
row	*document*
column	*field*
index	*index*

Just remember this…

Relational term	MongoDB equivalent
Database	Database
Tables	Collections
Rows	Documents



MongoDB

Facts that will blow your mind!

- No Schemas
- No transactions
- No joins
- Max document size of 16MB
  - Larger documents handled with GridFS

Mind blown!

BPP

MongoDB

Facts that might not blow your mind so much!

- Runs on most common OSs
  - Windows
  - Linux
  - Mac
  - Solaris

- Data stored as BSON (Binary JSON)

  - used for speed
  - translation handled by language drivers

Not so amazing facts!

Retrieving data

Statements and commands…

SQL Statement	MongoDB commands
SELECT* FROM table	**db.collection.find()**
SELECT* FROM table WHERE artist = ‘Nirvana’	db.collection.find**({Artist:”Nirvana”})**
SELECT* FROM table ORDER By Title	db.collection.find().**sort**(Title:1)
DISTINCT	.distinct()
GROUP BY	.group()
>=,<	$gte, $lt

SQL statements and MongoDB commands

Using MongoDB with Python

Let's understand more…

- The most basic type of query that can be performed in MongoDB is `find_one()`
- This method returns a single document matching a query (or `None` if there are no matches)
- It is useful when you know there is only one matching document, or are only interested in the first match

```
>>> from pymongo import MongoClient
```

```
>>> client = MongoClient('localhost', 27017)
```

```
>>> db = client.test_database
```

Examples of using MongoDB with Python

BPP

Understanding cursors in database queries

What you need to know…

Remember:

- FIND Queries return a cursor, which can be iterated over
- A cursor is a reference to the result set of a query

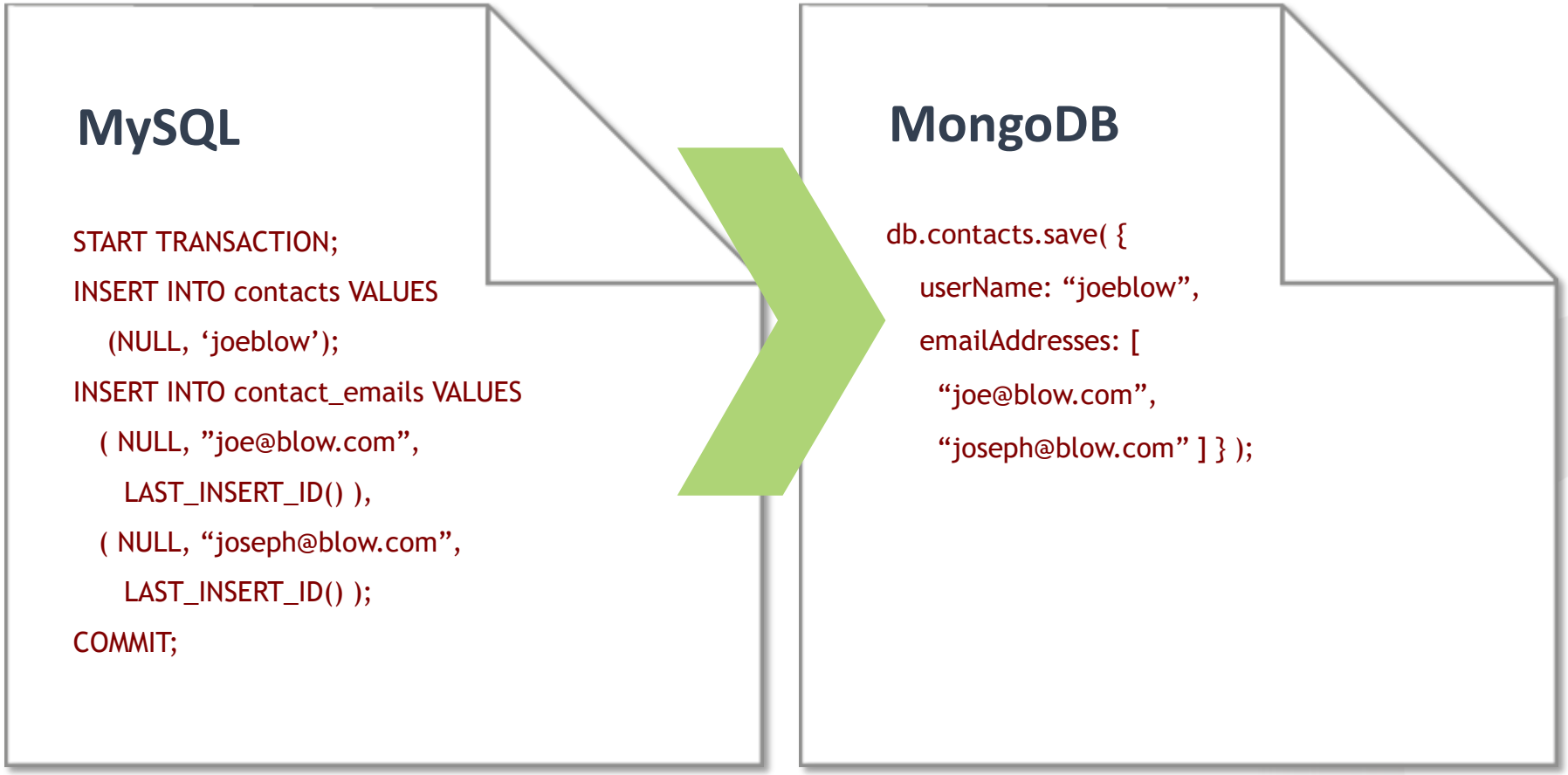
```
db.users.find({'last_name': 'Smith'})

// retrieve ssn field for documents where last_name ==
'Smith':
db.users.find({last_name: 'Smith'}, {'ssn': 1});

// retrieve all fields *except* the thumbnail field, for
all documents:
db.users.find({}, {thumbnail:0});

// retrieve all users order by last_name:
db.users.find({}).sort({last_name: 1});

// skip and limit:
db.users.find().skip(20).limit(10);
```

