UNIVERSITY OF TWENTE.

Database Management Systems

Presented by:

Mahdi Farnaghi

Assistant Prof.

Department of Geo-Information Processing

FACULTY OF GEO-INFORMATION SCIENCE AND EARTH OBSERVATION

OVERVIEW

- Databases and DBMS
  - Definitions and basic terminology
- Relational data model
  - Concepts and terminology
  - Domain, Attribute, Relation, Schema, SQL
  - Constraints

DATABASE

DATA, DATASET, DATABASE, DBMS AND DB SYSTEM

Data

- Is a **resource** held on paper or in digital format that serves to record or administer some **facts and descriptions of phenomena of interest**.

Data set (or dataset):

- A homogeneous **collection of data** normally describing a single kind of phenomenon

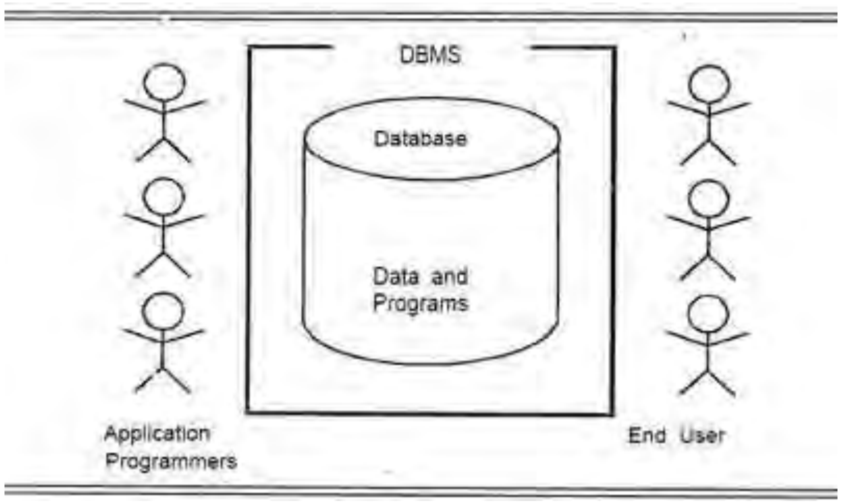
Database (DB)

- A **collection of interrelated data sets properly structured** by means of, and stored through a DBMS (Data Base Management System)

DATABASE
DATA, DATASET, DATABASE, DBMS AND DB SYSTEM

Database management system (DBMS)

- A **software package** that is designed for the purpose of managing databases. This means, DBMS allows to set-up, maintain and explore one or more databases.



Database system

- Combination of **a database and its DBMS**.

Question:

Why do we need database and database management system?

DATABASE MANAGEMENT SYSTEM (DBMS)

WHAT DOES A DBMS OFFER TO USERS?

- Supports the storage and manipulation of ***very large data sets***.
- Can be instructed to guard over ***data correctness***.
- Supports the ***concurrent use*** of the same data set by many users.
- Provides a high-level, declarative ***query language***.
- Includes data ***backup and recovery functions*** to ensure data protection and availability at all times.
- Allows the ***control of data redundancy***.

DATA MODEL

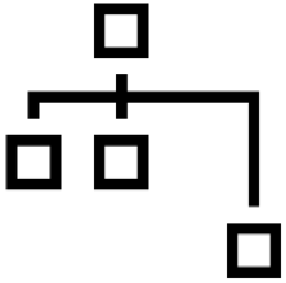
A DBMS supports the use of a **data model**.

It is a model that **organizes elements of data and standardizes how they relate to one another**

A data model is an integrated collection of:

- **Data structuring primitives,**
- **Rules of how to structure, and**
- **Mechanisms to handle the data**

In other words, a data model is a **toolbox** that allows us to **create/define** *a* database structure and **manipulate** the data stored in it.



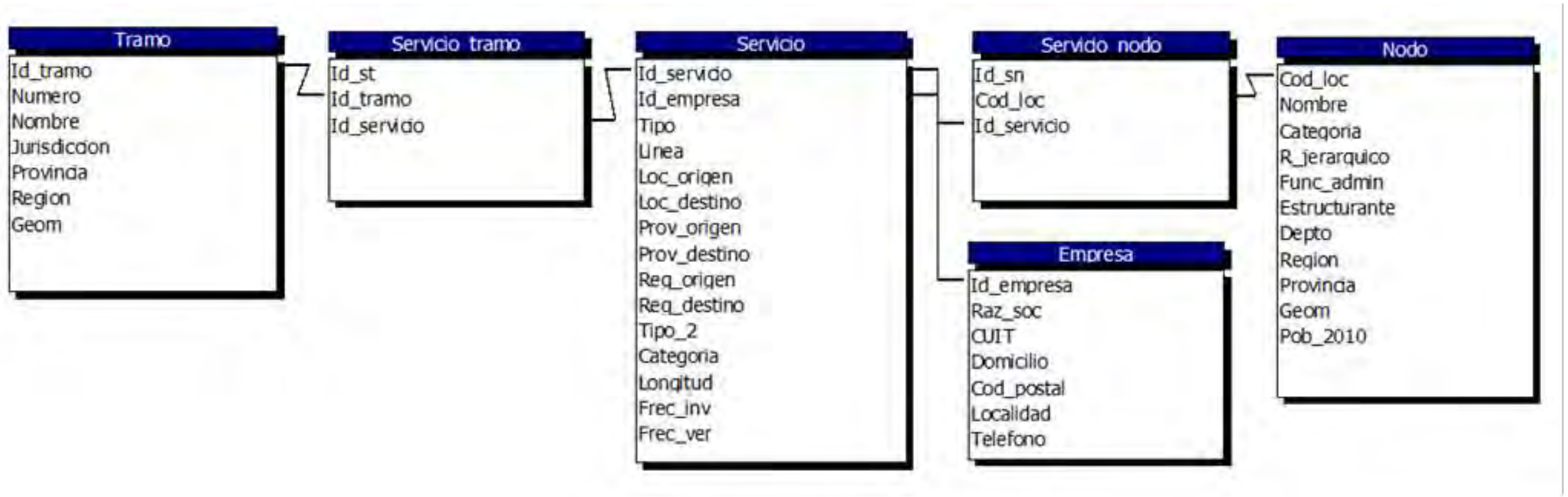
UNIVERSITY OF TWENTE.

