Paper 126-26

Migrate to ORACLE®? I need my SAS®!

Dianne Louise Rhodes,Westat, Rockville, Maryland

ABSTRACT

Recently we have heard SAS users asking, "My shop is moving to Oracle|Sybase|Informix. They tell me I can't use SAS anymore. They tell me I don't need SAS anymore. I don't know SQL and I don't want to write SQL pass through. I need my SAS! What can I do?" This paper introduces you to some basic relational database management systems (RDBMS) concepts. It discusses some design considerations used to construct RDBMS. And it answers the questions presented above. Why you still need SAS. How to use SAS with an RDBMS. Easy ways to write SQL and make the connection to Oracle; create SAS datasets from Oracle; and join SAS data and Oracle data.

INTRODUCTION

The ideas for this paper germinated at a Washington DC SAS Users Group (DCSUG) meeting. A big IT shop had just begun converting its legacy systems to Oracle, and management had told the SAS users they "didn't need SAS anymore." At another meeting, we heard users express confusion about relational database management systems (RDBMS). What are they good for? Can't we do the same things in SAS? Eventually, some of these questions showed up on the news group, SAS-L.

When it comes to current computer constructs, too often SAS programmers are behind the curve, particularly if it is a concept that they have not been exposed to by SAS Institute. This can be because they have come to programming through the back door of social science research, statitistics, or pharmaceuticals, instead of computer science. They are not familiar with some of the "hot buttons" like CASE tools, Entity Relationship Diagrams, and Relational or Object Oriented Databases.

RELATIONAL DATABASE MANAGEMENT SYSTEMS

In the early seventies, E.F. Codd described a procedure for producing an efficient database. This process was termed "normalization," the ideal for data storage being fourth normal form. Data in this type of structure was ideally stored in a relational database. Multics Relational Data Store was the first RDBMS, sold in 1978. Other RDBMS include Oracle, Sybase, Informix, Ingress, Microsoft Access, and IDMS

Why move to RDBMS? In the late 70's this was considered to be leading edge. We would expect the current movement to be to replace RDBMS with Object Oriented Databases supported by object-oriented programming. However, only about 1% of all the dollars spent on databases go for OODB. Although RDBMS are no longer "leading edge" most MIS shops have so much invested in legacy systems and the proliferation of C++ that they have little interest in converting to the newest industry standard. RDBMS remain a proven solution for data storage.

RDBMS provide a way of implementing normalization as described by Dr. Codd. However, implementing a RDBMS does not in of itself convey normalization. Normalization takes a lot of analysis and work. A database in fourth normal form is more efficient in the use of space and memory. Normalizing data takes one-dimensional; flat records and expands into two or more dimensions. So by example, it is like the difference between a COBOL record and an array.

RULES OF NORMALIZATION

FIRST NORMAL: REMOVE REPEATING GROUPS

Data before Normalization

PATNUM	DATE1	TEST1	DATE2	TEST2	DATE3	TEST3	DATE4	TEST4	DATE5	TEST5
001	01/01/98	20	01/08/98	19	02/02/98	17	03/01/98	12	04/30/98	20
002	01/01/98	18	01/14/98	16	02/02/98	11	03/05/98	19	05/01/98	14
003	01/01/98	20	01/04/98	19	02/02/98	18	03/11/98	17	04/30/98	15

Remove repeating or multi-valued attributes, name as a new entity-type (or table). For example, if you are tracking repeated measures for a patient create a table and keep one record for each patient for each test. Create a two-part key for it: first part will be the key from the entity-type whose attributes were removed, second party is a unique identifier. This creates a table with each row unique.

First Normal Form

PATNUM	TESTID	TEST	DATE
001	TEST1	20	01/01/98
001	TEST2	19	01/08/98
001	TEST3	17	02/02/98
001	TEST4	12	03/01/98
001	TEST5	20	04/30/98
002	TEST1	18	01/01/98
002	TEST2	16	01/14/98
002	TEST3	11	02/02/98
002	TEST4	19	03/05/98
	...		

SECOND NORMAL: REMOVE PARTIAL DEPENDENCIES

Remove and group attributes that are dependent just on the second part of the above two-part key, name as new entity type. In the example, you do not store information about the test in the table created in step one.

