07/07/2019

HackYourFuture Databases - Lesson 1

Class 4

Borja Romero

Lessons 1,2,3

Gina Stavropoulou

Lessons 1,2

(Geert Van Pamel)

Lessons 2,3

Who is who?

Borja Romero

- Spanish
- In Belgium since 2007
- Working for the European Institutions
- Building a data warehouse for Agricultural products
- Real Madrid supporter

Gina Stavropoulou

- Greek
- In Belgium since 2014
- Working for the Kapernikov
- Working mostly with LiDAR data in computer vision projects
- Film fanatic

Goals

By the end of this module, you should be familiar with the following:

- Entities & Fields
- The relational database model
- The Structured Query Language (SQL)
- The construction of a database system
- MySQL as an example of a relational database system

Course Overview

Week 1 (7/7): Introduction, Relational Databases, Entity Relationship Diagram (ERD), Basic SQL commands

Objective: In the first part of the class we will explain what a relational database is. We will look into Entity Relationship Diagrams and we will get you started with MySQL. By the end of this class you should be able to perform basic queries in a database and create your own tables.

Week 2 (14/7): Group by, Distinct, Having, Inner & Outer Joins

Objective: This class introduces more clauses (group by, having) in the select statement. MySQL joins (inner, self, left and right) should be explained with demonstration.

Week 3 (21/7): Database design, Normal Forms, SQL injection, NoSQL

Objective: In this class we will discuss again the Entity Relationship Diagram (ERD). Students should be able to explain their choices of entities, relationships, attributes etc. SQL injection should be explained with a demonstration (with a simple JS client). Concepts of database transaction, ACID properties, normal forms should be introduced with examples. Small introduction to NoSQL and revision of the material is also planned.

What is a database?

A collection of (organised) information

- Oxford dictionary:

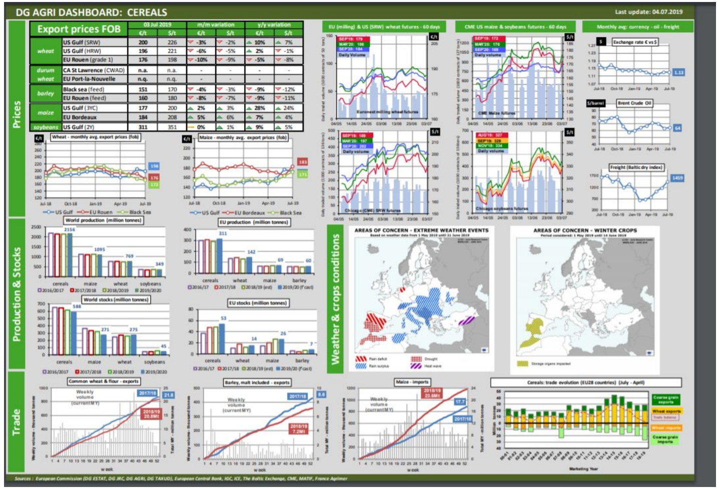
> “A structured set of data held in a computer, especially one that is accessible in various ways.
>
> *‘a database covering nine million workers’*

- Oracle, MySql, Postgres are Database Management Systems :)
- They are in your phone, in your laptop, spread across multiple servers.

Spreadsheets: World’s favourite “database”

What if we told you that this?

...comes from 20 files like this?

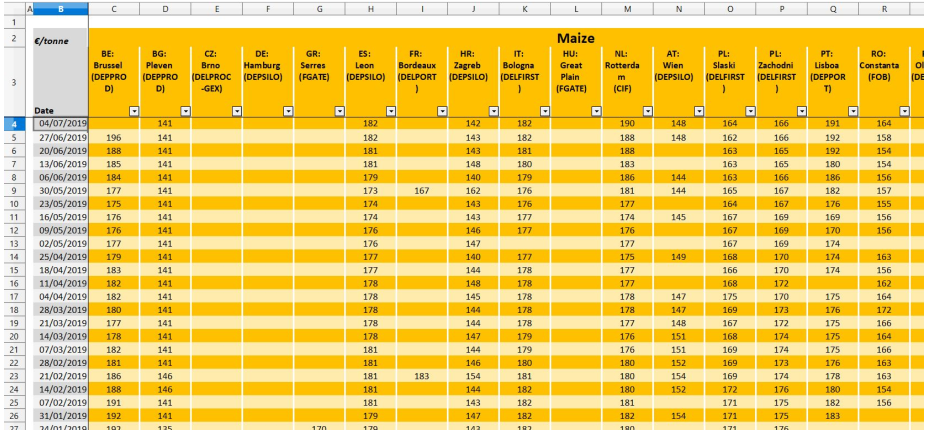


DG AGRI DASHBOARD: CEREALS

Last update: 04.07.2019

Export prices FOB		03 Jul 2019 €/t	03 Jul 2019 $/t	m/m variation €/t	m/m variation $/t	y/y variation €/t	y/y variation $/t
wheat	US Gulf (SRW)	200	226	▼ -3%	▼ -2%	▲ 10%	▲ 7%
	US Gulf (HRW)	196	221	▼ -6%	▼ -5%	▲ 2%	▼ -1%
	EU Rouen (grade 1)	176	198	▼ -10%	▼ -9%	▼ -5%	▼ -8%
durum wheat	CA St Lawrence (CWAD)	n.a.	n.a.	-	-	-	-
	EU Port-la-Nouvelle	n.q.	n.q.	-	-	-	-
barley	Black sea (feed)	151	170	▼ -4%	▼ -3%	▼ -9%	▼ -12%
	EU Rouen (feed)	160	180	▼ -8%	▼ -7%	▼ -9%	▼ -11%
maize	US Gulf (3YC)	177	200	▲ 2%	▲ 3%	▲ 28%	▲ 24%
	EU Bordeaux	184	208	▲ 5%	▲ 6%	▲ 7%	▲ 4%
soybeans	US Gulf (2Y)	311	351	— 0%	▲ 1%	▲ 9%	▲ 5%

Prices
Wheat - monthly avg. export prices (fob)
Maize - monthly avg. export prices (fob)
EU (milling) & US (SRW) wheat futures - 60 days
CME US maize & soybeans futures - 60 days
Monthly avg: currency - oil - freight
Exchange rate € vs $
Brent Crude Oil
Freight (Baltic dry index)
Production & Stocks
World production (million tonnes)
EU production (million tonnes)
World stocks (million tonnes)
EU stocks (million tonnes)
Weather & crops conditions
AREAS OF CONCERN - EXTREME WEATHER EVENTS
AREAS OF CONCERN - WINTER CROPS
Trade
Common wheat & flour - exports
Barley, malt included - exports
Maize - imports
Cereals: trade evolution (EU28 countries) (July - April)



