ANALYZING BIG DATA - I

Week 2 – Introduction to (Relational) Data Bases

Agenda

- Defining Data bases (DB's): History of DB's, Types of DB's, (Dis)Advantages of DB's
- Understanding relational DB's:
  - Elements of a relational-DB schema, tables, attributes, relations, primary keys, foreign keys
  - Basic modeling exercises
- Interacting with relational DB's: One will always need
  - A DBMS: We use MySQL workbench
  - A Programing language that is SQL – based: We use MySQL
    - History of MySql
    - SQL *vs* MySQL *vs* Postgre SQL *vs* Teradata SQL
  - We illustrate everything with: Geography DB
    - Create: Schemas, Tables, Relationships, Attributes, Add records to a talble
    - Drop: Tables, DB's
    - Very Basic Searches

DATABASES

Databases – History

- Storing and manipulating data has **always** been a major computer application
- Early1960's – Charles Bachman at General Electric designs the first Integrated Data Center (Store)
  - Nobel Prize in Computer Science…. Does not exist ☹
- Late 1960's –
  - IBM jumps in and develops the Information Management System (IMS)
  - The SABRE system is developed by American Airlines (IBM help). (Fly tickets)
- 1970's – Edward Codd proposes the relational model.
  - We will study the *relational data model* today for data storage
  - *Relational data model* became universal and Programing languages specialized to work with it are developed.
- 1990's – Today : The relational data model receives a new Boost. Why?

Databases – General Description

- Database: Any set of data held in a computer in such a way that can be easily accessed, managed or updated.
  - Definition is too wide (useless).
  - Solution: Further classify.

- Types of databases:

1. **Nonrelational DB's** (or NoSQL Based)
   - Examples
     - Key-value Stores: (Redis) and Amazon DynamoDB
     - Wide Column Stores: Cassandra, Scylla, and HBase
     - Graph Databases: (Neo4J, DataStax)
     - Document oriented DB's: MongoDB and Couchbase
     - Search engines: ElasticSearch
   - Advantages: Very Flexible
   - Disadvantages: Still at its infancy

2. **Relational DB's** (or SQL Based)
   - Based on Entities and Relationships

Databases – General Description

- Database: Any set of data held in a computer so that it can be easily accessed, managed and updated.
  - Definition is too wide (useless)
  - Solution: Further classify.

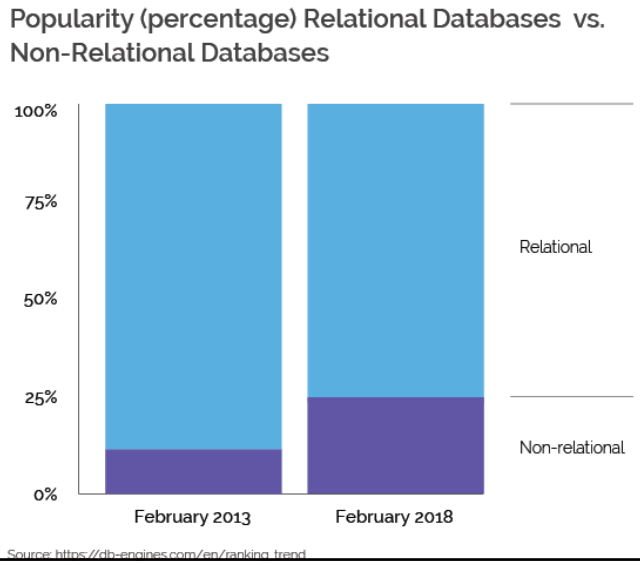
- Types of databases:

1. **Nonrelational DB's** (or NoSQL Based)
   - Examples
     - Key-value Stores: (Redis) and Amazon DynamoDB
     - Wide Column Stores
     - Graph Databases
     - Document oriented DB's
   - Advantages: Very Flexible
   - Disadvantages: Still at its infancy

2. **Relational DB's** (or SQL Based)
   - Based on Entities and Relationships
   - Rely on Database Management System

Databases – General Description

- Database: Any set of data held in a managed and updated.
  - Definition is too wide (useless)
  - Solution: Further classify.
- Types of databases:
  1. **Nonrelational DB's** (or NoSQL Bas
     - Examples
       - Key-value Stores: (Redis) and Amazon D
       - Wide Column Stores
       - Graph Databases
       - Document oriented DB's
     - Advantages: Very Flexible
     - Disadvantages: Still at its infancy
  2. **Relational DB's** (or SQL Based)
     - Based on Entities and Relationships
     - Rely on Database Management System



