Dealing with Data Warehouse Transaction Processing Truncated Data: Case Study of SQL Server 2000

Carin Chuang, Purdue University North Central, USA; E-mail: cchuang@pnc.edu
Kuan-Chou Chen, Purdue University Calumet, USA; E-mail: kchen@calumet.purdue.edu

ABSTRACT

In this paper, the cause of truncated data in the transaction processing will be discussed. Specifically, the main goal of this paper is to demonstrate the function of recovery model in the SQL Server 2000. In the paragraphs to come, we will discuss in a little more detail, the truncated data that can take place, along with the importance of data integrity. Afterwards, we will discuss two scenarios cases to implement the recovery model. In the case, we will begin by discussing the transaction processing in between data warehouse. Afterwards, we will discuss the potential causes of truncated data. The last part of this case details configuration of each recovery model; more specifically what types of functions are available in this particular setting, what requirements those functions place on to deal with specific problems and few of the benefits that come with having experience. This paper has the potential to provide the recovery model needs to database administrator to do advanced support task.

Keywords: CIS majors, math course, course design.

INTRODUCTION

Transaction management is one of the most crucial requirements for enterprise application development. Most of the large enterprise applications in the domains of finance, banking and electronic commerce rely on transaction processing for delivering their business functionality. Given the complexity of today's business requirements, transaction processing occupies one of the most complex segments of enterprise level distributed applications to build, deploy and maintain.

Enterprise data warehouse often require concurrent access to distributed data shared amongst multiple components, to perform operations on data. Such applications should maintain integrity of data (as defined by the business rules of the application) under the following circumstances:

1. Distributed access to a single resource of data, and
2. Access to distributed resources from a single application component.

In such cases, it may be required that a group of operations on (distributed) resources be treated as one unit of work. In a unit of work, all the participating operations should either succeed or fail and recover together. This problem is more complicated when a unit of work is implemented across a group of distributed components operating on data from multiple resources, and/or the participating operations are executed sequentially or in parallel threads requiring coordination and/or synchronization. Truncated data would be easy to be occurring in the process in either case. It is required that success or failure of a unit of work be maintained by the application. In case of a failure, all the resources should bring back the state of the data to the previous state (i.e., the state prior to the commencement of the unit of work).

In this paper, the cause of truncated data in the transaction processing will be discussed. Specifically, the main goal of this paper is to demonstrate the function of recovery model in the SQL Server 2000. In the paragraphs to come, we will discuss in a little more detail, the truncated data that can take place, along with the importance of data integrity. Afterwards, we will discuss two scenarios cases to implement the recovery model. In the case, we will begin by discussing the transaction processing in between data warehouse. Afterwards, we will discuss the potential causes of truncated data. The last part of this case details configuration of each recovery model; more specifically what types of functions are available in this particular setting, what requirements those functions place on to deal with specific problems and few of the benefits that come with having experience. This paper has the potential to provide the recovery model needs to database administrator to do advanced support task.

DATA WAREHOUSE AND TRANSACTION MANAGEMENT

Every operation involves data occurs within a transaction. The way in which a database handles transactions is a critical to the database industry as the aerodynamic curve of a wing is to the aircraft industry (Nielsen, 2003). Transaction processing is a reliable and efficient of large volumes of repetitive work. Database management systems ensure that simultaneous users do not interfere with each other and that failures do not cause lost work. There are three problems that can result because of simultaneous access to a database: [1] lost update, [2] uncommitted dependency, and [3] inconsistent retrieval (Mannino, 2004).

Data warehouse is a central repository for summarized and integrated data from operational databases and external data sources. In a data warehouse, data must be organized for rapid access to information for analysis and reporting, as this is the purpose of the data warehouse. Data integrity and performance of transaction processing requires that operational database be highly normalized. In contrast, data warehouses are usually demoralized from Normal Form to reduce the effort to join large tables. Most data warehouse processing involves retrievals and periodic insertions of new data. Because of difference processing requirements, different data models have been developed for operational data warehouse database. In general, the different processing may cause the truncated data.

DATA WAREHOUSE IMPLEMENTATION

Dimensional modeling is used in the design of data warehouse database to organize data efficiency of queries that analyze and summarize large volumes of data. The data warehouse schema is usually much simpler than the schema of an OLTP (On-Line Transaction Processing) system designed using entity-relation modeling. The verification tables used in OLTP systems that are used to validate data entry transactions are not necessary in the data warehouse database because the data warehouse data has been verified before it was posted to the data warehouse database and the data is not expected to change frequently once it is in the data warehouse (Agosta, 2000). Backup and restore strategies also differ in a data warehouse from those necessary fro an OLTP system. Much of the data in a data warehouse is unchanging and does not need repetitive backup. Backup of new data can be accomplished at the time of update, and in some situations it is feasible to do these backups from the data preparation database to minimize performance impact on the data warehouse database.

Data to be used in the data warehouse must be extracted from the data sources, cleaned and formatted for consistency, and transformed into the data warehouse schema. The data preparation area is a relational database into which data is extracted from the data sources, transformed into common formats, checked for

consistency and referential integrity, and made ready for loading into the data warehouse database. Performing the preparation operations in source database is not an option because of the diversity of data sources and the processing load that data preparation can impose on online transaction processing systems. Furthermore, attempting to transform data in the data source systems can interfere with online transaction processing (OLTP) performance, and the reconciliation of inconsistencies in data extracted from various sources cannot be accomplished until the data is collected in a common database, at which time data integrity errors can more easily be identified and recited. The data preparation area is a relational database that serves as a general work area for the data preparation operations. It contains tables that relate source sat keys to surrogate keys used in the data warehouse, tables of transformation data, and many temporary tables. It also contains the processes and procedures that extract data from source data systems.