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	
1																			
2		€/tonne	Maize																
3		Date	BE: Brussel (DEPPROD)	BG: Pleven (DEPPROD)	CZ: Brno (DELPROC-GEX)	DE: Hamburg (DEPSILO)	GR: Serres (FGATE)	ES: Leon (DEPSILO)	FR: Bordeaux (DELPORT)	HR: Zagreb (DEPSILO)	IT: Bologna (DELFIRST)	HU: Great Plain (FGATE)	NL: Rotterdam (CIF)	AT: Wien (DEPSILO)	PL: Slaski (DELFIRST)	PL: Zachodni (DELFIRST)	PT: Lisboa (DEPPORT)	RO: Constanta (FOB)	Ol (DE
4		04/07/2019		141				182		142	182		190	148	164	166	191	164	
5		27/06/2019	196	141				182		143	182		188	148	162	166	192	158	
6		20/06/2019	188	141				181		143	181		188		163	165	192	154	
7		13/06/2019	185	141				181		148	180		183		163	165	180	154	
8		06/06/2019	184	141				179		140	179		186	144	163	166	186	156	
9		30/05/2019	177	141				173	167	162	176		181	144	165	167	182	157	
10		23/05/2019	175	141				174		143	176		177		164	167	176	155	
11		16/05/2019	176	141				174		143	177		174	145	167	169	169	156	
12		09/05/2019	176	141				176		146	177		176		167	169	170	156	
13		02/05/2019	177	141				176		147			177		167	169	174		
14		25/04/2019	179	141				177		140	177		175	149	168	170	174	163	
15		18/04/2019	183	141				177		144	178		177		166	170	174	156	
16		11/04/2019	182	141				178		148	178		177		168	172		162	
17		04/04/2019	182	141				178		145	178		178	147	175	170	175	164	
18		28/03/2019	180	141				178		144	178		178	147	169	173	176	172	
19		21/03/2019	177	141				178		144	178		177	148	167	172	175	166	
20		14/03/2019	178	141				178		147	179		176	151	168	174	175	164	
21		07/03/2019	182	141				181		144	179		176	151	169	174	175	166	
22		28/02/2019	181	141				181		146	180		180	152	169	173	176	163	
23		21/02/2019	186	146				181	183	154	181		180	154	169	174	178	163	
24		14/02/2019	188	146				181		144	182		180	152	172	176	180	154	
25		07/02/2019	191	141				181		143	182		181		171	175	182	156	
26		31/01/2019	192	141				179		147	182		182	154	171	175	183		
27		24/01/2019	192	135			170	179		143	182		180		171	176			

Other “databases”

Javascript

```
const musicians = [
 "John Coltrane",
 "Miles David",
 "Thelonious Monk",
 "Sonny Rollins"];
```

Storing new information:

```
musicians.push("Steve Lehman");
```

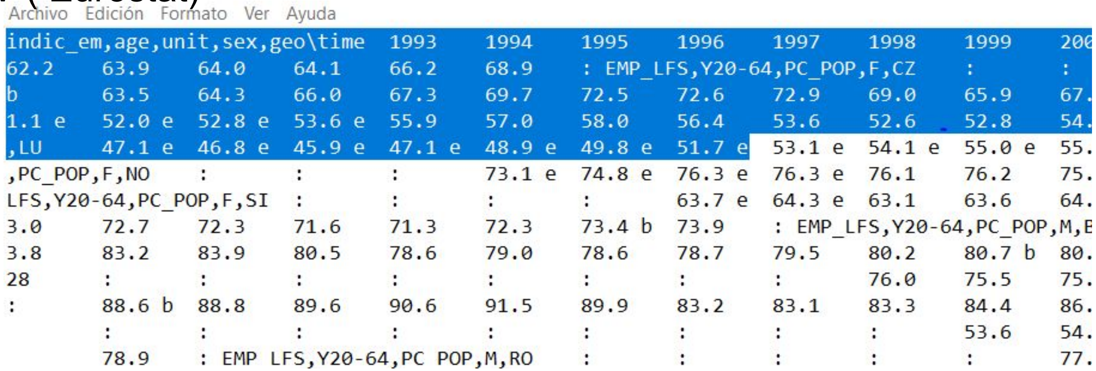
Information access:

```
console.log(musicians[0]);
```

JSON

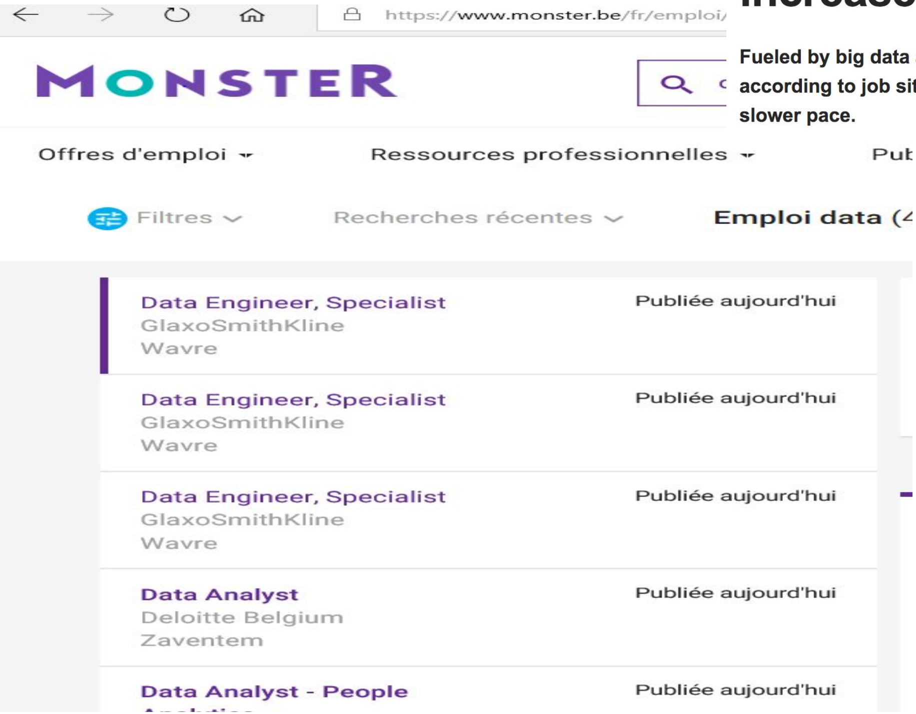
```
[
    {"name":"Ram", "email":"Ram@gmail.com"},
    {"name":"Bob", "email":"bob32@gmail.com"}
]
```

