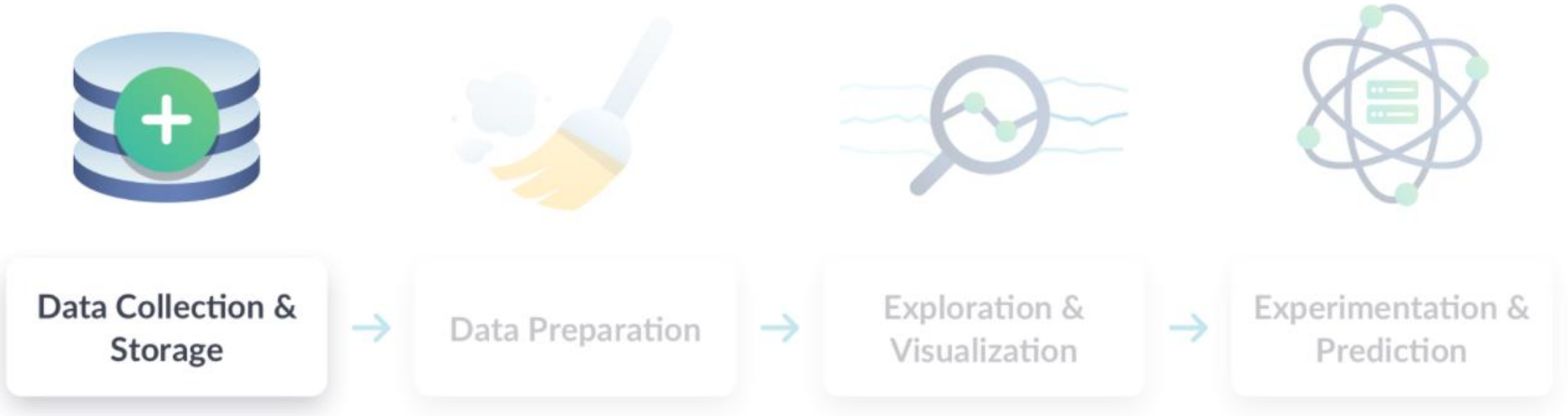
Data Collection, Storage and Retrieval

GGE5405/GGE6505 Introduction to Big Data & Data Science

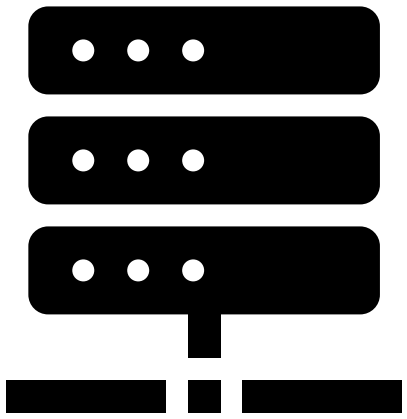
Winter 2022

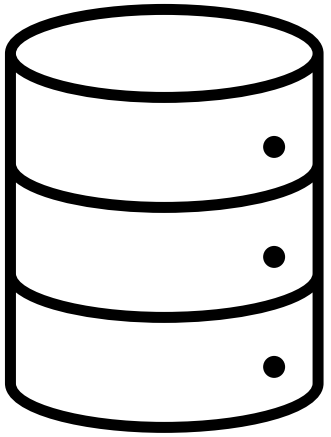
Data Science Workflow

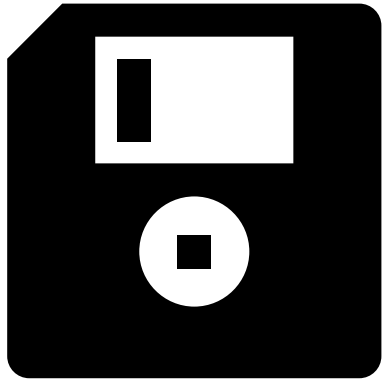


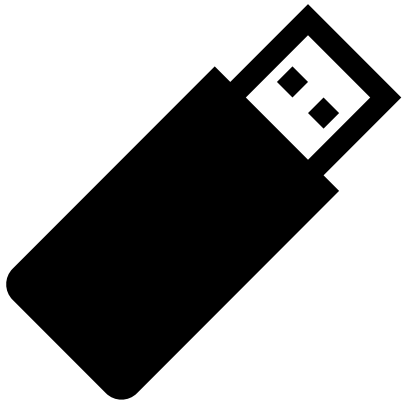
Outline

- When we are talking about data storage, we should pay attention to!
  - Data Sources and Data types
  - Data Storage
  - Data Retrieval









1.1 Data Source

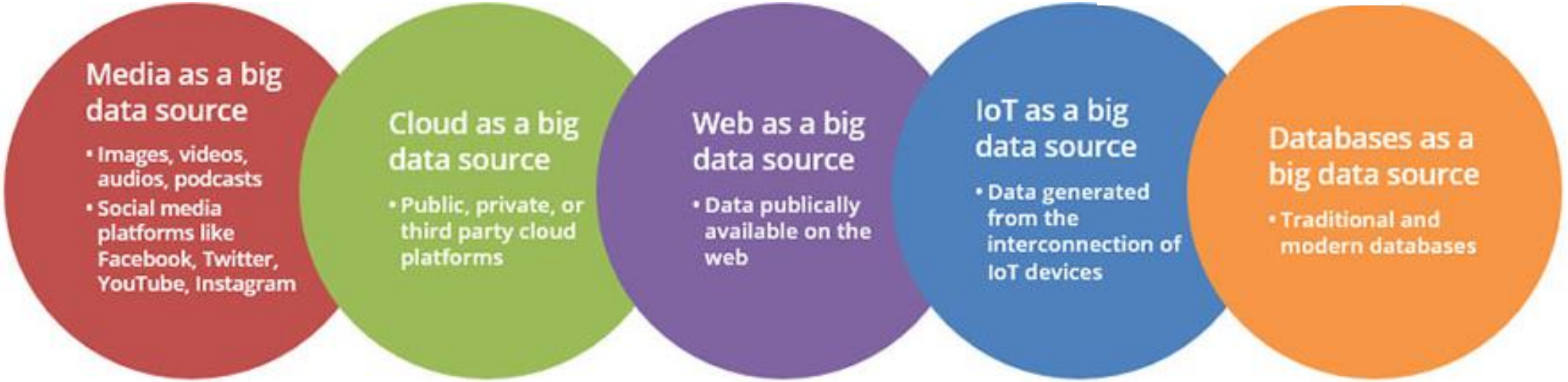
Data Sources

Company data

- Collected by companies
- Helps them make data-driven decisions
  - Web events
  - Survey data
  - Customer data

Open data

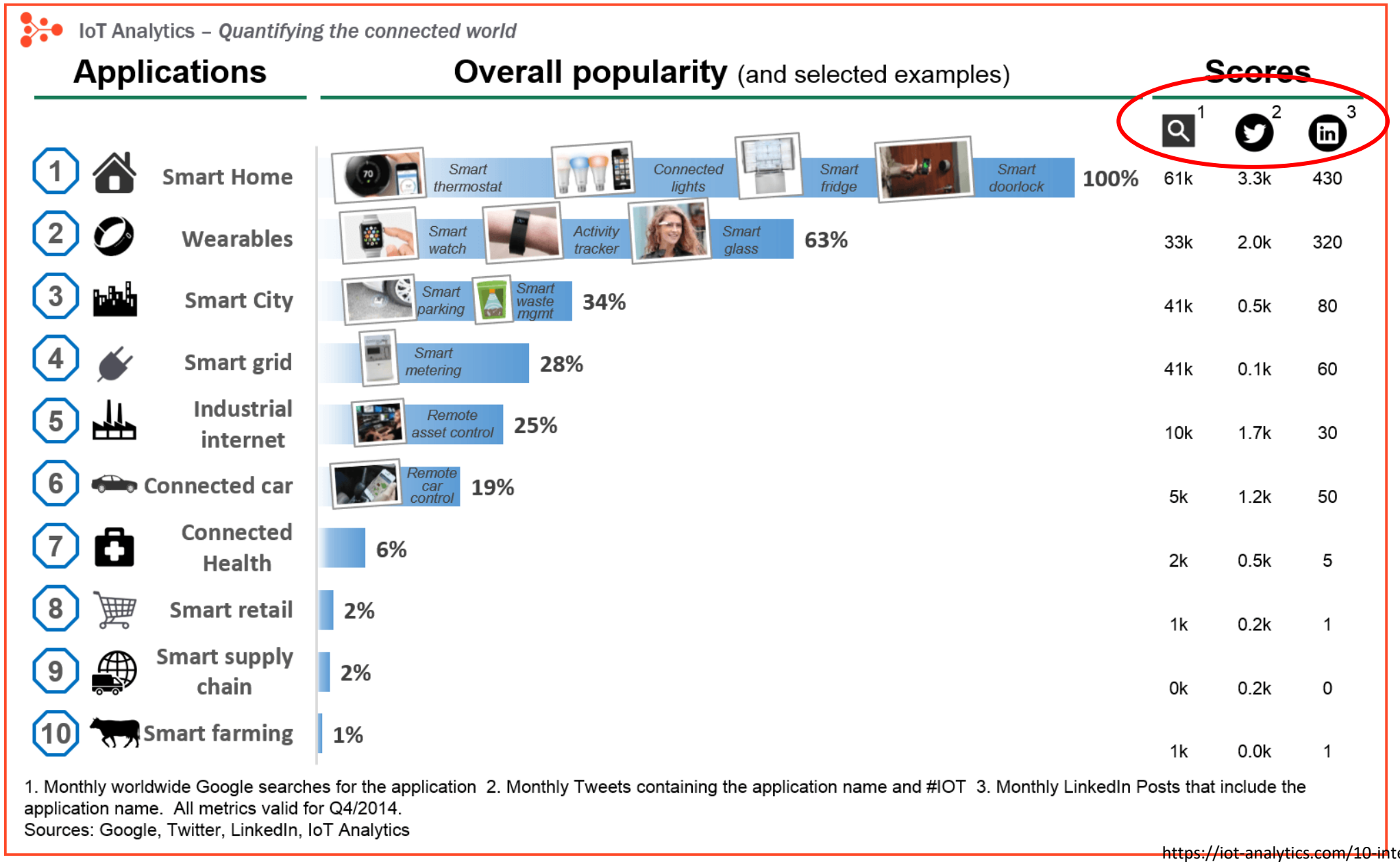
- Free, open data sources
- Can be used, shared, and built-on by anyone
  - Data APIs
  - Public records



The Internet of Things

The Internet of Things (IoT) in general refers to everything connected to the Internet but today it is increasingly used to define all “smart” devices embedded with sensors, software, and other technologies (watches, fridges, cars, etc) that "talk" to each other.
Such devices create a network of physical objects ("things") that are connecting and exchanging data with other devices and systems via the Internet.

