Building a Data Warehouse of Call Data Records
Solution Considerations
Many organizations that have established data warehouse solutions reach a point where the solution needs to be extended in an unusual direction. This challenge was given to us by one of our clients – a European mobile virtual network operator (MVNO) – that wanted to add both compliance and archiving functionality to its existing data warehouse.
An MVNO is a mobile phone operator that directly services its own customers but does not own the network assets, instead leasing these assets from a mobile (non-virtual) network operator. The company’s profits are derived from the margin between the price a subscriber pays the MVNO and the wholesale cost of purchasing capacity from the underlying host operator.

MVNOs differentiate themselves by very competitive pricing on certain parts of the call package – for example, ethnic MVNOs target ethnic communities by providing inexpensive calls to their home country. To track the costs of such offerings, especially when subscribers start using the service in unexpected and possibly costly ways, is a key business function, and this is where the data warehouse is used.

The primary role of the data warehouse is to profile subscribers with their associated attributes and their call usage patterns. Our client had this profiling built into a database that provided all of the subscriber information and three months of call detail records (CDRs) and Internet protocol detail records (IPDRs). Beyond the three-month limit there was aggregate data about calls but no direct access to the original call records.

This caused the business a number of challenges:

1. To provide a cost effective queryable archive of CDRs/IPDRs
   The original solution archived the incoming CDR files to disk and subsequently to CD. This meant that if specific sets of records were required, they had to be recovered, reloaded, and then queried.

2. To enable data retention law compliance
   The European Union – as well as other jurisdictions – requires telecommunication companies to retain call and data detail records for a period between 6 and 24 months. While the MVNO was in compliance with the regulations, much of the data was stored on CD and so suffered from the same problem as archived data.

3. To enhance CDR analysis in the existing data warehouse
   The business was looking to be able to analyze more of the CDR data on an ad hoc basis without the need to store more data in the main data warehouse. It also wanted to be able to change the stored aggregates quickly and easily when new data patterns were found. Previously this had required the recovery and reloading of archived data.

   The objective of the project was to design a solution that allowed CDRs to be stored for immediate access for at least two years and that could easily be extended to support up to seven years of data. As a consequence, it was also envisioned that the solution would improve the performance of aggregate load and rebuild when required because the CDRs would be available online.

SOLUTION CONSIDERATIONS

Given the business requirements, a number of technical aspects were then considered. These can be outlined as follows:

1. Should Hadoop or NoSQL solutions be used?
   Hadoop and other NoSQL solutions have grabbed a significant amount of attention over the past couple of years and in the right circumstances they offer a very cost-effective and high-performance solution. The discussion about the use of a Hadoop or NoSQL solution centered on the following aspects:
   a. A strength of Hadoop and NoSQL solutions is in allowing easy mapping of unstructured data. The data files in use have a strong predefined structure – and, importantly, any change to this structure would signify a material change across a number of business systems. There was no unstructured data that we needed to analyze.
   b. There are no existing skills within the IT organization. This in itself should not be a blocker to technological innovation within a company. However, this particular project is about maintaining business as usual and can’t be used as a vehicle for internal training.
   c. Not only does the IT department have no skills but our user community has a range of SQL tools. Therefore, if the NoSQL route was selected, then any data extraction would fall back onto the IT department rather than being self-service by business users.
d. The business has a range of business intelligence (BI) tools such as SAP BusinessObjects BI solutions and these require an SQL database. It was noted that at this point there are no significant BI tools that work with NoSQL solutions.

e. Some companies choose to be at the vanguard or leading edge of technology, others are the last to change or upgrade. This company favors the middle ground of choosing established technologies that offer a route to the future.

2. If it is a database, then what type of database?

Once the Hadoop debate had been put to rest the discussion turned to what type of database to use. Three options were considered:

a. Row-store databases

A row-store database is the type of database that most people are familiar with. Each record is stored as a row in a table and queried using SQL. The organization has a number of different row-store database platforms for operational systems, but the existing data warehouse is based on the SAP® Sybase® Adaptive Server® Enterprise (SAP Sybase ASE) so this would be the natural choice. The primary benefit of this would be to continue to use existing technology but, as a large database, it would not necessarily perform acceptably and would require significantly more disk space than the current raw files, as both data and index space would be required.

b. MPP-based data warehouse appliances

There are a number of massively parallel processing (MPP) architectures. These also store the data in rows, but they apply many CPUs with data on disks close to the CPU to speed up the problem. While these can be very attractive solutions they posed two specific problems to the project.

First, it would not be cost-effective to retain the existing data warehouse on the current platform and create an archive store on an MPP solution because MPP solutions generally match CPU and disk together for performance. Therefore, to get high performance there would be a significant underutilization of the available disk resource.