An example of find queries in MongoDB with Python

BPP

Writing data to MongoDB

An easy task!



Rules for building NoSQL data structuresqueries

What you need to know…

Rule 1: Every document must have an _id**.**

Rule 2: There is only one rule

- posts.**count_documents**({"author": "Mike"})
- *d = datetime.datetime(2009, 11, 12, 12)*
  posts.**find**({"date": {"$**lt**": d}}).sort("author")

Further useful MongoDB operations

BPP

Real-world application

Optimising databases for modern data…

The business benefits of NoSQL

According to a study by **Allied Market Research**, companies using NoSQL databases realised cost savings of over 25% compared to using traditional relational databases.

Some examples:

Twitter cut infrastructure costs by over $140 million annually after switching from a MySQL database to Apache Cassandra for managing tweets and user timelines

LinkedIn reduced query latency from hours to seconds and lowered costs by 80% by adopting Voldemort, a NoSQL key-value store, for their platform

Airbnb was able to achieve over $1 million in savings per year by switching to MongoDB and DynamoDB to manage their growing data needs

The Global Market Research Firm (AMR)

BPP

A focus on your experience

Which NoSQL database type is most relevant for your current or future job role and professional interests?

A. Document databases like MongoDB

B. Graph databases like Neo4j

C. Column-based databases like Cassandra

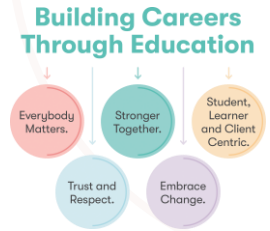
D. Key-value stores like Redis

E. I don’t expect NoSQL databases to be relevant in my role

Submit your responses to the chat

BPP

Knowledge check poll



Which of the following statements describes the key difference between SQL and NoSQL databases?

A. SQL databases are newer while NoSQL databases are older

B. SQL databases are unstructured while NoSQL databases are structured

C. SQL databases use JSON while NoSQL databases use tables

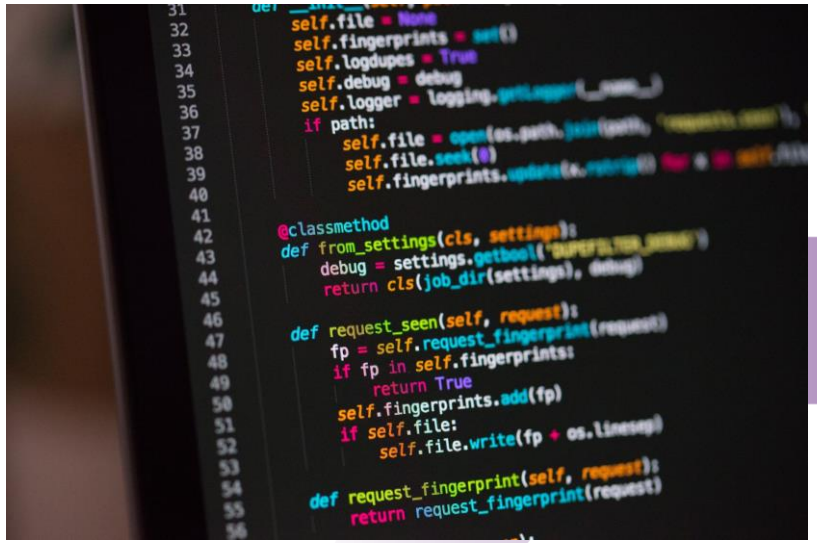
D. SQL databases are relational while NoSQL databases are non-relational

Submit your responses to the chat!