The Internet of Things Applications

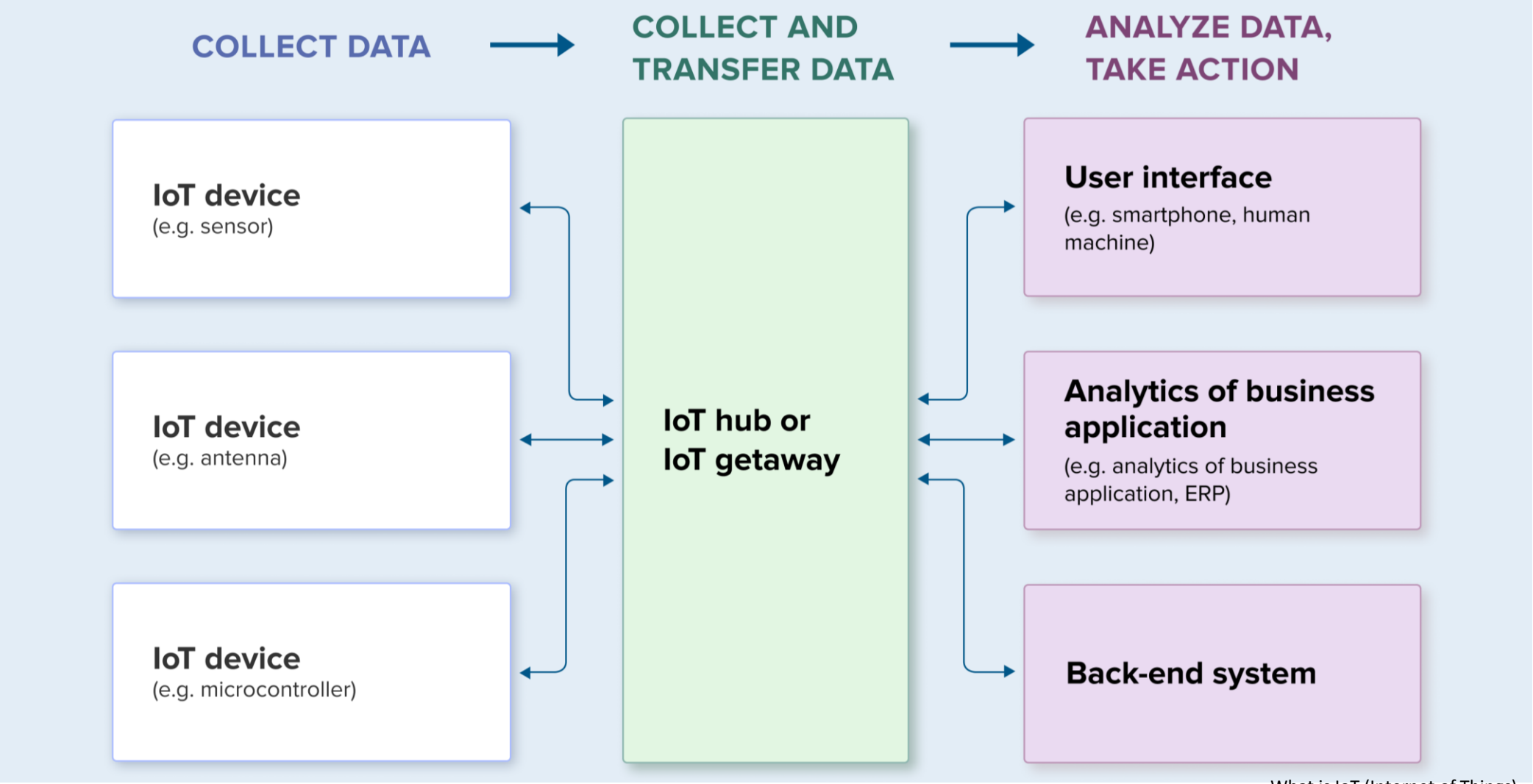


https://iot-analytics.com/10-internet-of-things-applications/

Key aspects when considering IoT

- **Range:** Are you deploying to a single office floor or an entire city?
- **Data Rate:** How much bandwidth do you require? How often does your data change?
- **Power:** Is you sensor running on mains or battery?
- **Security:** Will your sensors be supporting mission critical applications?
- **Privacy:** Are there privacy issues?

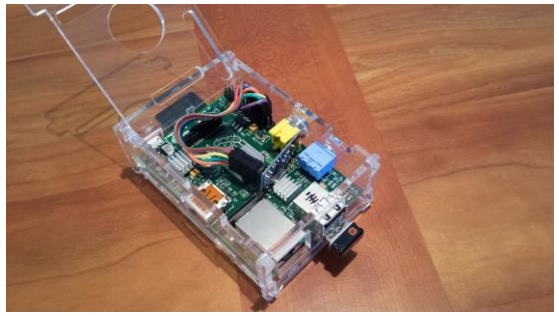
How it Works?



What is IoT (Internet of Things) - Explained by SumatoSoft

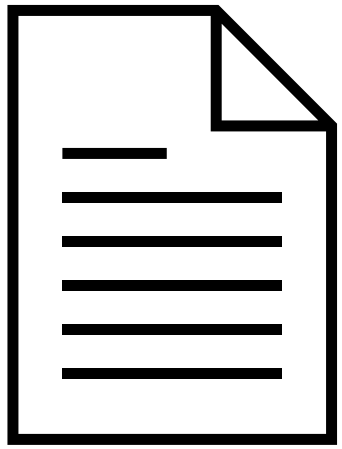
Gateways and Edge Note

Raspberry Pi Gateway



Smart Spot Gateway

- A gateway is a network element that acts as an entrance point to another network.
- Gateways are simple routers between two network architectures.
- It links two networks which have different communication protocols, different language and different architecture.



1.2 Data Type

Why data type is important?

- Important later on when:
  - Storing the data
  - Visualizing/analyzing the data

	A	B	C	D	E	F	G	H
1	Date	Time	Count	Status	Sensor	Type	Position	Location
2	4/29/2019	8:35:00	1	0	UP	Counter A	4	H2-L3
3	4/29/2019	9:00:00	1	0	UP	Counter A	4	H2-L3
4	4/29/2019	9:30:59	2	0	UP	Counter A	4	H2-L3
5	4/29/2019	9:34:59	1	0	UP	Counter A	4	H2-L3
6	4/29/2019	9:41:59	2	0	UP	Counter A	4	H2-L3
7	4/29/2019	9:47:59	1	16	UP	Counter A	4	H2-L3
8	4/29/2019	9:51:59	1	0	UP	Counter A	4	H2-L3
9	4/29/2019	9:56:59	1	0	UP	Counter A	4	H2-L3
10	4/29/2019	10:02:59	1	0	UP	Counter A	4	H2-L3
11	4/29/2019	10:13:59	1	0	UP	Counter A	4	H2-L3
12	4/29/2019	10:17:59	2	0	UP	Counter A	4	H2-L3
13	4/29/2019	10:18:59	1	0	UP	Counter A	4	H2-L3
14	4/29/2019	10:20:59	1	0	UP	Counter A	4	H2-L3

Single Observation

Variable/ Attribute

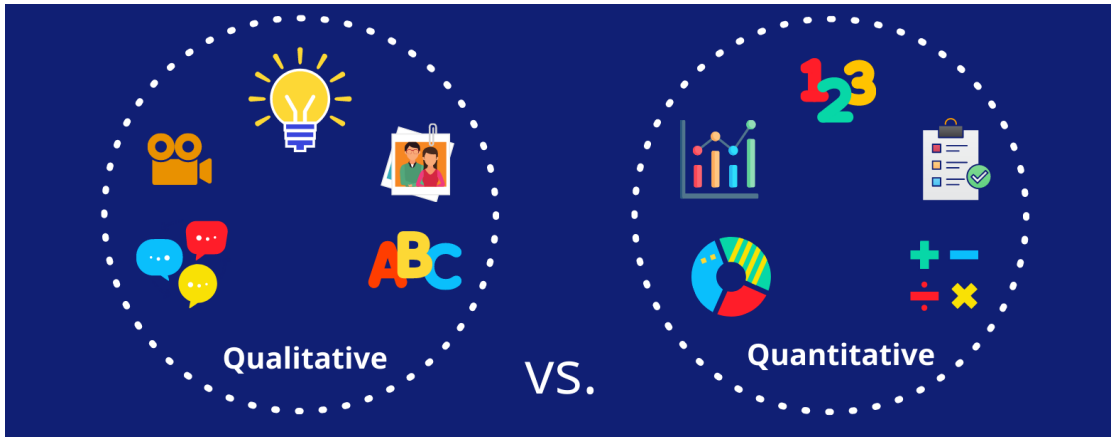
Data Types

Quantitative data

- Deals with numbers
- Data can be measured

Qualitative data

- Deals with descriptions
- Data can be observed but not measured



Qualitative and Quantitative data collection methods in M&E - TolaData

Quantitative vs Qualitative Data

Quantitative Data
(Numerical)

- Age
- Height
- Weight
- Income
- University size
- Group size
- Self-efficacy test score
- Percent of lecture attended
- Clinical skills performed
- Number of errors

Qualitative Data
(Categorical)

- Gender
- Religion
- Marital status
- Native language
- Social class
- Qualifications
- Type of instruction
- Method of treatment
- Type of teaching approach
- Problem-solving strategy used

Quantitative and Qualitative – what's the difference?

Types of Data

Quantitative

Data that can be measured with numbers, such as duration or speed

Discrete

Whole numbers that can't be broken down, such as a number of items

Continuous

Numbers that can be broken down, such as height or weight

Interval

Numbers with known differences between variables, such as time

Ratio

Numbers that have measurable intervals where difference can be determined, such as height or weight

Qualitative

Non-numerical data that is categorical, such as yes/no responses or eye colour

Nominal

Data used for naming variables, such as hair colour

Ordinal

Data used to describe the order of values, such as 1 = happy, 2 = neutral, 3 = unhappy

Types of Data and the Scales of Measurement | Quality Digest

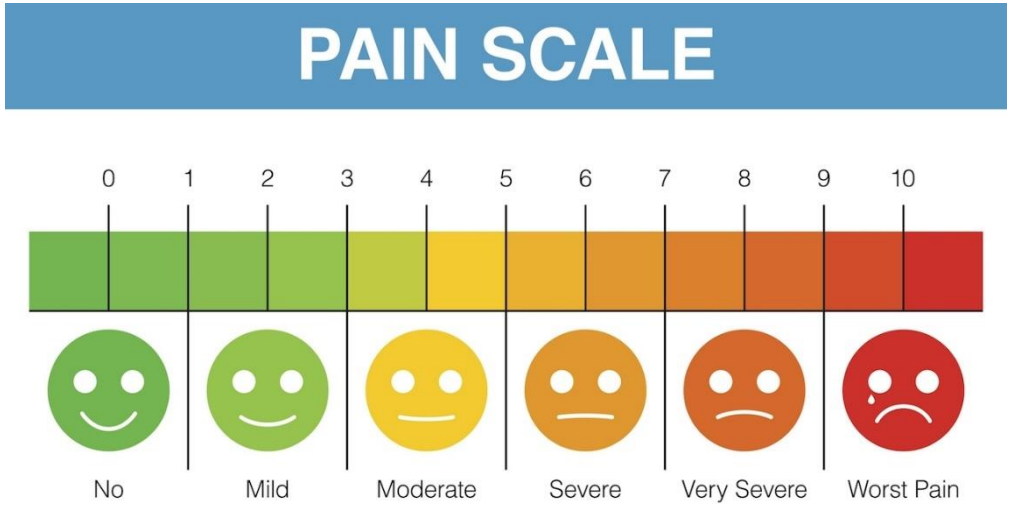
Quantitative Data: Ratio Scales

- The scale has a true, meaningful zero point rather than with respect to, for example, the mean.
- Only scale that allows to make ratio comparisons.

Can a map scale be a ratio scale?

Quantitative Data: Interval Scales

- In which both the order relationship and the concatenation of differences between objects is reflected in the relationships between measurements.



A patient providing information on a discrete (interval) scale: From 0 to 10 how severe your pain is? 0 is no pain, and 10 is the worst pain imaginable?

Qualitative Data: Nominal Scales

- In which the only empirical measurement is that objects have different values of the variable.
- Lowest level of measurement.
- Some data scientists regard this scale to be too limited to be dignified with the term measurement.

Qualitative Data: Ordinal Scales

- In which the only relationship between objects represented by the relationship between measurements is their order relationship.
- It makes no sense to add two numbers together since there is no meaningful notion of concatenating objects to create a harder object.

Douglas Sea Scale Degree	Height (m)	Description
0	No wave	Calm (Glassy)
1	0 – 0.1	Calm (Rippled)
2	0.1 – 0.5	Smooth
3	0.5 – 1.25	Slight
4	1.25 – 2.5	Moderate
5	2.5 – 4	Rough
6	4 – 6	Very Rough
7	6 – 9	High
8	9 - 14	Very High
9	14 +	Phenomenal