RELATIONAL DATA MODEL

WHY STUDY RELATIONAL DATA MODEL?

There are different data models: hierarchical, network, object-oriented, etc.

DBMS linked to most GIS packages make use of the **relational data model**

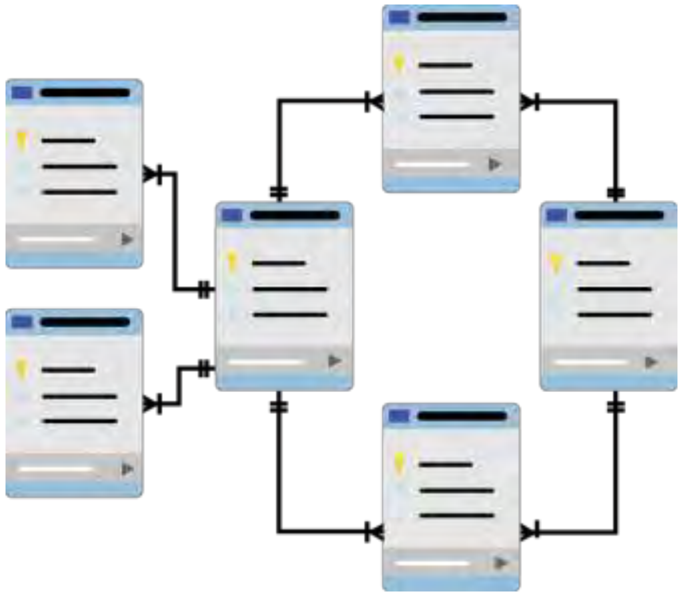


RELATIONAL DATA MODEL
WHAT IS THE RELATIONAL DATA MODEL?

The relational data model structures a DB as a **collection of inter-related tables (or *relations*)** → we model a piece of reality as linked tables

The introduction of relational data models (by Ted Codd in 1970) is considered as the most important event in the history of the database field.

It uses **Structured Query Language** (SQL)



THE LANGUAGE USED IN RELATIONAL DATA MODEL

Structured Query Language *(SQL)* – the relational database language:

- SQL is a DSL (*domain-specific language*) used in programming and **designed for managing data held in a relational DBMS**.
- Powerful and **natural language**
- First described in the 70's
- Became ISO standard in 1987
- **Particularly useful in handling structured data where there are relations between different entities/variables of the data.**

THE LANGUAGE USED IN RELATIONAL DATA MODEL

Find all cities (codes and names) that belong to the department called Tilcara

```
SELECT   Code, Name
FROM     A
WHERE    Dep = Tilcara
```

A

Code	Name	Dep
38077020	Cienega de Paicone	Santa Catalina
38077080	Paicone	Santa Catalina
38084020	Coranzuli	Susques
38084090	San Juan de Quillaques	Susques
38084080	Puesto Sey	Susques
38084040	Huancar	Susques
38084070	Pastos Chicos	Susques
38084060	Olaroz Chico	Susques
38084050	Mina Providencia	Susques
38084030	El Toro	Susques
38084055	Olacapato	Susques
38084045	Jama	Susques
38084100	Susques	Susques
38084010	Catua	Susques
38094050	Tilcara	**Tilcara**
38094040	Maimara	**Tilcara**
38094010	Colonia San Jose	**Tilcara**
38094030	Juella	**Tilcara**
38094020	Huacalera	**Tilcara**
38098010	Barcena	Tumbaya
38098050	Volcan	Tumbaya

UNIVERSE OF DISCOURSE

We always aim at representing **only a part of the real world**.

Universe of discourse

- A **part** of the real world that is **of interest**.

We use data models for representing the universe of discourse and storing that representation in a database.

A.K.A. = the **database miniworld**

It must be well understood by the designers of the DB in order to be able to properly represent it!