UNDERSTANDING RELATIONAL DB'S

Relational DB's

- Relational DB's: Collection of data describing the activities of an **specific** organization and it is composed by **Entities** and **Relationships**:
  - **Entities**: An **entity** is any object in the system that we want to model and store information about.
    - Movies DB: Genres, Actors, Directors, Movies
  - **Relationships**: Restrictions linking each of the entities
    - Movies DB: Movies has many actors and actors appear in many movies

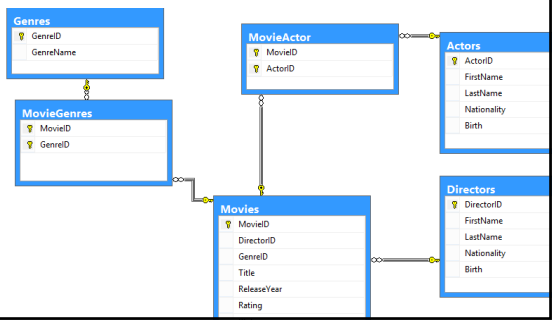
Relational DB's - Examples

See word document

Relational DB's - Schema

- The Key element of a Relational DB is its schema

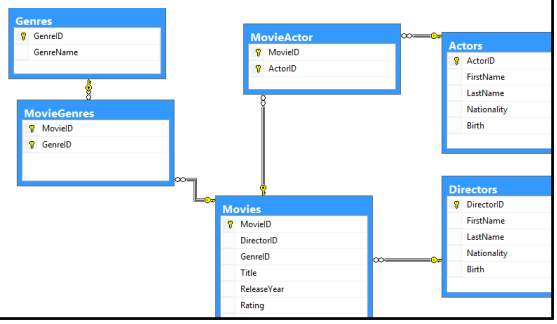
- It is graph which **must** include the following elements
  1. Entities (tables)
     - Attributes (or columns, or variables)
     - All the Primary keys
     - All the Foreign keys
  2. Relations
     - Relations **between real entities** must indicate its type: 1-to-many, many-to-many



Relational DB's - Schema - Tables

- The Key element of a Relational DB is its schema

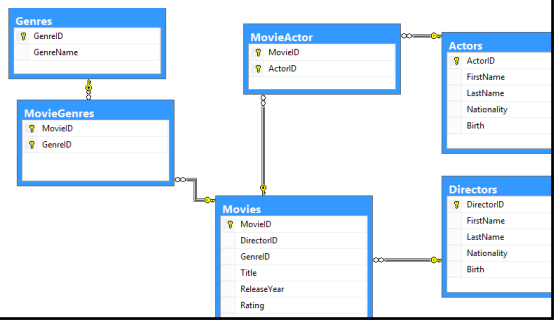
- It is graph which **must** include the following elements
  1. Entities (tables)
     - Attributes (or columns, or variables)
     - All the Primary keys
     - All the Foreign keys



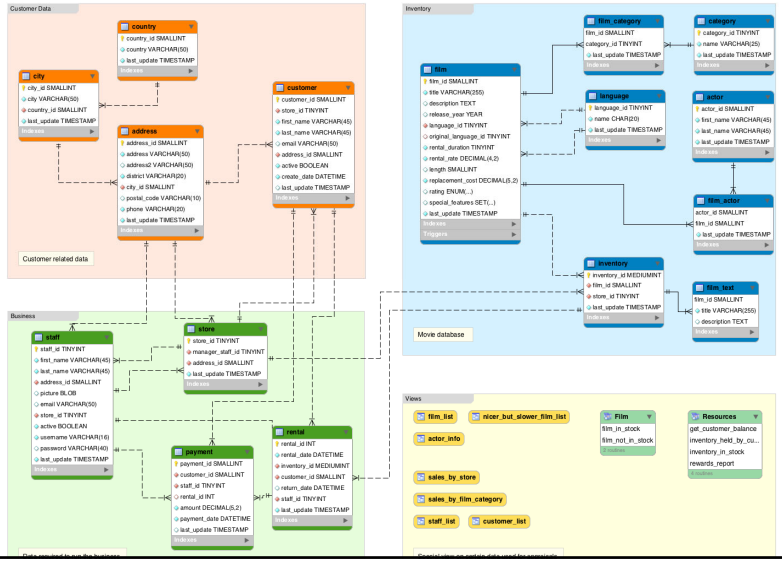
Relational DB's - Schema - Relations

- One-to-One: *Not a Problem (trivial)*
- One-to-many: Not a Problem
- Many-to-many: Almost Not a Problem

  -> Break it into two One-to-One Relationships
- Important: An Schema is a hierarchical structure