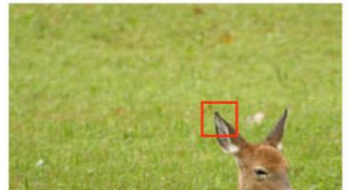
World Meteorological Organization

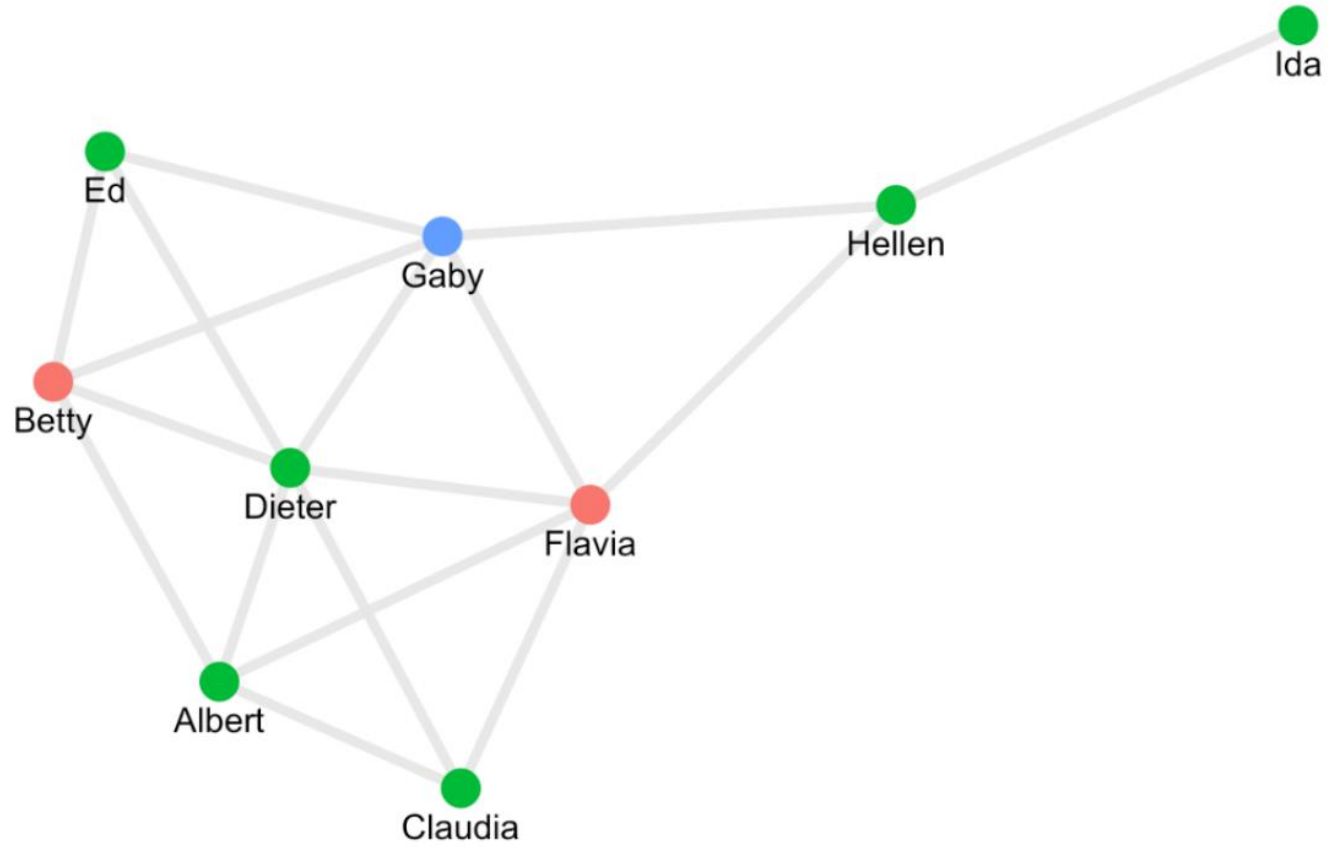
The order is meaningful, not the actual values.

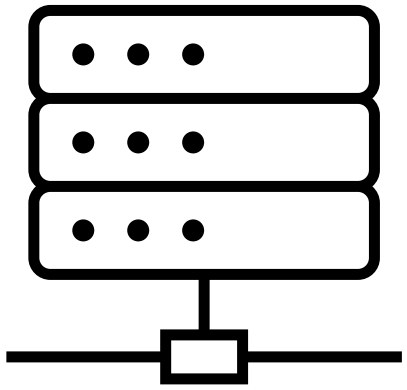
The gender of students in STEM fields	Qualitative Data
The maximum speed of formula 1 cars during the 2008 season	Quantitative Data
The individual weight of all the dogs in a shelter	Quantitative Data
The eye color of people participating in a study	Qualitative Data
The reviews for a property on Airbnb	Qualitative Data
Images of several cats	Qualitative Data
The daily average temperature in New York during 2019	Quantitative Data
The price of a cup of coffee in Parisian cafés	Quantitative Data

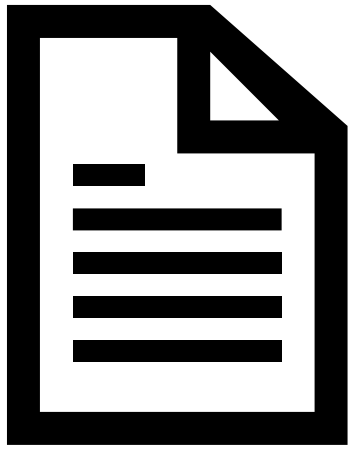
Other Data Types

- Image data
- Text data
- Geospatial data
- Network data
- ...

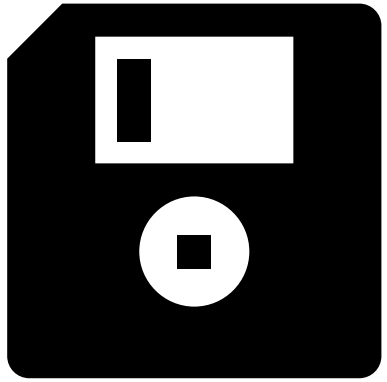


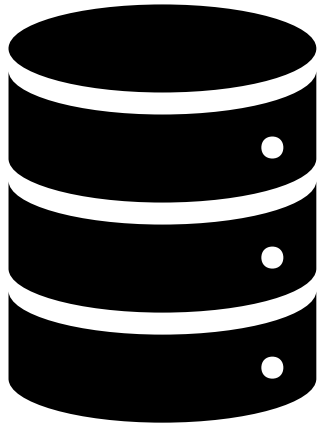






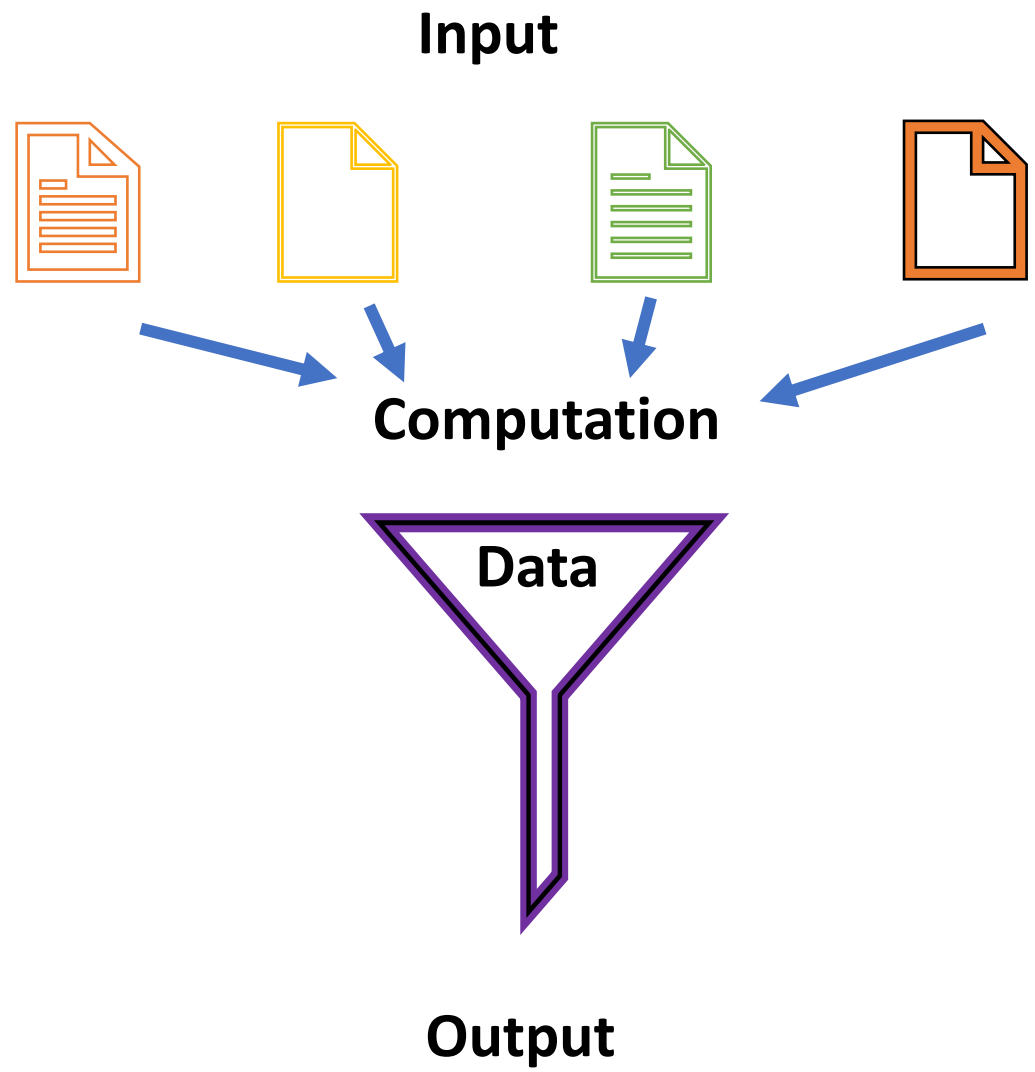
2. Data Storage





Data Systems

- Information systems (DBMS)
- Data analysis systems



Example of Data Systems

- Query 1: Is there an employee named “Joe”?
- Query 2: What is “Joe’s” salary?
- Query 3: How many departments are there in the company?
- Query 4: What is the name of “Joe’s” department?
- Query 5: How many employees are there in the “HR” department?

Employee

ID	Name	Dept ID	Salary	…
1000	Joe	123	110K	…
1001	Alice	55	80K	…
1045	Richard	204	140K	

Department

ID	Name	…
55	HR	…
123	Marketing	
204	IT	…

What is the need for a Database?

- The database systems that we use in our day-to-day life should be able to perform basic CRUD operations, i.e., the create, retrieve, update, and delete operations

- The various reasons a database is important are:
  - **Access** is about making data available to users.
  - **Integrity**: makes sure that the data is accurate and consistent
  - **Update:** easy to update data using various Data Manipulation languages
  - **Security:** Databases have various methods to ensure security of data

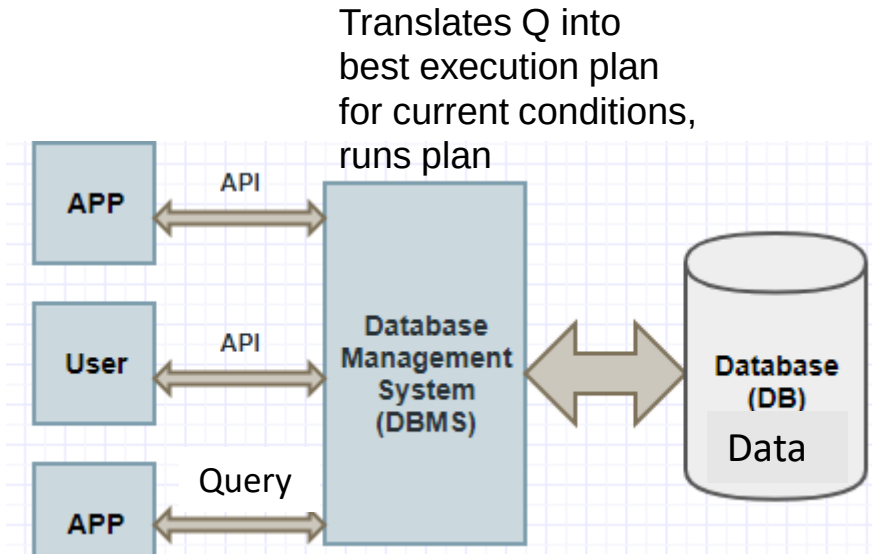
Database Management System (DBMS)

- DBMS is essentially nothing more than a computerized data-keeping system.