UNIVERSE OF DISCOURSE – *EXAMPLE*

Example:

- *We are interested in understanding if and how **crop production** relates to **population size and structure** in different **countries** of the world.*

Part of the real world that constitutes the *universe of discourse*:

- *Crop types*
- *Crop production*
- *Population (size and structure)*
- *Country*

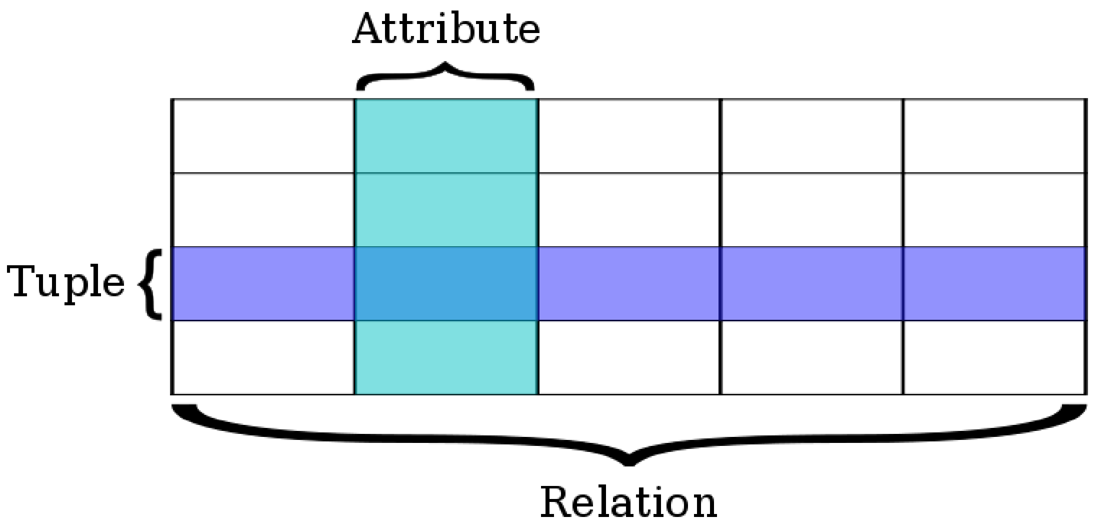
RELATIONAL DATA MODEL

The relational data model is an integrated collection of:

1. Data structuring primitives = **attributes, tuples,** and **relations**
2. Rules of how to structure = **data definition language**
3. Mechanisms to handle the data in a database = **data manipulation language**.

DATA STRUCTURING PRIMITIVES

- **Attribute**: properties that describe an entity
- **Tuple**: the entire row of attribute's values corresponding to a particular entity
- **Relation** *(or **table**)*: a collection of tuples that are similarly shaped



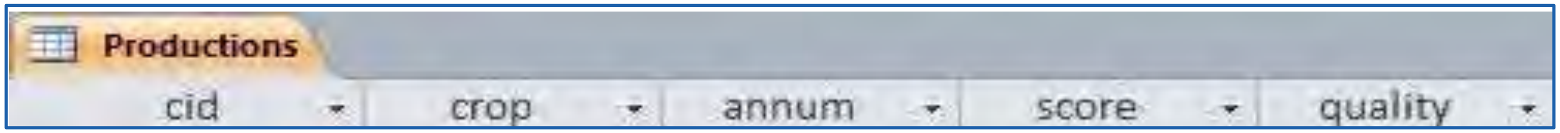
Source: wikipedia

ATTRIBUTE

Attribute

- In the relational data model, we represent **real-world objects** by **tuples** stored in **relations** (*tables*).
- These real-world objects have particular **properties** that describe them. These properties are called **attributes**.

The relation *Productions* has these attributes: *cid, crop*, *annum, score*, *quality*



Productions

cid	crop	annum	score	quality

UNIVERSITY OF TWENTE.

NULL VALUES

Some attributes may have **missing values**. An attribute value may be:

- ‘**unknown**’
- ‘**not applicable**’

$\Big\}$ In SQL we use **NULL** in both cases

NULL *(or Null)*

- A special marker used in SQL to indicate that a data value does not exist in the DB.
- It enables the representation of missing and inapplicable information
- **Not to be confused with a value of 0**

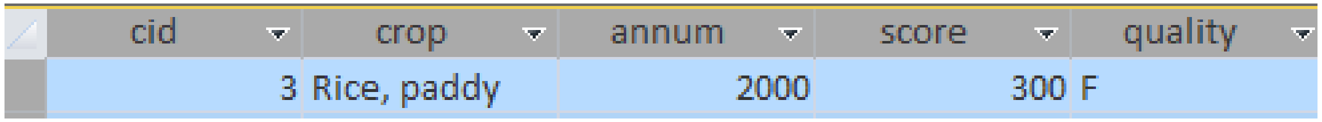
UNIVERSITY OF TWENTE.

TUPLE

Tuple

- A tuple is a **record**
- In the relational data model, tuples represent **real-world objects/phenomena** that have certain attributes
- A tuple can be defined as **a list of attribute values**

One tuple from the relation *Production* is: *(cid=3, crop="Rice,paddy", annum=2000, score=300, quality="F")*