Feedback

The correct answer is: **D** - The key difference is that SQL databases are relational with predefined schemas, while NoSQL databases are non-relational and have

NoSQL solutions



BPP

NoSQL sollutions

Why (and where) they might be suitable…

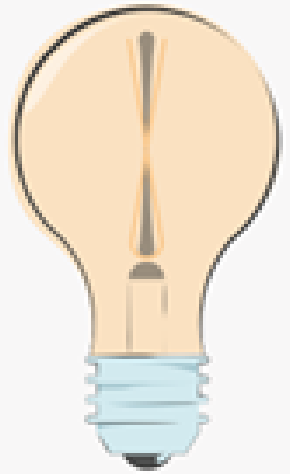
NoSQL stands for “not only SQL,” or “non-SQL."

→ not tabular - stores data differently from relational databases (no row/column/table design)

→ Won't use SQL to query data.

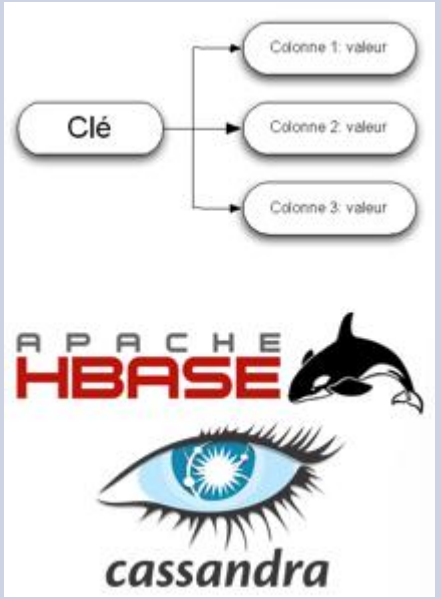
↓ Can store relationship data, but it is not split between tables

← offer flexible schemas and scale well for large amounts of data and higher user loads.



Major types of NoSQL databases

What you need to familiar with…

Key-value	Document based	Column based	Graph based
			

Building Careers Through Education

Everybody Matters. Stronger Together. Student, Learner and Client Centric. Trust and Respect. Embrace Change.

BPP

Redis

A very popular in-memory key-value database…

- Data is stored via **keys and values**, i.e. a key value database can be viewed as a kind of **dictionary**
- Values in Redis can be strings, lists, sets, ...
- Redis is an **in-memory database**, i.e. everything is stored in RAM
- This fact makes **reading and writing extremely fast**
- However, **only limited amount of data** can be stored in a Redis database
- To **persist storage and reduce risk**, data can be **written to disk** frequently

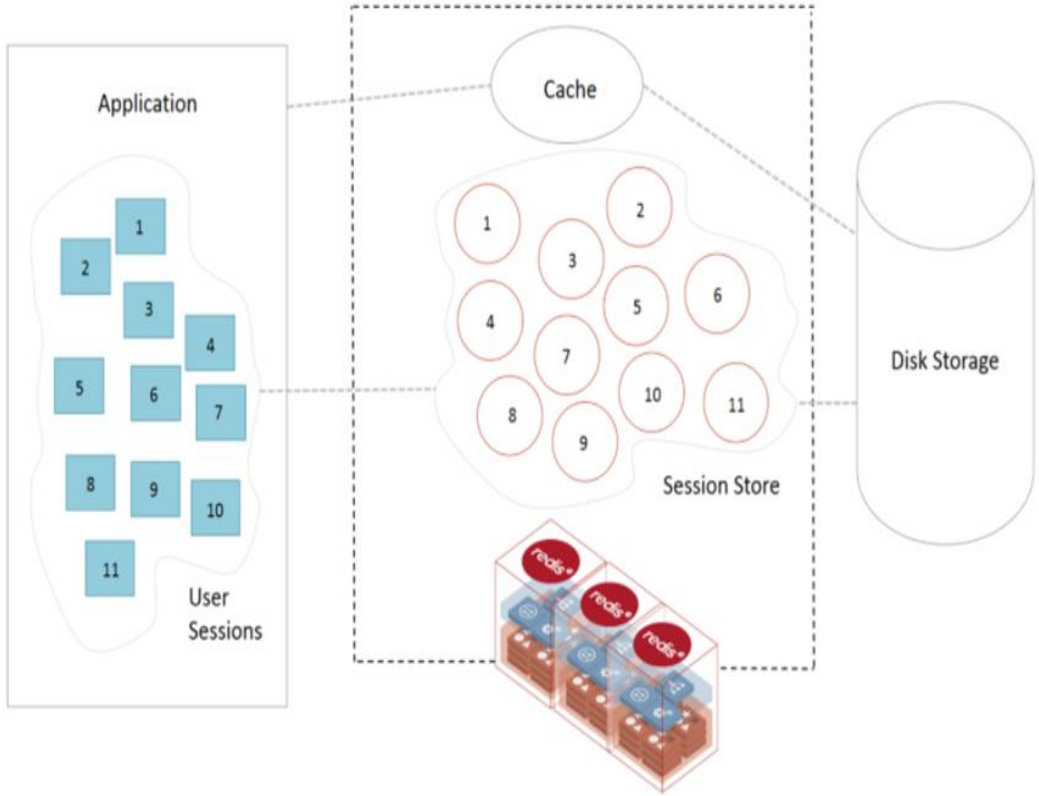
Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623

