School and Pool for Digital Talent

What types of storage do you know ?

Types of Data

Unstructured

Data does not conform to a rigid structure.

Semi- structured

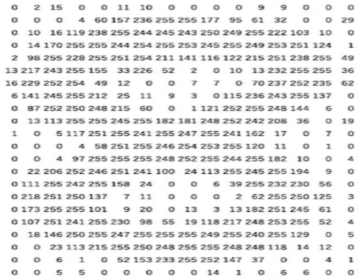
Data conforms to a schema but deviations are possible

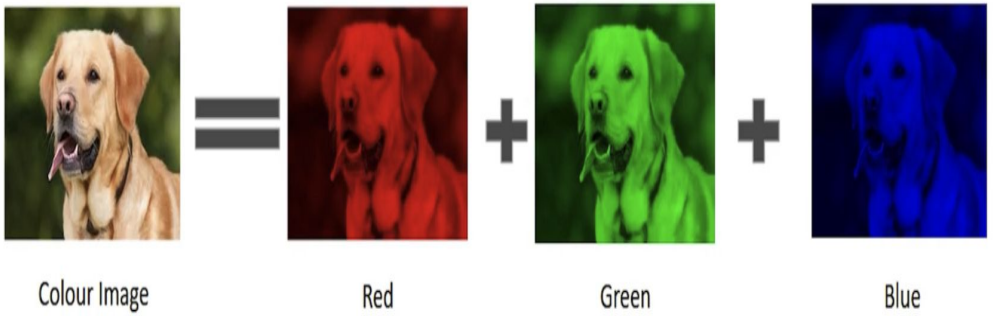
Structured

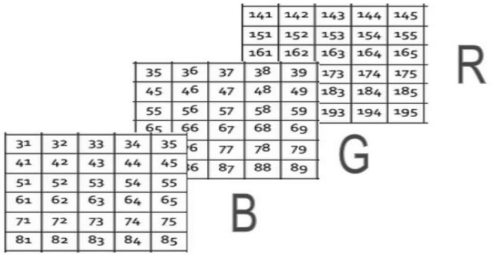
Schema and types of data are known and encoded in the system

Unstructured

- Free text - ‘lorem ipsum какой прекрасный сегодня день!’
- Images
- Audio files
- Can be stored in “object store” - AWS S3



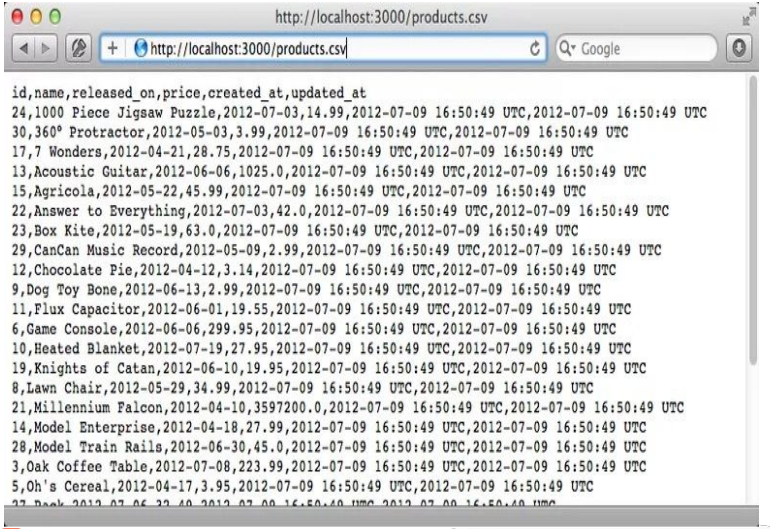




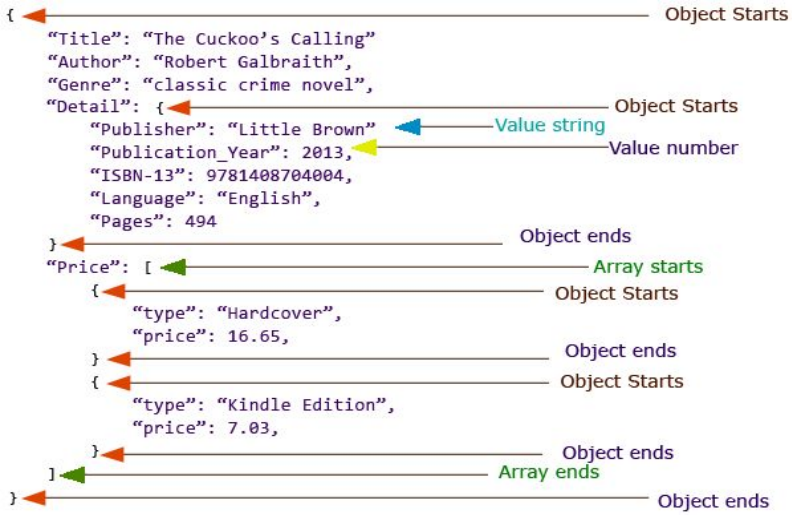
Semi-structured

- **Stored in document databases - “MongoDB”, “Elastic Search”**

- CSV - “comma separated values”



