



Retrieval Mechanisms within a Communications Data Retention Environment

A CopperEye Technical White Paper

November 2007

Table of Contents

INTRODUCTION.....	3
PROBLEM CHARACTERISTICS	3
FULL SCANNING	3
PARTITIONING	4
INDEXING	5
CONCLUSION	5
COMPANY INFORMATION.....	6

Introduction

This whitepaper looks at various retrieval mechanisms that may be employed for communications data. Data retrieval in communication generally involves queries that are acutely selective and typically return less than 0.001% of the data, from a large volume of (hundreds of billions) of call detail records. The mechanisms considered in this paper are those that are most prevalent in currently available solutions and include full scanning, partitioning and indexing. The paper demonstrates that indexing is the most appropriate, efficient and cost effective mechanism for such environments.

Problem Characteristics

The communications data retention and retrieval environment has a number of characteristics which distinguish it from conventional transactional or data warehousing problems. These include:

- **High Ingestion Rate.** GSM telecoms networks typically generate 25 million call record per day for each 1 million subscribers registered to the network. Therefore a medium sized GSM network of 10M subscribers will create approximately 250 million call detail records every day. This requires that the data store ingest and make this data accessible at very high speed.
- **Large Retention Volume.** Legal directives typically require a telecommunications network to retain call detail records for 1 year or more. Coupled with high ingestion rates, this generates extremely large retained data volumes. There will be of the order of 10 billion call records retained for each 1 million subscribers registered to the network. This results in 100 billion call records retained for a network with 10M subscribers. Call records typically average 250 bytes of storage which would demand a total storage requirement of 25Tbyte for 10M subscribers.
- **Selective Retrieval.** The subject of the data retrieval is usually limited to specific subscribers and/or specific equipment (such as handset) and/or location. These queries return a very small proportion of the overall data volume. Even retrieving the entire call history for a single subscriber from amongst 10M subscribers will return less than 0.00001% of the total call records retained.

Full Scanning

One approach to finding and retrieving specific call records is to scan across the entire retained volume and filter out the required records. Assuming the storage is spread over a number of 500Gbyte disks with an excellent disk transfer rate of 50Mbyte per second and with enough CPU resource to exercise all disks at peak transfer rate in parallel, it would still take 10,000 seconds to scan across all of the data to locate and return a result, regardless of the number of records sought and returned.

This time can be reduced by compressing the data, increasing the number of disk spindles and increasing the number of CPUs. However in order to obtain a retrieval response time of a minute or less, a performance of improvement of 3 orders of magnitude would have to be achieved. The best compression ratio that

can be expected is 10% of the original volume. To close the remaining performance gap, a 100 fold expansion in hardware (mostly disk spindles) would be required.

This clearly is an architecture which is inappropriate to the problem at hand. It is better suited to broad analytical queries which address a large proportion of the overall data volume and summarise the overall characteristics of the data queried. Analytical queries should be optimised for the best query throughput, while selective queries should be optimised for best response time. A solution that is architected for analytical queries is unlikely to offer optimal response times for selective queries. Aside from the enormous cost and reliability implications of such a vast hardware footprint it also has the following implications:

- The retrieval response time is linearly dependent on the total volume retained unless there is a commensurate upgrade in hardware resources.
- With all of the disks running at peak transfer rates, the retrieval response time is linearly dependent on the number of concurrent active queries and highly dependent on any data ingestion activity.

Partitioning solves a portion of the selectivity problem, but this too has its limitations.

Partitioning

Partitioning is an approach typically adopted by the appliance vendors and by database vendors to divide and/or parallelise potentially large searches.