BPP

Redis

An important use case for caching and session store…

- **Cache** stores non-user- specific data to serve future requests faster
- **Session store** stores **user-specific data**
- **Redis** can be used for **both use cases**
- **Session store** can be written to **disk storage** frequently

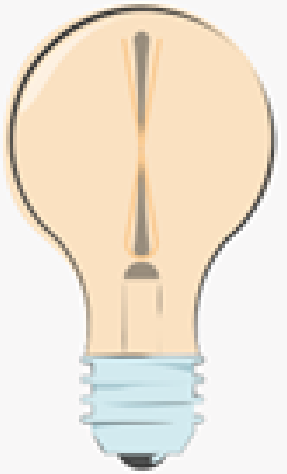


Caching and session store in Redis

Redis

When should you use it…?

- You deal with applications that require **very fast read and write**
- **Data structure** is **quite simple** and can be stored as key-value pair
- **Integrity** is important but **not critical**
- As a data engineer, you might want to use it when building a **dashboard** or an **API**



Potential drawbacks of noSQL solutions

Examples and drawbacks…

Potential drawbacks

- **Merges** between different tables are **difficult** to perform
- Therefore, data is **not stored** in a normalised way (also called denormalisation)
- This requires **quite some space** as e.g. attribute names and values are replicated
- Having **no schema** makes it sometimes **hard to understand** the data
- It is generally less efficient for repetitive tasks like reporting

Examples

mongoDB

elasticsearch

Amazon DocumentDB

BPP

Knowledge check poll

Which feature of document databases makes them flexible for working with unstructured data?

A. Tabular storage

B. Lack of schema

C. SQL storage

D. Complex relationships

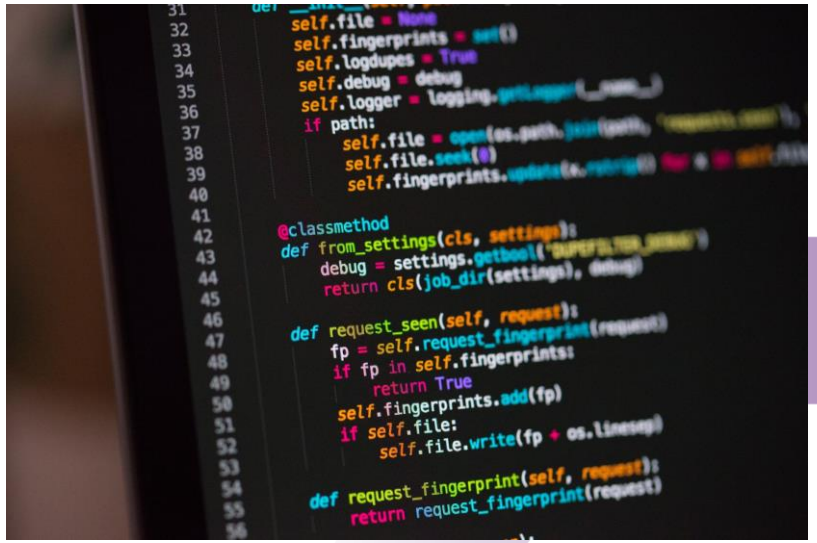
Submit your responses to the chat!

Feedback

The correct answer is: **B** - The lack of schemas in document databases provides flexibility for unstructured data.

