

Data Warehouse Newsletter #6

A Data Warehouse Generator

Overview

Version 1.0

16th October 2002

Author: Peter Nolan

pnolan@ozemail.com.au

Table of Contents

1. CHANGE CONTROL LOG	3
2. AUDIENCE	4
3. BACKGROUND TO THE DATA WAREHOUSE GENERATOR.....	5
3.1. In The Beginning	5
3.2. There Must Be a Better Way	5
3.3. Some Qualified Success	6
3.4. An Opportunity Presents Itself	6
3.5. Another Opportunity Presents Itself	7
3.6. The Generator Gets Shelved	8
3.7. The Generator Will Evolve – Just for Interest Sake.....	8
4. WHY HAVE THE DATA WAREHOUSE GENERATOR?	9
5. WHAT IS THE DATA WAREHOUSE GENERATOR?	10
5.1. What Do You Mean “Customise” the Data Warehouse Generator?	10
5.2. What is the Benefit of Using the Data Warehouse Generator?	11
5.3. Availability.....	11
5.4. How Do I get More Details?	11

1. CHANGE CONTROL LOG

#	Date	Name	Description
1.0	16/10/02	P. Nolan	Initial publication to the web.

2. AUDIENCE

The intended audience for this Newsletter is:

- IT Managers responsible for Data Warehouse Initiatives.
- Technical developers on the Data Warehouse project.

This document is intended to be read by IT Professionals who are:

- Considering building their first iteration of a Data Warehouse.
- In the learning phases of how to write the ETL for the Data Warehouse.
- In later iterations of the Data Warehouse but wondering if there is a better way to go about populating star schemas or time variant models.

Reading this document is still relevant if you are buying an ETL tool since this document will allow you to be clearer on what the ETL tools do not do.

3. BACKGROUND TO THE DATA WAREHOUSE GENERATOR

I said this web site was about being able to have my own rants and raves about Data Warehousing. Well, this is one of those places. This is a little bit of the history of how I came to build this Data Warehouse Generator.

3.1. In The Beginning

Many years ago, in what seems now like a long lost galaxy, I built my very first 'Data Warehouse'. I know it was in Q2 1991 because my daughter was born in the middle of the sales cycle for this project and I spent too much time at work and not enough time with my new baby. IT people can be like that.

To put it mildly, the first iteration of the data model was a disaster. Being a developer for 8 years previously, and steeped in IBM training (I was working for IBM at the time) I knew the right thing to do was to roll in the Bachman modeling tool (remember Bachman) and build a 3NF model because that's how data models are built, right.

After 3 months we got the system up and running. The database was DB2/MVS and the front end tool was the Data Interpretation System (DIS) from Metaphor. DIS is what got me hooked on Data Warehousing. We are yet to see the like of it in any other tool, certainly Business Objects and Microstrategy are a far cry from DIS. But that's another story.

The result of these three months work was disaster. When we started asking the database questions like 'demographic profiles of the policy holder base' every single question cost 1000+ CPU seconds on an IBM 3090-200J class machine. This was way, way too long and too expensive. Simply put, the customer could not afford the hardware that would be required to really make use of the system.

We were given three more months to make it go fast enough such that the customer would buy it. In that time I did some research on how DB2 accessed and processed data and how to make the model go faster. We reduced the 1000 CPU seconds per query down to 3-10 CPU seconds per query, the customer bought the whole package, made their money back within 3 months (yes, all their investment in the system was regained in 3 months, after that it was all profit) and we all went home happy.....well almost all of us.

3.2. There Must Be a Better Way

I had this nagging feeling there was a better way. Buried deep inside the Metaphor documentation there was 1½ pages (out of about 2,000-3,000 I think) which talked about this thing called a 'Star Schema' and that is how the database should look. I could not find another reference to 'star schema' in the documentation but a lot of the tools documentation talked about strange things called 'Facts' and 'Dimensions'. Indeed, a number of the tools said they could only be used on 'a dimensional model'.