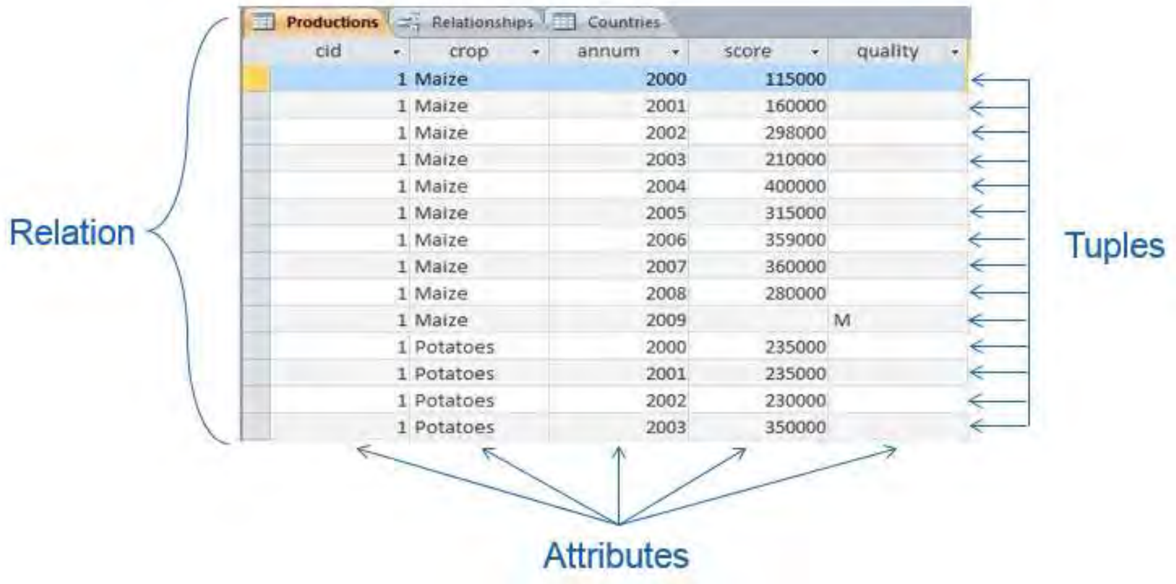
cid	crop	annum	score	quality
3	Rice, paddy	2000	300	F

RELATION (or TABLE)

Relation (*Table*)

- A **collection of tuples** that are **similarly shaped:** all tuples have the same attributes

The relation *Productions* in *FAOcrops.mdb*



cid	crop	annum	score	quality
1	Maize	2000	115000	
1	Maize	2001	160000	
1	Maize	2002	298000	
1	Maize	2003	210000	
1	Maize	2004	400000	
1	Maize	2005	315000	
1	Maize	2006	359000	
1	Maize	2007	360000	
1	Maize	2008	280000	
1	Maize	2009		M
1	Potatoes	2000	235000	
1	Potatoes	2001	235000	
1	Potatoes	2002	230000	
1	Potatoes	2003	350000	

RELATION SCHEMA AND RELATION INSTANCE

Relation schema

- Basic information describing a table or relation:
  - **Name** of the relation;
  - List of **attributes**;
  - **Domain** of each attribute

Relation instance

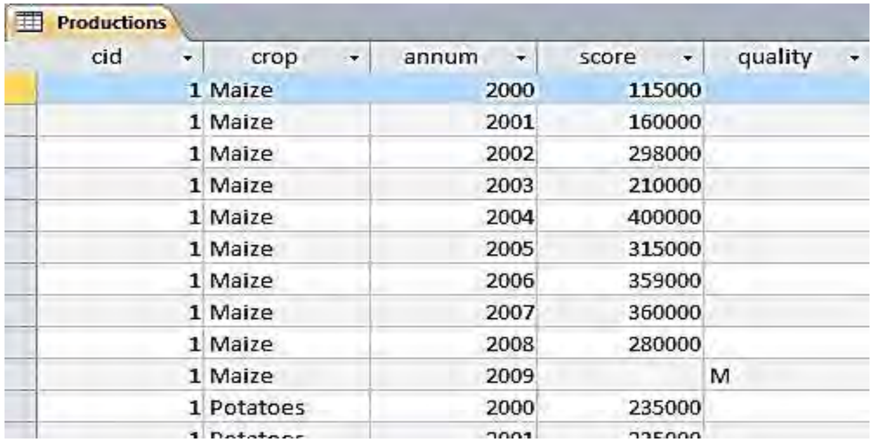
- Set of tuples that adheres to all the requirements that are formulated by the relation schema
- The **set of tuples in a relation at some point in time**

RELATION SCHEMA AND RELATION INSTANCE

- **Relation schema**

Productions (cid: integer, crop: varchar(255), annum: integer, score: integer, quality: varchar(5))

- **Relation instance –**



Productions

cid	crop	annum	score	quality
1	Maize	2000	115000	
1	Maize	2001	160000	
1	Maize	2002	298000	
1	Maize	2003	210000	
1	Maize	2004	400000	
1	Maize	2005	315000	
1	Maize	2006	359000	
1	Maize	2007	360000	
1	Maize	2008	280000	
1	Maize	2009		M
1	Potatoes	2000	235000	
1	Potatoes	2001	[illegible]	

on 18.09.2021 – 15:47 relation *Productions* had five attributes and 6930 tuples

RELATIONAL DATA MODEL

The relational data model is an integrated collection of:

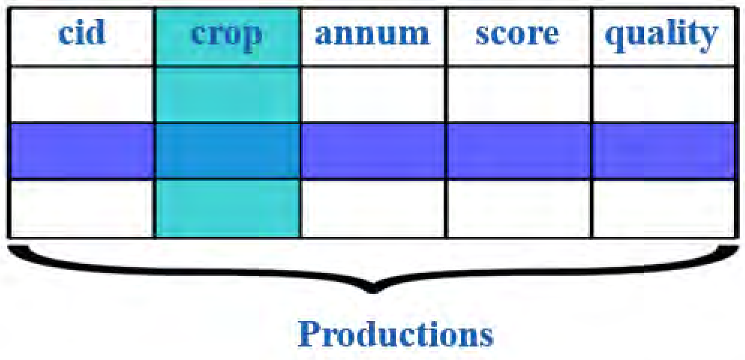
1. Data structuring primitives = **attributes, tuples,** and **relations**
2. Rules of how to structure = **data definition language**,
3. Mechanisms to handle the data in a database = **data manipulation language**.

