Residence Data Warehouse:
Pacific Bell “Jumps” to WARP Speed
Larry D. Bramblett, Pacific Bell, San Ramon, CA
Ann S. King, Pacific Bell, San Ramon, CA

1. Residence Data Warehouse

1.1 Abstract
The Residence Rerate system began as a means to produce a list of Pacific Bell customers who might benefit from optional calling plans. The business need was for a mailing list of one to two million residential customers. The system collects eligible call data and stores it, along with applicable customer account information. In order to accurately predict which customers would save money with a calling plan, and the amount they would save, three months of calling data was rerated at the proposed new rates and compared to the actual rated amount. This information was supplied to an external vendor for the mailing.

The Residence Rerate system has grown into a full-scale Residential Customer data warehouse. It is accessible through the industry-standard SQL language. The architecture of the system ensures that the data, although vast in amount, is maintained in small, manageable portions. The SAS® System was chosen as the platform for the Residence Data Warehouse because it is particularly good at data manipulation and managing information on large numbers of files.

1.2 Features of the System
The system is large, encompassing all the Pacific Bell direct dialed and credit card telephone calls and some statistics on calling patterns for all Residential customers in California. Its design is the result of extensive database tuning to minimize the use of both processor time and disk space.

The Residence Data Warehouse, while built for a very specific, one-time data pull, now supports multiple client data requests from several different platforms. These include direct sales and customer inquiry on a PL/1/DB2® platform, demographic reporting from an Oracle® database running on a Unix® system, promotional sales support for an outside vendor using Windows NT® as well as financial segmentation analysis via an on-line application developed with SAS/AF® software. The Residence Data Warehouse has proven to be easy to modify to support incremental development.

2. Challenges
The development team faced several challenges in handling the enormous volume of data to be processed and stored each day. The system tracks over 10.5 million residential customers in two processing centers. Approximately five percent of these accounts must be processed each day, along with the associated call detail. This translates to about 260,000 customer records and 800,000 call detail records each day in each center, with 450-500 bytes of data per customer record and approximately 120 bytes per detail record.

2.1 Storage
The first challenge facing the development team was storing 50+ gigabytes of data in each processing center so that it was readily accessible and recoverable. The goal was to store the data in a way that would not impact other critical systems. Ideally, the data would be on disk, in files no larger than 100 cylinders (for detail records) to 500 cylinders (for customer account records.)

The standard approach has been to put large files on tape and read them sequentially as masterfiles. Tape storage was not considered for the Residence Data Warehouse because the primary goal was to make the data easily and quickly accessible.
Since the system was originally designed to be a short-term repository to support a one-time data pull, the original request did not justify the time or cost required to model and build an RDBMS. Also, the processing time associated with the table joins necessary to access normalized data in relational DBMSs is prohibitive with large volumes of data.

These considerations led to the decision to store some data elements redundantly. The SAS® system was chosen as the vehicle for the project in part because it stores information about the data as part of its files and does not require extensive data modeling and normalization.

2.2 Processing

From a computer operations standpoint, it was necessary to minimize run times for individual jobs, ensuring that no single job ran for more than two hours. Under no circumstances could the Residence Data Warehouse cause delays for other systems.

In order to efficiently process the data, both throughput - how long processing would take - and recovery (e.g., bad input files or equipment I/O errors) time were considered. This determined that the data would have to be partitioned into many small files.

2.3 Segmentation

The development team had to devise a way to divide the data. The goal was to identify an element that would be consistent throughout the data (i.e., not missing or null for some records) and that would result in a uniform distribution of the data.

There were several data elements that were consistent but uniform distribution was harder to meet. The development team finally decided to try segmenting the data based on the last digit of the customer’s telephone line number. This resulted in an extremely uniform distribution of data for both customer account and call detail.

2.4 Backup and Recovery

Because of the nature of the data stored in the Residence Data Warehouse it was critical to ensure that lost or unusable files are easily recoverable within a fairly short time span.

While the operating system creates scheduled backups, the team decided that additional redundancy was desirable. The call detail and customer account tables are written to two separate sets of files. If the main file is damaged or deleted, the data can be restored from the auxiliary file. Damaged summary tables must be recreated from the call detail files.