TSV (Eurostat)



```
indic_em,age,unit,sex,geo\time  1993    1994    1995    1996    1997    1998    1999    200
62.2    63.9    64.0    64.1    66.2    68.9    : EMP_LFS,Y20-64,PC_POP,F,CZ    :       :
b       63.5    64.3    66.0    67.3    69.7    72.5    72.6    72.9    69.0    65.9    67.
1.1 e   52.0 e  52.8 e  53.6 e  55.9    57.0    58.0    56.4    53.6    52.6    52.8    54.
,LU     47.1 e  46.8 e  45.9 e  47.1 e  48.9 e  49.8 e  51.7 e  53.1 e  54.1 e  55.0 e  55.
,PC_POP,F,NO    :       :       :       73.1 e  74.8 e  76.3 e  76.3 e  76.1    76.2    75.
LFS,Y20-64,PC_POP,F,SI  :       :       :       :       63.7 e  64.3 e  63.1    63.6    64.
3.0     72.7    72.3    71.6    71.3    72.3    73.4 b  73.9    : EMP_LFS,Y20-64,PC_POP,M,E
3.8     83.2    83.9    80.5    78.6    79.0    78.6    78.7    79.5    80.2    80.7 b  80.
28      :       :       :       :       :       :       :       :       76.0    75.5    75.
:       88.6 b  88.8    89.6    90.6    91.5    89.9    83.2    83.1    83.3    84.4    86.
        :       :       :       :       :       :       :       :       :       53.6    54.
        78.9    : EMP LFS,Y20-64,PC POP,M,RO    :       :       :       :       :       77.
```

So… why bother?



Data Engineer, Specialist
GlaxoSmithKline
Wavre
Publiée aujourd'hui

Data Engineer, Specialist
GlaxoSmithKline
Wavre
Publiée aujourd'hui

Data Engineer, Specialist
GlaxoSmithKline
Wavre
Publiée aujourd'hui

Data Analyst
Deloitte Belgium
Zaventem
Publiée aujourd'hui

Data Analyst - People
Publiée aujourd'hui

Demand for data scientists is booming and will only increase

Fueled by big data and AI, demand for data science skills is growing exponentially, according to job sites. The supply of skilled applicants, however, is growing at a slower pace.

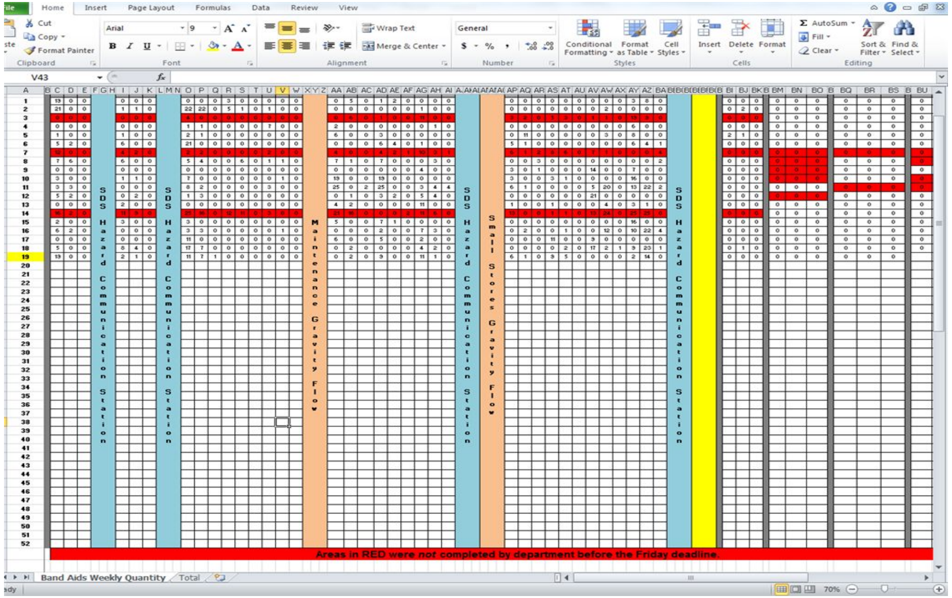
Data Analyst, the most in demand job of the coming years

Data Analysts seek to understand the origin of data and any possible distortions through the use of technology, and many believe this will be the job of the future. Five SMEs out of ten declare they intend to hire one in the next three years.

Ok, you convinced me but I already know Excel…

… and JSON, CSV files do not look so complex either…”

- What were your limitations?
    - Size
    - Ease of update
    - Collaboration
    - Accuracy
    - Security
    - Redundancy (backups)
    - Mix of code and data



Areas in RED were *not* completed by department before the Friday deadline.

Relational Databases

A relational database stores information in tables consisting of rows and columns; the **columns (fields/attributes)** are the properties of the item and the **rows (records, tuples)** represent individual items

Users Table

ID	Name	Last Name	email
got1	Daenerys	Targaryen	danny@gotmail.com
got2	John	Snow	john@gotmail.com
got3	Tyrion	Lannister	tyrion@gotmail.com
got4	Arya	Stark	arya@gotmail.com

Relational Databases

Products Table

ID	Product Name
prd1	Valyrian Steel
prd2	Arrows
prd3	Dog food

Customers Table

ID	Name	Last Name	email
got1	Daenerys	Targaryen	danny@gotmail.com
got2	John	Snow	john@gotmail.com
got3	Tyrion	Lannister	tyrion@gotmail.com
got4	Arya	Stark	arya@gotmail.com

Orders Table

ID	Customer_ID	Product_ID
ord1	got2	prd3
ord2	got2	prd1
ord3	got4	prd2

Relational Databases

Why not everything in one table?

Customers_and_Orders Table

Name	Last Name	email	Product
Daenerys	Targaryen	danny@gotmail.com	Arrows
John	Snow	john@gotmail.com	Arrows
John	Snow	john@gotmail.com	Dog Food

We want to avoid **data redundancy**

Identifying a record

Primary Key

A unique identifier: Cannot be repeated in the database.

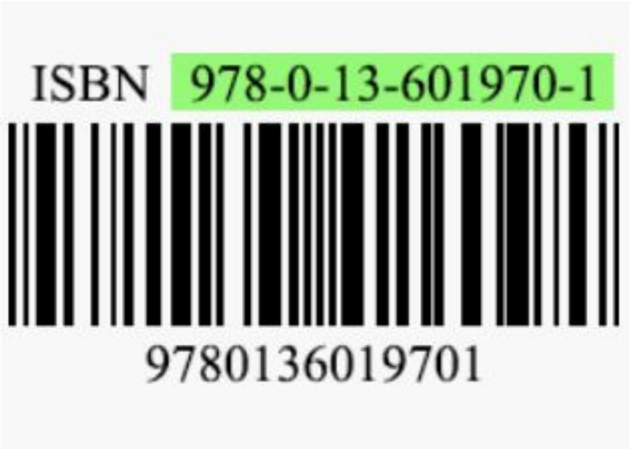
ID	Name	Last Name	email
1	Walder	Frey	walder19@gmail.com
2	Walder	Frey	wfrey@gmail.com
3	Walder	Frey	walderfrey@gmail.com
4	Walder	Frey	wfrey91@gmail.com