DATA DEFINITION LANGUAGE

Data Definition Language (DDL)

- A (subset of a) computer language used to create and modify the **structure** of database objects in a database:
- Set of commands that can be used to define the database **schema:** CREATE, ALTER, DROP

```
CREATE TABLE Productions(
                    cid single,
                    crop varchar(255), annum integer,
                    score integer, quality varchar(255)
                    )
```



RELATIONAL DATA MODEL

The relational data model is an integrated collection of:

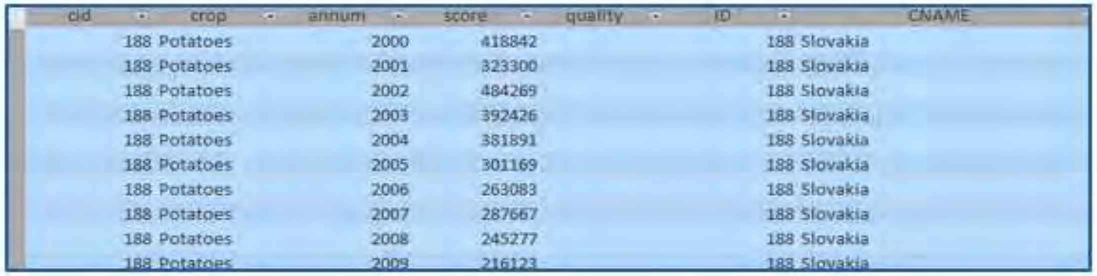
1. Data structuring primitives = **attributes, tuples,** and **relations**
2. Rules of how to structure = **data definition language**,
3. Mechanisms to handle the data in a database = **data manipulation language**.

DATA MANIPULATION LANGUAGE

Data Manipulation Language (DML)

- A (subset of a) computer language used for **adding** (*inserting*), **deleting**, and **modifying** (*updating*) **data** in a DB
- Set of commands that deals with the **manipulation of data:** SELECT, INSERT, UPDATE, DELETE

```
SELECT *
FROM Productions AS p, Countries AS c
WHERE c.ID=p.cid AND p.crop="Potatoes" AND
c.CNAME="Slovakia"
```



cid	crop	annum	score	quality	ID	CNAME
188	Potatoes	2000	418842		188	Slovakia
188	Potatoes	2001	323300		188	Slovakia
188	Potatoes	2002	484269		188	Slovakia
188	Potatoes	2003	392426		188	Slovakia
188	Potatoes	2004	381891		188	Slovakia
188	Potatoes	2005	301169		188	Slovakia
188	Potatoes	2006	263083		188	Slovakia
188	Potatoes	2007	287667		188	Slovakia
188	Potatoes	2008	245277		188	Slovakia
188	Potatoes	2009	216123		188	Slovakia

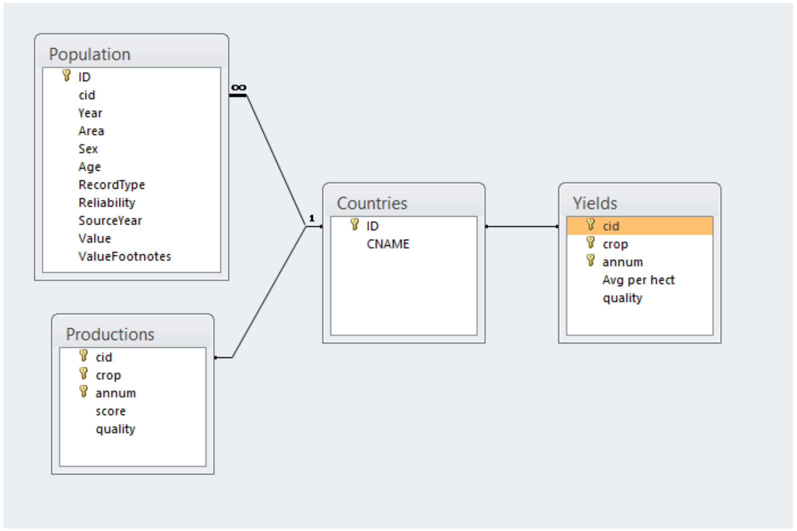
RELATIONAL DATABASE

Relational database

- A DB based on the **relational data model**
- A collection of relations (*tables*) **structured to recognize relations** *(links/associations)* between stored items of information.

Database schema

- The DB **structure** described in a formal language
- A **collection** of relation schemas and the **associations** amongst them
- The **skeleton structure** that represents the **logical view** of the entire database