- JSON - JavaScript object notation



Structured

- Most of the data generated by business processes.
- Enforcing the structure reduces flexibility but fosters reliability.
- Deviations from the structure mean :
  - The business process has changed
  - A bug has been introduced
- Data that is frequently needed can be loaded from disk to memory to be accessed faster

- **Structured data can be stored and validated by machines**

Why learn about Databases?

Why?

- Most of (large) data is stored in databases
- An analyst needs to be able to connect to a database and access its data in order to unlock insights

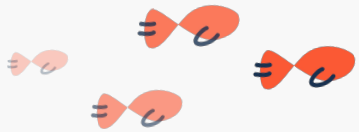
What?

- Understand the basics of databases
- Be able to connect to a database

How?

- Learn about database types and how they are structured
- Connect to a database and explore its content

Introduction to Databases



Databases

A systematic collection of data

Data is either stored on disk or in-memory (faster)

Support electronic storage and manipulation of data

Types of Databases

Structured data - **R**elational **D**ata**b**ase **M**anagement **S**ystems ("RDBMS" or "SQL" databases)

- **Use "SQL" (Structured query language)** to query RDBMS
- Have a predefined schema
- Data is stored in tabular form of columns and rows
- The relationship between data is **relational**
- Examples: Postgres, MySql, Oracle, SQLite

NoSQL Databases - "Not only SQL"

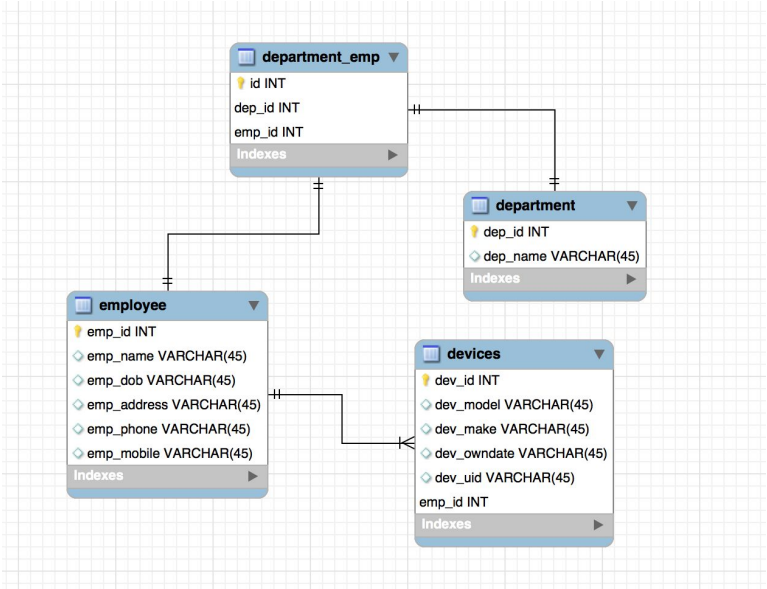
- **don't use SQL** as the primary language
- have no predefined schema
- Are there to deal with use cases that SQL databases are not good at
- Examples: Neo4j, Elasticsearch, MongoDB

RDBMS

- Many different types of databases exist and each uses a different flavour of SQL
- Their syntax can differ, but the core concepts are the same
- Some databases will implement a subset of the functionality
- Some DB will be optimized for speed of read, others for speed of write