This was all news to me and I started wondering what this dimensional model thing was all about. Since we (IBM) had made quite a bit of money on the DIS sale and I had somehow magically been given access to the executive floor and the customer was telling us that they managed to recoup the cost in 3 months we (IBM) decided to try selling DIS to some more customers. This meant that I started getting some access to people in Metaphor. As I got to know some of the people I found out that this 'Star Schema' thing was basically a trade secret and that most of the people in the company didn't know how they were designed or built. The people that implemented the applications using DIS saw a model presented by a tool called 'The Workstation Data Dictionary' which was an abstraction layer between the tools and the database. That tool told the developers if something was a fact or a dimension.

However, being the persistent person I was, and needing demonstrations to show our customers, I was eventually provided with some small sample databases and applications to show to our customers.

Funnily enough, even though I could see the database, and I could see exactly what a 'star-schema' was, I was totally baffled as to how this thing was ever designed and implemented. At that time, 1992, who ever heard of having a single central table with integer keys and having every other table hanging off it. It was very radical for us. Indeed when I was selling star schema databases I was shown the door by many a DBA telling me I was talking garbage and "that's a stupid way to build a database".

3.3. *Some Qualifed Success*

In 1993 I was involved in a head to head competition with NCR/Teradata. Since NCR was winning many of the data warehouse deals from IBM at the time I thought that project was as good a time as any to implement a banking star schema and to radically differentiate ourselves from NCR by doing so.

I was given some small amount of advice and guidance from Metaphor on how to design a prototype for a bank but Metaphor were most keen that their Data Management people wrote the code to build the database. Since IBM was doing the head-to-head for no charge no-one was willing to put in the money to build the database so it fell back to yours truly.

In hindsight, we built a good model, it worked well, but we were also outsold and I also made some mistakes in how I wrote the code to populate the model. I didn't know these were mistakes until about 6 months later when I started the whole process again.

3.4. *An Opportunity Presents Itself*

Around this time, the first customer who bought DIS was given a brief to experiment with and investigate 'Worlds Best Practice Direct Marketing in a Financial Services Company'. With this customer with me I had the very good fortune to be able to go to the 1993 Metaphor Users Conference in San Francisco. By good fortune, Bill Inmon was the key note speaker on the final day.

One of my main goals was to find someone from the 'Data Management Group' and find out how to design and build one of these star schema things properly. It was obviously a key component of the whole picture and it had not worked as well as I thought it must do in the head-to-head situation.

My customer was also very interested in this as he felt that these 'star-schema' databases were very likely a key to being able to implement the very complex analytics that he wanted that the current database just could not support.

We went to a presentation by one of the 'Data Management' people. A guy by the name of Steve Mongulla. Steve became something of a mentor for me on star schema designs and we remain good friends to this day. We walked through many of the concepts of how to build stars, why they were so powerful, how the optimizer resolved the queries etc. But Steve remained adamant that the Data Management Group could not provide details (like code fragments) on how to build these things.

Having taken all this in, my customer and I went back to Australia and we looked at the work ahead for experimenting with 'Worlds Best Practice Direct Marketing'. We had made many contacts while at the conference and as we talked with them it became clearer and clearer that a highly sophisticated analytical database would be one of the key components of what we would do.

It was decided, at great expense, to bring Steve out to Sydney and have him work with us to design the database. This he did, and he was kind enough to answer questions via email over the following periods. But we were still left with the question of how to build it, and there simply was not enough money to have it built by the Data Management Group.

To cut this part of the story short, it took us some 18 months elapsed time to write the code that was required to implement the features we wanted in the database we developed. In simple terms, we just could not understand how to do some things that were designed into the model. For example, multiple levels of summary data in the one fact table, incremental updates of all summary records with the days transactions, allocation of keys all fast enough such that the attribution process didn't take too long for the larger projects we had planned.

However, at the end of the 18 months we had a suite of code that would do what we wanted, it was elegant, fast, efficient and we were very, very pleased with what we had done. It was early 1995.

3.5. Another Opportunity Presents Itself

During this period IBM was having it's share of problems. In Australia it was decided to stop selling 'Information Warehouse' as we were not winning enough business from it to justify the effort. I was asked to sell office products like Lotus Notes. Notes was great to sell, it sold itself, but I wanted to do Data Warehousing. I had developed something of a passion for it.