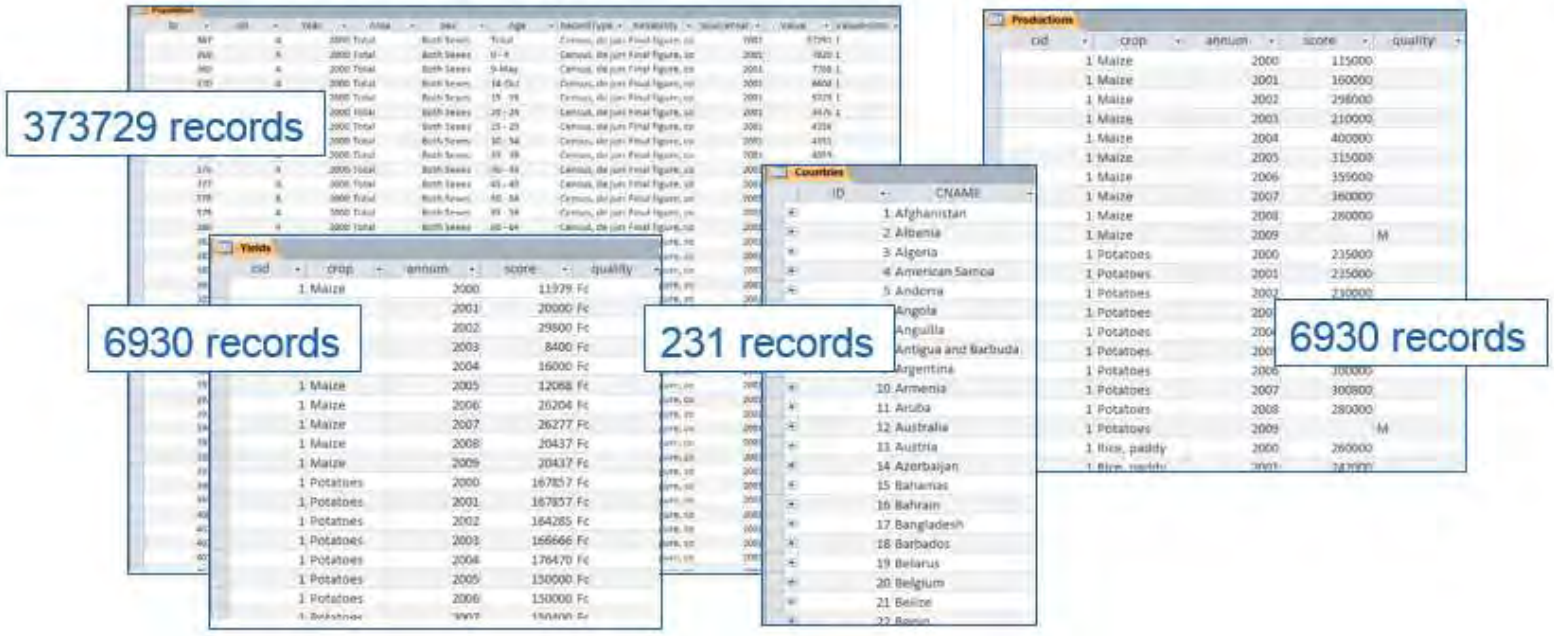


The GDB *FAOcrop.mdb* contains 4 relations: *(Productions, Yields, Countries, Population)*

DATABASE INSTANCE

Database instance

- A **collection of relation instances**, one for each relation in the database schema



Yields

cid	crop	annum	score	quality
1	Maize	2000	11979	Fc
[illegible]	[illegible]	2001	20000	Fc
[illegible]	[illegible]	2002	29500	Fc
[illegible]	[illegible]	2003	8400	Fc
[illegible]	[illegible]	2004	16000	Fc
1	Maize	2005	12088	Fc
1	Maize	2006	20204	Fc
1	Maize	2007	26277	Fc
1	Maize	2008	20437	Fc
1	Maize	2009	20437	Fc
1	Potatoes	2000	167857	Fc
1	Potatoes	2001	167857	Fc
1	Potatoes	2002	164285	Fc
1	Potatoes	2003	166666	Fc
1	Potatoes	2004	176470	Fc
1	Potatoes	2005	150000	Fc
1	Potatoes	2006	150000	Fc

Countries

ID	CNAME
1	Afghanistan
2	Albania
3	Algeria
4	American Samoa
5	Andorra
[illegible]	Angola
[illegible]	Anguilla
[illegible]	Antigua and Barbuda
[illegible]	Argentina
10	Armenia
11	Aruba
12	Australia
13	Austria
14	Azerbaijan
15	Bahamas
16	Bahrain
17	Bangladesh
18	Barbados
19	Belarus
20	Belgium
21	Belize

Productions

cid	crop	annum	score	quality
1	Maize	2000	115000	
1	Maize	2001	160000	
1	Maize	2002	298000	
1	Maize	2003	210000	
1	Maize	2004	400000	
1	Maize	2005	315000	
1	Maize	2006	359000	
1	Maize	2007	360000	
1	Maize	2008	280000	
1	Maize	2009		M
1	Potatoes	2000	235000	
1	Potatoes	2001	235000	
1	Potatoes	2002	210000	
1	Potatoes	[illegible]	[illegible]	
1	Potatoes	[illegible]	[illegible]	
1	Potatoes	[illegible]	[illegible]	
1	Potatoes	2006	300000	
1	Potatoes	2007	300800	
1	Potatoes	2008	280000	
1	Potatoes	2009		M
1	Rice, paddy	2000	260000	

FAOcrops.mdb instance on 18.09.2021 at 15:47

UNIVERSITY OF TWENTE.

INTEGRITY CONSTRAINTS

Integrity constraints

- A **set of rules** that are used **to maintain the quality of information** in a DB
- **Ensure** that the data manipulation is performed in such a way that **data integrity is not affected**.
- Used to **guard against accidental damage/errors** to/in the DB
- Integrity constraints are **specified** when the schema is defined, they are **checked** every time the database is modified and they must be **obeyed at all times**!
- A **valid** relation/database **instance satisfies all specified integrity constraints**.