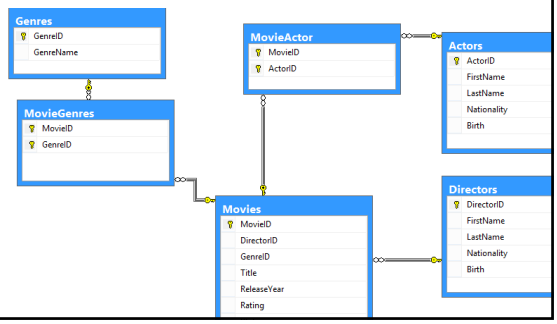
Relational DB's - Schema - Examples



Relational DB's - Schema: In Practice

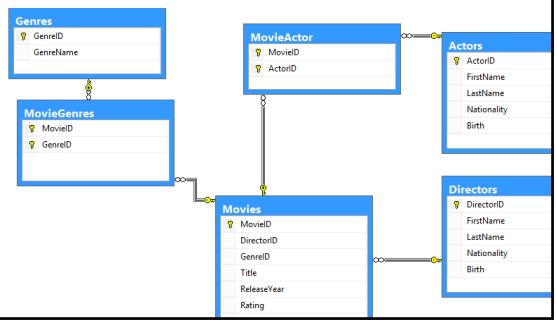
- Data Base is a whole field in Computer Science
- You don't need to know everything !!!!

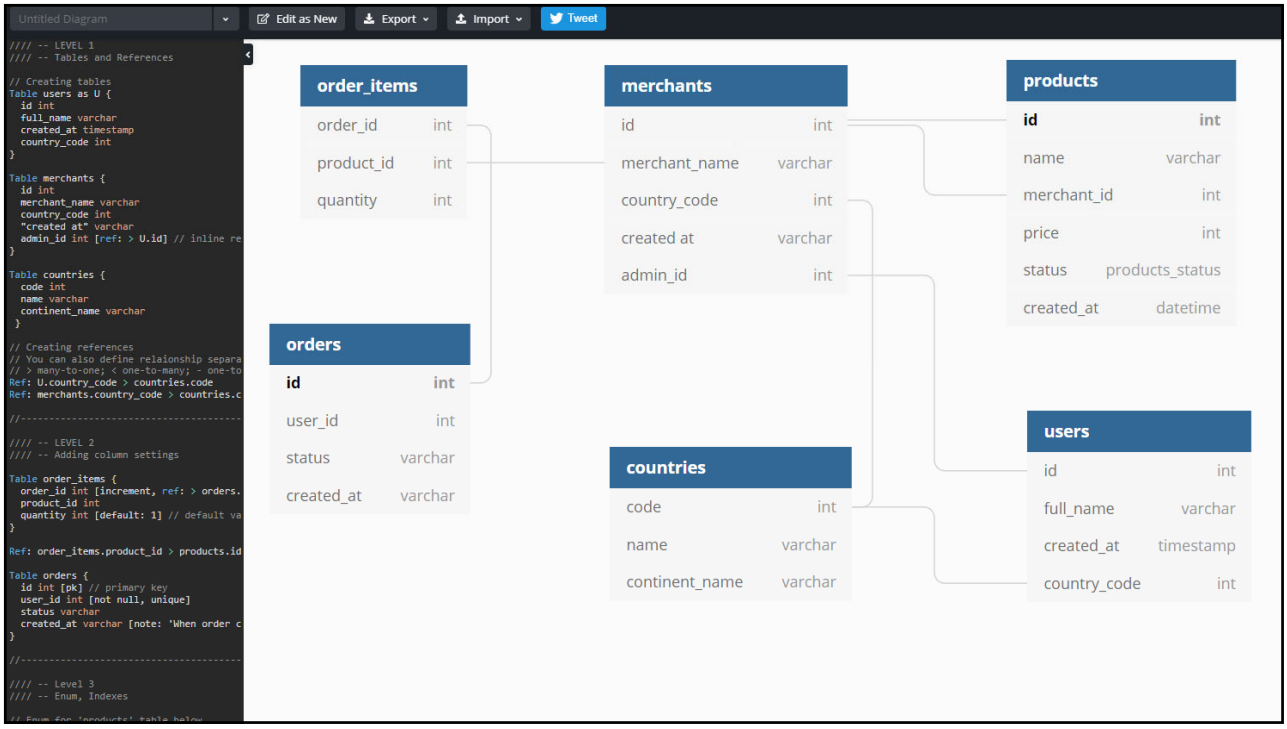
- As a data scientist **you just need to be able to**:
  - **Design** a basic schema for an organization based on a description of it.
  - **Describe** an organization based on the schema of the data base
  - **Recover an existing schema**:
    - My-SQL will do it for you
    - If your company does not use My-SQL. Still can be done
      - No worries ☺



