2016

Experimentation with Raw Data Vault Data Warehouse Loading

Connard Williams
Georgia Southern University, cw06893@georgiasouthern.edu

EXPERIMENTATION WITH RAW DATA VAULT DATA WAREHOUSE LOADING

Connard N. Williams
Georgia Southern University
cw06893@georgiasouthern.edu

ABSTRACT

The principal novelty in this work is raw Data Vault (DV) loads from source systems, and experiments with effects of allowing certain kinds of permissible errors to be kept in the Data Vault until correct values are supplied.

Keywords

Data warehouse testing, Data Vault, Data Mart, Raw Data Vault loads

INTRODUCTION

A number of factors such as increasing business mergers, data center migrations are fueling the need for more effective data warehouse testing, including coverage of loading strategies. Owing to the complexity of data warehouse projects (*Hughes, 2015*), great emphasis must be placed on an agile-based approach with properly developed and executed test plans throughout the various stages of designing, developing, and implementing the data warehouse to mitigate against budget overruns, missed deadlines, low customer satisfaction, and outright project failures. Testing is especially critical to success in data warehouse projects (*Golfarelli and Rizzi, 2009b*) as users need to trust in the quality of the information they access. Most of the Business Intelligence/Data Warehousing (BI/DW) systems are like black boxes to customers who primarily see the output reports/charts/KPIs, often overlooking the complex logic applied behind the scenes *(Kamal and Nakul, 2010*). Studies on testing data warehouses *(Mathen, 2010*), have provided insights into best practices, but research specific to testing DV-based DW is lacking.

Data Vault modeling was developed by Dan Linstedt and serves to structure the DW data as systems of permanent records, and to absorb structural changes without requiring any data alterations (*Jovanovic and Bojicic, 2012; Collins, Hogan, Shibley, Williams and Jovanovich, 2014*). The Data Vault (*Linstedt and Graziano, 2011; Graziano, 2011; Linstedt and Olschimke, 2015*) is comprised of Hubs, Links, and Satellites. In the context of this work, the EDW core historical data repository, consists of the Data Vault modeled tables and holds data over time at a granular level (raw data sets). The two-layer DW architecture (*Golfarelli and Rizzi, 2009a*) provides the backdrop against which this research is conducted. In our experimental implementation of the architecture, the source is loaded to the Data Vault, which feeds materialized Data Mart (DM) downstream. Transformations are kept to a minimum in the Extract-Transform-Load (ETL) process which loads the DV from the source. Major transformation activities are pushed further downstream to the next ETL process which loads and refreshes the Data Mart from the Data Vault.

PROBLEM STATEMENT

The traditional approach for dealing with errors during the data warehouse load normally results in one of the following actions:

(a) Fail the component/activity
(b) Ignore the error
(c) Redirect the error rows/records

Failing the component/activity aborts the batch load process and has the potential to cause delays due to tear down/rollback and restarts when errors are detected. Simply ignoring error without follow up can be problematic - there is no feedback to the source provider about the errors so that corrective actions can be taken. Moreover, depending on the nature of the error, it may severely impact the quality of data fed downstream to data marts for reporting and analyses. Redirecting the error rows/records to an error report file is prudent; however, this action by itself prevents the DV from carrying out its role as the "capture-all store-all" enterprise data warehouse. The data in the DV must be truly reflective of the source data (*Linstedt and Graziano,*

2011). Aggressive cleaning of the source will go against a basic requirement of the DV: a comprehensive DW staging area able to store not only current data, but also past data without any alterations or loss *(Jovanovic, Bojicic, Knowles and Pavlic, 2012*). Operational data errors must be kept on file in the DV and correction applied as new records instead of overwriting the errors - a crucial requirement for industries such as banking and healthcare. Finally, the traditional loading approach also limits the extent to which an agile methodology can truly be applied to the DW project, since Data Marts load and refresh testing are delayed until the staging and source loading issues have been resolved. A main goal of this research was to demonstrate a flexible approach, with repeatable load patterns for raw DV loads (both DV 1.0 and DV 2.0) from source systems, to address the issues stated.

