Les midis numériques

Data conservation - Perspectives, issues and solutions.

Miranda Bryant and Steve Vissault
s.vissault@yahoo.fr

January 30, 2014

All biologists collect data during their career but most of them are using inapropriate files, called *"flat files"*, to long term storage:

- Open Office or Microsoft spreadsheet
- text and CSV files

Some risks attributes at those practices: Overwriting file, lost the full dataset or some records.

Some disadvantage of classic storage file (i. e. Excel)

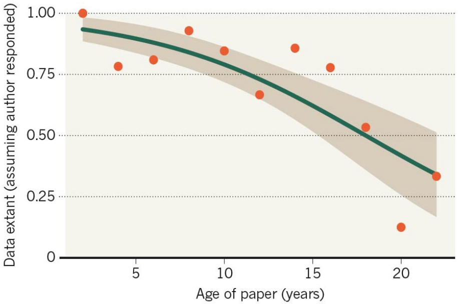
1. No dynamic query, only filters
2. Large dataset could be messy
3. Exportability : Files corrupted, plateform could be different between users
4. Absence of fonctionnality on manage multiple users

Introduction

Why do something different ?

MISSING DATA

As research articles age, the odds of their raw data being extant drop dramatically.



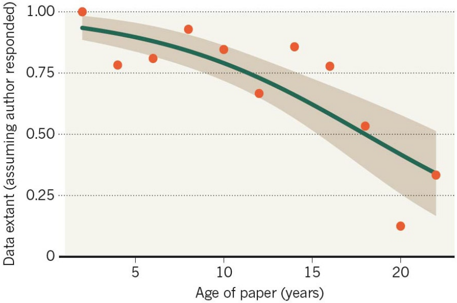
Data for almost all studies published just two years ago were still accessible, the chance of them being so fell by 17% per year (Vines et al., 2013)

Introduction

Why do something different ?

MISSING DATA

As research articles age, the odds of their raw data being extant drop dramatically.



Why ? Because researcher don’t think about long term usability of storage data.

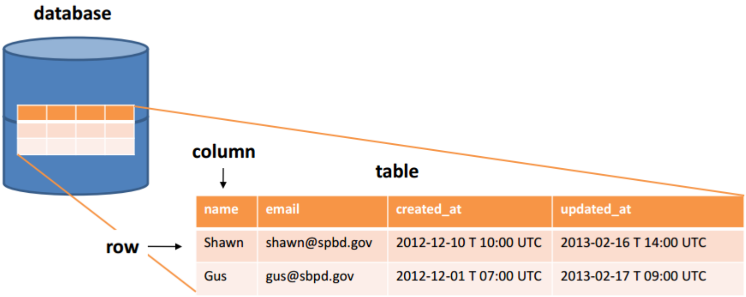
We need to keep focus on those points as a part of our biologist culture:

- All datasets containing specific information given a time and a location are usefull.
- 80% of datasets are built on public funding (Graham, 2013) and could be accessible publicly
- All datasets could be re-used, recycle or valorize (as the 3-R in waste management: Reduce, Reuse, Recycle)

Relational database

What is a Relational Database ?

- A database is basically ”tables”
- A Table goes down a row of items and across many columns of attributes. The data can be organized into different tables.
- The tables have “relations” within and to each other



name	email	created_at	updated_at
Shawn	shawn@spbd.gov	2012-12-10 T 10:00 UTC	2013-02-16 T 14:00 UTC
Gus	gus@sbpd.gov	2012-12-01 T 07:00 UTC	2013-02-17 T 09:00 UTC

Introduction

Why is a relational database a relevant solution for this context ?

Most of Relational databases include:

- **Metadata**: Authors, Year of creation, columns type and description
  - You ensure the happiness of users after 10 years of database no-used
- **Connectivity**: Users can get a remote secure access to your own data
  - You keep the control localy on your data and manage users
- **Exportability:** User can request data from different platforms and languages (i. e. C, C++, R etc...)
- **Provenence:** Store modifications and user-related data changes that allow for “roll back” or “updates” to the data

Relational database

What is a Relational Database ?

Four essential components:

❶ Row or Tuple : “A data set representing a single item”

❷ Column: “A labeled element of a row” such an address, name, etc.

