

Structuring the Unstructured: How to Dimensionalize Semi-Structured Business Data



The Business Intelligence industry is paying more and more attention to the ever-growing heap of unstructured and semi-structured business data. However, the price of structuring this data for the purposes of performing business analysis is quite steep. The focus of this paper is a fact-driven approach for converting semi-structured business spreadsheets into well-structured multidimensional models ready to be analyzed by any Business Intelligence system.

28 West 44th Street
New York, NY 10036
646-439-1618

www.interactiveedge.com

Semi-structured business spreadsheets

This whitepaper focuses on one specific part of business content – the semi-structured worksheets typically originating from collaboration, planning, reporting and other internal and external business processes. We will discuss the major problems that plague attempts to integrate these data islands with the enterprise data warehouse as well as propose a new approach to resolving these issues.

Let's start by examining the semi-structured characteristics of a typical business spreadsheet. As an example, we will use a 15-year mortgage series report provided by Freddie Mac.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1		Primary Mortgage Market Survey®																			
2		CONVENTIONAL, CONFORMING 15-YEAR FIXED-RATE MORTGAGE SERIES SINCE 1991																			
3	We make home possible™																				
4																					
5		1991		1992		1993		1994		1995		1996		1997							
6		Rate	Pts	Rate	Pts	Rate	Pts	Rate	Pts	Rate	Pts	Rate	Pts	Rate	Pts	Rate	Pts				
7																					
8	January	NA	NA	8.01	1.7	7.51	1.7	6.57	1.7	8.80	1.8	6.55	1.7	7.33	1.7						
9	February	NA	NA	8.38	1.8	7.17	1.5	6.66	1.7	8.46	1.8	6.56	1.7	7.15	1.7						
10	March	NA	NA	8.58	1.9	7.00	1.6	7.18	1.7	8.06	1.8	7.11	1.8	7.41	1.7						
11	April	NA	NA	8.47	1.7	6.94	1.6	7.80	1.7	7.88	1.8	7.44	1.7	7.68	1.7						
12	May	NA	NA	8.29	1.7	6.93	1.8	8.08	1.7	7.51	1.7	7.58	1.7	7.47	1.6						
13	June	NA	NA	8.08	1.7	6.92	1.6	7.91	1.8	7.06	1.7	7.83	1.7	7.24	1.7						
14	October	8.49	1.8	7.55	1.7	6.37	1.5	8.39	1.8	7.01	1.8	7.43	1.7	6.85	1.7						
15	November	8.33	1.7	7.80	1.8	6.69	1.5	8.67	1.8	6.89	1.8	7.14	1.7	6.76	1.7						
16	December	8.07	1.7	7.74	1.6	6.68	1.5	8.80	1.8	6.74	1.7	7.10	1.7	6.66	1.8						
17	Annual Avgs:	NA	NA	8.10	1.7	6.91	1.6	7.86	1.8	7.48	1.8	7.32	1.7	7.13	1.7						
18																					
19		1998		1999		2000		2001		2002		2003		2004							
20		Rate	Pts	Rate	Pts	Rate	Pts	Rate	Pts	Rate	Pts	Rate	Pts	Rate	Pts						
21																					
22	January	6.58	1.4	6.43	1.0	7.80	1.0	6.64	0.9	6.48	0.7	5.30	0.6	5.02	0.7						
23	February	6.64	1.2	6.44	1.0	7.93	1.0	6.64	0.9	6.38	0.7	5.22	0.6	4.94	0.7						
24	March	6.74	1.2	6.68	0.9	7.83	0.9	6.51	1.0	6.52	0.7	5.07	0.6	4.74	0.7						
25	April	6.78	1.0	6.53	0.9	7.80	1.0	6.60	1.0	6.48	0.7	5.12	0.6	5.16	0.6						
26	May	6.78	1.0	6.75	1.0	8.18	1.0	6.68	1.0	6.28	0.7	4.86	0.7	5.64	0.7						
27	June	6.67	1.0	7.18	1.0	7.99	0.9	6.70	1.0	6.11	0.6	4.63	0.6	5.66	0.6						
28	October	6.36	0.9	7.47	1.0	7.47	1.0	6.10	0.9	5.50	0.6	5.27	0.6	5.12	0.6						
29	November	6.51	0.9	7.36	1.0	7.42	0.9	6.15	0.8	5.46	0.6	5.27	0.7	5.14	0.6						
30	December	6.39	0.9	7.52	1.0	7.06	0.9	6.54	0.8	5.45	0.6	5.20	0.6	5.18	0.6						
31	Annual Avgs:	6.59	1.1	7.06	0.7	7.72	1.0	6.50	0.9	5.98	0.6	5.17	0.6	5.21	0.6						

This report represents the monthly mortgage rates in the USA at a national level over the course of seventeen years. From a Business Intelligence point of view, the report captures numerical facts about two measures: **Rate** – the mortgage rate itself, and **Pts** – the discount points, which guarantee the appropriate mortgage rate. In addition, each fact is captured in the context of a time period and all the facts in the report are related to a single product, which is identified in the header of the report – “Conventional, conforming 15-year fixed-rate mortgage”.