SOLUTION DESIGN

All tests and experiments were carried out on the same computer to ensure there was no bias for any given approach or test run. The hardware and software specifications were as follows:-

- *OS: Windows7 64 bit with 8.0 GB of internal memory*
- *Processor: Intel Core i7 4702MQ CPU @ 2.20 GHz*
- *DBMS: SQL Server 2012 for DW, MySQL Community Server 5.6.24 for generating test data*
- *ETL Tools: Microsoft SQL Server Integration Services with SQL Server Data Tools for VS 2012*
- *Test Data Generator: generate data tool downloaded from (www.generatedata.com).*

Data Models for Source, Data Vault, and Data Mart

The research conducted was based on a car rental company, which has multiple agents (locations) that rent vehicles to customers. The models adopted have been simplified to remove much of the unnecessary details in an effort to foster greater focus on the concepts. For example, a number of the attributes were omitted, and only the rental operation is considered. For the Data Vault, both DV 1.0 and DV2.0 were implemented. The only difference in the data models between DV 1.0 and DV 2.0 is that the 8 byte integer surrogate sequence keys in DV 1.0 (_Seq) are replaced by 32 byte character MD5 hash keys (_Hsk) in DV2. The source model, DV 2.0 model (in the de facto color scheme), and the dimensional star schema for Data Mart are shown below in figures 1, 2, and 3 respectively. Owing to similarity to DV 2.0 model and limited space, the DV 1.0 model is not shown.