Second, and as a consequence of the first issue, if the project were to move the existing data warehouse to an MPP platform, it would require porting/rewriting of the load processes (also known as ETL) that would add cost and time. This was exacerbated by the use of platform-specific user defined functions (UDF) in the current data warehouse that are not available on the MPP platforms.

c. Column-store databases

The third option was the choice of a column-store database. To the user a column-store database looks identical to a row-based database, but behind the scenes the data is actually stored in its columns with a record of data being made up of pointers to the data held in the columns.

Column-store databases consequently store only one copy of each value in a column. If you have a column in a table that contains the calling telephone number, then each telephone number will be repeated many times. Storing it only once and using a pointer to the string considerably reduces space, and because less space is used, less input/output is required to retrieve the data from the disk, and this also speeds up the query time. Both a smaller disk footprint and faster query time make column-store databases very attractive.

Column-store databases are considered more difficult than traditional row-based databases to update because the update has to check the data in the record map and in the column store itself. This is not an issue for data warehousing and archiving solutions where the data is usually loaded once and not updated.

Despite a lack of general awareness of column-store databases, they have been around since the mid-1970s and Wikipedia currently lists 21 commercial options and 8 open-source solutions. The most established column-store database in the market is SAP Sybase IQ, which fitted nicely with the existing data warehouse on SAP Sybase ASE.

Based on the choices, the client chose to go with a column-based database. The next stage of the project was to build a prototype SAP Sybase IQ store to understand what sort of performance and compression we obtained.
SOLUTION DESIGN

The solution design was to create a simple schema with two types of tables, one for the CDR data and one for the IPDR data. It was also decided that the tables would be split into multiple data sets based on the start date of the call or data session. As a result, we ended with a number of tables of the form CDR_2012, CDR_2011, and so on, and IPDR_2012, IPDR_2011.

Views were created over these tables, called CDR over the CDR_XXXX tables and IPDR for the IPDR_XXXX tables. A further view was created called DATA_RECORDS across all CDR_XXXX and IPDR_XXXX tables but selecting only the fields that were common to both — for example, start time, cost, record type. This was done for future ease of management, as it would be possible to remove an entire year of data simply by rebuilding the view to exclude the year and then dropping the table.

One year of call records consisted of around 500GB of raw data and 1 billion rows of CDRs. When loaded into the database, it took up less than 100GB, achieving a compression ratio of 5.7:1. Similar results were obtained for the IPDRs. Therefore a 2TB system (4TB when mirrored) could effectively hold seven years of history, even allowing for predicted growth.

The testing split the performance into two aspects — the load time and query performance.

The CDRs are delivered from the host operator on an hourly schedule as fixed format flat ASCII files that typically contained around 35,000 records, where each record was 583 characters long containing 80 fields. Loading an average file (typically 35K records in a 2MB file) took around one second, and since there are around 200 files a day this was not a burden on the system. It even allowed data to be loaded throughout the day rather than as a single nightly batch, thus allowing more immediate access to the CDRs.

Initially two years of data was loaded although this will be allowed to grow over the coming years. User query times directly on the SAP Sybase IQ database varied dramatically depending on the type and complexity but were typically 10 to 15 times faster than the equivalent SAP Sybase ASE query times. In addition the queries were working on a 24-month data set rather than on the original 3-month data set.

The existing data warehouse was left in place and using SAP Sybase ASE with SAP Sybase IQ permitted the users to access both data sets from a single unified interface. The aggregate load, held in the existing SAP Sybase ASE data warehouse, was also modified to build from the SAP Sybase IQ store rather than from the raw files. This reduced load complexity and allowed the IT team to be more responsive in delivering new aggregates.

In summary a CDR/IPDR archive was created on a second server with directly attached storage, roughly equivalent in size and performance to the original data warehouse platform. It was accessible via SQL, allowed users access from within the current environment, compressed the data by more the five times, and was typically ten times faster to query.

ABOUT THE AUTHOR

David Walker has been involved with business intelligence and data warehousing for over 20 years, first as a user, then with a software house, and finally with a hardware vendor before setting up his own consultancy firm Data Management & Warehousing (http://datamgmt.com) in 1995.

David and his team have worked around the world on projects designed to deliver the maximum benefit to the business by converting data into information and by finding innovative ways to help businesses exploit that information.

David’s project work has given him experience in a wide range of industries including telecommunications, finance, retail, manufacturing, transportation, and public sector as well as a broad and deep knowledge of business intelligence and data warehousing technologies.

David has written extensively on all aspects of data warehousing from the technical architecture and data model design through ETL and data quality to project management and governance.