Flatten from/to Relational

Jácome Cunha* João Saraiva* Joost Visser†

CIC 2007

22-23 October

*Universidade do Minho

†Software Improvement Group

Overview

Introduction

The Transformation

Example

Conclusions

Introduction

Motivation

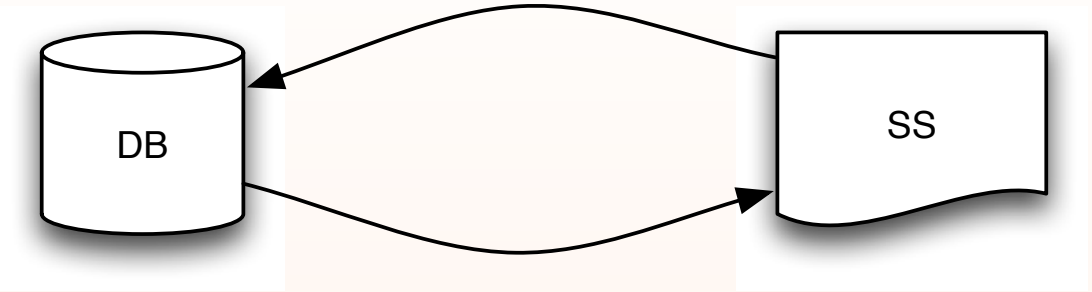
Spreadsheets are considered one of the largest programming languages in the world

Their languages/systems lack structured programming features

Programming in a spreadsheet environment is an error prone task

Data manipulation is not so supported as in other paradigms

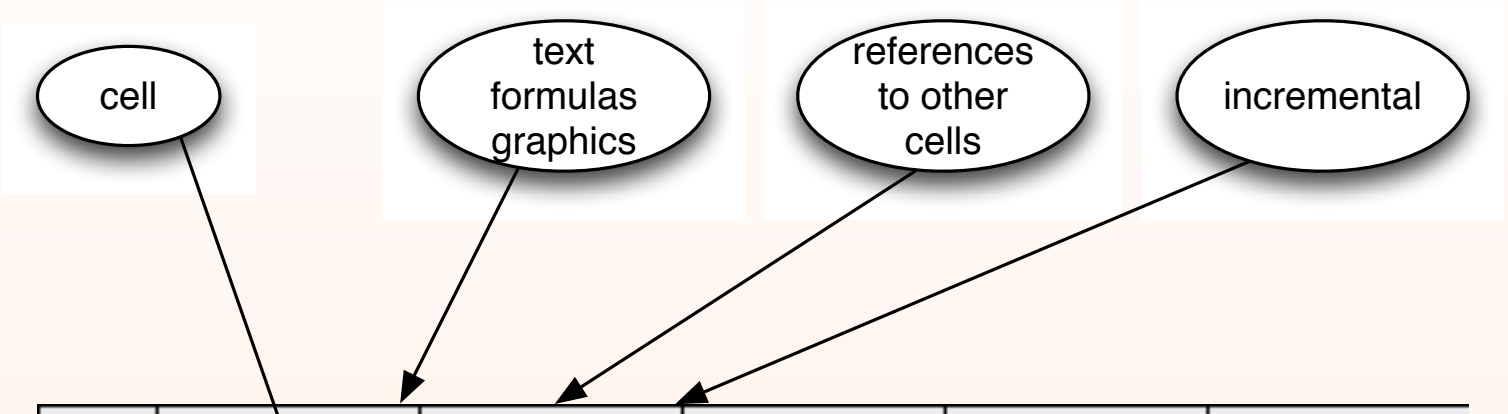
Bidirectional transformation



The flatten model of a spreadsheet is mapped into a relational database model

The other way around is also shown

What is a spreadsheet?



	A	B	C	D	E
1					
2		Name	Id	Age	
3		Joe	123	23	
4		Mary	221	20	
5		John	322	34	
6					

Relational database

It is a structured collection of records distributed over tables

Some concepts:

Primary key it is a set of attributes in the record that uniquely determine all the others

Functional dependency a set of attributes X *functionally determines* a set of attributes Y iff each X value is associated with only one Y value

Foreign key it is a reference to another entry in other table

Advantages of DB

The existence of database management system (DBMS) allows:

- insert, remove and update data in the DB
- chance the format of the DB
- ensures the integrity of the DB
- backup and replication
- give permissions to users or deny them

Examples:

- Oracle, Microsoft Access, MySQL, PostgreSQL

The Transformation

The big picture

$$\begin{pmatrix} a_1 1 & \dots & a_1 n \\ \vdots & & \vdots \\ a_m 1 & \dots & a_{mn} \end{pmatrix} \quad :: \quad (A_1 \times \ldots \times A_n)^*$$

$\searrow\downarrow$ extract FDs

$\downarrow$ 2LT $\qquad\qquad$ $\Pi(A_i \ldots A_k \rightharpoonup A_j \ldots A_l)$

$\downarrow$ normalisation

$$(a_m \ldots a_v \rightharpoonup a_n \ldots a_y)^* \quad :: \quad \Pi(A_m \ldots A_v \rightharpoonup A_n \ldots A_t)$$

Inferring FDs

We assume that the table has only data, not code or formulas or other things

Information about relation between columns is needed otherwise there will be too many FDs and will become useless

The FUN algorithm (Noel Novelli and Rosine Cicchetti) is used to infer the FDs

Normalisation

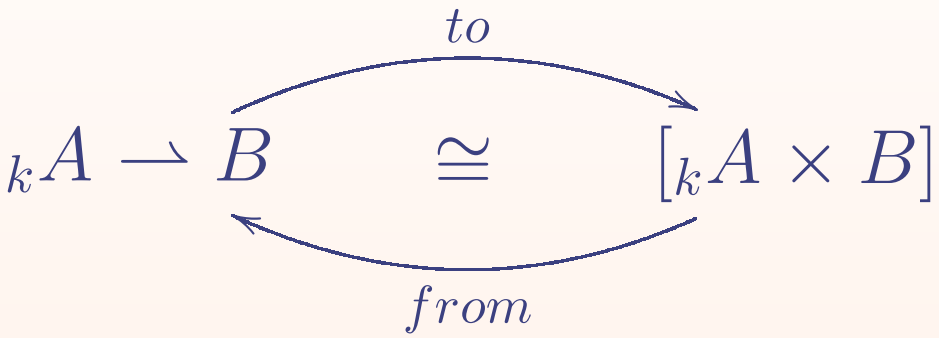
For each association of columns several FDs are yielded by the algorithm

One is chosen and a table is created according to it

The columns that are not referred are collected into a table where the primary key is a set constituted by the primary keys from other tables

Data migration

Performed by implementing refinement laws into 2LT



Abstraction (`from`) and *representation* (`to`) functions are needed to witness the refinement

Example

Internships

	A	B	C	D	E	F	G	H	I	J
1	companie	contacts	location	salary	status	description	time	st.number	st.name	graduation
2										
3	emp1	253555555	braga	400	canceled	tester	6 months	30000	antonio	lesi
4	emp1	253555555	braga	400	ok	programmer	half year	30001	joaquim	mcc
5	emp1	253555555	braga	250	ok	designer	3 mothes	30002	manuel	cc
6	emp2	253666666	porto	300	ok	html prog	5 monthes	30000	antonio	lesi
7	emp2	253666666	porto	500	ok	designer	half year	30003	jorge	lei
8	emp3	323232323	guimaraes	500	canceled	pm	3 months	30004	paulo	mcc
9	emp4	344444444	taipas	400	ok	tester	a year	30004	paulo	mcc
10	emp5	253666666	amares	300	ok	programmer	12 months	30005	joao	lesi
11	emp5	253666666	amares	200	ok	pm	5 monthes	30006	miguel	mcc
12	emp7	444444444	amares	500	ok	programmer	5 monthes	30007	carlos	cc

Internships

	A	B	C	D	E	F	G	H	I	J
1	companie	contacts	location	salary	status	description	time	st.number	st.name	graduation
2										
3	emp1	253555555	braga	400	canceled	tester	6 months	30000	antonio	lesi
4	emp1	253555555	braga	400	ok	programmer	half year	30001	joaquim	mcc
5	emp1	253555555	braga	250	ok	designer	3 mothes	30002	manuel	cc
6	emp2	253666666	porto	300	ok	html prog	5 monthes	30000	antonio	lesi
7	emp2	253666666	porto	500	ok	designer	half year	30003	jorge	lei
8	emp3	323232323	guimaraes	500	canceled	pm	3 months	30004	paulo	mcc
9	emp4	344444444	taipas	400	ok	tester	a year	30004	paulo	mcc
10	emp5	253666666	amares	300	ok	programmer	12 months	30005	joao	lesi
11	emp5	253666666	amares	200	ok	pm	5 monthes	30006	miguel	mcc
12	emp7	444444444	amares	500	ok	programmer	5 monthes	30007	carlos	cc