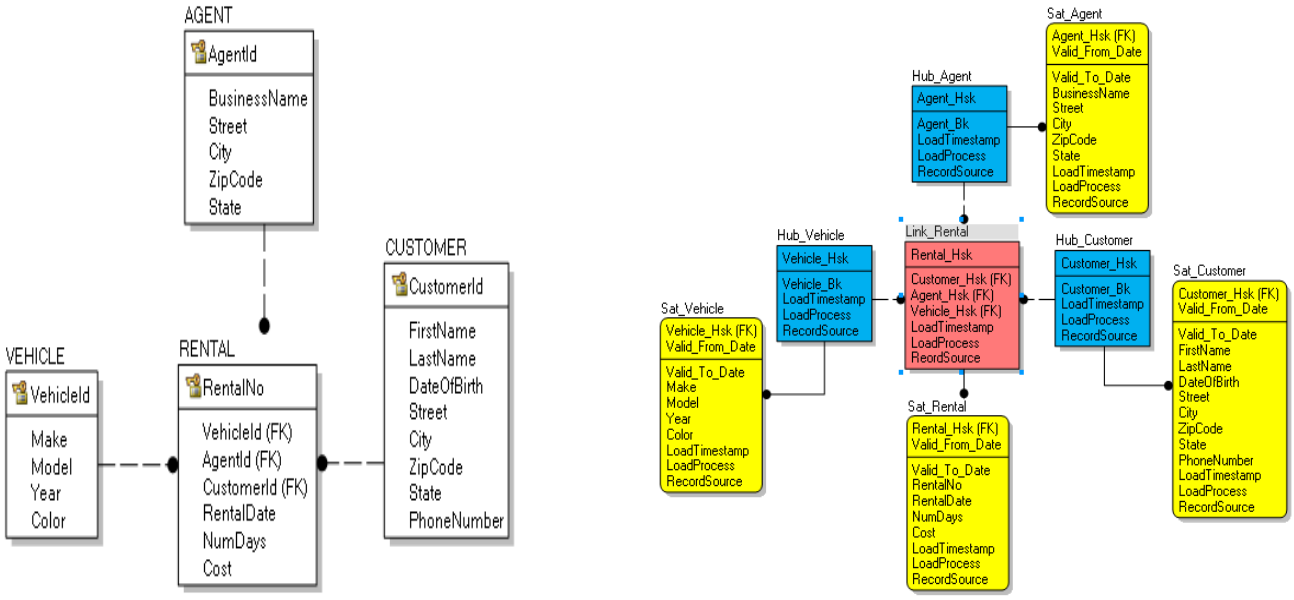

Figure 2. Data Vault 2.0 model

Figure 1. Relational model of operational source

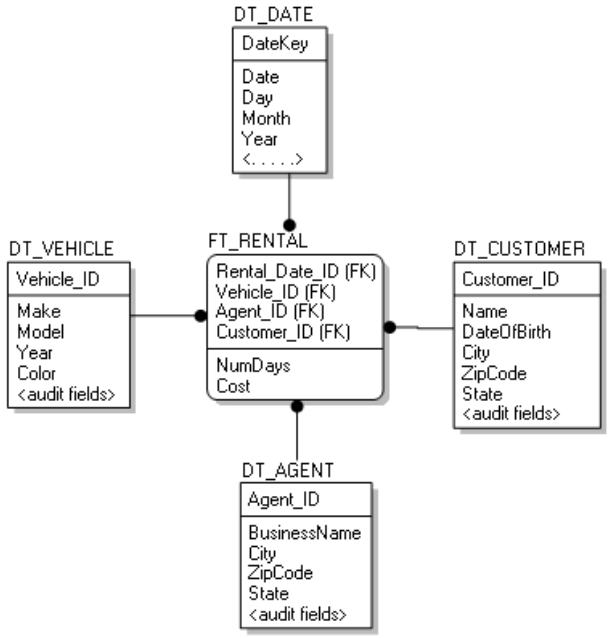

Figure 3. Star Schema for Data Mart

Data Scoping, Validation Rules, and Permissible Load Errors

Source data quality is dependent to a large extent on the governance of the schema and integrity constraints. For example, flat files, which have no schema, are more prone to erroneous data than their relational database counterparts (*Rahm and Do, 2000*). The concept of permissible load errors applies only to data errors caused from operational errors and not submission or system errors. Permissible load errors are essential for historical tracking and extended audit support. In contrast, a record with fatal errors such as missing business key cannot be permitted in the data warehouse because the business key is required for data integration. Since each DW implementation will have its own unique requirements, the set of permissible load errors will have to be determined on a case-by-case basis during the data scoping phase. The following business rules for permissible load errors were applied for this project:

- Customer – DateOfBirth field checked: Is customer at least 18 yrs. old at time or rental?
- Vehicle – vehicle's Year field checked: Is vehicle Year less than 3 years or more than 2 years
- Rental – RentalDays field checked: Is rental for a period of 0 to 30 days?

Data errors not listed above are treated as non-permissible errors and are rejected by both approaches.

EXPERIMENTS

The data generator was used to generate the following test records:

- Agent: 500 , Customer: 5,000, Vehicle: 10,000
- Rental: 1,000, 10,000, 50,000, 75,000 and 100,000 (for the five iterations of load strategy and DV model combinations), with 1% of records containing permissible load errors.

Only one test was run at any given point in time, and the average time for three runs was recorded.

Permissible load errors are kept in the DV for historical purposes, but cannot be transmitted to the DM as doing so would skew reporting/analyses results. Any dependent records, which may or may not themselves contain permissible errors, must also not be loaded or refreshed in the DM by the ETL job, as this could lead to sparse cube condition for any Relational Online Analytical Processing (ROLAP) implementation. In addition, each record in a fact table needs to have matching dimension records to

maintain referential integrity in the DM. To enforce this requirement, foreign key constraints were implemented on the Rental fact table.

After corrections are made, target records and all dependencies are sent to the DM in the next ETL load job. As an illustration of this principle, note below in figures 4 and 5 that customer Casey Mack's DateOfBirth was incorrectly captured as 1999-09-29 (instead of 1969-09-29), which would suggest that she was only 14 -15 yrs. old at the time or her vehicle rental activities. Even though there are no errors in these transaction records, these records must not be loaded in the Rental fact table in the DM until after the correct DateOfBirth is supplied to the DV and the Customer dimension table loaded or refreshed.

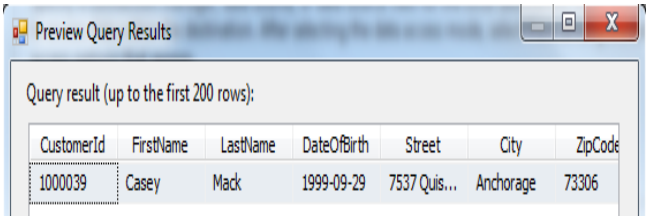

Figure 4. Query Preview Error File in SSIS ETL package

	RentalNo	VehicleId	AgentId	CustomerId	RentalDate	NumDays	Cost
1	RN1000050	500197	345	1000039	2014-08-21	18	298.70
2	RN1000060	500063	314	1000039	2014-01-25	1	28.40
3	RN1000451	500180	316	1000039	2014-03-15	8	139.70
4	RN1000460	500047	325	1000039	2014-01-11	8	139.70