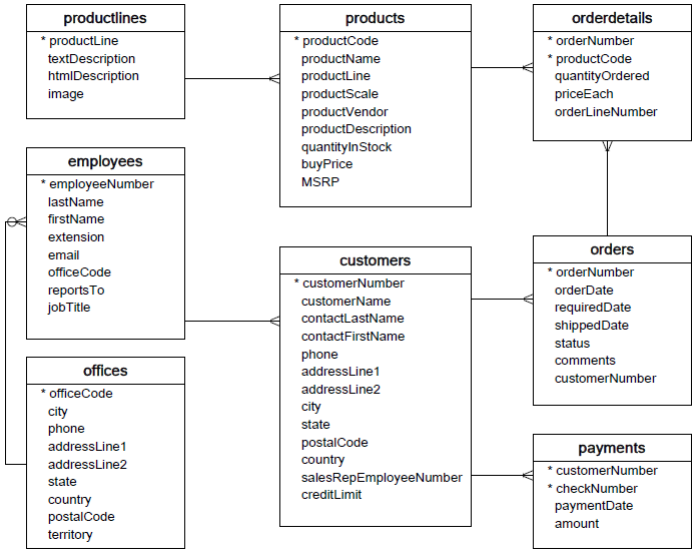
Relational DB's - Schema - Design

- Step one: List all the relevant entities
  - List all the attributes you want
  - Specify if they have to be stored as text or numerically
- Step two:
  - Identify as many relationships as possible between the entities.
  - Determine whether is 1 to many, 1 to 1 or many to many
    - If it is many to many -> Break it
- Go to MySQL or to https://dbdiagram.io/home



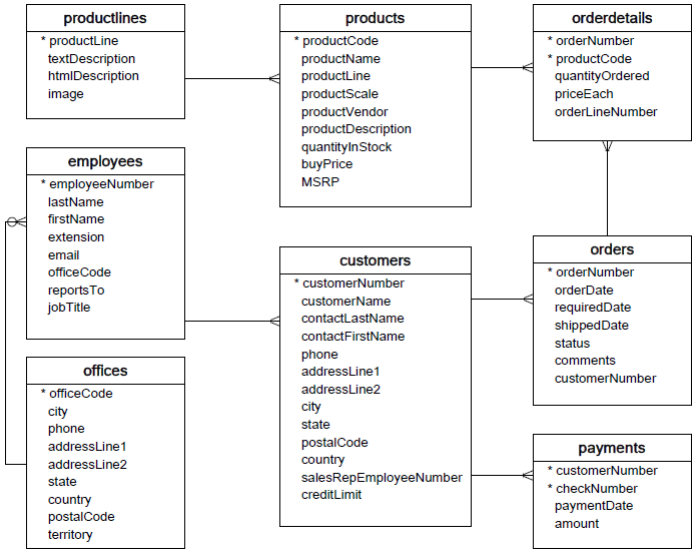

Untitled Diagram
Edit as New
Export
Import
Tweet
//// -- LEVEL 1
//// -- Tables and References
// Creating tables
Table users as U {
id int
full_name varchar
created_at timestamp
country_code int
}
Table merchants {
id int
merchant_name varchar
country_code int
"created at" varchar
admin_id int [ref: > U.id] // inline re
}
Table countries {
code int
name varchar
continent_name varchar
}
// Creating references
// You can also define relaionship separa
// > many-to-one; < one-to-many; - one-to
Ref: U.country_code > countries.code
Ref: merchants.country_code > countries.c
//----------
//// -- LEVEL 2
//// -- Adding column settings
Table order_items {
order_id int [increment, ref: > orders.
product_id int
quantity int [default: 1] // default va
}
Ref: order_items.product_id > products.id
Table orders {
id int [pk] // primary key
user_id int [not null, unique]
status varchar
created_at varchar [note: 'When order c
}
//----------
//// -- Level 3
//// -- Enum, Indexes
order_items
order_id int
product_id int
quantity int
merchants
id int
merchant_name varchar
country_code int
created at varchar
admin_id int
products
id int
name varchar
merchant_id int
price int
status products_status
created_at datetime
orders
id int
user_id int
status varchar
created_at varchar
countries
code int
name varchar
continent_name varchar
users
id int
full_name varchar
created_at timestamp
country_code int

