Live Aggregate Projections: Making Big Data Seem Small

Chuck Bear[1], Martin Arlitt[2], Brad Morrey[2]

[1]Vertica, [2]HP Labs

{firstname.lastname}@hp.com

Abstract

Today, many organizations have transitioned from making intuition-based decisions to data-driven decisions. These organizations are already collecting and analyzing their historical data, but now face increasing pressure to integrate data analytics into the operational aspects of their business. As this transition evolves, datasets are getting larger while decision-makers, regulators, etc., are demanding ever quicker analysis. For example, cellular providers are being mandated to alert users to excessive roaming charges within minutes. The HP Vertica Analytics Platform version 7.1 contains a new feature called "Live Aggregate Projections". This feature provides fast and accurate aggregate statistics, making fresh data available to such applications. We explain the design and capabilities of Live Aggregate Projections, and demonstrate via experimentation performance improvements of up to 500x for queries that use this new feature.

Problem statement

For decades, organizations have used data to influence business decisions. Initially, data was scarce and therefore was used mostly by decision-makers to complement their intuition when making decisions. "Data warehouses" were set up to archive the data, and practices like "overnight batch windows" were established in many organizations to permit slow queries to run to completion. Over time, many organizations have realized that substantial business value can be obtained from data. A shift to data-driven businesses is leading to significantly larger datasets and expectations of very quick query results. For example, it is no longer sufficient for cellular providers to generate monthly bills; they must alert users to excessive roaming charges within minutes. Similarly, financial analysts need to combine long-term trends with the current price spread, and power grid operators need to make long-term capacity planning decisions while also being alerted to demand anomalies in real-time. "Big Data Analytics Platforms" like HP Vertica were developed specifically to address this new paradigm.

Compared to traditional data warehouse technology, HP Vertica can store orders of magnitude more data and process queries orders of magnitude faster. Nevertheless, there are scenarios where the datasets are sufficiently large (and continuously growing) that the response times for certain resource-intensive queries may exceed expected limits. For example, the emerging "Internet of Things" market is expected to see billions of connected devices continuously collecting data that organizations want to convert into actionable information. For queries that involve calculating aggregate statistics (e.g., count, maximum, mean, minimum, sum, most recent, most relevant) the overhead can be quite high, and as a result, the response time longer than may be desired.

Our solution

Live Aggregate Projections (LAPs) are a new feature in HP Vertica that permits a projection to be created that contains a set of desired aggregate statistics for data in a table. As new data is loaded into the table, the aggregate statistics in the LAP are automatically updated. When a user writes a query that requires aggregate statistics about the data in the table, they can look up the aggregate statistics in the projection rather than perform a resource-intensive scan of the table to calculate the desired statistics.

Figure 1 illustrates the high-level design of the Live Aggregate Projection feature. Data from each load transaction is copied and stored in multiple "projections" (a). For example, a copy of the full data batch may be placed in a base projection, while a second copy undergoes aggregation (summation, counting, min/max, most recent, etc.) per key (meter ID, customer ID, date, etc.). Load batches for related projections are committed or rolled back as an atomic unit, ensuring transactional consistency. Background tasks (b) combine small partial aggregates into larger ones (by summing the sums, taking the maximum of the maximums, etc.), further aggregating the data. Lastly, if multiple partial aggregates remain, they are combined at query time (c). Stored data for LAPs is sorted by the aggregation key, so combining partial aggregates is an efficient "merge" operation applied to sequential streams, and data for individual keys can be found by searching, rather than scanning.

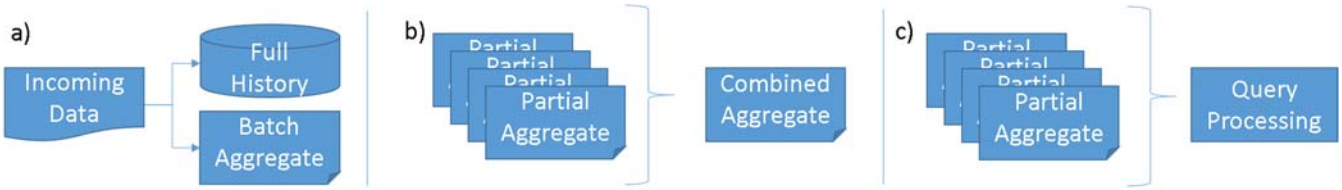

Figure 1. High-level design of the Vertica Live Aggregate Projections feature.