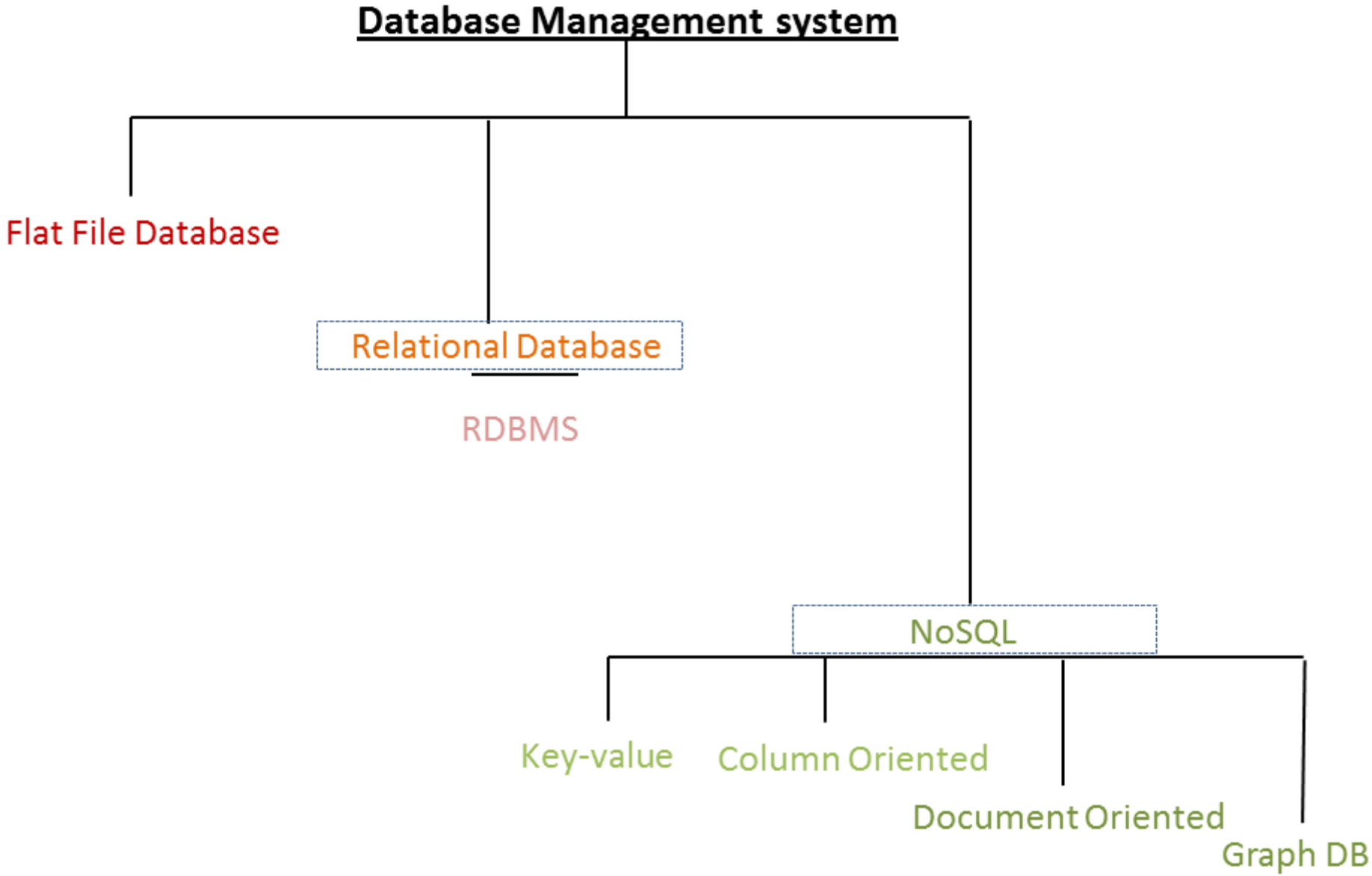
- Database vs. DBMS:

  - Database is collection of data related to particular subject or topic whereas DBMS is system or mechanism which allows us to create, organize and modify database and retrieve information

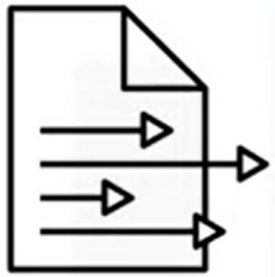
  - The example of database are dictionary, Telephone Directory, etc whereas The example of DBMS are Ms-Access , Oracle, MongoDB,FoxPro, MY SQL etc.



Keeps data safe and correct despite failures, concurrent updates, online processing, etc.


Database Management system
Flat File Database
Relational Database
RDBMS
NoSQL
Key-value
Column Oriented
Document Oriented
Graph DB

Flat File



Delimited text files
- CSV, TSV
- Store data in rows, items separated by a delimiter

Spreadsheets
- XLSX
- Store data in rows and columns

Language
- XML, JSON
- Transfer data
- Platform independent

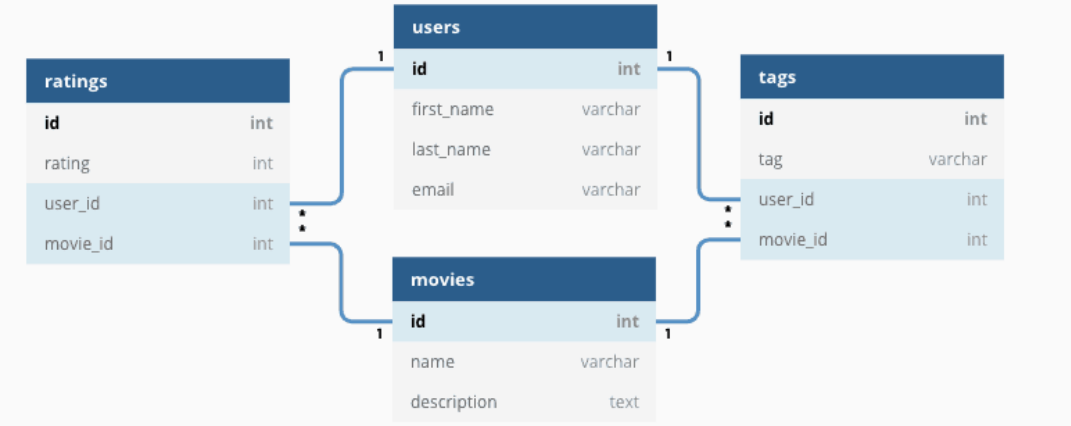
RDBMS or SQLDB VS. Non Relational or NoSQL

- **Data Model:** RDBMS databases are used for normalized structured (tabular) data strictly adhering to a relational schema. NoSQL datastores are used for non-relational data, e.g. key-value, document tree, graph.

- **Transaction Guarantees:** All RDBMS databases support **ACID** transactions, but most NoSQL datastores offer **BASE** transactions.

Relational Databases -RDBMS

- A relational database management system (RDBMS) is a system where data is organized in two-dimensional tables using rows and columns.
- This is one of the most popular data models which is used in industries. It is based on **SQL language**.
- Every table in a database has a key field which uniquely identifies each record.
- For example − Oracle Database, PostgreSQL, MySQL, Microsoft SQL Server etc.



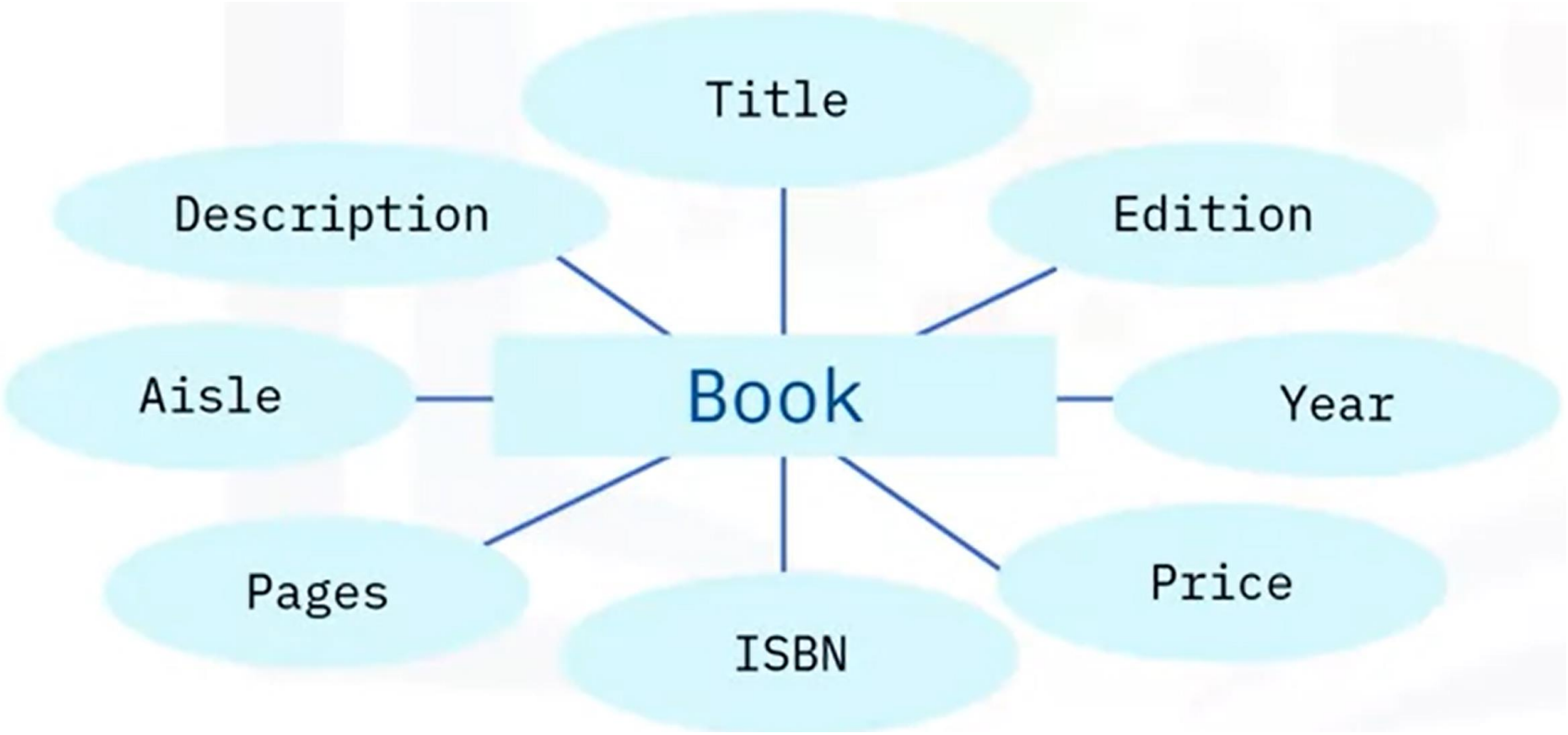
person

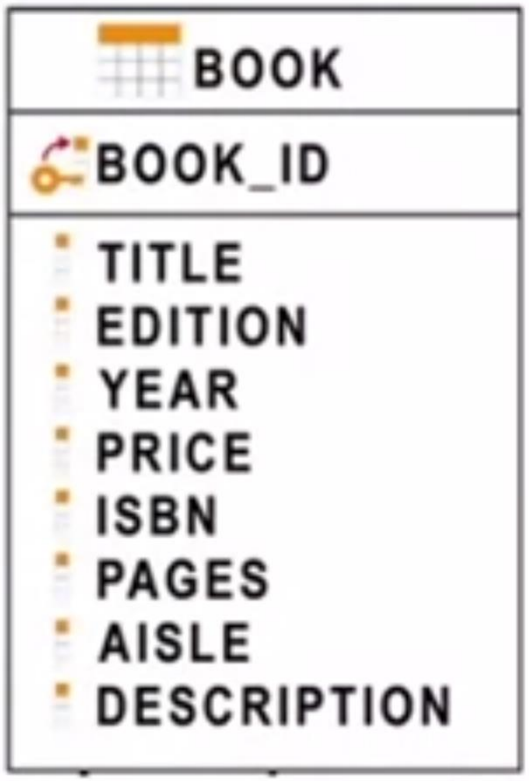
id	first_name	last_name	gender	age	car_id
1	Anne	Smith	FEMALE	44	
2	Jake	Jones	MALE	21	24
3	Andrew	Jacob	MALE	19	
4	Julia	Bravo	FEMALE	34	313

car

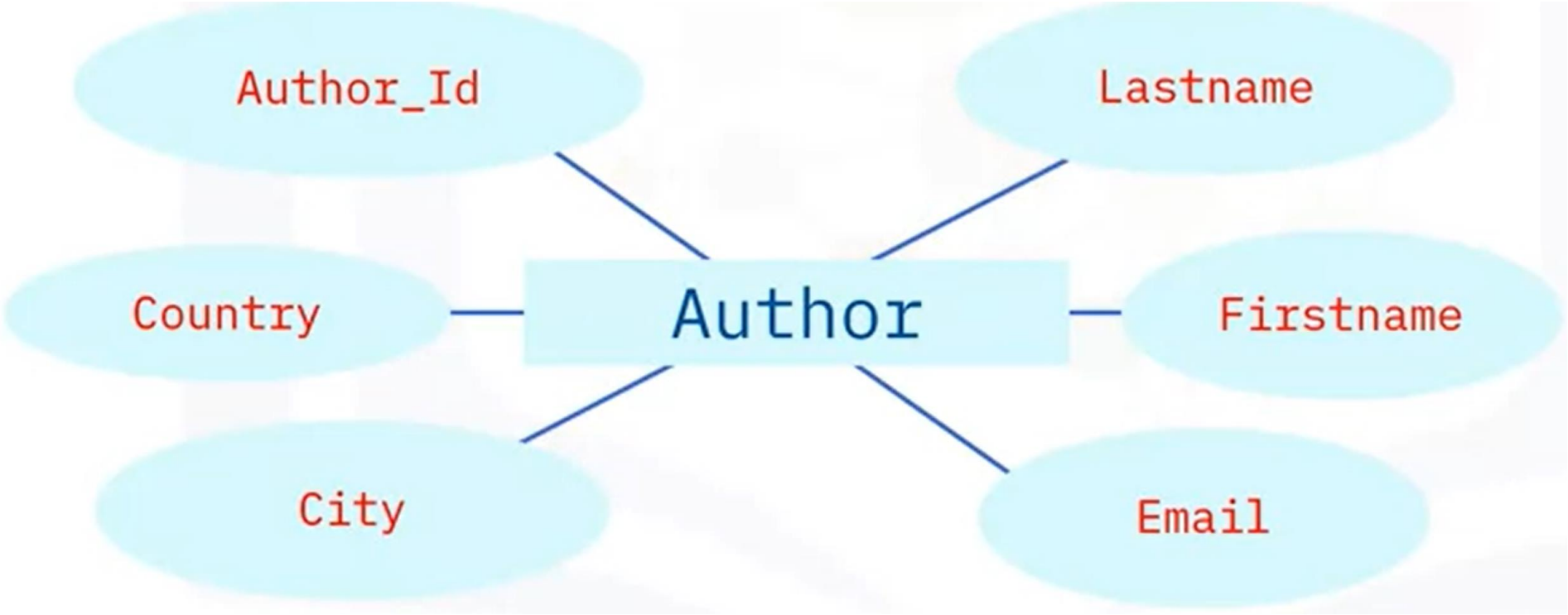
id	make	model	Price
24	Anne	Smith	44
313	Jake	Jones	21