Before Normalization

PATNUM	TESTID	TEST	DATE	TYPE
001	TEST1	20	01/01/98	HIV
001	TEST2	19	01/08/98	HLeV
001	TEST3	17	02/02/98	HA
001	TEST4	12	03/01/98	HIV
001	TEST5	20	04/30/98	HC
002	TEST1	18	01/01/98	HIV
002	TEST2	16	01/14/98	HLeV
002	TEST3	11	02/02/98	HA
...				

Create a unique key and store this data in a new table. Every non-key column must depend on the entire primary key. No non-key column can be a fact about a subset of the primary key.

Second Normal Form

TESTID	TYPE
TEST1	HIV
TEST2	HLeV
TEST3	HA
TEST4	HIV
TEST5	HC

THIRD NORMAL: REMOVE TRANSITIVE DEPENDENCIES

Remove attributes that are not totally dependent on the key, name as a new entity-type with the key being one of the removed attributes. As an example, if you were designing a table for tests administered and each test has a manufacturer, you would have an id for the manufacturer but other information, such as an 800 number, would be in a manufacturer table. That way if you need to change information about the manufacturer, you only need to change it in one place.

Before Normalization

TESTID	TYPE	EIGHTNO	MFG
TEST1	HIV	800-333-1234	SQUIB
TEST2	HLeV	800-555-1200	GLAXO
TEST3	HA	800-522-1600	BURRO
TEST4	HIV	800-333-1234	SQUIB
TEST5	HC	800-522-1600	BURRO

Third Normal Form

MFG_ID	MFG	EIGHTNO
20000001	BURRO	800-522-1600
20000003	GLAXO	800-555-1200
20000004	SQUIB	800-333-1234

TYPE	MFG_ID
HA	20000001
HC	20000001
HLeV	20000003
HIV	20000004

FOURTH NORMAL: REMOVE EXISTENCE DEPENDENCIES

Remove attributes that depend on the value of another attribute, name a new entity-type with the same key. There can be no independent one to many relationships between primary key and non-key columns. As an example, a researcher may have one or many lab assignments or one or many vacations planned but the data cannot be combined in one row in a table because it creates a spurious relationship between assignments and vacations.

Before Normalization

RESNUM	LAB	VACSTART	VACEND
001	CF1	04/08/2000	04/12/2000
001	CF2		
002	AB1	03/04/2000	03/10/2000
002		04/08/2000	04/12/2000

Fourth Normal Form

RESNUM	LAB
001	CF1
001	CF2
002	AB1

RESNUM	VACSTART	VACEND
001	04/08/2000	04/12/2000
002	03/04/2000	03/10/2000
002	04/08/2000	04/12/2000

This process takes data from a wide but relatively short in number set of records, and puts it into many narrow but longer sets of records, or tables. This allows for efficient storage and retrieval using indexing. This process is conducive to transaction processing. RDBMS usually allow for programmatically checking the data before it is actually committed or written to the database. Oracle uses commit, savepoint, and rollback statements. These features are implemented by the use of transaction logs. These logs also provide the capability of restoring a database after a systems crash.

Integrity constraints can be defined to ensure referential integrity. Referential integrity is a condition that exists when all references within a database are valid. Integrity problems can result in the loss of data, wasted storage, and inaccurate data. **Column constraints** are the most common way of enforcing integrity. **Primary key, unique**, and **not null** are used in the table schema to ensure these conditions exist References ensure that relative keys are present, and eliminate invalid relationships. All of these methods work by rejecting data that fails to meet the criteria specified, which can also result in the loss of data. Table constraints can be used to validate data, for example using a **Check** statement to check that date of birth is in a valid range. **Triggers,** or stored procedures associated with a specific operation on a specific table, can perform many of the same operations as constraints.

WHY SAS DATASETS AREN'T RDBMS

None of the above features are readily available, except indexing, in SAS. Although it is programmatically possible to create this functionality, with *SAS/Connect*® and *SAS/Access*® you can use the strength of an RDBMS and still use SAS**.**

Version 8 implements an audit trail, similar to transaction logs, in Beta test.