Since I was leaving IBM anyway, the DIS customer I was working for offered me a contract to do some of the work we had set up on the Direct Marketing project. This is how I came to be working on this code for a period of some 18 months.

My departure from IBM was so amicable that I was invited to put in a proposal for the development of a Data Warehouse for a retail banking system being developed in Sydney. I went to talk to the project manager, told him about my background, what I had done, what I believed was the best way to move forward and how I proposed to move the project forward. I was given the position of lead architect on the project.

The first piece of work was to analyse the defined requirements, take a look at the banking system data model, and develop the initial data model for the first portions of the Data Warehouse. Again, I proposed a star schema.

Once I had developed the model I noticed that the structure was pretty much the same as had been developed on the DIS project. Fact table, dimension tables, summaries etc. Except for the names of the columns and tables all was very similar. Even more co-incidental, the DIS customer I had been dealing with was outsourced to IBM Global Services during this period.

I told the IBM project manager that if we had access to the code that we had written on the first project we could save ourselves at least 3 months programming work because I would not have to write detailed specifications and then show people how to write the code. We would have 4 programmers working on the project. At a meeting it was decided to call IBM Global Services and see if they were willing to 'sell' their code to IBM. (IBM did not have any access to IBM Global Services intellectual property in Australia.) IBM decided they were willing to pay \$A30K to 'buy' the code. Being an independent consultant at the time what I heard at that meeting was if I had a suite of template code then IBM would be willing to pay \$A30,000 to gain access to it. Now that was interesting. So that very day I decided to find out if it was possible to write a suite of template code for a star schema data warehouse that could be easily customized. I went out and bought a cobol compiler, DB2/2, an extra drive etc. What I found out was, yes, it was entirely possible to write a suite of template cobol code that could be easily customized.

I was too late for the IBM project, but now I had in my pocket something that would greatly reduce the implementation time of any Star-Schema project. I later extended the template code to cover time variant models as well.

A little later the templates were changed so that they could be updated by a generator rather than changing the templates by hand.

With this ability to generate 95% of all the code that was required for a star schema data warehouse we implemented a number of star schema data warehouses in very quick time. We used this star schema generator a number of times when I was working at Hitachi. It proved a very effective tool to perform a large amount of coding very quickly. It also allowed us to bid on large fixed price systems integration deals knowing that we could write code faster than anyone else. In most bids we included Prism Warehouse Manager but where it was not selected we used our code generator.

As best I recall we used it on three major projects, and on two other projects where we were involved the customer selected Prism Warehouse Manager.

3.6. *The Generator Gets Shelved*

In 1998 I moved from Hitachi to PriceWaterhouse Coopers, and from PwC I moved to Ardent Software. PwC preferred not to be in 'software development' and Ardent Software had DataStage to sell. So in 1998 the Generator was put on the shelf and never used again. Today, of course, the ETL tools have come a long way forward and they have come down in price. Prism Warehouse Manager was expensive, but buying the base version of DataStage is not that expensive at all.

My view is that today there is no real sales opportunity for the Generator that I built. I believe there is no great lost potential income by publishing it. However, I do believe there is value in publishing it for the ideas that are imbedded in the templates. I know that many of the ideas have not been implemented on many of the Data Warehouses I have seen and these templates may just prove useful to some people out there.

3.7. *The Generator Will Evolve – Just for Interest Sake*

The other thing that I plan to do with this Generator is to convert it to C. The main reason being that I am interested in learning how to do a lot of things in C. I used to be a C programmer in another lifetime, but I have never written database processing programs in C. I figure that is something worth learning, just for fun.

4. **WHY HAVE THE DATA WAREHOUSE GENERATOR?**

Though we have moved a long way forward with the development of Star Schema data warehouses I so often come across data warehouses that are implemented in such a way as to have a great deal of processing time being consumed that is just not required. This is not a problem when the Data Warehouse is small, but in the large ones the extra processing defies the hardware budget.