❸ Table: Contains data items in rows and columns

❹ Relationships : Links between tables and within table

Each components is embedded in a design diagram

Relational database

Table example

ID PEP MES	YR	NO ARBRE	LATITUDE	LONGITUDE	DHPMM	CL DRAI	ESSENCE	ST CM2	AGE
710AB	1971	6	49.68091	-74.96537	92	NA	PIG	66.48	39
73096	2005	52	47.88118	-76.17882	125	30	EPN	122.72	93
70004	2015	31	45,53753	-71,08277	101	NA	SAB	80.12	47
76013	1992	2	48,24704	-69,67055	192	10	PET	289.53	52
75010	2007	49	48.43693	-75.91256	NA	30	NA	0.00	18
73008	1980	13	47.50123	-74.38295	187	0	RIR	274.65	62
72094	1987	15	47.13014	-76.02298	161	NA	EPN	203.58	55
70094	2007	10	48.28338	-71.73171	170	20		226.98	68
71006	1992	25	47.32089	-71.11494	162	20	SAB	206.12	45
76095	2003	17	47.38095	-71.72239	277	40	SAB	602.63	54

Design components

Wrong or messy informations

ID PEP MES	YR	NO ARBRE	LATITUDE	LONGITUDE	DHPMM	CL DRAI	ESSENCE	ST CM2	AGE
710AB	1971	6	49.68091	-74.96537	92	NA	PIG	66.48	39
73096	2005	52	47.88118	-76.17882	125	30	EPN	122.72	93
70004	2015	31	45,53753	-71,08277	101	NA	SAB	80.12	47
76013	1992	2	48,24704	-69,67055	192	10	PET	289.53	52
75010	2007	49	48.43693	-75.91256	NA	30	NA	0.00	18
73008	1980	13	47.50123	-74.38295	187	0	RIR	274.65	62
72094	1987	15	47.13014	-76.02298	161	NA	EPN	203.58	55
70094	2007	10	48.28338	-71.73171	170	20		226.98	68
71006	1992	25	47.32089	-71.11494	162	20	SAB	206.12	45
76095	2003	17	47.38095	-71.72239	277	40	SAB	602.63	54

Design components

Redundancy or calculated field

ID PEP MES	YR	NO ARBRE	LATITUDE	LONGITUDE	DHPMM	CL DRAI	ESSENCE	ST CM2	AGE
710AB	1971	6	49.68091	-74.96537	92	NA	PIG	66.48	39
73096	2005	52	47.88118	-76.17882	125	30	EPN	122.72	93
70004	2015	31	45,53753	-71,08277	101	NA	SAB	80.12	47
76013	1992	2	48,24704	-69,67055	192	10	PET	289.53	52
75010	2007	49	48.43693	-75.91256	NA	30	NA	0.00	18
73008	1980	13	47.50123	-74.38295	187	0	RIR	274.65	62
72094	1987	15	47.13014	-76.02298	161	NA	EPN	203.58	55
70094	2007	10	48.28338	-71.73171	170	20		226.98	68
71006	1992	25	47.32089	-71.11494	162	20	SAB	206.12	45
76095	2003	17	47.38095	-71.72239	277	40	SAB	602.63	54

What’s normalization ?

- Do not have the “one file” or “one table” mentality
- If you have redundancy in your table, you need to think about normalization (multiple table design)
- Stages of normalization 1-5 NF (Normal Forms) are the most commonly accepted
  - These are “technical”, but they describe the stages a database development will go through

Table contain keys:

Primary Keys A key that is unique to the table to help identify a record

Composite Keys A key that combines two or more columns to create a unique key into the table

Design components

Relationships

Different type of relationship:

1:1 Each key is linked with only one key in any other table

1:N Each key in one table may be linked to many other keys in another table

N:N One or more keys in a table can be linked to 0, 1, or many rows in another table