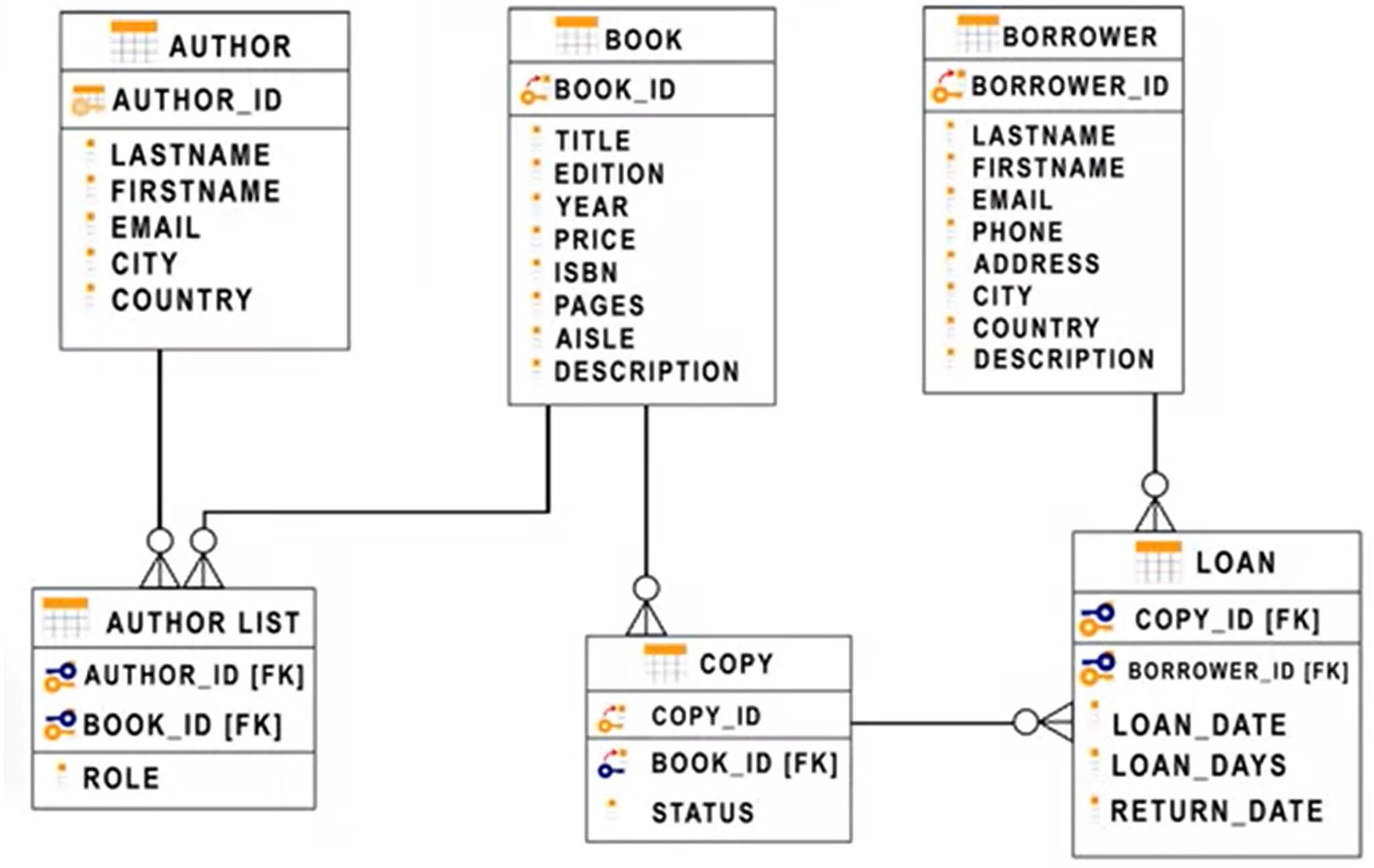
Library Management System





Library Management System




AUTHOR
AUTHOR_ID
LASTNAME
FIRSTNAME
EMAIL
CITY
COUNTRY
BOOK
BOOK_ID
TITLE
EDITION
YEAR
PRICE
ISBN
PAGES
AISLE
DESCRIPTION
BORROWER
BORROWER_ID
LASTNAME
FIRSTNAME
EMAIL
PHONE
ADDRESS
CITY
COUNTRY
DESCRIPTION
AUTHOR LIST
AUTHOR_ID [FK]
BOOK_ID [FK]
ROLE
COPY
COPY_ID
BOOK_ID [FK]
STATUS
LOAN
COPY_ID [FK]
BORROWER_ID [FK]
LOAN_DATE
LOAN_DAYS
RETURN_DATE

BOOK TABLE

Title	BookId	Year	Price	ISBN	Pages	Aisle	Description
Database Fundamentals	1	2010	24.99	978-0-98662 83-1-1	300	DB-A02	Teaches you the fundamentals of databases
Getting started with DB2 Express-C	2	2010	24.99	978-0-9866 283-5-1	280	DB-A01	Teaches you the essentials of DB2 using DB2 Express-C, the free version of DB2

Author TABLE

Author_ID	Lastname	Firstname	Email	City	Country
A1	Chong	Raul	rfc@ibm.com	Toronto	CA
A2	Ahuja	Rav	ra@ibm.com	Toronto	CA
A3	Hakes	Ian	ih@ibm.com	Toronto	CA
A4	Sharma	Neeraj	ns@ibm.com	Chennai	IN
A5	Perniu	Liviu	lp@univ.com	Transilvania	RO

PostgreSQL or Postgres

- a free and open-source relational database management system (RDBMS) emphasizing extensibility and SQL compliance

- tools for administering PostgreSQL include
  - **Psql:** The primary front-end for PostgreSQL is the psql command-line program, which can be used to enter SQL queries directly, or execute them from a file.
  - **pgAdmin:** The pgAdmin package is a free and open-source graphical user interface administration tool for PostgreSQL, which is supported on many computer platforms
  - **DataGrip and Python** also can be used.

PostgreSQL

NoSQL databases or Non-relational

Watch it: How do NoSQL databases work? Simply Explained! - YouTube

Product ID	name	Price
23	MacBook Pro 13"	$1032.21
87	USB Microphone	$78.21
23	Hand Sanitizer	$9.99

Key	Value
194252165973	{ name: "MacBook Pro 13", price: $1032.21, description: "laptop" }
42406659611	USB Microphone

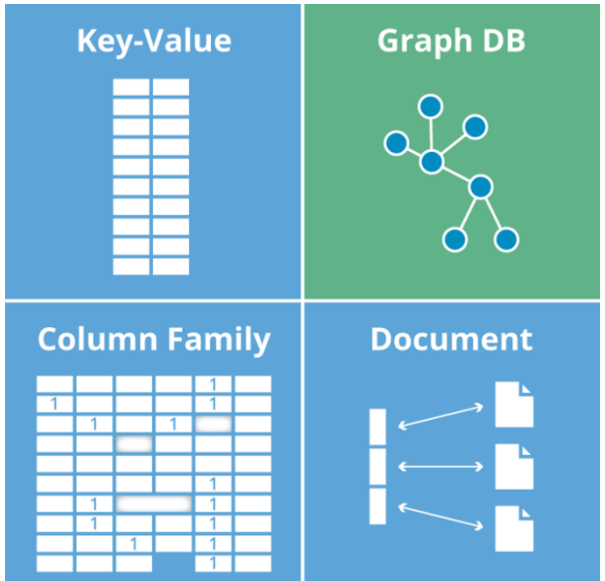
NoSQL Databases

Simply Explained

NoSQL databases

- Next Generation Databases mostly addressing some of the points: being **non-relational, distributed, open-source and horizontal scalable. The original intention has been modern web-scale databases.** The movement began early 2009 and is growing rapidly. **Often more characteristics apply as: schema-free, easy replication support, simple API, eventually consistent / BASE (not ACID), a huge data amount, and more.**

- The common structures adapted by NoSQL databases to store data are:
  - key-value
  - Column-based
  - Graph-based
  - Document-based



1- Key-value stores

- The data is stored as key- value pair and these databases are work on a simple data model that has a pair of unique key and a value. This is designed to handle heavy load of the data.
- In the key-value stores, key means a string of characters and the value is a series of uninterrupted bytes that are opaque to the database.
- The key-value stores have no query language. GET, PUT and DELETE commands are used to store, retrieve and update the data. This simple structure of this type makes key-value store fast, easy to access, scalable, portable and flexible.

- Examples:
  - Dynamo
  - Riak
  - Redis

Key	Value
Name	Joe Garden
Age	33
Job	Manager
Weight	75 kg

2- Column-based

- This type of database stores data in columns instead of rows. This type of database enables querying the large data sets faster than other conventional databases. Here the column don't have to consist across records and we can add a column to specific rows without having to add them to every single record.
- This type is commonly used for catalogs, manage data warehouses, Business intelligence and fraud detection. This database deliver the high performance on aggregation queries such as SUM, COUNT, AVG etc. as the data is readily available in a column.
- Example:
- HBase
- Cassandra
- Hypertable

Row-oriented

ID	Name	Grade	GPA
001	John	Senior	4.00
002	Karen	Freshman	3.67
003	Bill	Junior	3.33

Column-oriented

Name	ID
John	001
Karen	002
Bill	003

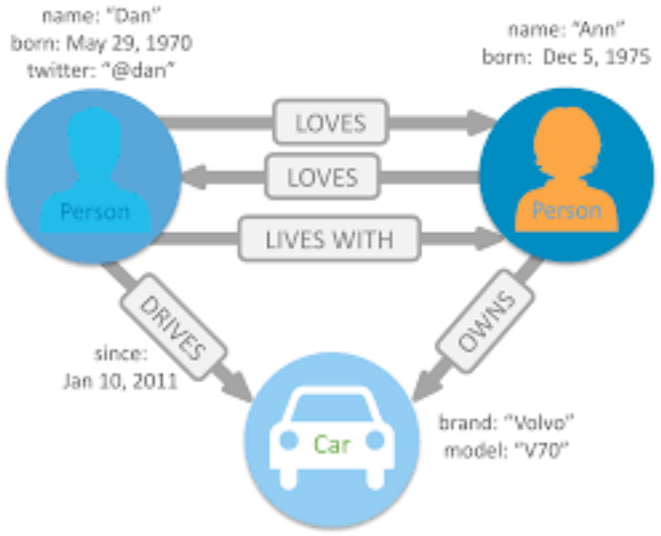
Grade	ID
Senior	001
Freshman	002
Junior	003

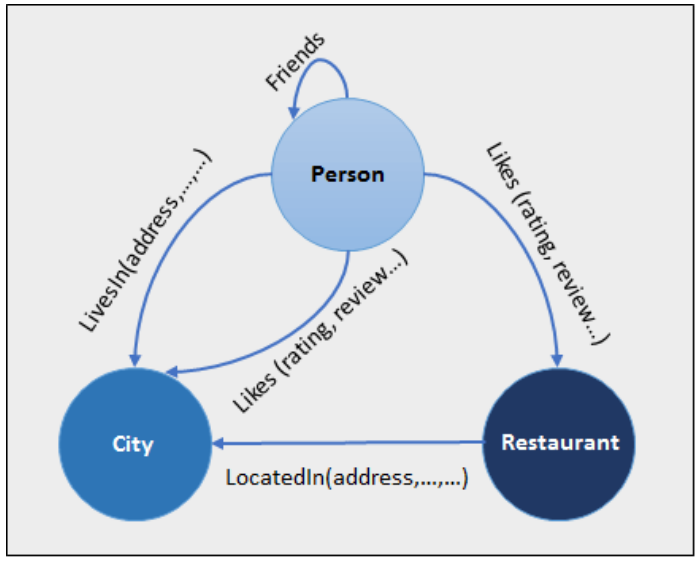
GPA	ID
4.00	001
3.67	002
3.33	003

Column Family			
Row Key	Column Name		
	Key	Key	Key
	Value	Value	Value
	Column Name		
	Key	Key	Key
	Value	Value	Value