BPP

Types and applications of NoSQL databases

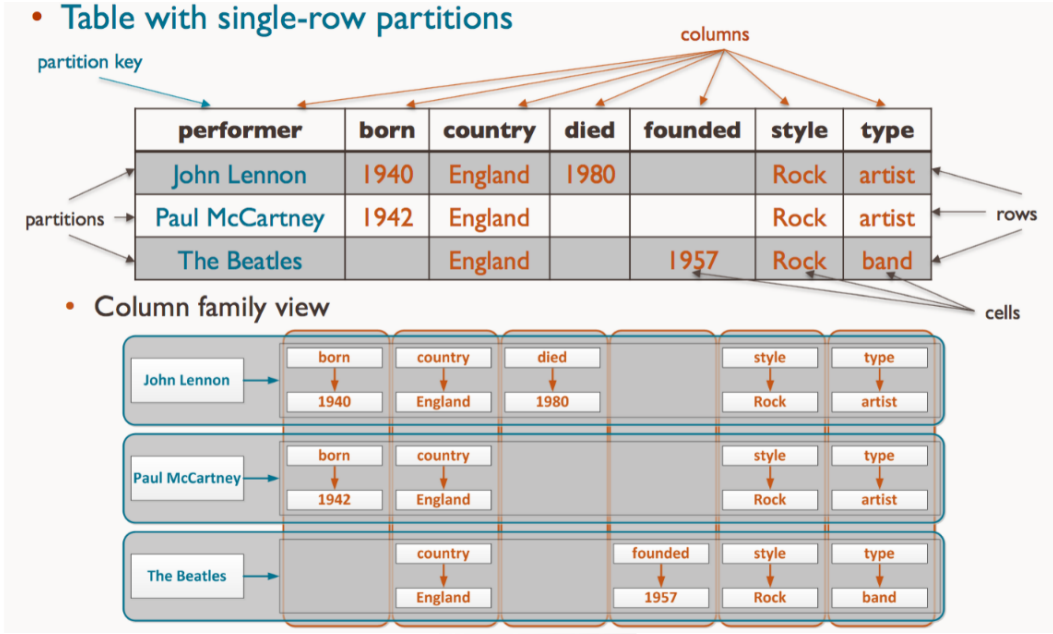


BPP

Column-based NoSQL databases

Wide-column stores…

- Similar to RDBMs because they store data in tables (rows and columns)
    - but differ from relational database because each row is not required to have the same columns
    - particularly suited for rapid collection of data and at high volumes
- Eg Cassandra and HBase.



Example wide-column store database

Industry applications of Column-store NoSQL

Types and use cases…

Spotify uses **Cassandra** to store user profile attributes and metadata about artists, songs, etc. for better personalisation

Facebook initially built its revamped Messages on top of **HBase**, but is now also used for other Facebook services like the Nearby Friends feature and search indexing

Outbrain uses **Cassandra** to serve over 190 billion personalized content recommendations each month

Column-based database

When you should use them…

- **Optimised for data warehousing:** Ideal for managing and querying large volumes of data efficiently, particularly in analytics and reporting.
- **Real-time big data analytics:** Facilitates rapid analysis and reporting, crucial for applications like financial trading and web analytics.
- **Scalability and flexibility:** Excellently suited for scalable, cloud-based applications with variable workloads due to efficient data distribution capabilities.
- **Efficient sparse data management:** Highly effective in handling datasets with many empty or null fields, common in telecommunications and IoT.

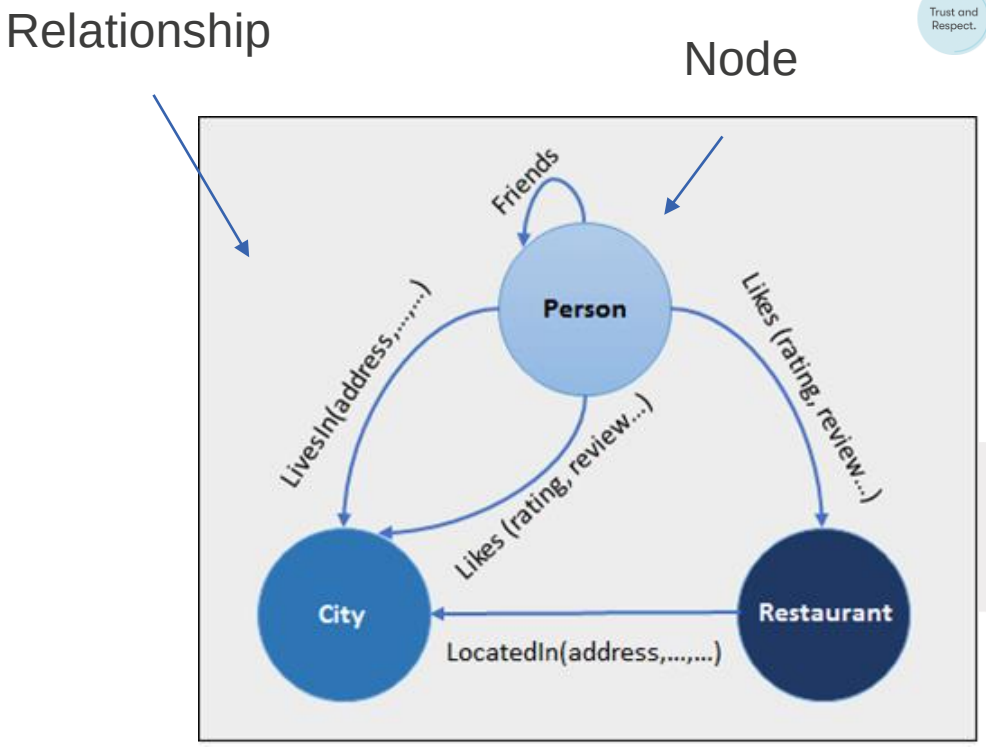
Column-based databases are optimised for data warehousing

BPP

Column-based NoSQL databases

Wide-column stores…

- Finding **all the friends of my friends** is quite a **complex task** in a normalised RDBMS
- This is a typical **graph problem**
- Graph database store **nodes and relationships**
- Easy and **fast graph traversal**
- Very **promising approach** as a lot of data is actually relationships between entities



Example wide-column store database

Industry use cases of Graph-based databases

Types and use cases…

Walmart

Walmart uses **Neo4j** to provide customers personalized, relevant product recommendations and promotions

Meta

Medium uses **Neo4j** to build their social graph to enhance content personalisation

outbrain

Cisco uses **Neo4j** to mine customer support cases to anticipate bugs

BPP

Knowledge check poll

Which NoSQL database type is optimised for handling large volumes of data and analytic workloads?