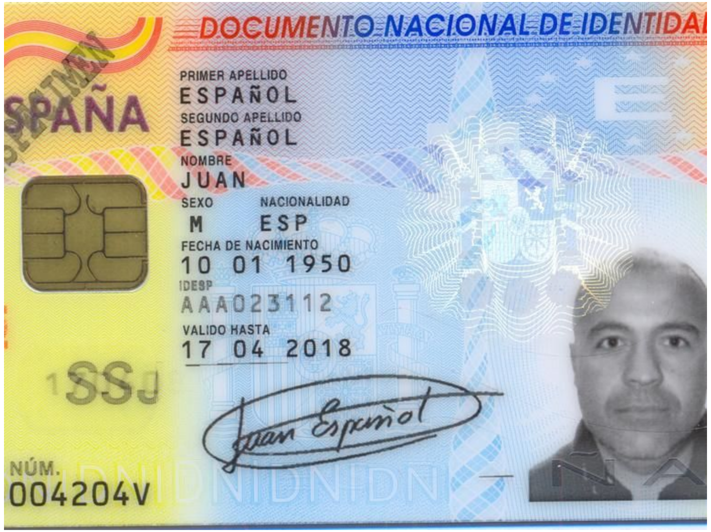
Primary Key

A unique identifier: Cannot be repeated in the database.

ID	Name	Last Name	email
got1	Daenerys	Targaryen	danny@gotmail.com
got2	John	Snow	john@gotmail.com
got3	Tyrion	Lannister	tyrion@gotmail.com
got4	Arya	Stark	arya@gotmail.com

Real Life (™) Primary keys





DOCUMENTO NACIONAL DE IDENTIDA

SPECIMEN

SPAÑA

PRIMER APELLIDO
ESPAÑOL

SEGUNDO APELLIDO
ESPAÑOL

NOMBRE
JUAN

SEXO M

NACIONALIDAD ESP

FECHA DE NACIMIENTO
10 01 1950

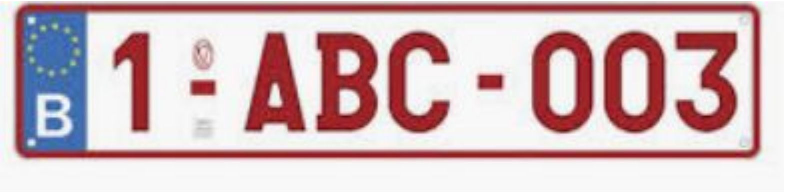
IDESP
AAA023112

VALIDO HASTA
17 04 2018

SSJ

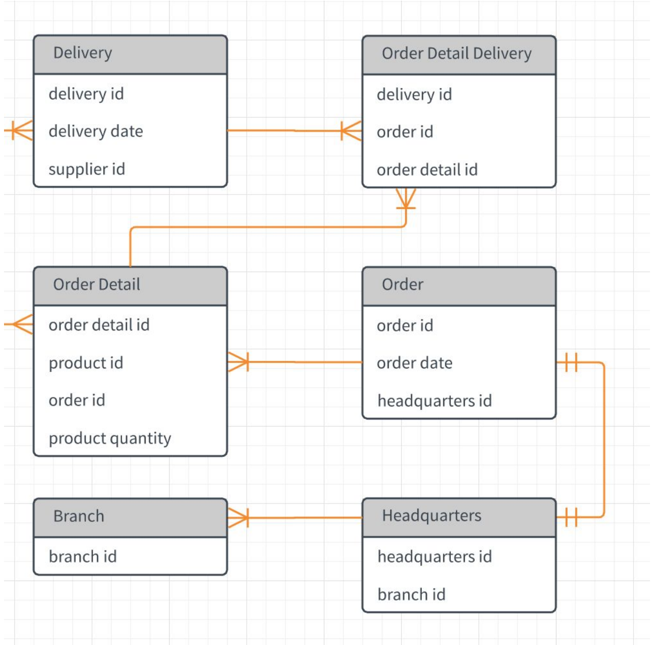
Juan Español

NÚM.
004204V



Entity Relationship Diagram

Helps to understand how the different element of the database interact with each other



Delivery
- delivery id
- delivery date
- supplier id

Order Detail Delivery
- delivery id
- order id
- order detail id

Order Detail
- order detail id
- product id
- order id
- product quantity

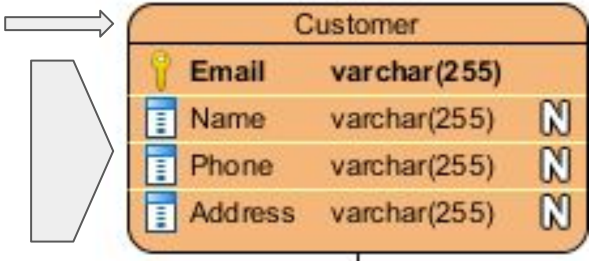
Order
- order id
- order date
- headquarters id

Branch
- branch id

Headquarters
- headquarters id
- branch id

Entity

Fields



Customer		
Email	varchar(255)	
Name	varchar(255)	N
Phone	varchar(255)	N
Address	varchar(255)	N

Relationships:

- one-to-one
- one-to-many
- many-to-many

Tools: Lucidchart, draw.io, MS Visio ($$), Enterprise Architect ($$$), a piece of paper ;)