There should be "enough" cases to represent the reality

Company's table

`company`, `contacts` and `location` are related

```
company -> contacts, location
contacts -> company, location
```

Company's table

`company`, `contacts` and `location` are related

```
company -> contacts, location
contacts -> company, location
```

Student's tables

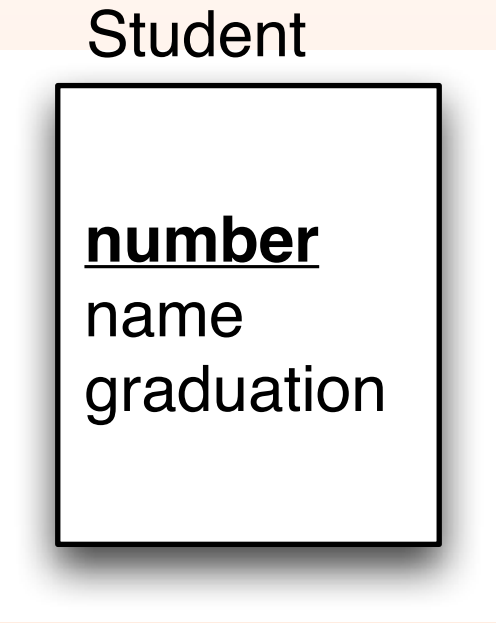
`st.number, st.name` and `graduation` are also related

```
st.number -> st.name, graduation
```

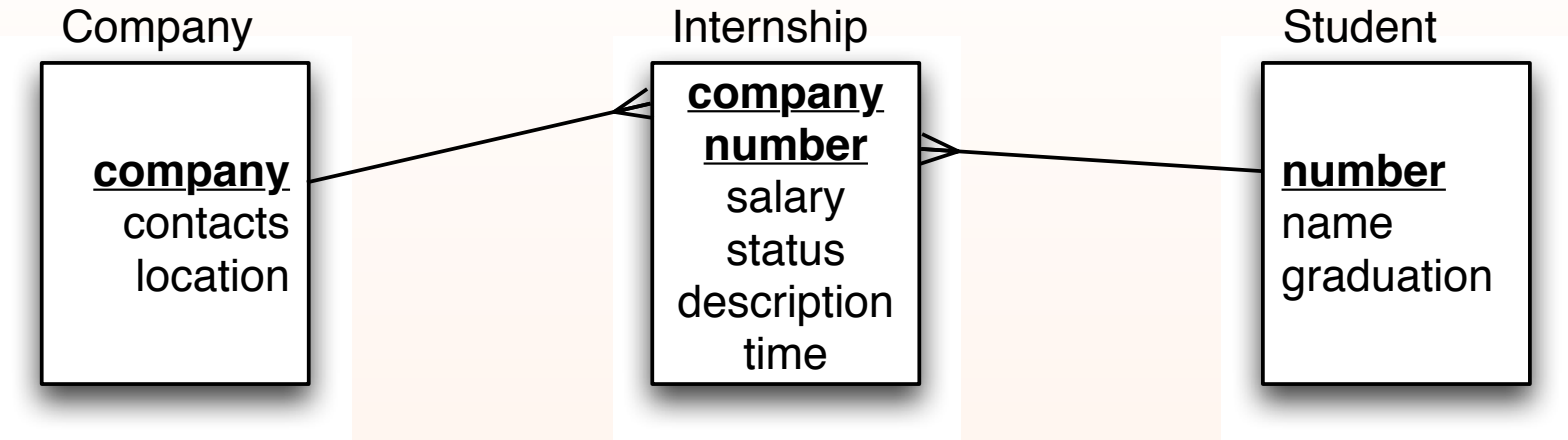
Student's tables

`st.number, st.name` and `graduation` are also related

```
st.number -> st.name, graduation
```



DB model




```
Company (_company, contacts, location)

Student (_st.number, st.name, graduation)

Internship (_company, _st.number, salary, status, description, time)
  foreign key (company)
    references Company(company)
  foreign key (st.number)
    references Student(st.number)
```

Conclusions

Conclusions

Start with a flatten data table, a spreadsheet

Determine FDs (automatically) and the relational DB schema (not so automatically)

Allows the migration to a well known and supported paradigm

Future work

Import a spreadsheet (and other formats) to HASKELL

Tune the FDs inference/normalisation

Integration into 2LT framework

Create ways of export the new model to SQL or to spreadsheet