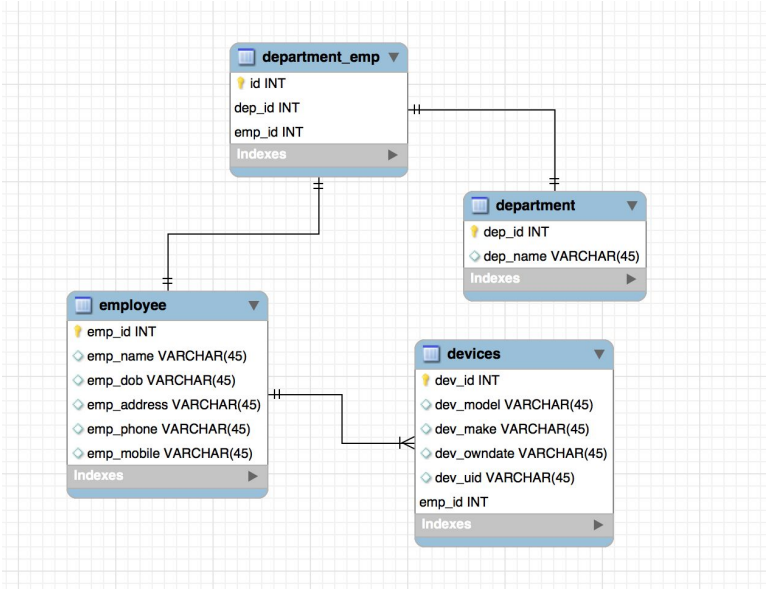
Database Structure

- A database consists of one/multiple schemas
- Schemas consist of tables
- Tables consist of columns and rows
- A column is a variable and has a unique name
- A row is an observation
- Every cell is a single value



Entity-Relationship model

- Data consists of entities of type object, class, person or place
- The property of an entity is described through their attribute(s)
- Relationships describe the relation between entities
- Different types of relationship exist

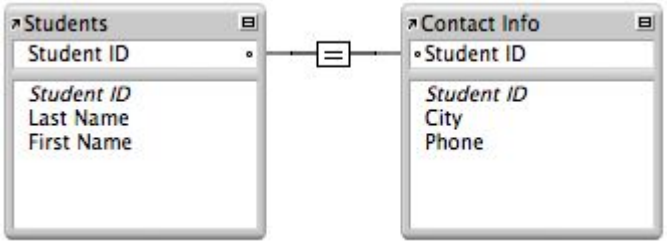


Entity-Relationship types

- One-to-one (1:1)
- One-to-many (1:n) / Many-to-one (n:1)
- Many-to-many (n:n)

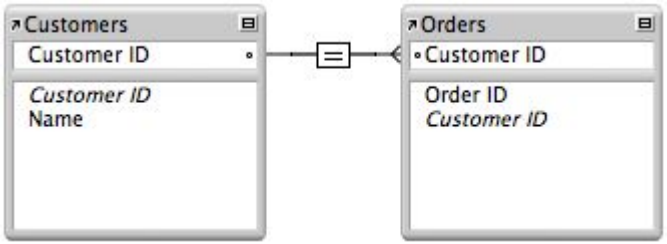
Entity-Relationship types

- One-to-one (1:1)
- One-to-many (1:n) / Many-to-one (n:1)
- Many-to-many (n:n)



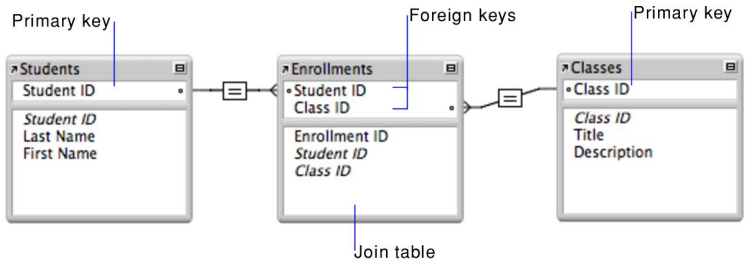
Entity-Relationship types

- One-to-one (1:1)
- One-to-many (1:n) / Many-to-one (n:1)
- Many-to-many (n:n)