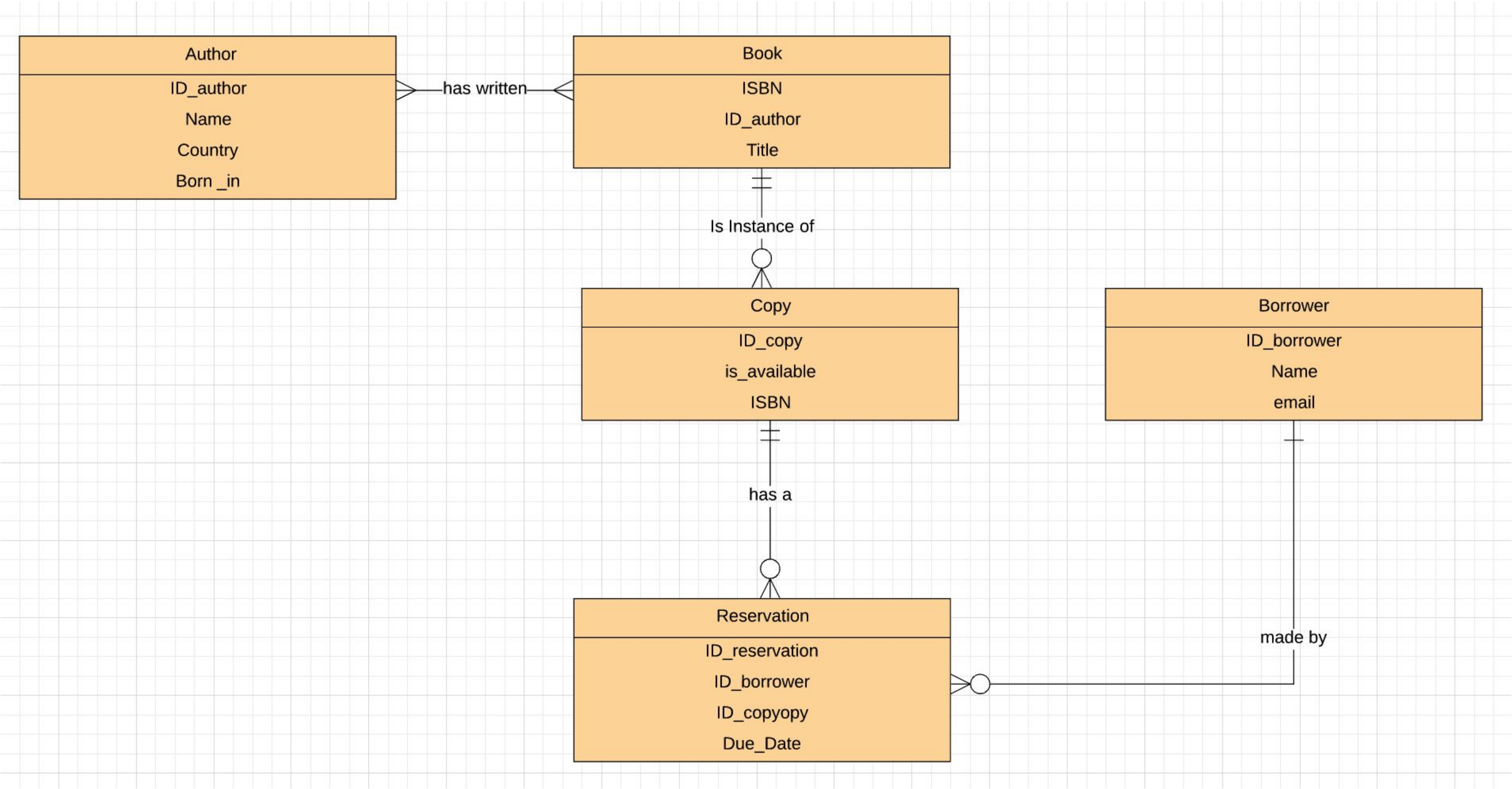
Lucidchart

Enough of slides PLEEEEASE!

LET’s START PLAYING

https://www.lucidchart.com

Lucidchart (Solution)



Author

ID_author

Name

Country

Born _in

has written

Book

ISBN

ID_author

Title

Is Instance of

Copy

ID_copy

is_available

ISBN

has a

Borrower

ID_borrower

Name

email

Reservation

ID_reservation

ID_borrower

ID_copyopy

Due_Date

made by

Getting Started

1. Install MySQL using the following official docs

   (MAC users may use brew install mysql)

2. Download some sample libraries from here.

   Put them in a folder in your desktop.

3. Open a terminal and connect to mysql:

   mysql -u root

4. Load the sample databases:

   SOURCE /path/to/the/databases/folder/imdb.sql

   SOURCE /path/to/the/databases/folder/world.sql

   SOURCE /path/to/the/databases/folder/musicians.sql

Break time!

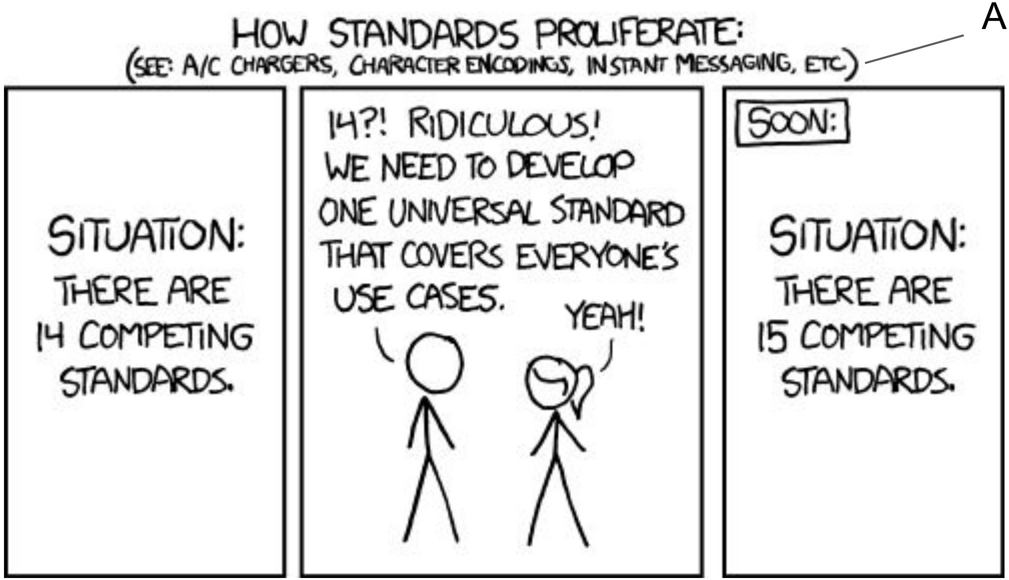
What is SQL?

ESS KEW ELL
SE- QUEL
SKEWL

Structured Query Language

A language used to pull(query) data out of a database.

SQL differences



HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)
SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

And SQL language

MySQL

- One of the first open source databases, developed in the 90’s
- A Relational Database Management System (RDBMS)
- Uses SQL
- Allows data handling, storing, modifying, deleting in a form of tables.