Relational DB's – Describe an Organization based on the Schema



Relational DB's - Recovering Schemas

- Done in My-SQL
- When you get to your first job, your boss may not give you the schema.
- They may just give you access to the database



Relational DB's – Advantages

- Mature Technology
- Data Integrity and Security
- Efficient (and concurrent) Data Access
- Data Independence
- Data administration

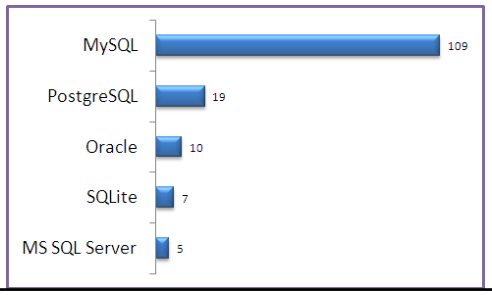
Relational DB's – Disadvantages

- Relational databases are VERY costly to install

- DBMS are **EXTREMLY** costly to change
  - Switching costs can become a serious problem
  - DBMS create "*Path dependence*"
    - IBM
    - Oracle
    - Amazon

INTERACTING WITH RELATIONAL DB'S

Welcome to *SQL - Pronounced Sequel* ☺



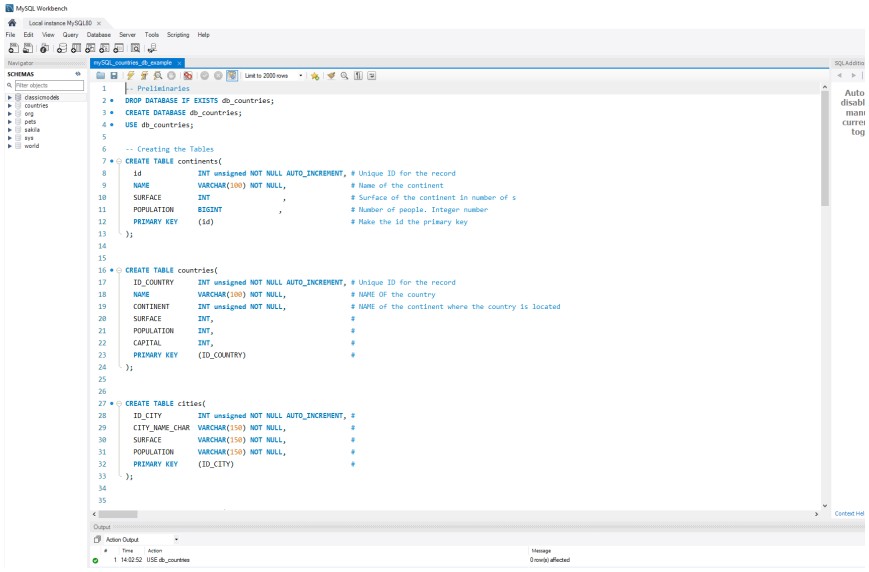
SQL – History

SQL - Programing Language many dialects

SQL – From SQL to My-SQL

- In what follows we switch to My-SQL we are going to work directly with the DBMS
- The goal is to replicate all what he have done in a hands-on approach using My-SQL

Visiting My-SQL workbench



Creating a DB

- On My-Sql workbench

Creating a Table

- On My-Sql workbench

Attribute type - NUMERIC

- Integers
  - TINYINT/ BOOL/BOOLEAN
  - SMALLINT
  - MEDIUMINT
  - INT/INTEGER
  - BIGINT
- Floating point
  - FLOAT: single precision 7 digits
  - DOUBLE: double precision 15 digits.
- Fixed Point
  - DECIMAL(6,2) represents 6 digits of which 2 will be decimal points.

Attribute type - STRINGS

- Character Strings:
  - Fixed length: CHAR (for example CHAR(8)).
  - Variable length: VARCHAR (p.e. VARCHAR(255)).
  - Very long text: TEXT
- Un-ordered Sets: ENUM
  - ENUM('Dean','Full-Professor','Associated-Professor','Assistant-Professor')
- Ordered Sets: SET
  - SET('red','green','yellow')

Attribute type - DATE

- DATE: just the date
- DATETIME: Calendar time with hours and minutes and
- YEAR: 4 digit year
- YEAR(2): 2 digit year (don’t use it)
- TIMESTAMP:

Altering Tables – Add/Remove Column

- On My-Sql workbench

Altering Tables – Add/Remove Foreign Key

- On My-Sql workbench

Reverse Engineering an Schema

- On My-Sql workbench

Adding a Column to Table

- On My-Sql workbench

Reverse Engineering an Schema

- On My-Sql workbench

Introducing HW 2