Partitions will be arranged so that they can be eliminated at query time by virtue of the query criteria. Effectively the partitions are acting as indexes, analogous to coarse-grained indexed organised tables (content stored by access dimension). One typical partitioning dimension is transaction time, with daily partitions to ease data management. Here, a retrieval of a subscriber's data for one month will address at least 28 of the 365 (28 days in a month of 365 days in the year) possible partitions which results in 8% selectivity. This is still 800,000 times worse than the selectivity required to yield the best possible query speed – which is 0.00001% selectivity to return all records for a single subscriber across a year

Clearly it is possible to further sub-partition each partition to an arbitrary depth to offer more selective partition elimination. For example, each daily partition can be sub-partitioned by subscriber identity range (say with 1000 partitions) and the partition selectivity becomes much better. Of course this assumes that the query criteria match the partition schema exactly. Partitioning cannot help retrievals with criteria that differ from the partition arrangement. For example, partitioning by date and then subscriber cannot assist a query purely predicated around equipment or location. In these circumstances, retrieval performance reverts back to that of a full scan described above.

With multiple orthogonal retrieval criteria, the only partitioning solution is to replicate the data across multiple partitioning schemas with a separate schema for each combination of retrieval predicates. Clearly this leads to an explosive growth in storage requirements, potentially taking it into petabyte territory, and adds the complication of ensuring all replication schemas provide a consistent view at any one time.

Indexing

A selective index is an index that provides direct access to the data required and minimises the disk I/O workload for a selective retrieval. Each selective index is navigated by key value (such as subscriber identity) and yields the precise addresses for each of the communications records sought. This reduces the disk I/O cost to one disk I/O for each call record returned.

Multiple selective indexes can be built on top of the data records to afford the retrieval dimensions required. For example, a selective index can be built for call date/time, subscriber identity, dialled number and for handset serial number, etc., to allow retrieval by any of these or by any combination of these. Thus, a subscriber index can provide the addresses of a single subscriber's call records from all of the retained records allowing each record to be retrieved by a single disk I/O. While index navigation adds some overhead itself, it is typically a small proportion of the disk I/O incurred by the retrieval overall.

For data management purposes, data and indexes are typically organised into rolling day partitions to allow data beyond the retention window to be dropped easily. While this adds to the index I/O overhead because it requires multiple index partitions to be scanned, it is still a minimal overhead compared to the cost of retrieving the call records overall.

For example, to retrieve a subscriber's records for a single month (30 days x 25 records per day) by selective index access will require approximately 150 disk I/Os to navigate the daily index partitions (30 days x 5 per index) and 750 disk I/Os to retrieve all of the communication records. This total of 900 disk I/Os can be easily serviced by a single disk in a handful of seconds. Moreover, this response time is governed by the volume of records sought rather than the volume of records retained and multiple disks can service multiple concurrent queries with minimal contention. Contrast this behaviour with that of scanning with or without partitioning.

Conclusion

For applications that require very targeted queries, rather than broad scale searches, selective indexes provide far better performance than other retrieval algorithms. In order to use the other mechanisms and achieve the performance required, the limitations in the software must be overcome with significant hardware augmentation.

Company Information

CopperEye is a leading provider of data retention and retrieval solutions designed to meet the requirements of organizations challenged with explosive growth in the volume of data they must manage. CopperEye's software helps companies cost-effectively capture, quickly store, and instantly access selective transactions from tens or hundreds of terabytes, whether the data is seconds or decades old. This is increasingly important as data volumes more than double every year and regulations require organizations to keep information longer.

In addition, CopperEye's patented indexing technology approach delivers a quantum leap in performance in an open access format. While eliminating vendor and technology dependency, it supports rapid data retrieval from enormous volumes of information at a fraction of the cost of traditional database and data warehouse approaches.

CopperEye has dual headquarters in Bath, UK, and Stamford, Connecticut, USA.

UK Corporate Headquarters

CopperEye Ltd, Suite 47, Aztec Centre, Almondsbury, Bristol BS32 4TD
t +44 (0) 1454 203610 f +44 (0) 1454 203330

US Headquarters

CopperEye, 101 Federal Street, Suite 1900, Boston, MA 02110, USA
t +1-617-342-7173 f +1-617-342-7080

e contact@coppereye.com
w www.coppereye.com