Data Modeling (DDL vs DML)

DDL Data Definition Language

CREATE

DROP

DESC

TRUNCATE

ALTER

DML Data Manipulation Language

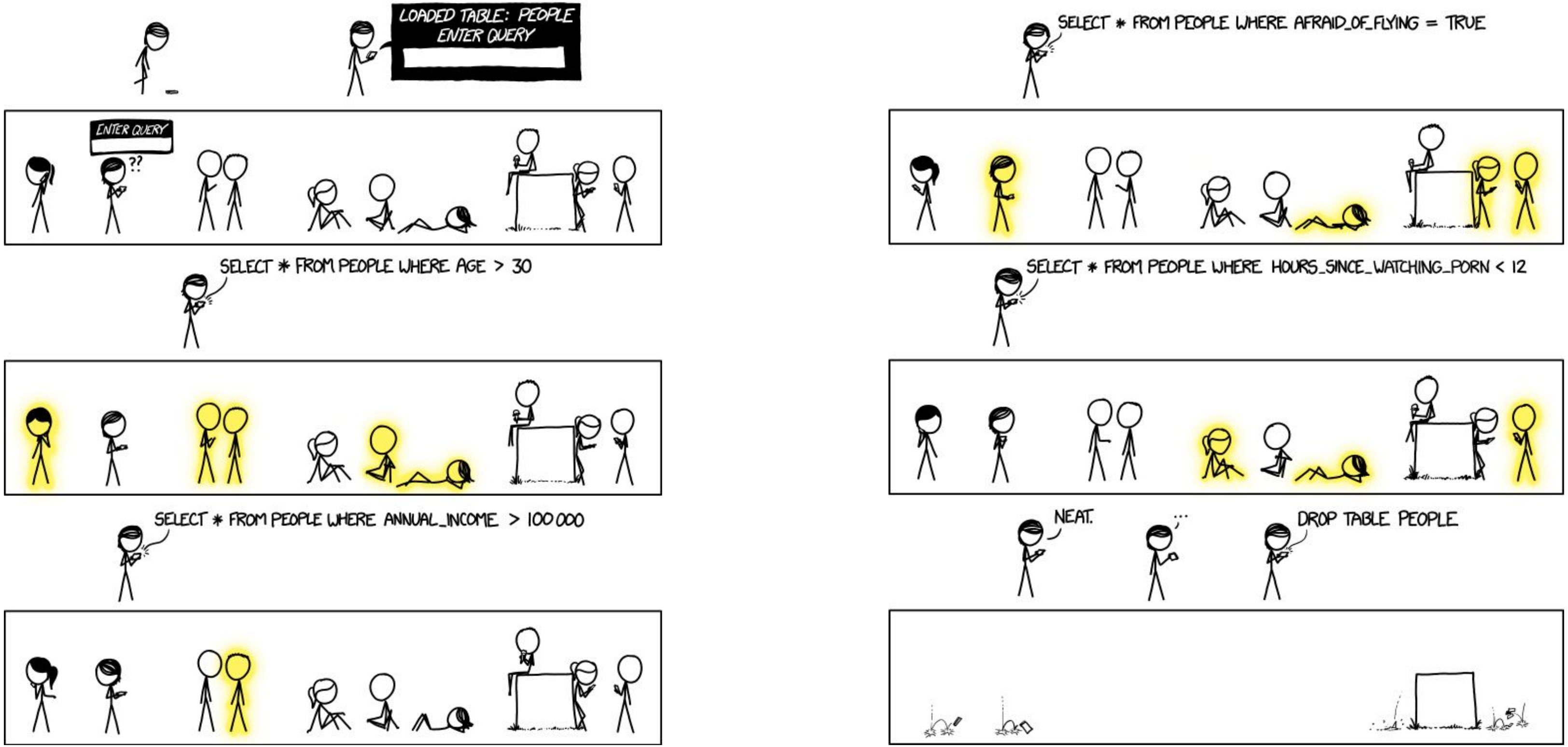
SELECT

INSERT

UPDATE

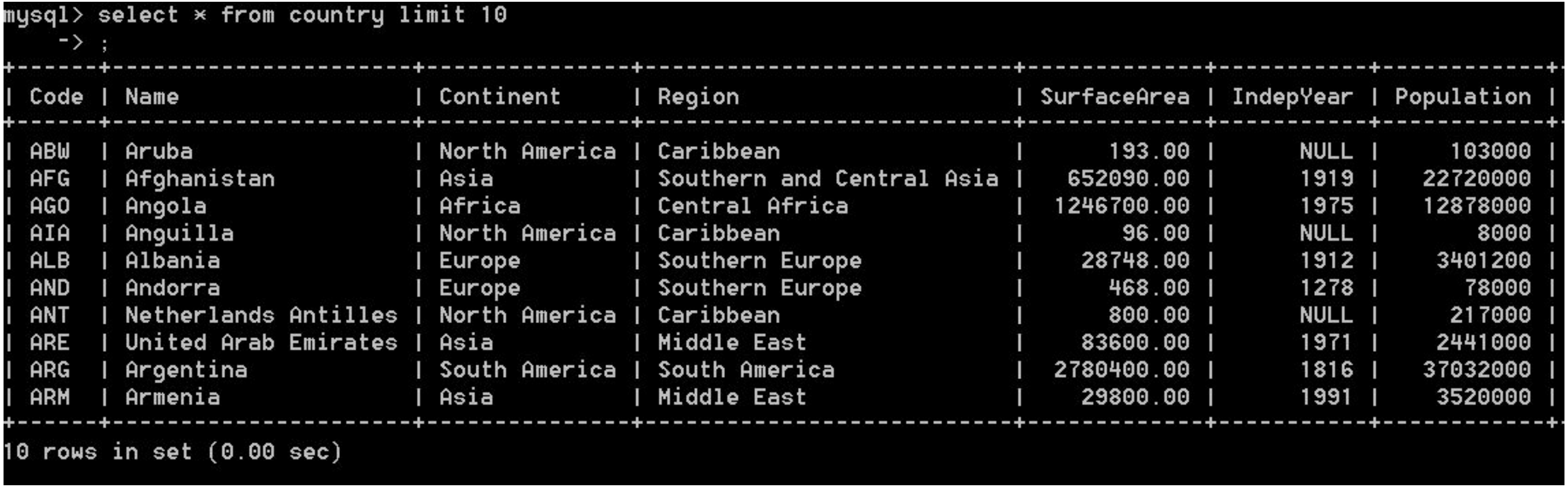
DELETE

SELECT


LOADED TABLE: PEOPLE
ENTER QUERY
ENTER QUERY
??
SELECT * FROM PEOPLE WHERE AGE > 30
SELECT * FROM PEOPLE WHERE ANNUAL_INCOME > 100 000
SELECT * FROM PEOPLE WHERE AFRAID_OF_FLYING = TRUE
SELECT * FROM PEOPLE WHERE HOURS_SINCE_WATCHING_PORN < 12
NEAT.
...
DROP TABLE PEOPLE