USAGE

Structured Query Language is used to access an RDBMS. You can access multiple tables using joins and multiple joins. This enables what is termed "dynamic database management"; you create the view of the data you want to use on the fly. How can you learn to use SQL? Oracle SQL is easily learned from the manuals and tutorials. You can use the Oracle Query Builder to build queries without using SQL. SAS SQL can be learned from reference books, papers presented at SUGI, NESUG, etc. You can also use the SAS Access Window to create SQL without learning to write SQL.

LIMITATIONS OF SQL UNDER ORACLE

One limitation is efficiency in terms of time. Most RDBMS are painfully slow to load, particularly if they have large numbers of indices. They are also slow to retrieve data. The complexity of the data scheme contributes to this, and some indexing schemes are better than others are. In one paper presented at SUGI 23 the authors discussed methods of optimizing queries in SAS to Oracle databases. They found that SAS was faster than Oracle in accessing Oracle databases. The authors recommend storing data in SAS datasets for the best performance. (Somerville & Cooper, 1998). Other data schemas may produce better performance under Oracle.

The major limitations are reporting limitations. Oracle SQL does not have the functionality to produce complex statistics, which are easily done with SAS. Some quotes from users contemplating the move to RDBMS (via SAS-L):

My client is an Insurance company. The SAS main users are the actuaries. We mostly do all sorts of analysis on the premiums and claims. I'm quite sure SAS is the best tool for that, but I can't show any figure or facts that would prove that...

We also produce hundreds of reports for management, accounting, claim division, marketing, etc. Since we are moving from legacy systems to Oracle, we will have to convert hundreds of reports. I don't think they could be easily converted to an Oracle report writer product because of the complexity. (Bernard Tremblay, formerly of La Capitale)

I have theoretically tried to pull a random sample from Oracle tables, and there are no random number functions available in SQLPLUS to assign to observations in order to pull a truly random sample.

As with the carpenter's workbench - there are different tools for different purposes. I can't imaging doing a Least Squares Fit or a Time Series Analysis in SQLPLUS...(Michael Hines, Perdue University)

The main strength of RDBMS is as an On-Line Transaction Process (OLTP). Most analysis wants On-Line Analytical Process (OLAP), and you will find many presentations at SUGI 26 on this subject.

A LITTLE BIT ABOUT OLAP

An OLAP is constructed to exist separately from your production Databases. They allow the user to "Slice and Dice" your data. It returns the data to a cube format. E.F. Codd saw that OLAP needed to be supported by a multidimensional data model. In 1993 he wrote (with Salley) "Providing OLAP to User-analysts: An IT Mandate" which described the 12 rules of an OLAP database.

The OLAP Report takes these twelve rules and condenses them into five, under the acronym FASMI: Fast, Analytic, Shared, Multidimensional, and Information.

FAST: the system is targeted to deliver most responses to users within about five seconds, with the simplest analyses taking no more than one second and very few taking more than 20 seconds.

ANALYTIC: the system can cope with any business logic and statistical analysis that is relevant for the application and the user, and keep it easy enough for the target user. The programmer/analyst should not need to know the details of how the data is constructed - a drawback of OODB's.

SHARED: the system implements all the security requirements for confidentiality. If multiple write access is needed, concurrent update locking at an appropriate level. The system should be able to handle multiple updates in a timely, secure manner. This is a major area of weakness in many OLAP products, which tend to assume that all OLAP applications will be read-only, with simplistic security controls. Of all the OLAP products evaluated, only SAS can merge and update.

MULTIDIMENSIONAL is the key requirement. If you had to pick a one-word definition of OLAP, this is it. The system must provide a multidimensional conceptual view of the data, including full support for hierarchies and multiple hierarchies (drill down), as this is certainly the most logical way to analyze businesses and organizations. The underlying database technology is not relevant, provided that the user gets a truly multidimensional conceptual view

INFORMATION is all of the data and derived information needed wherever it is and however much is relevant for the application. The capacity of the product in terms of how much input data they can handle is the key, not how many Gigabytes they take to store it. The capacities of the products differ greatly - the largest OLAP products can hold at least a thousand times as much data as the smallest. There are many considerations here, including data duplication, RAM required, disk space utilization, performance, integration with data warehouses and the like.