The data preparation area should also contain the processes that are used to extract the data from the data sources, the processes that transform and cleanse the data, and the processes that load the data to the data warehouse. The processes may be in the form of SQL quires, stored procedures, DTS packages, or documents of manual instructions. As in the development of any database system, the objective is to automate as much of the process as possible and to mange and maintain the automated tools developed. Storing and maintaining the transformation processes in the data preparation area permits the use of standard database backup and restore mechanisms to preserve them. The database schema must support complex data representations. Database must contain data that are aggregated and summarized.

CASE STUDY OF SQL SERVER 2000

Scenario

As an administrator of a SQL Server 2000 computer, the job responsibility is to create a database to use an intermediate data store for a data warehouse. Each night the administrator must import daily sales data into the database from SQL Server 2000 computers in 120 locations. After the data is moved into the data warehouse, the tables are truncated. The database schema is shown in the Figure 1. For the purpose of minimizing the time to import the sales data and administering the database, there are several procedures for an administrator to configure the data import processes.

Implementation

We want to optimize or speed and at the same time keep administration down. We accomplish this by

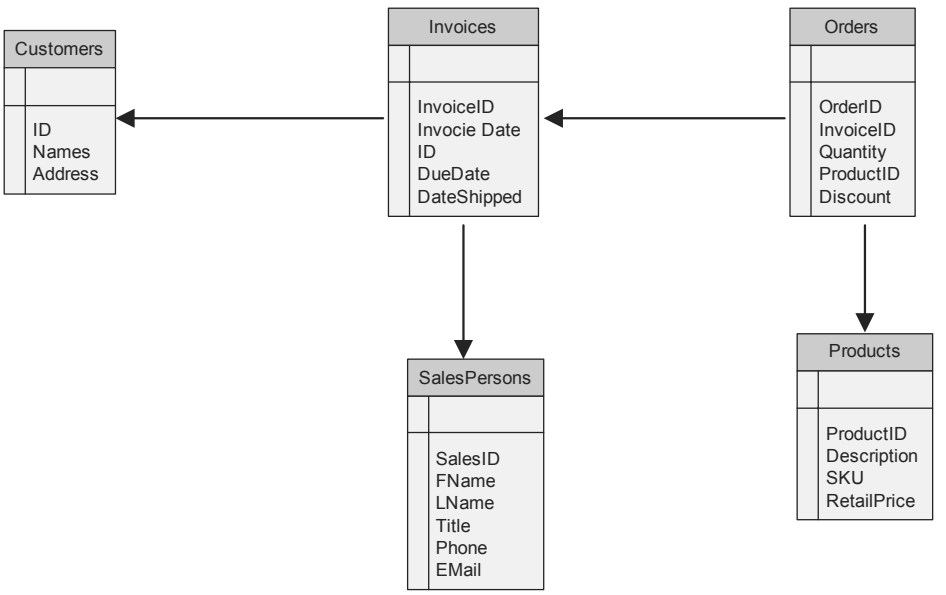
1. Use the simple recovery model and the FOR LOAD option to create the database. Use the simple recovery model. It will avoid logging of the bulk import process. This will increase speed.
2. Create a Data Transformation Services package that uses the BULK INSERT statement to copy the sales Data. Create a DTS package with BULK INSERT to copy sales data. BUL INTERT is fast way to import data and a DTS package can be reused and requires little administration.

A SQL Server 2000 provides three recovery models that can be used to recover database data in case of hardware failure or other eventualities that may compromise data integrity. These recovery models are: the Simple Recovery model, the Full Recovery model and Bulk-Logged Recovery model. With the Simple Recovery model, the database can be recovered to the point of the last backup but not to the point of failure or to a specific point in time. To do that, choose either the Full Recovery or Bulk-Logged Recovery model. Furthermore, the FOR LOAD clauses of the CREAT DATABASE statement are used for compatibility with earlier versions of SQL Server. The database is created with the dbo use only database option turned on, and the status is set to loading. The simple recovery model fits this scenario well since the tables are truncated every night.

REFERENCES

1. Agosta, L. (2000). The essential guide to data warehousing. New York: Prentice Hall.
2. Barquin, R. & Edelstein, H. (1997). Planning and designing the data warehouse. New York: Prentice Hall.
3. Bock, B. (2004). SQL for SQL Server. New York: Prentice Hall.
4. Chaffin, M., Knight, B. & Robinson, T. (2003). Professional SQL Server 2000 DTS. New York: Wiley Publishing.
5. Connolly, T. & Carolyn, B. (2005). Database systems: a practical approach to design, implementation, and management. New York: Addison-Wesley.
6. Mannino, M. (2004). Database: desing, application development, and administration. 2nd Ed. New York: McGraw-Hill.
7. Maraks, G. (2003). Modern data warehousing, mining, and visualization. New York: Prentice Hall.
8. Nielsen, P. (2003). Microsoft SQL Server 2000 Bible. New York: Wiley Publishing.
9. O'Brien, J. A. & Maraks, G. (2007). Management information systems 8th Ed. New York: Irwin/McGraw Hill.
10. Post, G. (2005). Database management systems. New York: McGraw-Hill.
11. Raftree, M. F. (2002). SQL Server 2000 Administration. Boston: Course Technology.

Figure 1. Database schema



12. Rob, P. & Coronel, C. (1995). Database systems: design, implementation, and management. Danvers: Boyd & Fraser Publishing.
13. Turban, E., Leidner, D., & McLean, E. R. (2006). Information Technology for Management. Edison: John Wiley & Sons.
14. Zachman, J. A. & Alexander, T. (1997). Data warehouse: practical advice from the expert. New York: Prentice Hall.