A. Document database

B. Graph database

C. Column-based database

D. Key-value store

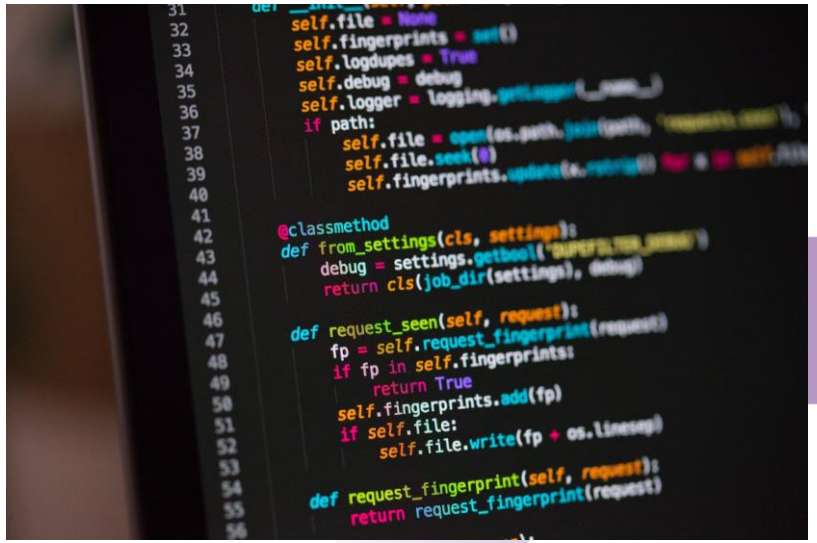
Feedback

The correct answer is: **C** – Column-based databases like Cassandra are optimised for handling large data volumes and analytic workloads efficiently.

Submit your responses to the chat!

BPP

A gentle intro to data pipelines

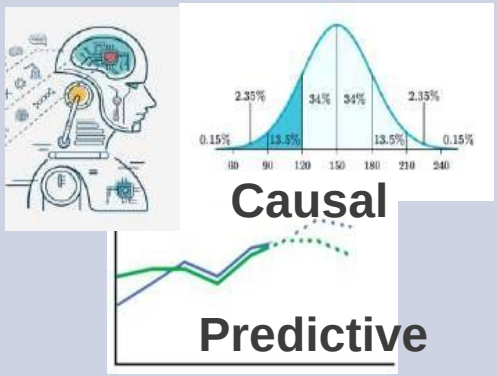


BPP

What are data pipelines?