WHAT ORACLE SAYS

In the book, Oracle Unleashed, you will find a discussion of OLAP in the chapter "Introducing Oracle Discoverer". This is another product that is used in data warehousing using a star schema.

If you go to the Oracle web site, www.oracle.com, you will find that they are touting Oracle 9I as "the first and only OLAP-ready relational database." On further investigation, you will find that they want to sell you another product, Express Server/ Analyzer. Express Analyzer is an object-oriented tool for navigating, accessing, and analyzing summary data in a data warehouse or data mart. Sound familiar?

But who is the recognized industry Super Achiever for building data warehouses, datamarts, and OLAP? If you are going to have to buy additional products, why not stick with the products that are recognized by the industry as the leaders? SAS Institute has won industry recognition for their data warehousing products.

With SAS one can build a warehouse by moving your Legacy Data into SAS, and from there it can be loaded into an RDBMS (using SAS to load RDBMS).

ENOUGH ABOUT OLAP, I JUST NEED TO RUN SOME REPORTS!

Let's be practical. You have enough ammunition to convince the powers that be that you still need SAS as you are currently using it. You do not want to have to rewrite all your SAS programs in another language and buy yet another tool. Strong supports for sticking with SAS are its ability to manipulate data on the fly and the fact that legacy programs exist.

RECOMMENDATIONS

We would follow the recommendations of Somerville and Cooper. Please note that these are based on SAS Version 6.12; Version 7 and 8 still need to be researched. The first recommendation is to create SAS datasets that mirror your Oracle database using Connect and SQL pass through. Your analytical data should always be separate from your production database. Commonly used joins can be created and stored in views. Once the SAS datasets are created, to get them into the right format for your legacy reports, you can use a combination of data step programming, SQL, and other SAS procedures.

To join SAS data and Oracle data, you must first build a sasfile from Oracle. Then in after you disconnect from Oracle, you can join the two tables. But they both have to be SAS or Oracle; you can't mix them.

```
proc sql;
connect to oracle(user=rhodes_d orapw=***
path="mypathway");

create table extract1 as
   select *
    from connection to oracle
 (select id, age1x, age2x, age96x,
         begrefm1, begrefd1, begrefy1,
         begrefm2, begrefd2, begrefy2,
         begrefm3, begrefd3, begrefy3,
         endrefm1, endrefd1, endrefy1,
         endrefm2, endrefd2, endrefy2,
         endrefm3, endrefd3, endrefy3
      from oracletab1)
   as sasfile;

disconnect from oracle;

/*Limit to ids we want to keep */
create table saslib.match as
  select unique sbdhosp.id
  from saslib.sbdhosp ;
;
create table saslib.extids as
  select * from extract1, saslib.match
   where match.id=extract1.id
   order by id
    ;
quit;
run;
```

If you have SAS/Access®, you can use the database query tools to build SQL for you. There is a good paper on this subject in the SUGI 24. Briefly, you open the Access window, select the tables you want to join. Define the joins and any other conditions. Than you can save the SQL to submit or modify as you choose.

Use PROC Transpose to denormalize your data, that is, to turn instances into variables. And back again. Don't forget about PROC SUMMARY or PROC MEANS using the NWAY option for creating summary records for analysis.

SAS VERSION 7 AND 8

In Versions 7 and 8, you can define a library with an Oracle engine. This allows you to access any Oracle table or view as a SAS database. If your Oracle were on a different platform, e.g. sitting on a Unix box, you would still need to use Connect to get to your data. An example:

```
LIBNAME libref ORACLE user=rhodes_d orapw=***
path='@..' schema=… ;
```

ORACLE VERSION 8 AND 9I

Oracle version 8 has some additional capabilities, which would allow you to pull a random sample from an Oracle table using Oracle SQL. In Version 7, you would have to use SAS.

The Oracle website claims that Oracle 9I has the "full range of mathematical, statistical, financial, and time series functions required by high-end analytical applications."