Also, there is little published information about exactly how to build the code for populating star schema data warehouses. When I worked at Ardent I was surprised to find that there were no 'templates' for building star schema databases.

I gave one of the 'best and brightest' people in the company my cobol templates and asked him to convert them to DataStage jobs. We found that it was not practical to do. I recently worked on an Informatica project, and in my spare time I did some research to see if such a set of templates could also be generated in Informatica. Funnily enough, I believe generating these templates is also not possible in Informatica, or at least, it is so difficult that it would require such a significant development effort and the maintenance of the maps would be so high that any benefit in generating the maps would be lost.

So, some of the reasons I think people will like to read these templates are as follows. It will show people:

- How to build Star Schema DWs in the worlds most common computing language.
- Features of large star implementation that are difficult to implement in todays tools.
- How to manage multiple levels of aggregation.
- How to manage adjustments to transactions.

And who knows, someone might just use this code again one day. Certainly I think that if it is converted to C (no-one under 30 seems to know cobol any more) a few more people might look into how such code works. Also, it will be callable from inside DataStage and Informatica and some things that might be useful to do could be included into these tools.

5. WHAT IS THE DATA WAREHOUSE GENERATOR?

The Data Warehouse Generator consists of the following components:

- A Set of Cobol ANSI SQL template programs that have a series of parameterised variables within the code.
- A code generator written in ANSI Cobol that reads in parameter files and the template code and produces the required code to run a large star schema data warehouse.
- A code generator that reads the star schema data model out of an simple repository and generates the required DDL for the target database management system. Current target databases are Oracle and DB2.
- A set of twenty five Oracle/DB2 Tables that define a simple Star Schema such that you can learn from the experts..
- The actual Oracle/DB2 Database in a backed up format.

Thus with the Data Warehouse Generator we start with a complete prototype Data Warehouse which can be installed and run on a COBOL Oracle/DB2 machine so that you can learn and then extend the prototype to your business. If you choose to run the real warehouse on a different database manager there will be little effort in converting across as long as the other database manager uses standard ANSI SQL.

5.1. What Do You Mean “Customise” the Data Warehouse Generator?

It is true that everyone's Data Warehouse is different. Yet, there are some strong similarities. The similarities are what allowed me to build the Data Warehouse Generator.

For example, consider a Star Schema database. Though each company has different field names and different dimension names all Star Schemas have detailed fact tables, summary fact tables, snapshot fact tables and dimension tables. The manner in which the transactions are attributed and placed into the Data Warehouse is very nearly the same. Thus, if one has a template of how this all works one can simply make the minor adjustments necessary to customise the code.

Let's take a real example. The Data Warehouse Generator supports as many dimensions as the Database Manager can manage. Let's say you only want 8 dimensions. You would do this by setting a parameter file to say that there are 8 dimensions for this particular star. You then generate the code to support this. Simple as that!!

Let's take another example. Let's say you want detailed data to have many levels of summary, 30-50 is not uncommon in very large sites) and because of the depth of the index trees you want to store some of the summaries in one table and you want a few other summary fact tables with their summaries. Thus you want 30-50 summaries spread across 5 tables. What you would do in this case is define each of the summary tables in the repository, define each of the fact tables in the parameter files and then generate the code for all five fact tables. You would then also have to update the summary control table to define the levels of aggregations that exist in each of the five fact tables.

5.2. *What is the Benefit of Using the Data Warehouse Generator?*

Today I expect that most companies will buy an ETL tool of some sort. So the advantage of downloading and looking at this code is simply to understand what is necessary to be done in the ETL tool of choice.

5.3. *Availability*

Just download it. What I have published is the generated code. I have not published the input templates or the generator because I don't think they are useful. You would only need the generator if you intended to actually use the Generator on a project. If you want to do that I want to know about it. So if you want to actually build a data warehousing using the Generator feel free to contact me on pnolan@ozemail.com.au.

5.4. *How Do I get More Details?*

The Data Warehouse Newsletter #7 is named "The Data Warehouse Generator - Details". It contains a significant level of detail for the Data Warehouse Generator. You are welcome to download it for more information.