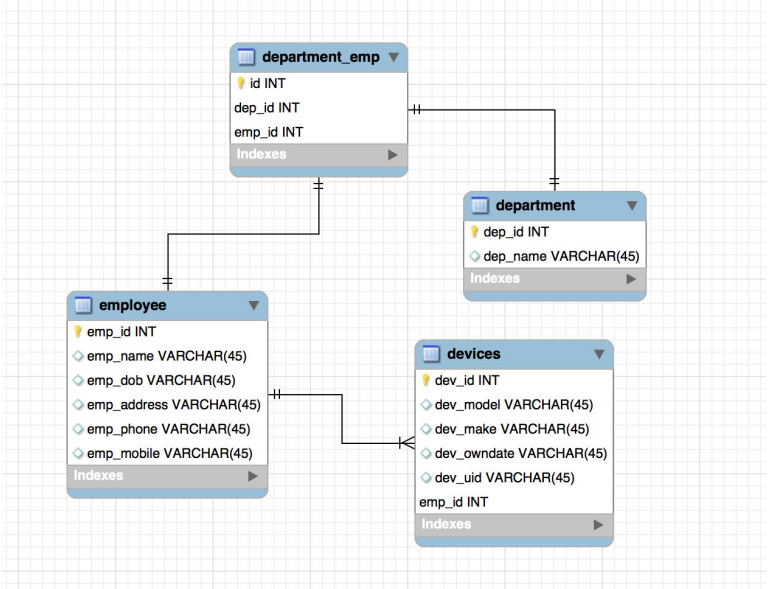
Entity-Relationship types

- One-to-one (1:1)
- One-to-many (1:n) / Many-to-one (n:1)
- Many-to-many (n:n)

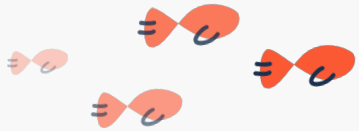


Relational Databases

- Tables are related via primary and foreign keys
- Each table has **one** primary key that is unique for each record
- A foreign key is a field (or collection of fields) in one table, that refers to the primary key in another table



ERM Exercise

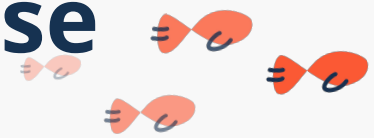


Group Work

1. Split up in 4 groups
2. Open the Miro Board we send in Slack
3. Create between 4 and 8 entities
4. Write down properties (columns) of entities
5. Model dependencies (1:1, 1:m, m:n)
6. Present your final diagram and explain decisions
7. You have 30 Minutes

Topic
Airport
Chocolate Factory
Roller Coaster Park
Restaurant

Connecting to a Database



SQL Client / Database IDE

- IDE = **I**ntegrated **D**evelopment **E**nvironment
- Powerful software that can be used to connect to a database and retrieve and visualise data (and more!)
- Local or in the cloud
- Collection of open-source, free and paid software available

Local SQL Clients

- Installed and run locally on your machine

Examples:

Cloud SQL Clients

- Deployed in the cloud and accessed via a web-interface

Examples:

DBeaver

- In this course we will use DBeaver

Why?

- Free
- Easy to use
- Works for many different types of databases
- Cross platform (Windows, Linux, Mac OS, Solaris)

Installing DBeaver

Choose one:

1. Run the following command in your console:

```
brew install --cask dbeaver-community
```

OR

2. Download from their official website DBeaver Mac OS X (dmg) and install to your Applications folder

Setting up DBeaver

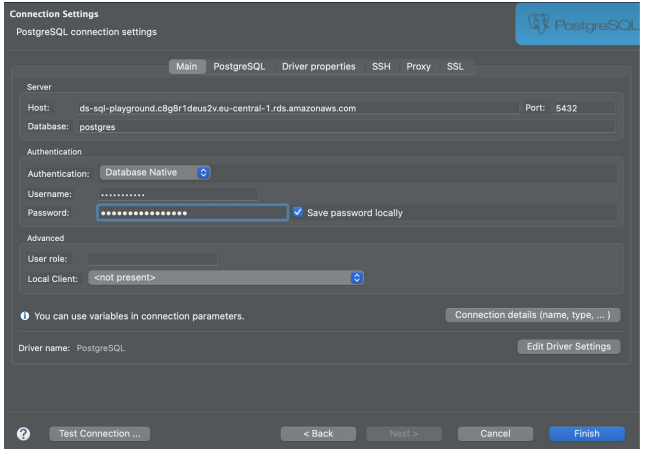
Open DBeaver > Preferences > Editors

- Enable upper case: SQL Editor > Formatting > Keyword Case > Set to: Upper
- Add line numbers: Text Editors > Show line numbers > Tick box

Connect to a PostgreSQL database

1. Click on “New Database Connection”
2. Search for and select PostgreSQL
3. Enter the connection details below

Host	ds-sql-playground.c8g8r1deus2v.eu-central-1.rds.amazonaws.com
Port	5432
Database	postgres
Username	Will be posted in Slack/Zoom Chat
Password	Will be posted in Slack/Zoom Chat



Exploring the database

Find the database connection in your “Database Navigator” pane on the left
Expand it to postgres > postgres > Schemas > public > Tables
Check out the introduction’s tables and answer the following questions:

1. What happens if you double click on a table?
2. What is the first and last entry listed in the records table?
3. What data type is the column ‘record_date’ in the record table?