As added insurance, the team also implemented a second backup of the critical files. While the PROC COPY procedure was the initial choice for doing this redundant backup, the runtime was too long. An IBM® system utility, DFDSS, is used to copy the SAS libraries to tape.

3. “JUMP” Tables

3.1 Navigation and Control

With over 800 data files created each month for the initial implementation (22 or 23 work days x 4 files x 10 partitions), the biggest challenge for the development team was navigating through the files to find the exact location of current data for any customer and controlling the number of files which had to be searched. It would be excessive in terms of both processor time and elapsed (human) time to search all of the files for an individual entry.

To simplify navigation through the hundreds of files created each month to store the Residence Data Warehouse information, the development team created “jump” tables. The “jump” tables are a road map to the exact location of current data for any customer. They are Metatables that contain information about other tables, much the same as Metadata is information about the data.

The “jump” tables are partitioned by the last digit of the customer’s billing telephone number and indexed by the billing number. Each entry points to the processing cycle number for the last time the customer’s telephone number billed. This allows inquiries to allocate only the file pertinent to the information being requested. (See Figure 1.) The data files are all indexed by the billing telephone number, making access extremely fast.
3.4 Sample MACRO Code

@MACRO custcyc;

******************************************************************************
** this macro creates the **
** jump tables **
******************************************************************************

OPTIONS compress=no;
DATA tmc&digit.;
   LENGTH row_cycl 4;
   row_cycl=&cycl.;
SET
tmd12&digit.90.tmc&cycl.&digit.
   (KEEP=cust_id);

btn_no=floor(cust_id/1000);
KEEP btn_no row_cycl;
LABEL BTN_NO = 'BTN NUMBER'
ROW_CYCL = 'BILLING CYCLE';
RUN;

PROC SORT data=tmc&digit.;
   BY btn_no;
RUN;

DATA tmdl0191.tmc&digit.;
   LENGTH row_cycl 4;
MERGE tmd191p.tmc&digit.
   tmc&digit;
   ;
   BY btn_no;
RUN;
PROC SQL;
   CREATE INDEX btn_no ON
   tmdl0191.tmc&digit. (btn_no);
RUN;
%MEND;

3.5 Reliability

The Residence Data Warehouse has run every
day since its move to production without
intervention and without a single failure. The
extensive testing and tuning that the system
underwent during development is a primary
reason for its reliability. It also incorporates the
WAIT option on the LIBNAME statements that
allocate files from within the SAS® code. This
prevents batch jobs from ABENDING if a
requested file has been migrated to tape by the
system backup and archive facility.
4. The Next Step

4.1 Extendibility
Two major projects are now underway to expand the scope of the Residence Data Warehouse. The first project will expand the account-level information stored for each customer and extend the summary tables to carry six months of history instead of three. The second project includes additional call detail records so the database will carry all Pacific Bell calls. This effectively doubles the size of the Residence Data Warehouse. Both projects are being implemented within a very short timeframe.

4.2 Reusability
The “jump” table concept has been extended to another system to track every call made from a vendor-owned coin phone in California. The system processes approximately 30 million calls each day. It uses data-driven macros and custom formats to insert detail call records into weekly files based on the call date. The “jump” tables for the coin system track by calling telephone number and call date and total calls per number per day. This allows the marketing product manager to immediately detect coin phones which have ceased to generate traffic and contact the vendor/owner of the phone.

5. Conclusion
It is possible to build extremely large data warehouses using SAS® software without sacrificing control or efficiency. The key is keeping individual data files and tables small and balanced and then building Metatables as a navigational guide. The concept is deceptively simple yet very powerful. It requires good data analysis to define and create the underlying structure, but can then be implemented with a minimum of effort.

6. Acknowledgments
The authors would like to acknowledge the following members of the original project team:

Lee Ann Koppel, Pacific Bell
Merrie Lawson, Pacific Bell
Lisa Frakes, SETKA
Anthony Hughes, Computer Power Group
Sandra Morgan, San Francisco Systems
Demetris Papalacovou, Philos Computing

SAS, and SAS/AF are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

Larry D. Bramblett
2600 Camino Ramon, Rm. 2W700E
San Ramon, CA 94583
(510) 823-8320
LDBRAM1@PB1.PACBELL.COM

Ann S. King
2600 Camino Ramon, Rm. 2W600W
San Ramon, CA 94583
(510) 901-8818
ASKING@PB1.PACBELL.COM