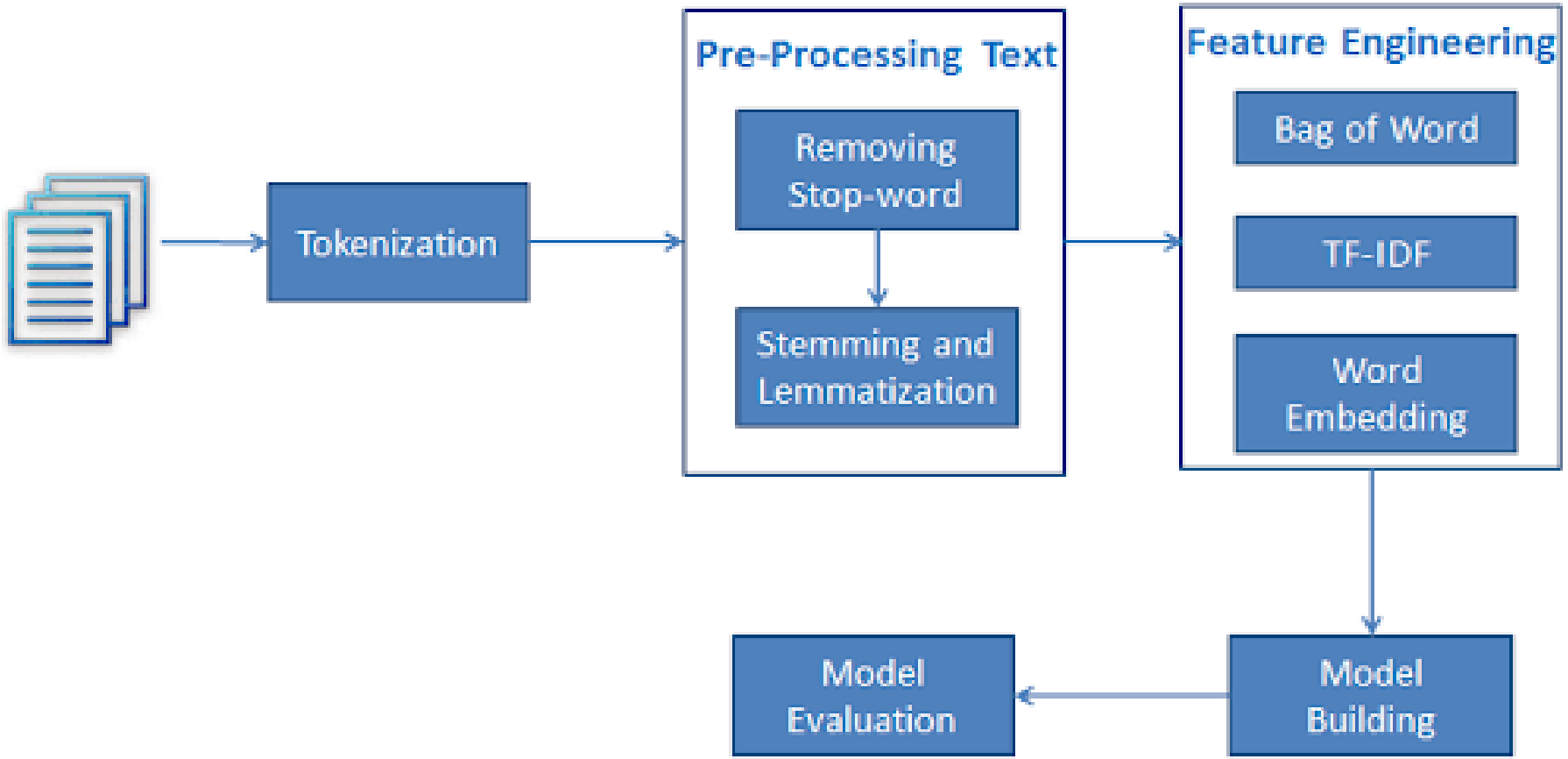
What you need to familiar with…

- Structured and unstructured data is created and stored to extract business value
- In the context of unstructured data, we must design a pipeline from extraction to value creation

Extract	Process/Transform	Analyse/Model	Present
			Descriptive Exploratory 

NLP Pipeline

An example…



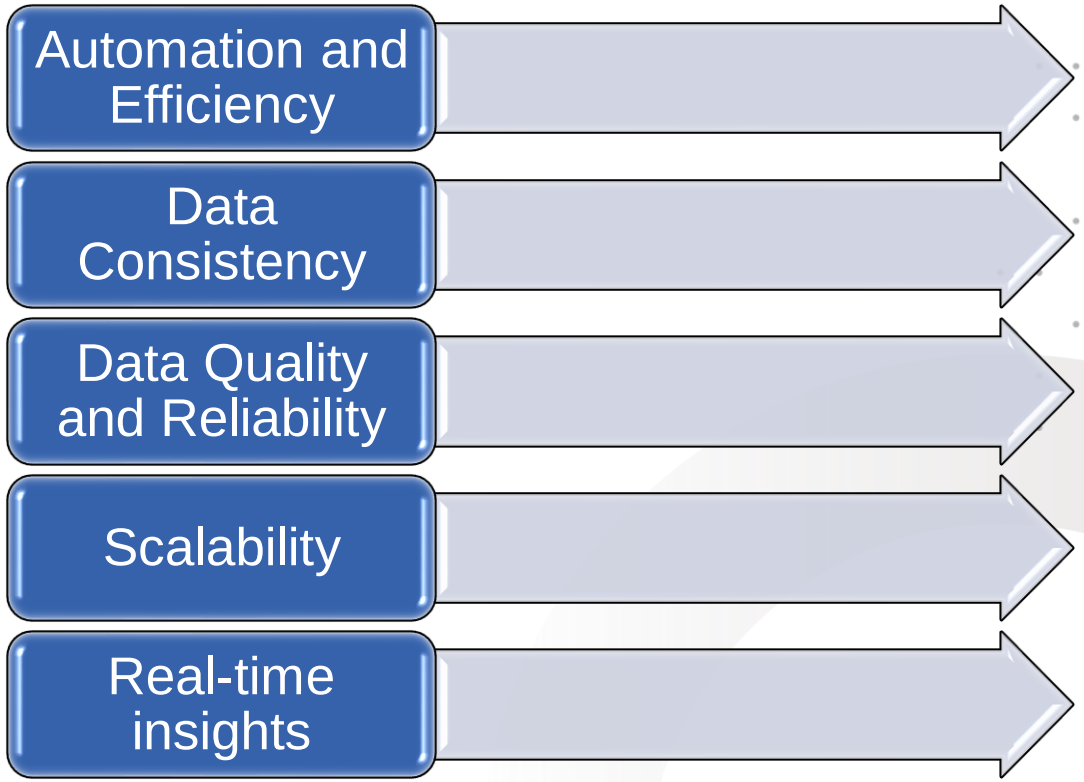
The architecture of an example NLP pipeline

Column-based NoSQL databases

Wide-column stores…

Pipelines play a pivotal role in automating and streamlining data movement and transformation processes.

