Relational Algebra for Excel 2.0

matti@belle-nuit.com 4.4.2017

Introduction

- Relational Algebra for Excel is a collection of custom functions to make calculations with relations or, rephrased, use Excel as a database.

- You can use these functions to query data in sour Excel Sheets with the same expressive power as query languages like SQL.

- The function can handle tables with 500-4000 rows.

Why use it?

- Excel provides filters for data, which is powerful, but not persistent. You loose a query, when you make the next one. Also, you can search only in one table

- Pivot tables can combine data from multiple tables, but is neither intuitive nor flexible nor persistant.

- There are SQL plugins, but they act as macro commands and are static. Relational Algebra for Excel uses functions, the query results update dynamically when you edit cells.

Installation

- All VBA code is in one module. You can either add the module to your sheet or you can install the add-in and use the functions on all sheets on your computer.

- You must save your sheet as Excel sheet with macros and you must enable macros to use it.

- If you use the add-in, you must give it also to the people when you share the sheet.

- Once installed, you can use the functions. They all have the prefix "rel".

Set theory

- Relational Algebra has evolved from the set theory you may have experienced in school.
- A set is a collection zero of more elements, where each element is unique.
- S = {A, B} is a set with the elements A and B.
- A ε S A is an element of S
- {A} ⊂ {A,B} is a subset
- {} or Ø is an empty set.

Relation

- A **relation** is a set of zero or more tuples that share the same properties.

- The **cardinality** of a relation is the number of tuples. The empty relation {} or ∅ has no tuples and the cardinality 0

- A **tuple** is a set of zero or more property-value pairs. Each property has its domain. A domain is the set of all possible values. $\mathbb{N}$ is a domain for example.

- The **arity** is the number of properties of the tuples in a relation. The properties do not have a particular order.

Tables

	A	B	C
1	id	title	country
2	1001	Ma vie de Courgette	CH
3	1002	Elle	FR
4	1003	Toni Erdmann	DE
5	1004	Above And Below	CH

- In Excel are the tuples and columns are the properties.
- Row and column order are not significant and that each row ist unique.
- Tables have always a column header.
- Name the cell ranges before you start. **Films** is more readable than **$A1:$C5**

Internal representation of the relation

```
filmid::title::country
1001::Ma vie de Courgette::CH
1002::Elle::FR
1003::Toni Erdmann::DE
1004::Above And Below::CH
```

- Relational algebra operates on relations and the result is always a relation.
- All functions work on a single string. It uses the separator "::" for the properties and space+newline for the tuple.
- Set cell wrap to see multiple lines.

Limitations

- All properties have the same domain: string. The following characters cannot be used in a value because they are separators: newline and "::"'
- The property names must start with a letter and do not have spaces.
- Excel limitation: A relation in a cell cannot have more than 32K characters.

Convert between relation and cells

- **relRange(range)** reads a range of cells into a relation.
- Most functions convert a cell range implicitly into a relation
- **relCell(relation, row, column, isNumber, noError)** reads a single value out of a relation
- **relCellArray(relation)** used as array function reads a relation into a cell array
- **relFilter** can return directly a single value if the relation is a single column and a single row.

Use of the functions

You can work in two ways:

- Use the various functions (relSelect, relProject, relJoin) individually and combine them as Excel functions.

- Use the **relFilter** as single function and pile all operators on a stack. relFilter handles better data volume. The 32k limit only applies on the end result but not on the intermediate data.

Internal representation of the relation

```
filmid::title::country
1001::Ma vie de Courgette::CH
1002::Elle::FR
1003::Toni Erdmann::DE
1004::Above And Below::CH
```

- Relational algebra operates on relations and the result is always a relation.
- All functions work on a single string. It uses the separator "::" for the properties and space+newline for the tuple.
- Set cell wrap to see multiple lines.

Union

fiction	doc	fiction ∪ doc
filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE	filmid::title::country 1004::Above And Below::CH	filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH

- = relUnion(fiction,doc)
- = relFilter(fiction, doc, "U")
- SELECT filmid, title, country FROM fiction UNION SELECT filmid, title, country FROM doc
- Both relations must have the same arity and the same properties

Intersection

fiction	swissfilms	fiction ∩ swissfilms
filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE	filmid::title::country 1001::Ma vie de Courgette::CH 1004::Above And Below::CH	filmid::title::country 1001::Ma vie de Courgette::CH

- = relIntersect(fiction,swissfilms)
- = relFilter(fiction, swissfilms, "I")
- SELECT filmid, title, country FROM fiction INTERSECT SELECT filmid, title, country FROM swissfilms
- Both relations must have the same arity and the same properties

Difference

films	swissfilms	films - swissfilms
filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH	filmid::title::country 1001::Ma vie de Courgette::CH 1004::Above And Below::CH	filmid::title::country 1002::Elle::FR 1003::Torni Erdmann::DE

- = relDifference(films,swissfilms)
- = relFilter(films, swissfilms, "D")
- SELECT filmid, title, country FROM films DIFFERENCE SELECT filmid, title, country FROM swissfilms
- Both relations must have the same arity and the same properties

Selection

films	$\delta_{\text{country="CH"}}$films
filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH	filmid::title::country 1001:::Ma vie de Courgette::CH 1004::Above And Below::CH

- = relSelect(films,"$country=""CH""")
- = relFilter(films,"S $country=""CH""")
- SELECT filmid, title, country FROM films WHERE country = 'CH'
- Selection expression can use any column, Excel formula and cell references and must evaluate to true or false.
- Data type ad hoc: A column preceded by $ is used as string, preceded by % is used as number.
- Use double quotes when needed. Keep cell references outside the quoted text, so that they are updated.