Why do we consider this spreadsheet to be semi-structured?

Let's take a look at the formal definition of semi-structured data. According to database theorists, semi-structured data is any data exhibiting one or more of the following characteristics:

- Structure is implicit and irregular
- No fixed schema (e.g. not relational)
- Data is nested

Structure is implicit and irregular

To the human eye, there is a clear structure to our sample spreadsheet. However, this structure is implicit and parts of it are not contained within the spreadsheet itself. For instance, we clearly see that the spreadsheet contains a header, which includes the name of the product. The header is immediately followed by vertical sub-pages – each page encapsulating up to seven years worth of data. The time period of the facts is distributed both across the columns (the year row in the header of each sub-page) and down the rows (the months in the left-most column of the spreadsheet).

We make all these distinctions based on our implicit knowledge and the visual guides embedded in the spreadsheet. Our knowledge is obviously external to the spreadsheet. As for the visual guides, yes, they are part of the spreadsheet, but these guides alone do not contribute to the formal structure of the document. What's missing is a system of rules which explicitly defines this structure.

No fixed schema

The spreadsheet has no fixed schema (from a relational point of view) because it may grow horizontally and because the columns of the report have no fixed type. If we remove the header of the report and slice it down to its separate vertical sub-pages, we will discover that these sub-pages do not have a fixed schema either.

Each sub-page would contain data recorded in the course of up to seven years. But what does this mean? Can we have less than seven years on a page? And what if we wanted to expand our worksheet to show blocks of nine years? Would that alter the essence of the data model itself? No, but it will certainly change the “relational” schema of the spreadsheet.

As for data type, we can barely find a column in this worksheet which contains a single data type. Text and numbers are mixed with blank cells and pseudo NULL values (see “NA” cells in columns B and C). Even though we can clearly see that the measures are noticeably typed, we can't say the same for the columns of the worksheet in general.

This is where the culprit is – the spreadsheet is not data-normal. The measures are not located in their own separate typed columns, nor are the dimension attributes. If we were to “structure” this spreadsheet for the purposes of loading it into a relational data warehouse schema, we would need to “normalize” it first.

Data is nested

Data nesting can be observed at many different levels in this report. To begin with, the vertical sub-pages are nested in the spreadsheet header. The data stored in all vertical sub-pages is relative to the single product located in the header. In addition, the measure names are nested within the years. Each cell which contains a year marks the beginning of a horizontal group of measures which are related to this year. Essentially the years split the vertical sub-pages to yet another level of horizontal sub-sub- pages. This type of nesting is probably the most difficult issue to resolve when attempting to structure such business content yet it is quite common in typical semi-structured business spreadsheets.

Back to hand-coded ETL

So, how are these issues addressed by today's ETL and data integration platforms and solutions? How does one extract, normalize and dimensionalize the data captured in such a spreadsheet using an ETL platform? Most of these products are designed to extract data from structured data sources (relational databases and normalized spreadsheets) and do not ship with support for semi-structured data. The data integration developer using a typical ETL platform is usually forced to write hand-coded extraction scripts whenever a complex business spreadsheet needs to be integrated in a business intelligence solution. The cost of writing these scripts usually is high and accumulates over time – a hand-coded solution is frequently error prone, fragile and difficult to maintain due to the common complexities of writing scripts in the first place. It is fragile, because in order to cut cost, the developers of a typical hand-coded solution make a lot of assumptions about the data source's layout and schema, which are often incorrect. A hand-coded solution is difficult to maintain because like any other source code, it requires documentation, adherence to coding conventions and lots of communication between team members.

There are a few ETL platforms that ship with out-of-the-box semi-structured data extraction capabilities. This functionality is usually provided through XML engines which require the spreadsheets to be converted to XML documents and then mapped through the regular data discovery and transformations provided by the appropriate engine. While this works very well for HTML-scraping and enterprise application integration (EAI), this approach is questionable when applied to semi-structured spreadsheets. Even though it may yield more robust results than a hand-coded solution, the steps needed to set up the extraction, transformation and normalization procedure of an XML-driven solution are so intricate, that its complexity is doubtfully lower.