3- Graph Based

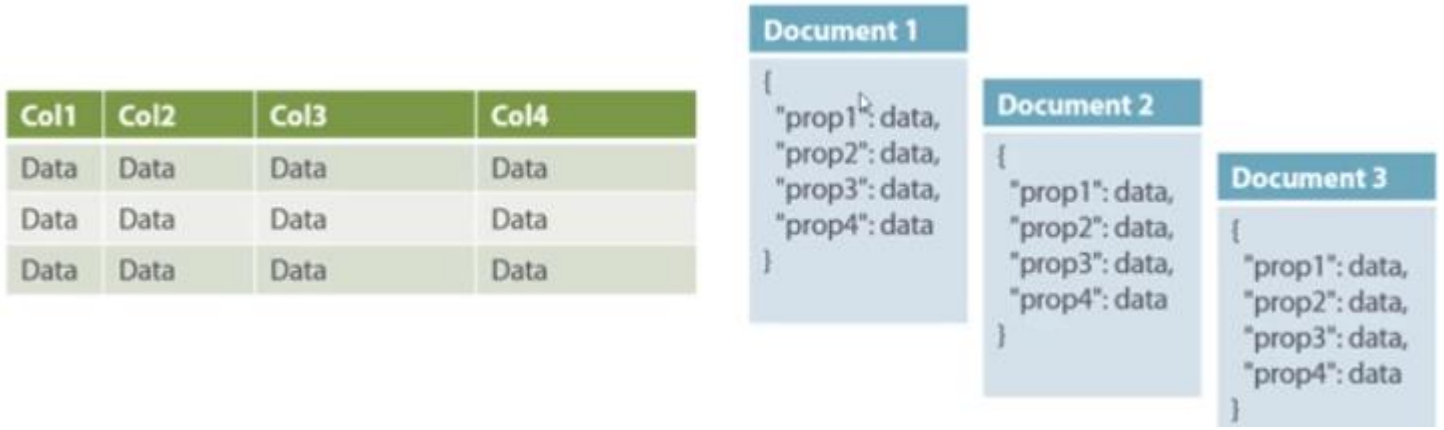
- This type is slightly different from other types, because here, the data is represented as a graph of entities and the relationship between the entities, which each node in the graph a free-form chunk of data. The edges represents the relationship between the nodes. Every edge and node has the unique identifier.
- When comparing to the relational or traditional database where tables are loosely connected, a graph database is considered as multi-relational in nature. So traversing relationship is fast. This type of databases are mostly used for social networks, logistics, spatial data.
- Examples :
- Neo4j
- OrientDB
- FlockDB





4- Document based

- Here, the data is stored as a key value pair but the value part is stored as a document or free-form JSON structures, where the data can be anything from integers to strings to free-form text.
- This model is fir with use cases such as catalogs, real-time analytics and e-commerce applications, content management system where each document is unique. This is not well fit for the complex transactions which have multiple operations or queries.
- Example:
- CouchDB
- MongoDB
- Riak



MongoDB

- MongoDB is an open-source cross-platform document-oriented database program.
- Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas to store structure and unstructured .
- MongoDB offers faster query processing but with an increased load and system requirements.

- Document-oriented. MongoDB stores data as documents that tend to have all data for a given record in a single document.

- Dynamic. Having a predefined schema or structure for your data is not necessary with MongoDB. You can create documents without first defining the data structure (e.g., fields, types of their values) and you can easily change the structure of documents by adding new fields or deleting existing ones.

```
C:\Program Files\MongoDB\Server\4.0\bin>mongod
2018-10-13T14:00:52.895+0200 I CONTROL  [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'
2018-10-13T14:00:53.392+0200 I CONTROL  [initandlisten] MongoDB starting : pid=4248 port=27017 dbpath=C:\data\db\ 64-bit host=DESKTOP-K5I5P0L
2018-10-13T14:00:53.401+0200 I CONTROL  [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2018-10-13T14:00:53.409+0200 I CONTROL  [initandlisten] db version v4.0.3
2018-10-13T14:00:53.415+0200 I CONTROL  [initandlisten] git version: 7ea530946fa7880364d88c8d8b6026bbc9ffa48c
2018-10-13T14:00:53.423+0200 I CONTROL  [initandlisten] allocator: tcmalloc
2018-10-13T14:00:53.435+0200 I CONTROL  [initandlisten] modules: none
2018-10-13T14:00:53.468+0200 I CONTROL  [initandlisten] build environment:
2018-10-13T14:00:53.472+0200 I CONTROL  [initandlisten]     distmod: 2008plus-ssl
2018-10-13T14:00:53.483+0200 I CONTROL  [initandlisten]     distarch: x86_64
2018-10-13T14:00:53.487+0200 I CONTROL  [initandlisten]     target_arch: x86_64
2018-10-13T14:00:53.494+0200 I CONTROL  [initandlisten] options: {}
2018-10-13T14:00:53.510+0200 I STORAGE  [initandlisten] exception in initAndListen: NonExistentPath: Data directory C:\data\db\ not found., terminating
2018-10-13T14:00:53.529+0200 I NETWORK  [initandlisten] shutdown: going to close listening sockets...
2018-10-13T14:00:53.539+0200 I CONTROL  [initandlisten] now exiting
2018-10-13T14:00:53.565+0200 I CONTROL  [initandlisten] shutting down with code:100
```

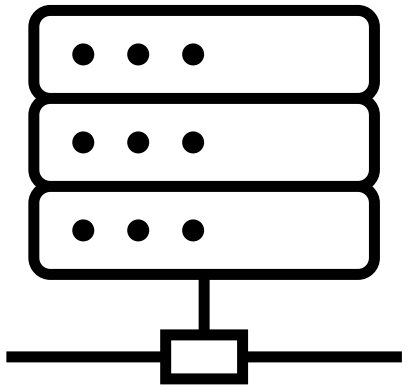
Advantages and Disadvantages of NoSQL

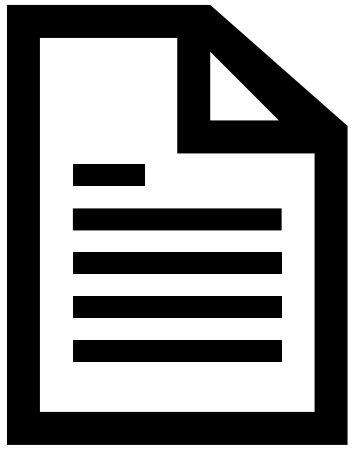
- Advantages
- No Single point of failure
- Big data capability
- Simple to implement when comparing with relational database.
- Provides fast performance
- Horizontal scalability.
- Suitable for structured, unstructured and semi-structured data.
- Elastic scalability
- Disadvantages
- Less community support
- No advance expertise
- Limited query capabilities
- No standard rules.

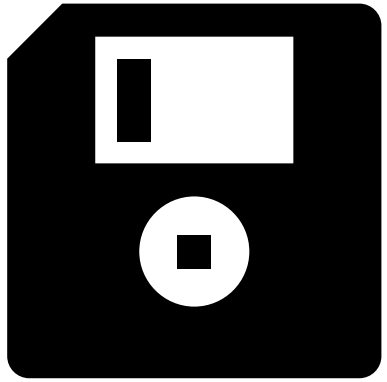
Difference between SQL and NoSQL

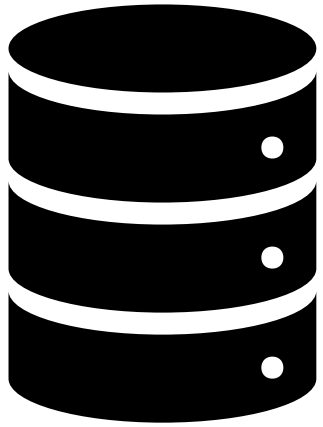
- SQL refers to the Structured Query Language. This is a relational database that stores data into tables and SQL created an interface to interact with it. NoSQL database do not follow the rules of relational database. It does not store data in a traditional tables and also it does not use SQL to query the data.
- SQL database have a pre defined schema but NoSQL use dynamic schema for unstructured data.
- SQL database is vertically scalable while NoSQL database is horizontally scalable. SQL databases is scaled by increasing the horse power of the hardware. It means by increasing CPU, RAM the scalability of SQL is increased. while NoSQL is scaled by increase the database servers.
- SQL uses structured query language to defining and manipulating the data in the database. But in NoSQL database, queries are focused on collection of documents. It is also called as Unstructured query language.
- SQL database is most suitable for the complex queries. And NoSQL is not fir for the complex queries. because NoSQL don't have interfaces to perform the complex queries and also the queries of NoSQL are not much powerful as SQL queries.
- NoSQL database is the best choice for storing the huge data set that is big data. SQL performance becomes slow when it stores the big data.

	Relational databases (RDBMS)	Non-relational databases (non-RDBMS)
Data	Structured data is stored in tables	Unstructured data is stored according to different models (key-value, document-oriented, graph, wide-column store, and multi-model)
Schema	Supports rigid (pre-defined) data schema	Supports dynamic data schema
Scalability	Vertically scalable	Horizontally scalable
Language	Structured query language	Unstructured query language
Transaction	ACID-compliant (atomicity, consistency, isolation, durability)	CAP theorem (consistency, availability and partition tolerance), may be ACID-compliant
Best for	Complex queries, database transactions and routine data analysis	Storing and modeling structured, semi-structured and unstructured data
Examples	Amazon Redshift, Azure Synapse Analytics, Microsoft SQL Server, Oracle Database, MySQL, IBM DB2, etc.	Amazon DynamoDB, Azure Cosmos DB, Amazon Keyspaces, Amazon DocumentDB, Oracle NoSQL database, etc.









3. Data Retrieval

1- Structured Query Language or SQL

- Catalog
  - A set of schemas that constitute the description of a database
- Schema
  - The structure that contains descriptions of objects created by a user (base tables, views, constraints)
- Data Definition Language (DDL)
  - Commands that define a database, including creating, altering, and dropping tables and establishing constraints
- Data Manipulation Language (DML)
  - Commands that maintain and query a database such as insert, update, delete.
- Data Control Language (DCL)
  - Commands that control a database, including administering privileges and committing data

Basic Data Types

- **Numeric** data types
  - Integer numbers: `INT`, `INTEGER`, `SMALLINT`, `BIGINT`
  - Floating-point (real) numbers: `REAL`, `DOUBLE` , `FLOAT`
  - Fixed-point numbers: `DECIMAL(`*n*`,`*m*`)`, `DEC(`*n*`,`*m*`)`, `NUMERIC(`*n*`,`*m*`)`, `NUM(`*n*`,`*m*`)`
- **Character-string** data types
  - Fixed length: `CHAR(`*n*`)`, `CHARACTER(`*n*`)`
  - Varying length: `VARCHAR(`*n*`)`, `CHAR VARYING(`*n*`)`, `CHARACTER VARYING(`*n*`)`, `LONG VARCHAR`
- **Large object** data types
  - Characters: `CLOB`, `CHAR LARGE OBJECT` , `CHARACTER LARGE OBJECT`
  - Bits: `BLOB`, `BINARY LARGE OBJECT`
- **Boolean** data type
  - Values of `TRUE` or `FALSE` or `NULL`
- `DATE` data type
  - Ten positions
  - Components are `YEAR`, `MONTH`, and `DAY` in the form YYYY-MM-DD

Create Tables

STEP 01

```
CREATE TABLE tableName (
    column1 data_type,
    column2 data_type,
    ....);
```

Insert Data

STEP 02

```
INSERT INTO table
(column1, column2, ...)
VALUES (value1, value2...);
```

Select Data

STEP 03

```
SELECT * FROM table_name
WHERE condition;
```

Aggregate Data

STEP 04

```
SELECT SUM(column)
FROM tablename
```

Join Tables

STEP 05

```
SELECT table1.*, table2.*
FROM table1
LEFT JOIN table2
ON table1.column1 =
table2.column2
```

Create Tables

- Specify a new relation
  - Provide name
  - Specify attributes and initial constraints

Base tables (base relations)

- Relation and its tuples are physically stored and managed by DBMS

CREATE TABLE COMPANY.EMPLOYEE ...

or

CREATE TABLE EMPLOYEE ...