Projection

films	$\pi_{country}$ films
filmid::title::country	country
1001::Ma vie de Courgette::CH	CH
1002::Elle::FR	FR
1003::Torni Erdmann::DE	DE
1004::Above And Below::CH	

- = relProject(films,"country")

- = relFilter(films,"P country")

- SELECT country FROM films

- Projection list can have multiple columns, separated by ::

Rename

films	$\delta_{\text{filmid isan}}$ films
filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH	isan::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH

- = relRename(films,"filmid id")
- = relFilter(films,"R filmid isan")
- SELECT filmid as isan, title, country FROM films
- The rename operator will be important for joins
- Multiple renames are possible separated by ::

Natural Join

films	theatres	films ⋈ theatres
filmid::title::country	theatreid::theatre::filmid	filmid::title::country::theatreid::theatre
1001::Ma vie de Courgette::CH	21::Corso::1003	1001::Ma vie de Courgette::22::Apollo
1002::Elle::FR	22::Apollo::1001	1001::Ma vie de Courgette::23::Metropol
1003::Torni Erdmann::DE	23::Metropol::1001	1002::Elle::24::Le Paris
1004::Above And Below::CH	24::Le Paris::1002	1003::Toni Erdmann::21::Corso

- = relJoin(films,theatres,"NATURAL")
- = relFilter(films, theatres, "J NATURAL")
- SELECT filmid, title, country FROM films JOIN SELECT theatreid, theatre, filmid JOIN ON filmid
- Natural Join is based on common properties

Theta Join

films	theatres	films $_{\theta\ \text{id = filmid}}$ theatres
id::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH	theatreid::theatre::filmid 21::Corso::1003 22::Apollo::1001 23::Metropol::1001 24::Le Paris::1002	id::title::country::theatreid::theatre::filmid 1001::Ma vie de Courgette::22::Apollo::1001 1001::Ma vie de Courgette::23::Metropol::1001 1002::Elle::24::Le Paris::1002 1003::Toni Erdmann::21::Corso::1003

- = relJoin(films,theatres,"%id=%filmic")
- = relFilter(films, theatres, "J %id=%filmid")
- SELECT filmid, title, country, theatreid, theatre FROM films, theatres WHERE filmid = id
- Theta Join allows any expression like the select expression
- Table namespace is not supported. You may need to rename before join

Cross Product

films	theatres	films × theatres
Id::title::country	theatreid::theatre::filmid	id::title::country::theatreid::theatre::filmid
1001::Ma vie de Courgette::CH	21::Corso::1003	1001::Ma vie de Courgette::21::Corso::1003
1002::Elle::FR	22::Apollo::1001	1001::Ma vie de Courgette::22::Apollo::1001
1003::Torni Erdmann::DE	23::Metropol::1001	1001::Ma vie de Courgette::23::Metropol::1001
1004::Above And Below::CH	24::Le Paris::1002	1001::Ma vie de Courgette::24::Le Paris::1002
		1002::Elle::21::Corso::1003
		1002::Elle::22::Apollo::1002 and 10 others

- = relJoin(films,theatres,"TRUE")
- = relFilter(films, theatres, "J TRUE")
- SELECT filmid, title, country, theatreid, theatre FROM films, theatre

Other joins

- Left Join ⋉
- Right Join ⋊
- Outer Join
- Left Semi Join
- Right Semi Join
- Left Anti Semi Join
- Right Anti Semi Join

Aggregation (not relational)

films	
filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH	country::filmid_count CH::2 FR::1 DE::1

- = relProject(films,"country::filmid COUNT")
- = relFilter(films,"P country::filmid COUNT")
- SELECT country, COUNT(filmid) FROM films GROUP BY country
- Other aggregators: SUM, MIN, MAX, AVG

Order (not relational)

films	
filmid::title::country	filmid::title::country
1001::Ma vie de Courgette::CH	1004::Above And Below::CH
1002::Elle::FR	1001::Ma vie de Courgette::CH
1003::Torni Erdmann::DE	1003::Torni Erdmann::DE
1004::Above And Below::CH	1002::Elle::FR

- = relOrder(films,"country::title")
- = relFilter(films,"O country::title")
- SELECT filmid, title, country FROM films ORDER BY country, films
- Multiple columns are separated by ::
- Order can be specified with modifiers: A Z 1 9 for alphabetic or numeric, normal or reverse