INTEGRITY CONSTRAINTS

Integrity constraints

- There are various types of integrity constraints that a DB provides

  We will discuss 3 types:

  1. **Domain** integrity constraint
  2. Entity integrity constraint: **Primary key**
  3. Referential integrity constraint: **Foreign key**

INTEGRITY CONSTRAINTS

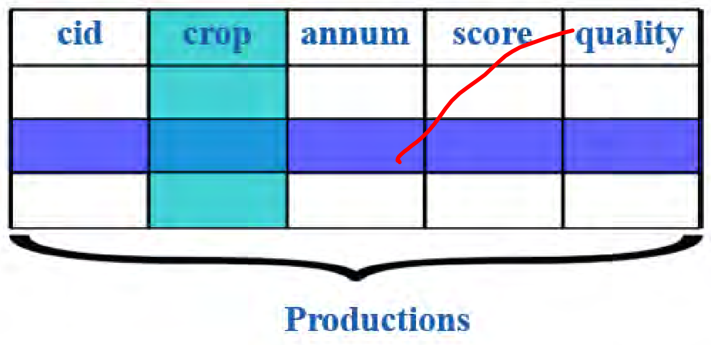
DOMAIN CONSTRAINTS

Domain integrity constraints

- Refers to the **definition of a valid set of values for an attribute**
- In order to add a new value, it needs to meet the criteria defined for that attribute. Otherwise, we get an error

```
CREATE TABLE Productions(
                         cid single,
                         crop varchar(255), annum integer,
                         score integer, quality varchar(255)
                         )
```

primary key



INTEGRITY CONSTRAINTS
PRIMARY KEY

Primary key

- The primary key *K* of a relation *R* is one of (or a combination of) R's attributes, such that:

  1. It is **unique** – there are no two distinct tuples of R that have the same attribute value.
  2. It is **minimal** – there is no proper subset of K that is unique.

- If there is more than one key for R, the database designer chooses one of them to become the ***primary key***. The other remain *candidate keys*.

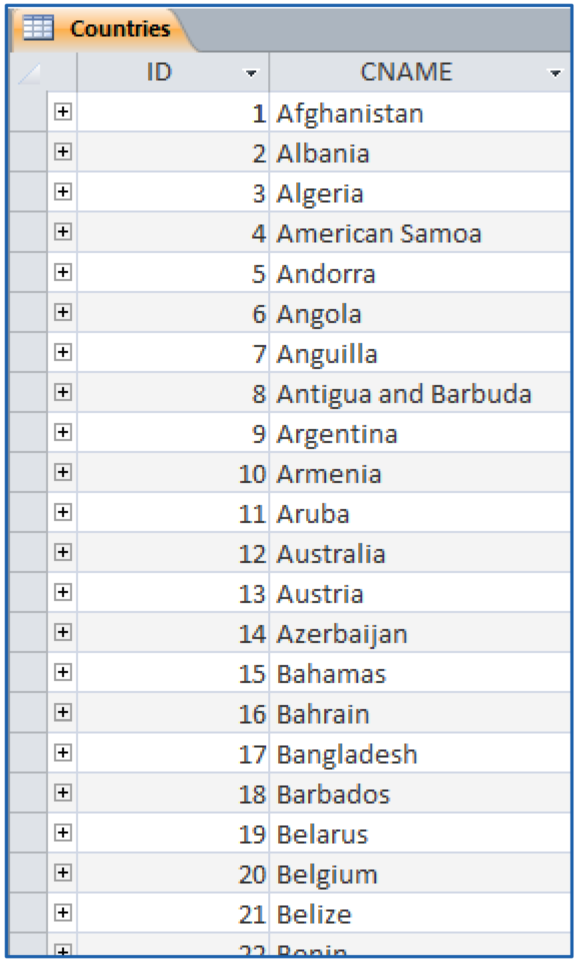
INTEGRITY CONSTRAINTS

PRIMARY KEY

Example – For the relation *Countries* in *FAOcrops.mdb* we can think of three candidate keys:

a. ID
b. CNAME
c. ID+CNAME

Which one would you choose as the primary key? Why?



Countries

ID	CNAME
1	Afghanistan
2	Albania
3	Algeria
4	American Samoa
5	Andorra
6	Angola
7	Anguilla
8	Antigua and Barbuda
9	Argentina
10	Armenia
11	Aruba
12	Australia
13	Austria
14	Azerbaijan
15	Bahamas
16	Bahrain
17	Bangladesh
18	Barbados
19	Belarus
20	Belgium
21	Belize
22	Benin

INTEGRITY CONSTRAINTS
PRIMARY KEY

Whether one (or a set) of the attributes can become a primary key or not can **never** be judged from looking at a relation instance.

- If the attributes seem to have unique combinations of values amongst tuples this may just be a coincidence!

Golden rule of keys:

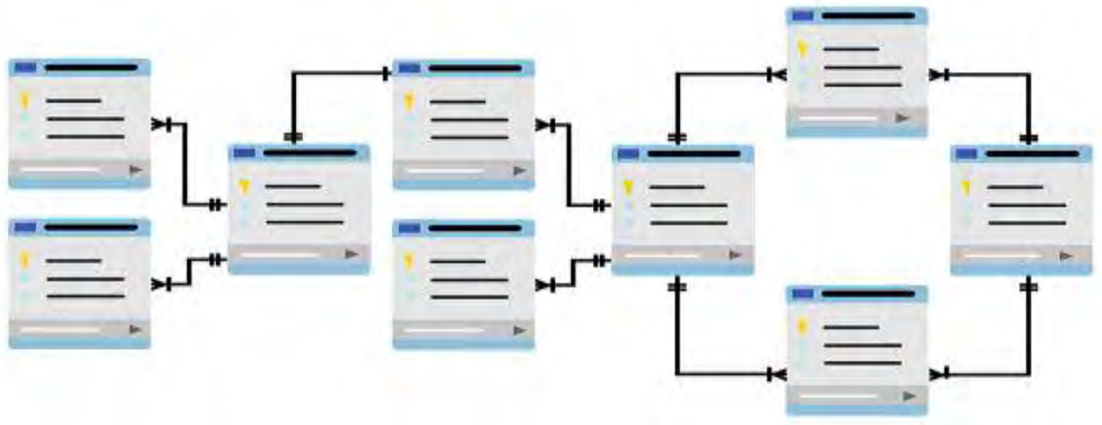
- The suitability to become a primary key must be ascertained by a careful **analysis of the semantics** of the involved relation and attributes, independently of the data that is currently available.
- Each stored relation in a DB **must have** a defined primary key.
- Primary keys are conceived and defined (or even created!) when designing the database schema