SAS INSTITUTE AND ORACLE PARTNERSHIP

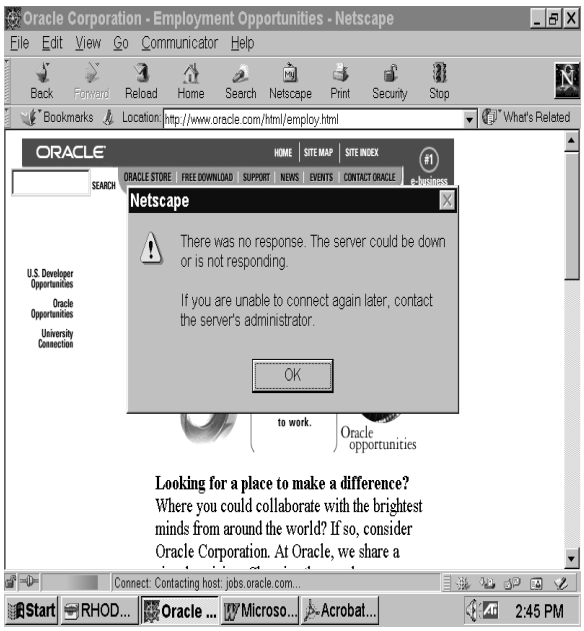
SAS Institute and Oracle have entered into a partnership to develop software for the pharmaceutical industry, so we should look forward to increasing compatibility of the two vendors' products.

PROGRAMMING IN THE REAL WORLD - WHAT THE JOB MARKET SAYS

What are MIS shops using and what kinds of work are real people doing out there? A search of the Washington Post on line ads on Software Developer, Oracle AND SAS produced 16 listings, including Johns Hopkins University Applied Physics Lab, Marriott International, and Fannie Mae.

The Fannie Mae ad specified: The candidate must be able to demonstrate success in developing and implementing projects using relational database tools such as Oracle PL/SQL, SQL*Loader, or related Sybase tools. The candidate should also be able to demonstrate success using SAS 6.08 (or higher) and SAS macros under the MVS operating system, and SAS 6.12 on the client/server platform. They should also be proficient in JCL, mainframe utilities, and mainframe SAS utilities. Must have proven demonstrated success in a SAS and relational database-programming environment. An updated version of this ad pulled in January 2001 stated: The position requires the candidate must have proven demonstrated success managing SAS and Oracle development projects in a very large database or data warehousing environment. The ideal candidate has experience working with SAS version 6.08 or above (mainframe and Unix) Oracle DBMS, and Oracle development tools such as, Oracle PL*SQL, SQL*Loader, and SQL*Plus.

SEARCHING THE ORACLE EMPLOYMENT SITE



CONCLUSION

Stick with SAS.

REFERENCES

Emerson, Sandra L., Darnovsky, Marcy, and Bowman, Judith of Sybase, Inc. (1989). The Practical SQL Handbook: Using Structured Query Language. Reading, Massachusetts: Addison-Wesly Publishing Co.

Sams Publishing (1997). Oracle Unleashed, Second Edition. Indianapolis: Sams Publishing.

Somerville, Clare and Cooper, Clive (1998). "Optimizing SAS® Access to an Oracle Database in a Large Data Warehouse", in *Proceedings of the Twenty-Third Annual SAS Users Group International (SUGI) Conference*. pp.511-519.

ACKNOWLEDGEMENTS

The author would like to thank Bernard Tremblay of Imaginasys, enr., for encouraging me to write this paper. A special thanks to my colleague, Ian Whitlock, for his time spent reviewing my SUGI 25 draft and his valuable suggestions. A round of applause and a big thank you to Paul Dorfman for bringing up my paper and the ideas expressed there in on SAS-L and contributing some real-life examples of OLAP. A smile and thanks to Ron Fehd, for consistently showing up (no fast forward).

AUTHOR CONTACT

Comments, questions, and additions are welcomed. Contact the author at:

Dianne Louise Rhodes
WESTAT
An Employee-Owned Research Corporation
1650 Research Blvd.
Rockville, MD 20850
Phone: (301) 315-5977
Email: diannerhodes@westat.com

TRADEMARKS