Figure 5. Query of transaction source

Traditional approach –No Load Errors permitted in Data Vault

Records detected with errors during DW load were reported in an Excel spreadsheet for error correction and resubmission. Since the load job failed causing the process to rollback, the entire dataset with the corrections included was resubmitted for loading into the DV. The corrected dataset was subsequently used to feed the data mart downstream in the DM initial load. The load times with increasing number of Rental transactions are shown below in tables 1 and 2:

	DV Load		DM Load	**Total**
Rental Rows	Initial	Resubmit	Initial	
1,000	16.273	13.213	5.210	**34.696**
10,000	28.595	27.223	5.803	**61.621**
50,000	207.672	207.575	8.572	**423.819**
75,000	424.667	420.812	11.372	**856.851**
100,000	743.781	743.392	14.932	**1502.105**

Table 1. Load times for DV 1.0-based DW using "No Load Errors Allowed" strategy

	DV Load		DM Load	**Total**
Rental Rows	Initial	Resubmit	Initial	
1,000	7.934	5.913	6.942	**20.789**
10,000	8.659	6.880	7.269	**22.808**
50,000	103.803	103.522	10.405	**217.73**
75,000	232.831	231.974	11.138	**475.943**
100,000	424.986	421.187	14.258	**860.431**

Table 2. Load times for DV 2.0-based DW using "No Load Errors Allowed" strategy

Alternative– Keeping permissible load errors in Data Vault until corrections are provided

Records detected with errors during DW load were reported in an Excel spreadsheet for error correction and resubmission, as in previous approach. However, errors predetermined as permissible/allowable errors during DV load were accepted and held for future correction – a stark contrast to the previous approach. Additionally, the resubmitted dataset included only the corrected records. The load times (in seconds) with increasing number of Rental rows are shown below in tables 3 and 4:

	DV Load		DM Load		**Total**
Rental Rows	Initial	Retry	Initial	Refresh	
1,000	15.849	1.389	3.822	1.700	**22.760**
10,000	26.941	1.607	5.382	2.487	**36.417**
50,000	200.555	4.509	8.003	6.133	**219.200**
75,000	435.727	8.860	10.389	8.502	**463.478**
100,000	733.686	12.792	15.168	10.874	**772.52**

Table 3. DV 1.0- "Permissible Load Errors" strategy

	DV Load		DM Load		**Total**
Rental Rows	Initial	Retry	Initial	Refresh	
1,000	6.017	1.154	4.804	1.685	**12.76**
10,000	7.269	1.981	4.961	2.528	**16.739**
50,000	103.787	3.962	9.251	6.240	**123.24**
75,000	236.151	5.725	10.890	8.455	**261.221**
100,000	420.625	9.906	15.959	10.904	**457.394**

Table 4. DV 2.0- "Permissible Load Errors" strategy

ANALYSIS OF RESULTS

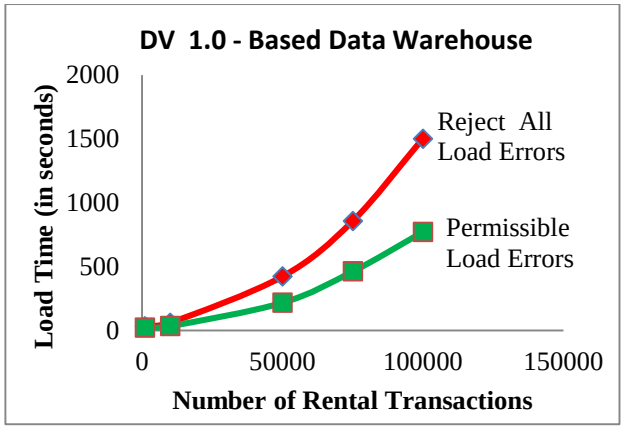

Figure 6. DV 1.0 – based DW Load times for "Reject All Load Errors" vs "Permissible Load Errors"

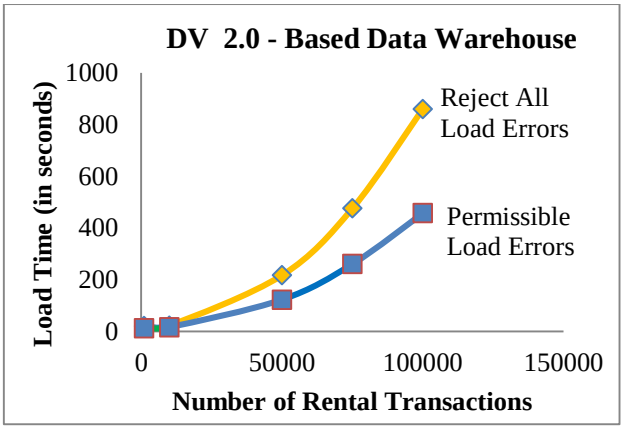

Figure 7. DV 2.0 – based DW Load times for "Reject All Load Errors" vs "Permissible Load Errors"