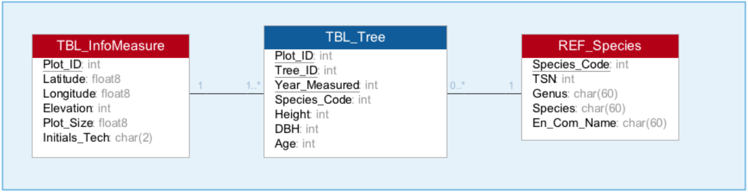
Design components

Last example not normalized

ID PEP MES	YR	NO ARBRE	LATITUDE	LONGITUDE	DHPMM	CL DRAI	ESSENCE	ST CM2	AGE
710AB	1971	6	49.68091	-74.96537	92	NA	PIG	66.48	39
73096	2005	52	47.88118	-76.17882	125	30	EPN	122.72	93
70004	2015	31	45,53753	-71,08277	101	NA	SAB	80.12	47
76013	1992	2	48,24704	-69,67055	192	10	PET	289.53	52
75010	2007	49	48.43693	-75.91256	NA	30	NA	0.00	18
73008	1980	13	47.50123	-74.38295	187	0	RIR	274.65	62
72094	1987	15	47.13014	-76.02298	161	NA	EPN	203.58	55
70094	2007	10	48.28338	-71.73171	170	20		226.98	68
71006	1992	25	47.32089	-71.11494	162	20	SAB	206.12	45
76095	2003	17	47.38095	-71.72239	277	40	SAB	602.63	54

New and cleanest design

Need to be transform to the cleanest design without redundancy



TBL_InfoMeasure
Plot_ID: int
Latitude: float8
Longitude: float8
Elevation: int
Plot_Size: float8
Initials_Tech: char(2)

TBL_Tree
Plot_ID: int
Tree_ID: int
Year_Measured: int
Species_Code: int
Height: int
DBH: int
Age: int

REF_Species
Species_Code: int
TSN: int
Genus: char(60)
Species: char(60)
En_Com_Name: char(60)

What does SQL mean?

- “Structured Query Language”
- This language was designed to be cross-platform to handle data in relational databases
- The International Organization for Standardization the language in 1981
- Provides cross-platform consistence (most of the time)

How to create this design ?

Create table

Table example creation:

SQL Code

```
CREATE TABLE "TBL-Tree" (
        ”Plot-ID” int NOT NULL,
        ”Tree-ID” int NOT NULL,
        ”Year-Measured” numeric(4) NOT NULL,
        ”Species-Code” char(3) NULL,
        ”Height” float8 NULL,
        ”DBH” numeric(5) NULL,
        ”Age” int NULL,
PRIMARY KEY (”Plot-ID”, ”Tree-ID”, ”Year-Measured”)
);
```

Basic manipulation

Update

Simple to update to a large amount of data:

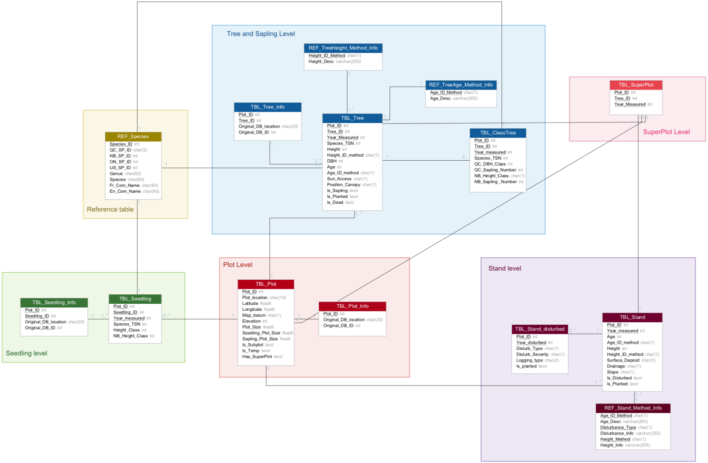
SQL Code

```
UPDATE Tree-Species
      SET genus-name = ‘Acer’
      WHERE species-id = ’sacchaurm’;
```

Every update is recorded so if there is an error, it can be “rolled back” (undo)

QUICC-FOR Example

Diagram



Tree and Sapling Level

REF_TreeHeight_Method_Info
Height_ID_Method: char(1)
Height_Desc: varchar(255)

REF_TreeAge_Method_Info
Age_ID_Method: char(1)
Age_Desc: varchar(255)

TBL_Tree_Info
Plot_ID: int
Tree_ID: int
Original_DB_location: char(20)
Original_DB_ID: int