Include information for each column (attribute) plus constraints

- Column name
- Column type (domain)
- Key, uniqueness, and null constraints

Specifying Key Constraints

- **`PRIMARY KEY`** clause
  - Specifies one or more attributes that make up the primary key of a relation

```
Dnumber INT NOT NULL PRIMARY KEY,
```

  - Primary key attributes must be declared `NOT NULL`
- **`UNIQUE`** clause
  - Specifies alternate (candidate) keys

```
Dname VARCHAR(15) UNIQUE;
```

  - May or may not allow null values, depending on declaration

- If no key constraints, two or more tuples may be identical in *all* columns.
  - SQL deviates from pure relational model!
  - Multiset (bag) behaviour

Basic SQL Retrieval Queries

- All retrievals use `SELECT` statement:

```
SELECT  <return list>
FROM    <table list>
[ WHERE   <condition> ] ;
```

where

`<return list>` is a list of expressions or column names whose values are to be retrieved by the query

`<table list>` is a list of relation names required to process the query

`<condition>` is a Boolean expression that identifies the tuples to be retrieved by the query

- Example

```
SELECT title, year, genre
FROM Film
WHERE director = 'Steven Spielberg' AND year > 1990;
```

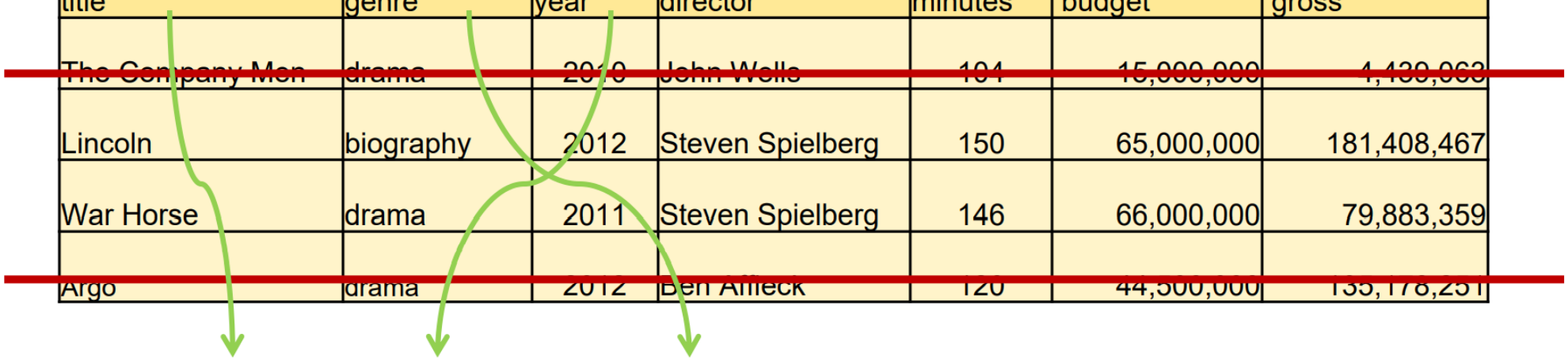
- Omitting WHERE clause implies all tuples selected.

1. Start with the relation named in the `FROM` clause
2. Consider each tuple one after the other, eliminating those that do not satisfy the `WHERE` clause.
   - Boolean condition that must be *true* for any retrieved tuple
   - Logical comparison operators

     =, <, <=, >, >=, and <>
3. For each remaining tuple, create a return tuple with columns for each expression (column name) in the `SELECT` clause.
   - Use `SELECT *` to select all columns.

Film

title	genre	year	director	minutes	budget	gross
~~The Company Men~~	~~drama~~	~~2010~~	~~John Wells~~	~~104~~	~~15,000,000~~	~~4,439,063~~
Lincoln	biography	2012	Steven Spielberg	150	65,000,000	181,408,467
War Horse	drama	2011	Steven Spielberg	146	66,000,000	79,883,359
~~Argo~~	~~drama~~	~~2012~~	~~Ben Affleck~~	~~120~~	~~44,500,000~~	~~135,178,251~~



SELECT-FROM-WHERE Semantics

- What if there are several relations in the `FROM` clause?

1. Start with cross-product of all relation(s) listed in the `FROM` clause.
   - Every tuple in R_1 paired up with every tuple in R_2 paired up with …
2. Consider each tuple one after the other, eliminating those that do not satisfy the `WHERE` clause.
3. For each remaining tuple, create a return tuple with columns for each expression (column name) in the `SELECT` clause.

Steps 2 and 3 are just the same as before.

```
SELECT actor, birth, movie
FROM Role, Person
WHERE actor = name and birth > 1940;
```

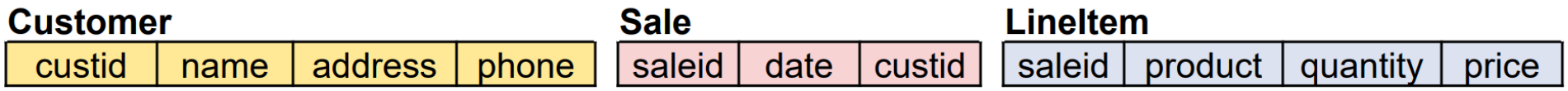
Role

actor	movie	persona
Ben Affleck	Argo	Tony Mendez
Alan Arkin	Argo	Lester Siegel
Ben Affleck	The Company Men	Bobby Walker
Tommy Lee Jones	The Company Men	Gene McClary

Person

name	birth	city
Ben Affleck	1972	Berkeley
Alan Arkin	1934	New York
Tommy Lee Jones	1946	San Saba

- Same name may be used for two (or more) columns (in different relations)
  - Must **qualify** the column name with the relation name to prevent ambiguity




```
SELECT name, date, product, quantity
FROM Customer, Sale, LineItem
WHERE price > 100 AND Customer.custid = Sale.custid AND
      Sale.saleid = LineItem.saleid;
```

- Note
  - If SELECT clause includes custid, it must specify whether to use Customer.custid or Sale.custid *even though the values are guaranteed to be identical*. ☹

```
SELECT award, actor, persona, Role.movie
FROM Honours, Role
WHERE category = 'actor' AND winner = actor
      AND Honours.movie = Role.movie
```

Honours

movie	award	category	winner
Lincoln	Critic's Choice	actor	Daniel Day-Lewis
Argo	Critic's Choice	director	Ben Affleck
Lincoln	SAG	supporting actor	Tommy Lee Jones
Lincoln	Critic's Choice	screenplay	Tony Kushner
War Horse	BMI Flim	music	John Williams

Role

actor	movie	persona
Ben Affleck	Argo	Tony Mendez
Tommy Lee Jones	Lincoln	Thaddeus Stevens
Daniel Day-Lewis	The Boxer	Danny Flynn
Daniel Day-Lewis	Lincoln	Abraham Lincoln

Honours.movie	award	category	winner	actor	Role.movie	persona	
Lincoln	Critic's Choice	actor	Daniel Day-Lewis	Ben Affleck	Argo	Tony Mendez	✗
Lincoln	Critic's Choice	actor	Daniel Day-Lewis	Tommy Lee Jones	Lincoln	Thaddeus Stevens	✗
Lincoln	Critic's Choice	actor	Daniel Day-Lewis	Daniel Day-Lewis	The Boxer	Danny Flynn	✗
Lincoln	Critic's Choice	actor	Daniel Day-Lewis	Daniel Day-Lewis	Lincoln	Abraham Lincoln	☑
Argo	Critic's Choice	director	Ben Affleck	Ben Affleck	Argo	Tony Mendez	✗
Argo	Critic's Choice	director	Ben Affleck	Tommy Lee Jones	Lincoln	Thaddeus Stevens	✗
Argo	Critic's Choice	director	Ben Affleck	Daniel Day-Lewis	The Boxer	Danny Flynn	✗
Argo	Critic's Choice	director	Ben Affleck	Daniel Day-Lewis	Lincoln	Abraham Lincoln	✗
Lincoln	SAG	supporting actor	Tommy Lee Jones	Ben Affleck	Argo	Tony Mendez	✗
Lincoln	SAG	supporting actor	Tommy Lee Jones	Tommy Lee Jones	Lincoln	Thaddeus Stevens	✗
Lincoln	SAG	supporting actor	Tommy Lee Jones	Daniel Day-Lewis	The Boxer	Danny Flynn	✗
...							

PostgreSQL and Python

- **Psycopg** is the most popular PostgreSQL database adapter for the Python programming language.
- Its main features are the complete implementation of the Python DB API 2.0 specification and the thread safety (several threads can share the same connection).
- It was designed for heavily multi-threaded applications that create and destroy lots of cursors and make a large number of concurrent “INSERT”s or “UPDATE”s.

2- Access Data in MongoDB

- MongoDB Query Language (MQL): MongoDB queries are based on JavaScript.
- BSON(Binary JSON): is a binary serialization of JSON-like documents.
- Works on the concept of Collection and Documents
  - A collection in MongoDB is similar to table in SQL
  - A Document is similar to row (record) in SQL

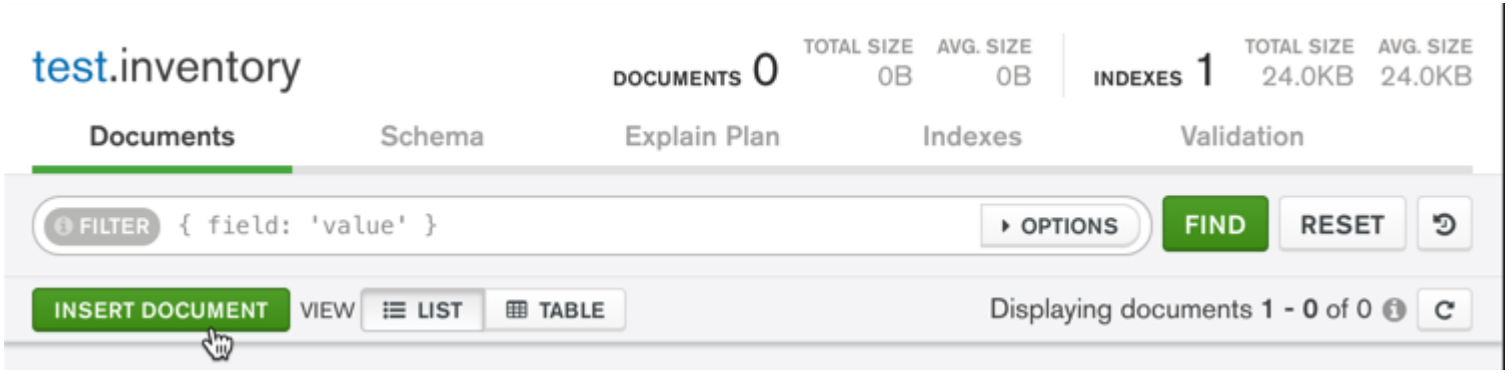
2- Access Data in MongoDB

Insert data

- Using MongoShell

```
db.inventory.insertOne(
   { "item" : "canvas",
     "qty" : 100,
     "tags" : ["cotton"],
     "size" : { "h" : 28, "w" : 35.5, "uom" : "cm" }
   }
)
```

- Using Compass
- Using Python
- Using DataGrip



2- Access Data in MongoDB

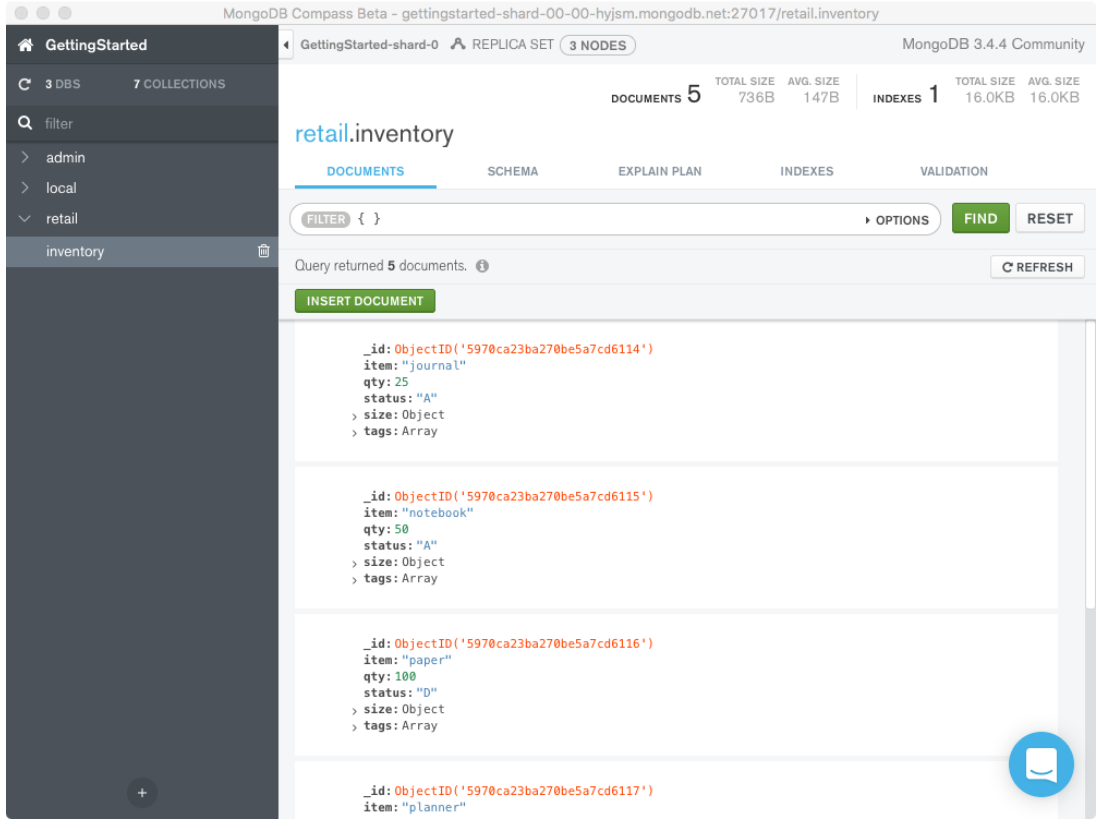
Read data

- Using MongoShell

```
from connect import Connect
from pymongo import MongoClient

connection = Connect.get_connection()
```