SELECT

How does a select look like?



```
mysql> select * from country limit 10
    -> ;
+------+----------------------+---------------+---------------------------+-------------+-----------+------------+
| Code | Name                 | Continent     | Region                    | SurfaceArea | IndepYear | Population |
+------+----------------------+---------------+---------------------------+-------------+-----------+------------+
| ABW  | Aruba                | North America | Caribbean                 |      193.00 |      NULL |     103000 |
| AFG  | Afghanistan          | Asia          | Southern and Central Asia |   652090.00 |      1919 |   22720000 |
| AGO  | Angola               | Africa        | Central Africa            |  1246700.00 |      1975 |   12878000 |
| AIA  | Anguilla             | North America | Caribbean                 |       96.00 |      NULL |       8000 |
| ALB  | Albania              | Europe        | Southern Europe           |    28748.00 |      1912 |    3401200 |
| AND  | Andorra              | Europe        | Southern Europe           |      468.00 |      1278 |      78000 |
| ANT  | Netherlands Antilles | North America | Caribbean                 |      800.00 |      NULL |     217000 |
| ARE  | United Arab Emirates | Asia          | Middle East               |    83600.00 |      1971 |    2441000 |
| ARG  | Argentina            | South America | South America             |  2780400.00 |      1816 |   37032000 |
| ARM  | Armenia              | Asia          | Middle East               |    29800.00 |      1991 |    3520000 |
+------+----------------------+---------------+---------------------------+-------------+-----------+------------+
10 rows in set (0.00 sec)
```

SELECT

```
SELECT column1, column2, …
FROM table1;
```

```
SELECT FirstName
  FROM Musicians;
+------------+
| FirstName  |
+------------+
| Thelonious |
| Sonny      |
| Steve      |
+------------+
3 rows in set (0.00 sec)
```

```
SELECT *
FROM table1;
```

SELECT … WHERE...

```
SELECT column1, column2, …
FROM table1;
```

```
SELECT FirstName
  FROM Musicians;
+------------+
| FirstName  |
+------------+
| Thelonious |
| Sonny      |
| Steve      |
+------------+
3 rows in set (0.00 sec)
```

```
SELECT *
FROM table1;
```

```
SELECT column1, column2, …
FROM table1
WHERE condition
    AND/OR another_condition
    AND/OR …;
```

```
SELECT *
FROM Musicians
WHERE FirstName = 'Thelonious'
+----+------------+----------+------+
| Id | FirstName  | LastName | Born |
+----+------------+----------+------+
|  1 | Thelonious | Monk     | 1917 |
+----+------------+----------+------+
1 row in set (0.00 sec)
```

SELECT (Operators, BETWEEN, IN, NOT IN)

Operator	**Condition**	**SQL Example**
=, !=, < <=, >, >=	Standard numerical operators	col_name **!=** 4, col_name **=** “abc”
BETWEEN … AND …	Number is within range of two values (inclusive)	col_name **BETWEEN** 1.5 **AND** 10.5
NOT BETWEEN … AND …	Number is not within range of two values (inclusive)	col_name **NOT BETWEEN** 1 **AND** 10
IN (…)	Number or string exists in a list	col_name **IN** (2, 4, 6), col_name I**N** (“a”, “b”)
NOT IN (…)	Number or string does not exist in a list	col_name **NOT IN** (1, 3, 5), col_name **NOT IN** (“a”, “b”)

Source: SQLBolt

SELECT (Exercise time!)

Switch to the imdb database.

See which tables it includes.

Try to answer the following queries:

1. Find all the first and last name of all the actors.
2. Find the first name of the actor with ID = 3.
3. Find all the actors whose name is “Jennifer”.
4. Find the name and biographies of all the actresses.
5. Find the first and last name of the actor whose age is above 50.
6. Find the titles of the films with ratings between 6 and 8.
7. Find the titles and the ratings of the films that were NOT released between 1990 and 2000.

SELECT (Solutions)

1. Find all the first and last name of all the actors.

   ```
   select fname, lname from actors;
   ```

2. Find the first name of the actor with ID = 3.

   ```
   select fname from actors where aid=3;
   ```

3. Find all the actors whose name is “Jennifer”.

   ```
   select fname, lname from actors where fname = "Jennifer";
   ```

4. Find the last name and biographies of all the actresses.

   ```
   select lname, biography from actors where gender="f";
   ```

SELECT (Solutions)

5. Find the first and last name of the actor whose age is above 50.

```
select fname, lname from actors where age>50;
```

6. Find the titles and ratings of the films with ratings between 6 and 8.

```
select mname, rating  from films where rating between 6 and 8;
```

7. Find the titles and year of the films that were NOT released between 1990 and 2000.

```
select mname, rating from films where year not between 1990 and 2000;
```

SELECT (LIKE, ORDER BY, LIMIT, COUNT)

Operator	Condition	Example
LIKE	Case insensitive exact string comparison	col_name **LIKE** "abc"
NOT LIKE	Case insensitive exact string inequality comparison	col_name **NOT LIKE** "abc"
%	Used anywhere in a string to match a sequence of zero or more characters (only with LIKE or NOT LIKE)	col_name **LIKE** "%TO%" (results in "TOMATO", "POTATO", "TO","TOP")

Source: SQLBolt

```
SELECT column1,column2, …
FROM table1
WHERE condition(s)
ORDER BY column ASC (or DESC);
```

```
SELECT column1,column2, …
FROM table1
WHERE condition(s)
ORDER BY column ASC (or DESC)
LIMIT number_of_results;
```

```
SELECT COUNT(*) FROM table1;
```

SELECT (Exercise time!)

Now try these queries:

1. Find all the information about the actors whose first name starts with an A.
2. Find all the movie titles that contain the word “club”.
3. Find all the films that do not contain the word “games”.
4. Find all the film title and ratings in descending order of rating.
5. Find all the information about the 3 films.
6. Find the titles of the 3 most recent films.
7. Count all the films in the database.

SELECT (Exercise time!)

1. Find all the information about the actors whose first name starts with an A.

   ```
   select * from actors where fname like "A%";
   ```

2. Find all the movie titles that contain the word “club”.

   ```
   select title from films where title like "%club%";
   ```

3. Find all the films that do not contain the word “games”.

   ```
   select title from films where title not like "%games%”;
   ```

4. Find all the film title and ratings in descending order of rating.

   ```
   select title, rating from films order by rating desc;
   ```

SELECT (Exercise time!)

Now try these queries:

5. Find all the information about 3 films only.

   ```
   select * from films limit 3;
   ```

6. Find the titles of the 3 most recent films.

   ```
   select mname from films order by year desc limit 3;
   ```

7. Count all the films in the database.

   ```
   select count(*) from films;
   ```

SELECT SYNTAX

(OMG!) How to read a statement syntax:

```
SELECT [ALL | DISTINCT | DISTINCTROW ]

      [HIGH_PRIORITY ] [STRAIGHT_JOIN ][SQL_NO_CACHE ] [SQL_CALC_FOUND_ROWS ]

    select_expr [, select_expr ...]

    [FROM table_references

      [PARTITION partition_list]

    [WHERE where_condition]

    [GROUP BY {col_name | expr | position}, ... [WITH ROLLUP]]

    [HAVING where_condition]

   [ORDER BY {col_name | expr | position} [ASC | DESC], ...]

    [LIMIT {[offset,] row_count | row_count OFFSET offset}]
```

CREATE

```
CREATE DATABASE db_name;
```

```
CREATE TABLE table1 [IF NOT EXISTS] (

column1_name data_type [NOT NULL] [DEFAULT default_value],

column2_name data_type,…, PRIMARY KEY (column2_name);
```

CONSTRAINTS:
The `NOT NULL` indicates that the inserted value cannot be `NULL`.
The `DEFAULT value` is used to specify the default value of the column.
The `PRIMARY KEY` specifies that values are the unique identifiers.