Examination of tables 1, 2, 3, and 4 reveals that for both strategies, the load times for the data vault are much smaller for DV 2.0 than DV 1.0 for a given rental data set. This can be attributed to fact that DV 2.0 uses hash keys, generated from the business keys, thereby obviating the need for expensive lookups when loading satellites and links, and enabling more parallelism in DV loads. As seen in tables 1 and 2, the "No Load Errors Permitted" strategy generally requires more time for the initial DV load attempt than the resubmit attempt. This is due to the additional expense of writing out error report to Excel file. Even though general loading of DV2.0 is faster than loading DV 1.0, the "Permissible Load Errors" strategy is noticeable more expensive for loading of the Data Mart from DV 2.0 than from DV 1.0. This can be explained by the joins required by the ETL logic, which uses a 32-byte hash key in DV 2.0 which is slower than using the 8-byte integer surrogate sequence used in DV 1.0. Finally, the total load times in the tables, complimented by the graphs in figures 6 and 7, reveal that the "Permissible Load Errors" strategy consistently provides significantly better load times than the traditional strategy for both DV 1.0 and DV 2.0. Moreover, larger datasets provide even greater gains in load times.

CONCLUSION

In this paper we have shown that by adopting a Data Vault-based Enterprise Data Warehouse we can simplify and enhance various aspects of testing, and curtail delays that are common in DW projects. Additionally, using raw DV loads, keeping transformations to a minimum in the ETL process which loads the DV from the source, and allowing "Permissible Load Errors" in the DV, we can avoid the constant building up and tearing down associated with traditional staging. This strategy also fosters a more agile approach of getting DV up quickly and incrementally building out the individual Data Marts. The load patterns used are flexible, restart able, and reusable. The load patterns should be systematically extended for DWs containing other types of data; however, further research needs to be conducted to confirm this conjecture.

REFERENCES

1. Collins, G., Hogan, M., Shibley, M., Williams, C. and Jovanovich, V. (2014). Data Vault and HQDM Principles, *Proceedings of the Southern Association for Information Systems*, Paper 3.
2. Golfarelli, M., & Rizzi, S. (2009a). Data Warehouse design: Modern principles and methodologies. McGraw-Hill, Inc.
3. Golfarelli, M., & Rizzi, S. (2009b). A comprehensive approach to data warehouse testing. In *Proceedings of the ACM twelfth international workshop on Data warehousing and OLAP* (pp. 17-24). ACM.
4. Graziano, K. (2011). Introduction to Data Vault Modeling. *True Bridge Resources, White paper*.
5. Hughes, R. (2015). *Agile Data Warehousing for the Enterprise: A Guide for Solution Architects and Project Leaders.* Morgan Kaufmann.
6. Jovanovic, V., & Bojicic, I. (2012). Conceptual data vault model. In *SAIS Conference, Atlanta, Georgia: March* (Vol. 23, pp. 1-6).
7. Jovanovic, V., Bojicic, I., Knowles, C., Pavlic, M., (2012). Persistent staging area models for Data Warehouses. *Issues in Information Systems*, *13*(1), 121-132.
8. Kamal, R., & Nakul M. (2010). Adventures with Testing BI/DW Application: On a crusade to find the Holy Grail, http://msdn.microsoft.com/en-us/library/gg248101.aspx. Retrieved January 20, 2015.
9. Linstedt, D., & Graziano, K. (2011). Super Charge Your Data Warehouse: Invaluable Data Modeling Rules to Implement Your Data Vault. CreateSpace.
10. Linstedt, D., & Olschimke, M. (2015). Building a Scalable Data Warehouse with Data Vault 2.0: Implementation Guide for Microsoft SQL Server 2014. Morgan Kaufmann.
11. Mathen, M. P. (2010). Data Warehouse Testing. Infosys White paper published in the DeveloperIQ Magazine,
12. Rahm, E., & Do, H. H. (2000). Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, *23*(4), 3-13.