- Using Compass



- Using Python

- Using DataGrip

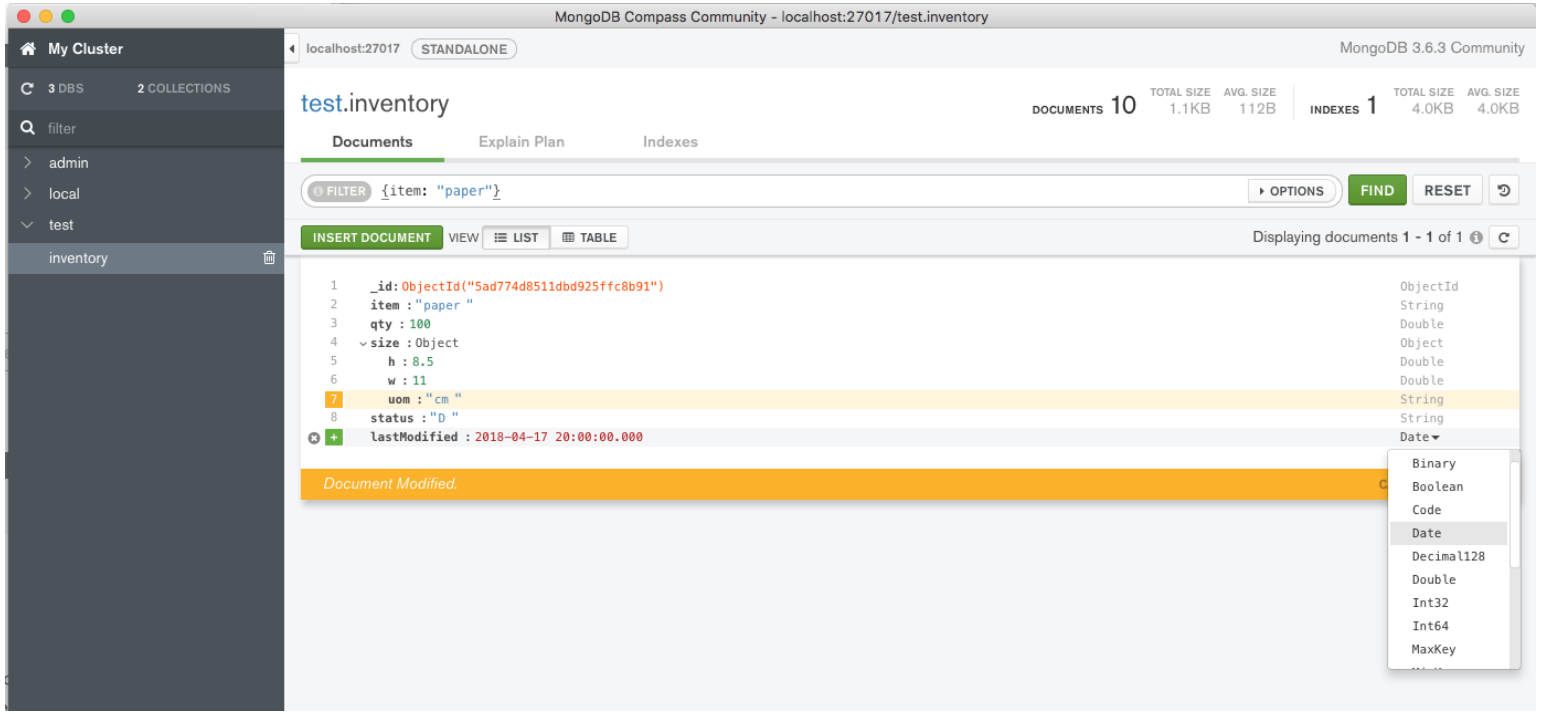
2- Access Data in MongoDB

Update data

- Using MongoShell

```
db.inventory.update_one(
    {"item": "paper"},
    {"$set": {"size.uom": "cm", "status": "P"},
     "$currentDate": {"lastModified": True}})
```

- Using Compass



- Using Python

- Using DataGrip

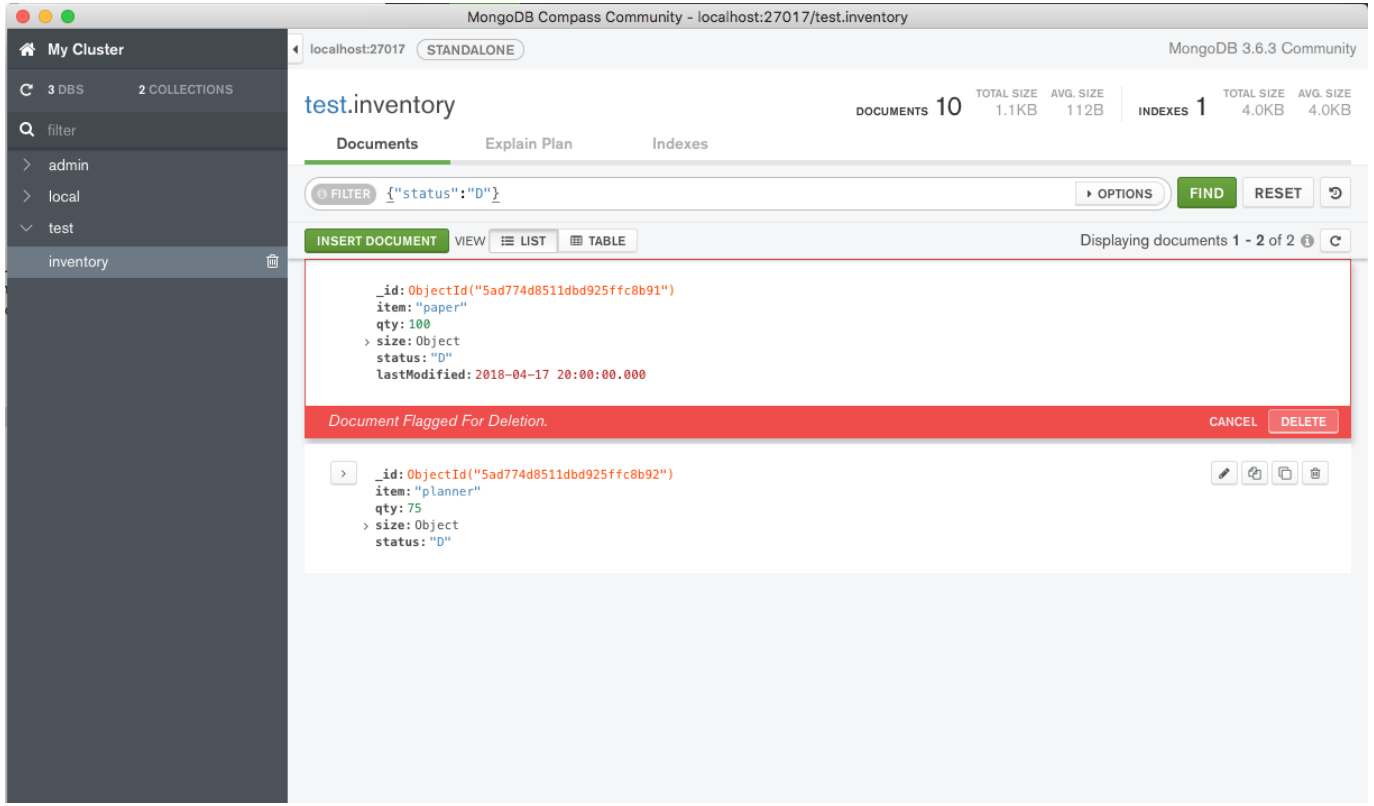
2- Access Data in MongoDB

Delete data

- Using MongoShell

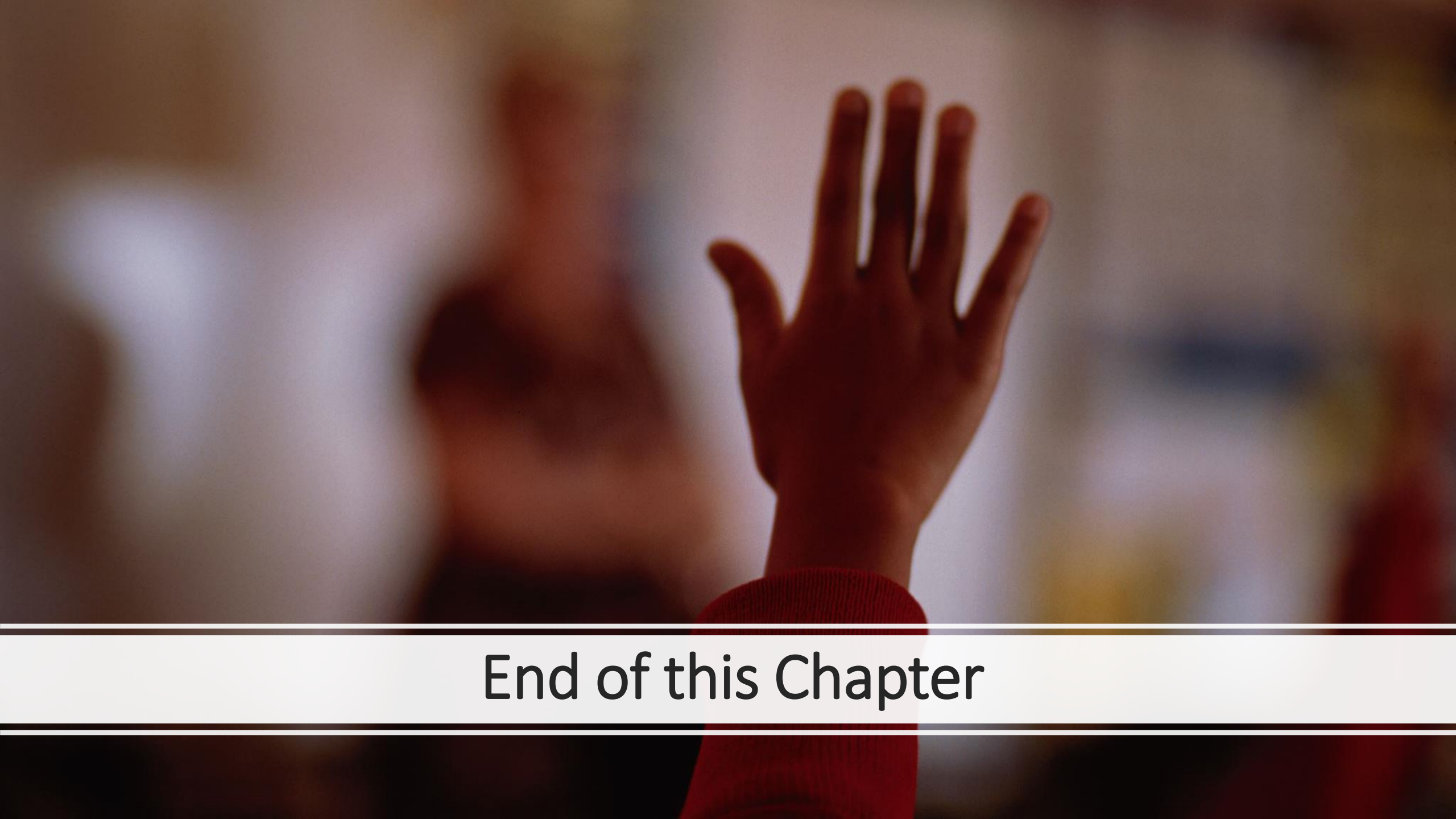
```
db.inventory.delete_one({"status": "D"})
```

- Using Compass



- Using Python

- Using DataGrip



End of this Chapter