As an example of how to use a Live Aggregate Projection, consider an organization that records data about each of its sales; when it was made, by whom, and for how much. This timeseries data can be stored in a simple table like "sales_data" (Figure 2). To create a LAP for this data, the first step is to create an anchor table ("sales_anchor") that includes all of the data (this requirement may be removed in the future). Next, the LAP is defined. In this example, a monthly summary of the number of sales and total revenue for each employee is determined ("sales_lap"). The LAP is updated as data is loaded into the main table ("sales_table"). As the example query shows, users make use of the data in Live Aggregate Projections just like they would a table.

Sales Data

```
2014-01-01 12:42:00|alice|300.00
2014-01-15 13:42:00|alice|600.00
2014-01-21 10:42:00|bob|150.00
2014-01-31 19:42:00|alice|400.00
2014-02-01 11:42:00|alice|500.00
2014-02-11 09:42:00|bob|100.00
2014-02-28 11:42:00|alice|1000.00
```

Table Definitions:

```
CREATE TABLE sales_data (
date_of_sale TIMESTAMP,
employee VARCHAR(30),
sales_amount FLOAT);

CREATE PROJECTION sales_anchor AS
SELECT * from sales_data
SEGMENTED BY HASH(employee) ALL NODES KSAFE;
```

Live Aggregate Projection:

```
CREATE PROJECTION sales_lap(
   employee ENCODING RLE,
   month ENCODING RLE,
   num_sales ENCODING DELTAVAL,
   revenue ENCODING DELTAVAL)
AS SELECT employee,
 EXTRACT(year from date_of_sale)*100
+ EXTRACT(month from date_of_sale) as month,
 COUNT(sales_amount) as num_sales,
 SUM(sales_amount) as revenue
FROM sales_data
GROUP BY employee, month;
```

Example Query

```
SELECT employee, month, revenue
FROM sales_lap
GROUP BY month, employee, revenue
ORDER BY month asc, revenue desc;
```

Query Output

```
employee | month | revenue
---------+--------+---------
alice    | 201401 |    1300
bob      | 201401 |     150
alice    | 201402 |    1500
bob      | 201402 |     100
```

Figure 2. An example of how to use a Live Aggregate Projection.

Evidence the solution works

To demonstrate the effectiveness of Live Aggregate Projections, we use IoTAbench [1] to conduct an experimental evaluation.The system under study is a cluster of three HP ProLiant DL320 g8 servers. Each server has a single Intel Xeon E3-1270v2 CPU, 32 GB memory, four 4TB 7.2K SAS hard drives, and 1 Gb/s NIC. The internal cost of this system is ~$10,000. RHEL 6.5 and Vertica 7.1 (Dragline) is installed on each of the servers.

To demonstrate the capabilities of this system, we consider a smart meter use case involving 10 million meters recording a consumption value every 15 minutes for a one month period. This are the parameters used in a 2012 Oracle white paper [3]. To calculate "bill determinants" for Time of Usage billing, Oracle used a 42U rack full of servers. The Oracle system required 20 hours to calculate the monthly bill determinants for all 10 million meters. With Vertica 7.1 running on three DL320s ($1/14^{th}$ of a 42U rack), the same calculation took 18 minutes – without using LAPs. LAPs enable the bill determinants to be calculated in **24 seconds**, 3,000x and 45x faster respectively.

To further highlight the benefits of LAPs, we conducted two experiments. In the first experiment, we vary the number of distinct items that we must keep aggregate statistics on (all other variables are fixed; in particular, the meter reading frequency is set to twice per hour). Figure 3(a) shows the results for two different queries: Time of Usage Billing (bill determinant calculation) and Top Consumers (sorted list of the meters by total consumption). The LAP version of each query is more than an order of magnitude quicker (the axes are log-scale) than the "normal" query implementations, even as the number of meters is increased by two orders of magnitude.
Our second experiment considers the case where the number of readings to be aggregated for each meter increases. We anticipate this will happen as organizations seek to extract more business value from sensor data. For example, with very frequent readings, it is possible to extract per-appliance electricity consumption from aggregate smart meter data [2]. With "normal" queries, increasing the number of readings per sensor means significantly more work to be completed when the query is submitted. With LAPs, the additional data is processed when it is loaded, enabling the subsequent queries to complete very quickly. Figure 3(b) shows that as the reading frequency increases from twice per hour to 60 times per hour, the time to complete the "normal" queries increases linearly with the amount of data. In contrast, the time to perform the LAP version of each query remains constant, as the amount of work to be done at query time is the same. When there are relatively few datapoints to aggregate

for each distinct meter, LAP queries complete about one order of magnitude faster. The performance improvement widens arbitrarily as the number of datapoints per meter grows. For example, when each meter has a reading every minute, Live Aggregate Projections enabled the Time of Usage billing query to complete 259x faster, and the Top Consumers query to complete 486x faster than the "normal" versions of those queries.