The modern ETL systems provide the following two extremes and nothing in between:

- Simple out-of-the box solutions to extract data from structured data sources
- Complex generic platforms to extract data from pretty much anything

Here we propose an elegant solution to the problem of integrating business content spreadsheets with any business intelligence system. It revolves around the concept of recognizing the dimensionality of the data captured within the business content and sets the stage for a data extraction procedure which generates normalized structured multi-dimensional models ready to be loaded into any analytical system.

A fact-driven solution

The idea behind the proposed solution is simple – given a business spreadsheet, determine the areas that contain facts and then indicate the location of dimensions and attributes relative to these facts. Having such a data mapping schema can help us devise an extraction and normalization engine which can scan all the facts of the data source, determine their dimensional context and pump out the data in the form of a dimensional model which could be consumed by any ETL tool or analytical system. The primary challenge here is how to define the data mapping schema such that it would be flexible enough to sustain expected changes to the data source without breaking.

Let's take a closer look at our mortgage report. The following image highlights the fact areas as well as the context of the dimensions and measures surrounding them:

	1991		1992		1993		1994		1995		1996		1997	
	Rate	Pts												
January	NA	NA	8.01	1.7	7.51	1.7	6.57	1.7	8.80	1.8	6.55	1.7	7.33	1.7
February	NA	NA	8.38	1.8	7.17	1.5	6.66	1.7	8.46	1.8	6.56	1.7	7.15	1.7
March	NA	NA	8.58	1.9	7.00	1.6	7.18	1.7	8.06	1.8	7.11	1.8	7.41	1.7
April	NA	NA	8.47	1.7	6.94	1.6	7.80	1.7	7.88	1.8	7.44	1.7	7.68	1.7
May	NA	NA	8.29	1.7	6.93	1.8	8.08	1.7	7.51	1.7	7.58	1.7	7.47	1.6
June	NA	NA	8.08	1.7	6.92	1.6	7.91	1.8	7.06	1.7	7.83	1.7	7.24	1.7
October	8.49	1.8	7.55	1.7	6.37	1.5	8.39	1.8	7.01	1.8	7.43	1.7	6.85	1.7
November	8.33	1.7	7.80	1.8	6.69	1.5	8.67	1.8	6.89	1.8	7.14	1.7	6.76	1.7
December	8.07	1.7	7.74	1.6	6.68	1.5	8.80	1.8	6.74	1.7	7.10	1.7	6.66	1.8
Annual Avgs:	NA	NA	8.10	1.7	6.91	1.6	7.86	1.8	7.48	1.8	7.32	1.7	7.13	1.7

Divide and conquer

We can start building the mapping schema by dividing the spreadsheet into simpler structurally equivalent sub-pages. Those pages will contain clean non-broken fact areas surrounded by context, relevant only to those facts.

For example, we can state that the spreadsheet is partitioned vertically into pages, which begin with a row, which contains exactly four digits in its second cell and end on the first row thereafter, which contains the words "Annual Avgs":

	1991		1992		1993		1994		1995		1996		1997	
	Rate	Pts												
January	NA	NA	8.01	1.7	7.51	1.7	6.57	1.7	8.80	1.8	6.55	1.7	7.33	1.7
February	NA	NA	8.38	1.8	7.17	1.5	6.66	1.7	8.46	1.8	6.56	1.7	7.15	1.7
March	NA	NA	8.58	1.9	7.00	1.6	7.18	1.7	8.06	1.8	7.11	1.8	7.41	1.7
April	NA	NA	8.47	1.7	6.94	1.6	7.80	1.7	7.88	1.8	7.44	1.7	7.68	1.7
May	NA	NA	8.29	1.7	6.93	1.8	8.08	1.7	7.51	1.7	7.58	1.7	7.47	1.6
June	NA	NA	8.08	1.7	6.92	1.6	7.91	1.8	7.06	1.7	7.83	1.7	7.24	1.7
October	8.49	1.8	7.55	1.7	6.37	1.5	8.39	1.8	7.01	1.8	7.43	1.7	6.85	1.7
November	8.33	1.7	7.80	1.8	6.69	1.5	8.67	1.8	6.89	1.8	7.14	1.7	6.76	1.7
December	8.07	1.7	7.74	1.6	6.68	1.5	8.80	1.8	6.74	1.7	7.10	1.7	6.66	1.8
Annual Avgs:	NA	NA	8.10	1.7	6.91	1.6	7.86	1.8	7.48	1.8	7.32	1.7	7.13	1.7

If for some reason the header of the report or the whitespace between the subpages grows, this page-slicing rule will still apply and will correctly capture the pages in the new report.