CREATE

Each column has a name and a data type. For some data types you can also specify a maximum length

Data type
INTEGER, BOOLEAN
FLOAT, DOUBLE, REAL
CHARACTER(max_length), VARCHAR(max_length), TEXT
DATE, DATETIME

EXAMPLE:

```
CREATE TABLE `actors` (
  `aid` int(11) NOT NULL AUTO_INCREMENT,
  `fname` varchar(50) DEFAULT NULL,
  `lname` varchar(50) DEFAULT NULL,
  `biography` text,
  `age` int(2) DEFAULT NULL,
  `sex`  varchar(1) DEFAULT NULL,
  PRIMARY KEY (`aid`)) ;
```

More data types:
https://dev.mysql.com/doc/refman/8.0/en/data-types.html

CREATE (Exercise time!)

Create a database named “class4”.

Create a table “students”. We would like to store first and last name, email, country, age and height. Each student should also have an unique identifier. First and last name should always be inserted.

CREATE (Solution)

```
create database class4;
```

```
create table students(fname varchar(50) not null, lname varchar(50) not null, email
text, country varchar(50), height float, age integer default 0, id int, primary key(id));
```

CREATE SYNTAX

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS]
tbl_name
    (create_definition,...)
    [table_options]
    [partition_options]
create_definition:
    col_name column_definition
  | {INDEX|KEY} [index_name] [index_type]
(key_part,...)
      [index_option] ...
  | [CONSTRAINT [symbol]] PRIMARY KEY
      [index_type] (key_part,...)
      [index_option] ...
  | [CONSTRAINT [symbol]] UNIQUE [INDEX|KEY]
      [index_name] [index_type]
(key_part,...)
      [index_option] ...
  | [CONSTRAINT [symbol]] FOREIGN KEY
      [index_name] (col_name,...)
      reference_definition
  | check_constraint_definition
```

```
column definition:
    data type [NOT NULL | NULL] [DEFAULT {literal |
(expr)} ]
      [AUTO INCREMENT] [UNIQUE [KEY]] [[PRIMARY] KEY]
      [COMMENT 'string']
      [COLLATE collation name]
      [COLUMN FORMAT {FIXED|DYNAMIC|DEFAULT}]
      [STORAGE {DISK|MEMORY}]
      [reference definition]
      [check constraint_definition]
  | data type
      [COLLATE collation name]
      [GENERATED ALWAYS] AS (expr)
      [VIRTUAL | STORED] [NOT NULL | NULL]
      [UNIQUE [KEY]] [[PRIMARY] KEY]
      [COMMENT 'string']
      [reference definition]
      [check_constraint_definition]
```

INSERT

```
INSERT INTO table1
VALUES (value_or_expr,
another_value_or_expr, …),
       (value_or_expr_2,
another_value_or_expr_2, …),
       …;
```

EXAMPLE:

INSERT INTO `actors` VALUES (1,'Brad','Pitt','lot of adopted children',55, 'm'),(2,'Orlando','Bloom','Cool guy',42, 'm'));

INSERT (Exercise time!)

Time to populate the table you just created. Fill in the data for yourself and for the person next to you.

INSERT (Solution)

```
insert into students value("harry","potter","harry@hogwarts.com", "Uk", 1.65, 14, 1);
```

INSERT SYNTAX

```
INSERT [LOW PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE]
    [INTO] tbl name
    [PARTITION (partition name [, partition_name] ...)]
    [(col name [, col name] ...)]
    {VALUES | VALUE} (value list) [, (value list)] ...
    [ON DUPLICATE KEY UPDATE assignment_list]
```

```
INSERT [LOW PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE]
    [INTO] tbl name
    [PARTITION (partition_name [, partition_name] ...)]
    SET assignment list
    [ON DUPLICATE KEY UPDATE assignment_list]
```

```
INSERT [LOW PRIORITY | HIGH_PRIORITY] [IGNORE]
    [INTO] tbl name
    [PARTITION (partition name [, partition_name] ...)]
    [(col name [, col_name] ...)]
    SELECT ...
    [ON DUPLICATE KEY UPDATE assignment_list]
```

More material

Hack Your Future (Databases Week 1) https://github.com/HackYourFuture/databases/tree/master/Week1

SQL Bolt (SQL) https://sqlbolt.com/

W3 Schools (SQL) https://www.w3schools.com/sql/

MySQL https://www.tutorialspoint.com/mysql/mysql-introduction.htm

And ALWAYS ALWAYS:

www.google.com and stackoverflow.com

I want to practice more: https://www.hackerrank.com/domains/sql

Homework

1. **Build you own ER diagram:**

   Imagine the database of a travel agency. It offers trips around the world and now it is in the process of creating an online system for reservations. Each customer can make reservation for a trip, which will have a start date and an end date. What will be the possible entities and what will be their fields? Draw an ER diagram with at least 5 entities to explain.

   You can use Lucidchart or the software of you choice. The final form should be submitted in pdf. Be careful to use the correct relationships between the entities. Keep it simple :)

Homework

2. **Create your database:**

   Create a database for the library based on the EDR diagram that we made today. Create the tables and populate them. You can choose to how extended your database will be but create at least 4 tables with 3 or 4 rows in each. Make sure that you use the correct data types (and that you use at least 3 different ones, eg: text, number, date). Keep in mind, a Library can have more than one copies of a book. There should be a mechanism to know if a copy is available or not. *(HINT: Use the sample databases you downloaded to help you)*

   Optional: Try to think what would be a good identifier (primary key) in each of your tables.

Homework

3. **Write queries to retrieve data that answers the following questions:**

Use world.sql db.

1. What are the names of the countries with population greater than 8 million
2. What are the names of the countries that have “land” in their names ?
3. What are the names of the cities with population in between 500,000 and 1 million ?
4. What are the names of all the countries on the continent ‘Europe’ ?
5. List all the countries in the descending order based on their surface areas.

If you have time left and want more practice you can try these optional homework exercises:

1. Write queries that answer the following questions:
   - i. What are the names of all the cities in the Netherlands?
   - ii. What's the population of Rotterdam?
   - iii. What's the top 10 countries based on surface area?
   - iv. What's the top 10 cities with the highest population?
   - v. What's the population of the world ?

Homework

Optional: Try to connect your node.js to mysql.

The end!

Thank you :)