Limit (not relational)

films	
filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH	filmid::title::country 1002::Elle::FR 1003::Torni Erdmann::DE

- = relLimit(films,2,2)
- = relFilter(films,"L 2 2")
- SELECT filmid, title, country FROM films START 2 LIMIT 2
- Order before you limit

Extend (not relational)

films	
filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH	filmid::title::country::sfid 1001::Ma vie de Courgette::CH::1 1002::Elle::FR::2 1003::Torni Erdmann::DE::3 1004::Above And Below::CH::4

- = relExtend(films,"sfid","%filmid - 1000")

- = relFilter(films,"E sfid %filmid - 1000")

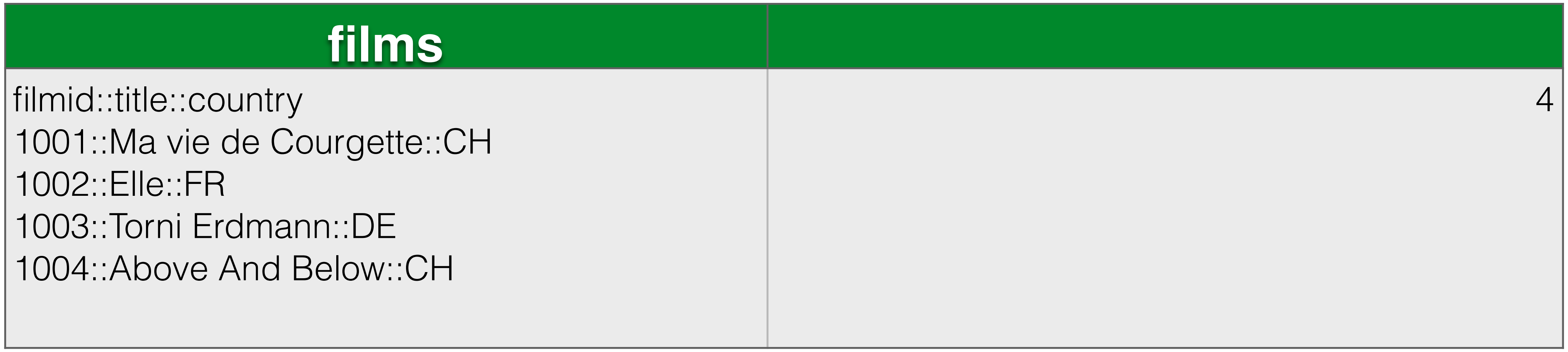
- SELECT filmid, title, country, (filmid-1000) as sfid FROM films

- Extension expression can use any column, Excel formula and cell references and must evaluate to true or false.

- Data type ad hoc: A column preceded by $ is used as string, preceded by % is used as number.

- Use double quotes when needed. Keep cell references outside the quoted text, so that they are updated.

Return single value (not relational)



films	
filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH	4

- = relFilter(films,"P filmid COUNT","Z")

- If the operators return a relation with only one row and one column, you can drop the header and return directly the value

- "Z" value as number

- "K" value as text

- "C" value automatic depending if there is a numeric e

Example 1

films	π title δ country="CH" films
filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH	title Ma vie de Courgette Above and Below

- Return the title of all Swiss movies
- = relProject(relSelect(films,"$country=""CH"""),"title")
- = relFilter(films, "S $country=""CH""", "P title")

Example 2

films	theatres	$\pi_{\text{title, theatre}}$ films ⋈ theatres
filmid::title::country	theatreid::theatre::filmid	title:theatre
1001::Ma vie de Courgette::CH	21::Corso::1003	Ma vie de Courgette::Apollo
1002::Elle::FR	22::Apollo::1001	Ma vie de Courgette::Metropol
1003::Torni Erdmann::DE	23::Metropol::1001	Elle::Le Paris
1004::Above And Below::CH	24::Le Paris::1002	Toni Erdmann::Corso

- Show title of all films and the name theatres they are shown
- = relProject(relJoin(films,theatres,"NATURAL"),"title::theatre")
- = relFilter(films, theatres, "J NATURAL","P title::theatres)

Example 3

films	theatres	π_{filmid} **films -** π_{filmid} **films ⋈ theatres**
filmid::title::country 1001::Ma vie de Courgette::CH 1002::Elle::FR 1003::Torni Erdmann::DE 1004::Above And Below::CH	theatreid::theatre::filmid 21::Corso::1003 22::Apollo::1001 23::Metropol::1001 24::Le Paris::1002	filmid 1004

- Show the title of the films that are not shown
- = relDifference(relProject(films,"filmid"), relProject(relJoin(films,theatres,"NATURAL"),"filmid"))
- = relFilter(films, "P filmid", films, theatres, "J NATURAL","P filmid,"D")
- = relFilter(films, theatres, "J leftantisemi", "P filmid")

Other functions

- Rotate
- Fixpoint
- Assert
- Special operators in relFilter
  - # starts a comment
  - ! stops execution (debugging)