Furthermore, we can split the resulting pages into horizontal sub-sub-pages, which can be defined like this:

Each sub-sub-page begins with a column, which contains four digits in its first cell and end on the last column thereafter, which has non-empty second cell:

	(1991)		(1992)		(1993)		(1994)		(1995)		(1996)		(1997)	
	Rate	Pts												
January	NA	NA	8.01	1.7	7.51	1.7	6.57	1.7	8.80	1.8	6.55	1.7	7.33	1.7
February	NA	NA	8.38	1.8	7.17	1.5	6.66	1.7	8.46	1.8	6.66	1.7	7.15	1.7
March	NA	NA	8.58	1.9	7.00	1.6	7.18	1.7	8.06	1.8	7.11	1.8	7.41	1.7
April	NA	NA	8.47	1.7	6.94	1.6	7.80	1.7	7.88	1.8	7.44	1.7	7.68	1.7
May	NA	NA	8.29	1.7	6.93	1.8	8.08	1.7	7.51	1.7	7.58	1.7	7.47	1.6
June	NA	NA	8.08	1.7	6.92	1.6	7.91	1.8	7.06	1.7	7.83	1.7	7.24	1.7
October	8.49	1.8	7.55	1.7	6.37	1.5	8.39	1.8	7.01	1.8	7.43	1.7	6.85	1.7
November	8.33	1.7	7.80	1.8	6.69	1.5	8.67	1.8	6.89	1.8	7.14	1.7	6.76	1.7
December	8.07	1.7	7.74	1.6	6.68	1.5	8.80	1.8	6.74	1.7	7.10	1.7	6.66	1.8
Annual Avgs:	NA	NA	8.10	1.7	6.91	1.6	7.86	1.8	7.48	1.8	7.32	1.7	7.13	1.7

This page-border recognition rule is flexible as well. It makes no assumptions about the number of years contained in the report. If a new report appears with 12 years per page, the rule split the sub-page into 12 sub-sub-pages – one per year.

The rule captures a variable number of measures as well. Imagine that a new report comes in with one more measure next to Mortgage Rate and Discount Points; say number of deals closed for this time period. This will add a new column for each year sub-sub-page. Since the rule expects no particular measure at the end of each page, it will correctly include all the measures contained within a sub-sub- page, even if we have a different set of measures for every page.

After all this slicing, we are down to the level of a basic page, which contains opaque facts surrounded by some context. We can easily identify the fact area as the area of cells enclosed within the fourth and second-to last rows of this atomic subpage:

1992	
Rate	Pts
8.01	1.7
8.38	1.8
8.58	1.9
8.47	1.7
8.29	1.7
8.08	1.7
7.55	1.7
7.80	1.8
7.74	1.6
8.10	1.7

We don't want to include the numbers in the last row, because these are pre-aggregate sub-totals, which could be calculated more accurately by the targeted Business Intelligence system.

Make sense of the facts

We managed to boil down the worksheet to a pattern of clean repetitive fact areas. However, these facts are meaningless outside their complete context. The atomic sub-pages contain only partial context – namely the year and the measures, which the facts have been recorded for. But what about the month and the product? As we were slicing and zooming into the sub-pages of the worksheet, we left those pieces of context in higher zoom levels.

We can simply carry them over from those higher levels using stable rules such as “bring in N-number of rows from the previous zoom level”. This way we can build a series of virtual atomic spreadsheets, which not only contain opaque and non-broken facts, but also the entire dimensional context, which applies to these facts:

CONVENTIONAL_CONF		
	1991	
	Rate	Pts
January	NA	NA
February	NA	NA
March	NA	NA
April	NA	NA
May	NA	NA
June	NA	NA
July	NA	NA
August	NA	NA
September	8.69	1.8
October	8.49	1.8
November	8.33	1.7
December	8.07	1.7

CONVENTIONAL_CONF		
	1993	
	Rate	Pts
January	7.51	1.7
February	7.17	1.5
March	7.00	1.6
April	6.94	1.6
May	6.93	1.8
June	6.92	1.6
July	6.72	1.6
August	6.63	1.5
September	6.43	1.5
October	6.37	1.5
November	6.69	1.5
December	6.68	1.5

CONVENTIONAL_CONF		
	1992	
	Rate	Pts
January	8.01	1.7
February	8.38	1.8
March	8.58	1.9
April	8.47	1.7
May	8.29	1.7
June	8.08	1.7
July	7.67	1.6
August	7.49	1.6
September	7.41	1.6
October	7.55	1.7
November	7.80	1.8
December	7.74	1.6

Putting it all together

From that point on, defining the rules that bind the facts with their respective context is straightforward:

The product of each fact cell is located in cell A1 of the respective virtual spreadsheet; the month of each fact cell is located in cell B2 of every virtual spreadsheet; the month of each fact cell is located in the first cell of the same row where the fact is located, while the measure is positioned in the third cell of the column where the fact cell resides.