TBL_Tree
Plot_ID: int
Tree_ID: int
Year_Measured: int
Species_TSN: int
Height: int
Height_ID_method: char(1)
DBH: int
Age: int
Age_ID_method: char(1)
Sun_Access: char(1)
Position_Canopy: char(1)
Is_Sapling: bool
Is_Planted: bool
Is_Dead: bool

TBL_ClassTree
Plot_ID: int
Tree_ID: int
Year_measured: int
Species_TSN: int
QC_DBH_Class: int
QC_Sapling_Number: int
NB_Height_Class: char(1)
NB_Sapling _Number: int

SuperPlot Level

TBL_SuperPlot
Plot_ID: int
Tree_ID: int
Year_Measured: int

Reference table

REF_Species
Species_ID: int
QC_SP_ID: char(3)
NB_SP_ID: int
ON_SP_ID: int
US_SP_ID: int
Genus: char(60)
Species: char(60)
Fr_Com_Name: char(60)
En_Com_Name: char(60)

Seedling level

TBL_Seedling_Info
Plot_ID: int
Seedling_ID: int
Original_DB_location: char(20)
Original_DB_ID: int

TBL_Seedling
Plot_ID: int
Seedling_ID: int
Year_measured: int
Species_TSN: int
Height_Class: int
NB_Height_Class: int

Plot Level

TBL_Plot
Plot_ID: int
Plot_location: char(10)
Latitude: float8
Longitude: float8
Map_datum: char(1)
Elevation: int
Plot_Size: float8
Seedling_Plot_Size: float8
Sapling_Plot_Size: float8
Is_Subplot: bool
Is_Temp: bool
Has_SuperPlot: bool

TBL_Plot_Info
Plot_ID: int
Original_DB_location: char(20)
Original_DB_ID: int

Stand level

TBL_Stand_disturbed
Plot_ID: int
Year_disturbed: int
Disturb_Type: char(1)
Disturb_Severity: char(1)
Logging_type: char(2)
Is_planted: bool

TBL_Stand
Plot_ID: int
Year_measured: int
Age: int
Age_ID_method: char(1)
Height: int
Height_ID_method: char(1)
Surface_Deposit: char(3)
Drainage: char(1)
Slope: char(1)
Is_Disturbed: bool
Is_Planted: bool

REF_Stand_Method_Info
Age_ID_Method: char(1)
Age_Desc: varchar(255)
Disturbance_Type: char(1)
Disturbance_Info: varchar(255)
Height_Method: char(1)
Height_Info: varchar(255)

QUICC FOR – A real-life example

- Data from two countries, over many years and many sources
- Data is in varying formats, all publicly available
- Created a normalized Database to store data
- Created a schema to merge and clean data
- Then worked to create a new schema that incorporates all data

Some disadvantages:

- New system to learn that is not always intuitive
- System has to be setup to run a database
- Poor database design could be “worse” than a basic flat file
- Maintenance cost can be higher

Some advantages:

- “Coarse” data is never touched except by certain users
- Most users are always working on a query
- Data is publicly available but more secure
- Multiple users can access data
- Log of all modifications with username
- Ability to “roll back” changes
- Size of data only limited by storage
- Secure, long distance connections to data

Where can SQL be used ?

- R
- Java
- C
- C++
- Perl
- Python
- .Net

What's metadata ?

Defined: "Metadata is data about data"

Examples: Author, Journal Title, Edition, Publication Date or Tree description, date updated, collection date

According to Goodman et al. (2014), we need to keep focus on those points:

1. Love your data and help others love it too
2. Share your data online with a permanent identifier
3. Conduct science with a particular level of reuse in mind

A. Goodman, A. Pepe, and A. Blocker. 10 Simple Rules for the Care and Feeding of Scientific Data. *arXiv preprint arXiv: . . .* , pages 1–9, 2014.

D. Graham. Academic Publishing - Survey of funders supports the benign Open Access outcome priced into shares. Technical report, HSBC - Global Research, 2013.

T. Poisot, R. Mounce, and D. Gravel. Moving toward a sustainable ecological science: don't let data go to waste! *Ideas in Ecology and Evolution*, 6(2): 11–19, 2013. ISSN 19183178. doi: 10.4033/iee.2013.6b.14.f.

T. Vines, A. Albert, R. Andrew, and F. Débarre. The availability of research data declines rapidly with article age. *Current Biology*, 2013.