The benefits are as follows:

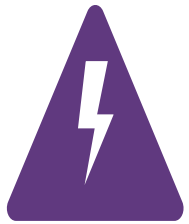


The benefits of data pipelines

Building high-quality pipelines

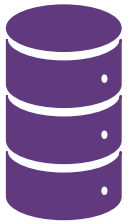
Key features…

A high-quality pipeline is resilient, idempotent, and scalable, qualities that are critical for reliable data operations.



Resiliency features to handle routine failures.

Idempotency ensures consistent output regardless of reruns.



Designed for scalability to handle large datasets.

Allows for high frequency of invocations.

BPP

Knowledge check poll

Which of the following is NOT a key characteristic of a high-quality data pipeline?

A. Resiliency

B. Idempotency

C. Scalability

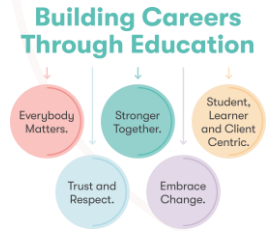
D. Complexity

Submit your responses to the chat!

Feedback

The correct answer is: **D** – High-quality data pipelines should aim to be resilient, idempotent, and scalable.

Scenario discussion



Imagine the following:

A social media company needs to select a NoSQL database type to store data about users, posts, follows/connections etc.

Discuss which NoSQL database type would be most appropriate for this use case and why?

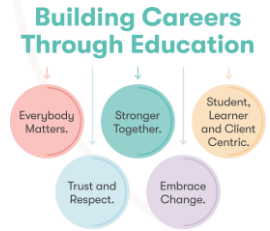
Key considerations:

- Need to store data on users, posts, and complex connections/relationships between users
- Relationships are at the core - who is connected to who, which users interact with the same posts, etc
- Data structure evolving rapidly - new post types emerging, new ways users interact
- Large volume of data that needs to scale as user base grows
- Need for low latency queries to serve content in user feeds

Pair and share discussion

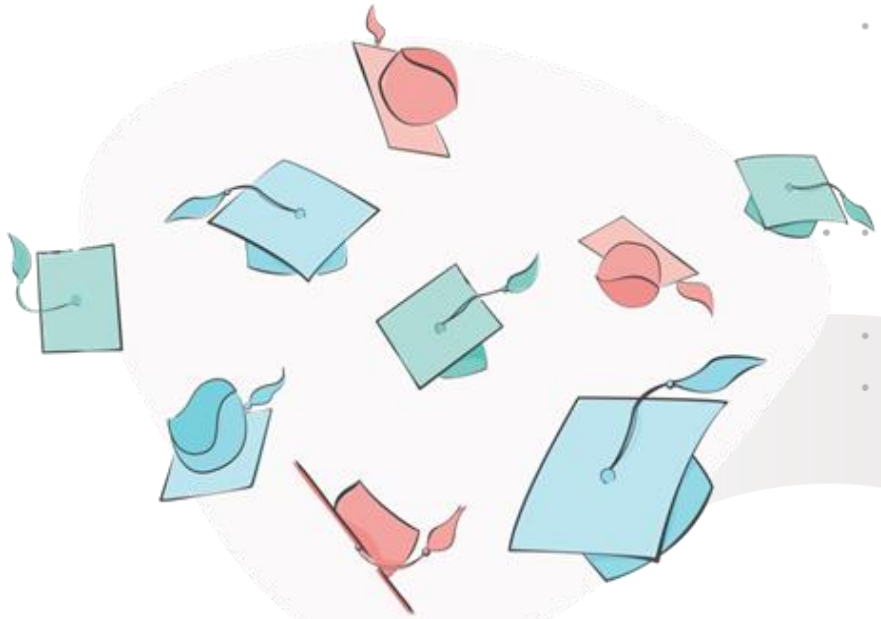
BPP

Key learning summary



The key takeaways from this lesson are as follows:

- NoSQL databases provide flexibility, scalability and performance for modern big data needs
- Each NoSQL database type serves specific use cases like documents, graphs, analytics
- Data pipelines automate and scale data flows from source to destination
- Choosing the right technologies is crucial for managing unstructured data



BPP

Thank you

Do you have any questions, comments, or feedback?

How confident do you now feel about your knowledge of programming and scripting essentials following this webinar?

- **A:** Very confident
- **B:** More confident than before this webinar
- **C:** What was this webinar session even about?!

Submit your responses to the chat!