To sum it up, we just defined a system of flexible rules that map out the schema of a whole class of Freddie Mac reports. The resulting schema consists of the following:

CONVENTIONAL_CONF		
	1993	
	Rate	Pts
January	7.51	1.7
February	7.17	1.5
March	7.00	1.6
April	6.94	1.6
May	6.93	1.8
June	6.92	1.6
July	6.72	1.6
August	6.63	1.5
September	6.43	1.5
October	6.37	1.5
November	6.69	1.5
December	6.68	1.5

- The definition of a formal dimensional model (product, time period and measures)
- A description of the worksheet’s physical layout, which breaks it down to atomic clean fact areas
- A set of rules, which define the location of the dimensional context relative to the facts

Now let's revisit the main characteristics of semi-structured spreadsheets, this time in the light of such a dimensional mapping schema:

- **Structure is implicit and irregular**
There is nothing implicit or irregular about the dimensional mapping schema. It clearly indicates the structure of a spreadsheet, its headers, footers, page construction rules, fact areas and context locations.
- **No fixed schema**
Laying the dimensional mapping schema on top of a semi-structured spreadsheet transforms it from a business document with implicit schema to an explicit relational schema with well defined fields, attributes and data types
- **Data is nested**
Data nesting is eliminated by means of defining patterns of low-level fact areas and binding relations between those facts and their context. The exercise of slicing the spreadsheet into its composite parts flattens it to a point where regular structured data relations can be defined.

The cost of data

A key aspect of the dimensional mapping schema we defined is the fact that it is declarative. The rules and conditions are defined and stored outside the body of a formal data extraction script.

This makes it possible to develop a variety of automated processes and build a robust software system, which could help us resolve the data extraction solution issues discussed earlier.

Such a system can introduce user interfaces designed around the notion of formulating a dimensional mapping schema in terms of declaring rules based on the visual representation of sample sets of worksheets. This will help tremendously to drive down the cost of developing semi-structured spreadsheet extraction solutions. There will be no hand-coded scripting or complex functional-language transformations involved.

The maintenance of these solutions could be streamlined as well. Data issues could be spotted easily by automatic data validation processes. The original intent of the schema designer could be conveyed by automatically generated verbose schema documentation.

And last, but not least, given such a declarative schema, an automated process could be invented to extract and dimensionalize batches of semi-structured spreadsheets and transform them into well structured relational data models. These models could then be loaded into any data warehouse, data mart, and OLAP cube or in fact into any data analytical system.

When all's said and done, the problem of unstructured data is often a problem of pure economics. A cost-effective approach like the one proposed here can get us one step closer to structuring the unstructured.

About DataDefractor

DataDefractor (www.datadefractor.com) by Interactive Edge is the leading platform for semi-structured spreadsheet normalization. It helps data warehouse and data integration developers extract, normalize and dimensionalize hundreds of complex semi-structured business spreadsheets.

DataDefractor is a visual data mapping and processing tool designed to automate the dimensionalization of complex Excel and CSV spreadsheets into structured data. It transforms business content, including financial spreadsheets, HTML tables, medical, insurance or real estate invoices into data suitable for loading into any Business Intelligence system, data visualization application, data warehouse or OLAP cube.

The concepts of flexible declarative schemas and automatic dimensional processing of semi-structured worksheets lay the foundation of DataDefractor's architecture. The product features intuitive user interfaces for mapping out rule-based schemas based on sample input spreadsheets. It also includes an extraction and dimensionalization engine, which processes semi-structured spreadsheets into well defined structured relational star schemas ready to be loaded into any Business Intelligence system.

About Interactive Edge

Interactive Edge develops and markets software products and solutions designed to put the right information in the right hands at the right time. Interactive Edge is a Microsoft Certified Partner. Year after year, Interactive Edge is the recipient of numerous awards including Consumer Goods Technology Magazine's "Best in Class" and "#1 in Customer Experience". Products by Interactive Edge are leveraged to drive bottom line growth at some of the largest and most influential enterprises in Consumer Goods, Life Science and other industries.