INTEGRITY CONSTRAINTS
REFERENTIAL INTEGRITY

Foreign key

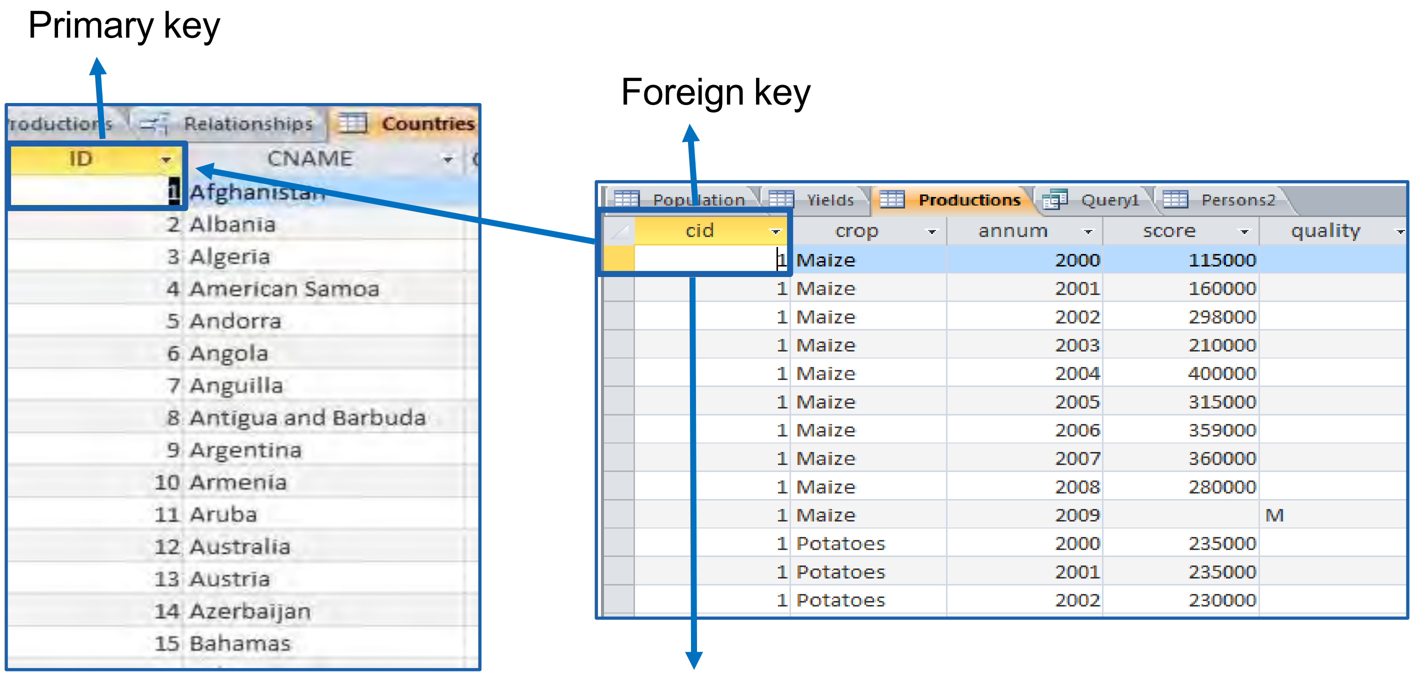
- A set of attributes that is used to **refer to** a tuple in another relation.
- It must correspond with a primary key value in the second relation.
- A foreign key behaves like a 'logical pointer'.

- Links between **primary** and **foreign** keys determine the relationships amongst tables in a DB



INTEGRITY CONSTRAINTS

REFERENTIAL INTEGRITY



ID	CNAME
1	Afghanistan
2	Albania
3	Algeria
4	American Samoa
5	Andorra
6	Angola
7	Anguilla
8	Antigua and Barbuda
9	Argentina
10	Armenia
11	Aruba
12	Australia
13	Austria
14	Azerbaijan
15	Bahamas

cid	crop	annum	score	quality
1	Maize	2000	115000	
1	Maize	2001	160000	
1	Maize	2002	298000	
1	Maize	2003	210000	
1	Maize	2004	400000	
1	Maize	2005	315000	
1	Maize	2006	359000	
1	Maize	2007	360000	
1	Maize	2008	280000	
1	Maize	2009		M
1	Potatoes	2000	235000	
1	Potatoes	2001	235000	
1	Potatoes	2002	230000	

The attribute *cid* is a foreign key in the table *Productions* pointing at the attribute *ID* of the table *Countries*. It only allows:

- ✓ Existing values in *ID*
- ✓ NULL values

UNIVERSITY OF TWENTE.

QUESTIONS?