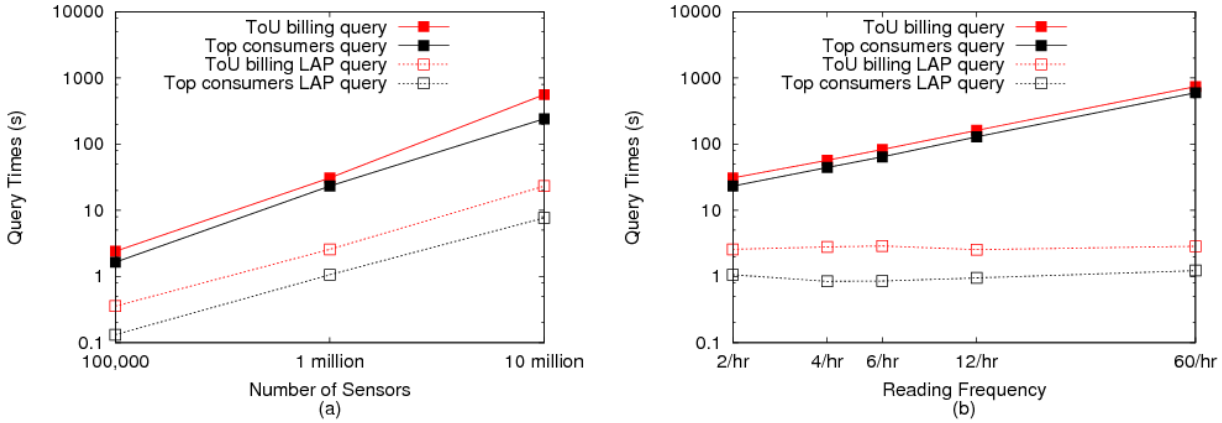

Figure 3. Examining the performance benefits of Live Aggregation Projections.

Competitive approaches

As the problem of improving query performance on large datasets is not new, there have been several approaches developed over the years. For example, "Cubes" are structures built from the data that can answer aggregate queries quickly. However, most organizations build these nightly, and therefore they are unsuitable for real-time queries on fresh data. Newer techniques that utilize distributed in-memory storage and realtime ingestion and computation have been proposed [4]. These are separate solutions that are disjoint from the source database. "Materialized views" are another common approach intended to improve query performance. However, none of the implementations simultaneously provide transactional consistency, fast query on well-organized and compressed data, and low system overhead. Our solution addresses these needs and also provides in-memory operations for small datasets, graceful degradation for large datasets, and the ability to delete old data.

Current status

Live Aggregate Projections were added as a feature in version 7.1 of the HP Vertica Analytics Platform, which was released in August 2014. We expect to have production customer adoption after the first maintenance version is released, and anticipate having telco, financial and/or equipment data (or other use cases) by TechCon.

Next steps

The current implementation of Live Aggregate Projections requires an "anchor projection" that is segmented around the cluster using a subset of the aggregation keys. We are considering lifting this requirement in a future release. The Vertica query optimizer does not automatically re-write queries to use LAPs; this is being worked on. We are working with HP Labs on an in-memory B-Tree structure to hold LAPs [5]. This would improve the freshness of the data on tiny transactions, and could be used to trigger events or notifications. Lastly, we are integrating LAPs with a direct query API. This is expected to support large numbers of point queries, enabling end users to obtain personal analytics on their aggregate data in real-time.

References

[1]. M. Arlitt et al., "IoTAbench: an Internet of Things Analytics benchmark", presentation at HP TechCon 2014.
[2]. M. Marwah et al., "Unsupervised disaggregation of low frequency power measurements", poster at HP TechCon 2011.
[3]. Oracle white paper, "Meter-to-Cash performance using Oracle Utlities applications on Oracle Exadata and Oracle Exalogic", January 2012.
[4]. L. Abraham et al., "Scuba: Diving into Data at Facebook", PVLDB, August 2013.
[5]. C. Bear et al., "Towards enabling small queries in Vertica", HP TechCon 2013 submission.