VER 1
2017

Dimensional INSIGHT

Spectre™: First Look

DIVER PLATFORM 7.0 DATA PROCESSING ENGINE

JAMES CLARK & JEAN COLLINS

Spectre: First Look

Table of Contents

Introduction

Version 7.0 of Diver Platform, Dimensional Insight's award-winning business intelligence platform, includes the new powerful Spectre engine. The Dimensional Insight Spectre engine combined with cBase, the latest columnar database technology, delivers robust business intelligence for information analysts, managers, and consumers.

Spectre was designed from the ground up to optimize performance and perfect ease-of-use features for all users. Spectre makes the latest version of Dimensional Insight's flagship product fast, scalable, and manageable to meet the needs of today's enterprise customers.

Evolution of the Dimensional Insight engine

Dimensional Insight's classic Model engine has been in the vanguard since it was first introduced. The Model engine optimizes performance on servers with more limited specifications than today's servers. Remember when a 32-bit OS with a dual core processor, 1 gigabyte of RAM, and a couple gigabyte hard drive was standard? Servers with fewer cores and less memory and disk space present a design challenge when the goal is to deliver the speed needed for business intelligence.

By design, the Model engine does a lot of the work up front to give good performance on limited hardware. Model pre-calculates many dives during the build process, saves the tables, and at run-time loads the results when requested. The Model engine makes end-user performance fast, even without fast hardware.

However, enterprise-level businesses are seeing rapid changes in the business environment, such as the increase in the quantity and diversity of data in varied formats from different platforms and the increased need to combine views. Spectre takes advantage of the latest hardware advances, such as faster core speeds and multiple cores for built-in parallel processing, large amounts of memory, solid state disk (SSD), and advanced compiler technology, to radically boost performance in these environments.

[Table 1 gives a side-by-side comparison of the Model and Spectre engines, highlighting areas where Spectre excels.

[Table 1: Model and Spectre: Side-by-side comparison]

Model	Spectre	Notes
Precomputes dive tables at build time for fast run-time performance.	Builds in-memory cBase columnar databases that stores values in binary format. The build does not precompute dive tables.	The Spectre cBase columnar database does not need to precompute dive tables, so it provides very rapid calculations with greatly reduced build time and smaller on-disk size.
Specifies an upper limit on the number of dimensions.	Supports unlimited dimensions.	Spectre does not need to prespecify which columns are dimensions, summaries, or info fields.
Uses consolidated multimodels, for example, separate models for each fiscal year.	Does not need to use consolidated multimodels.	Spectre builds are faster, and cBases are smaller than models, so consolidated multimodels generally are not needed.

Overview of Spectre

Spectre and cBase, Dimensional Insight's completely reengineered columnar database technology, give the greatest increase in speed and efficiency to features that are most used by Dimensional Insight power analysts. The most important fundamental change is the new column-oriented, shareable database storage format optimized for query-time calculations instead of build-time calculations. The new design takes advantage of hardware innovations and analysis practices to better handle new user behaviors and queries.

Columnar database design

Spectre uses an in-memory, binary-format columnar database to optimize the most commonly used functions. So, just what is columnar database technology and why is it faster? Spectre's speed comes from perfectly matching the design with function. Mining actionable information from diverse sources requires consolidation of large amounts of data and the ability to quickly access the data in the aggregate as well as the underlying details.

Here's where the design is critical. Typically, a relational database stores fields in a record together, like rows in a table. The records have multiple fields that are potentially of different data types, such as string, integer, fixed, or double. Each record might use a block of memory, or there might be multiple records in a block. Regardless, all of the fields for a record are stored consecutively. This is a great design when you want to retrieve all the fields of a record every time the record is accessed. However, business intelligence queries typically need to access only one or a few fields of each record. For these queries, the row-oriented design is not very efficient.

Contrast the columnar database design with the row-oriented design. In the columnar database design, instead of storing all of the fields for each record together, the records are broken up. The "like" fields for all records, or each column of a table, are stored together in blocks of memory. Now, when you want to perform calculations over the data, such as a SUM, MAX, MIN, COUNT, or AVG, only the relevant columns need to be accessed, thereby making calculations very fast.

Let's use a simplified sales example to show the differences. Suppose you have a table as shown below with quarterly sales figures for four products.

[Table 2: Sample quarterly product sales figures]

Product	Q1	Q2	Q3	Q4
Product A	5000.00	5200.00	4700.00	7000.00
Product B	1400.00	1100.00	900.00	1200.00
Product C	10000.00	10000.00	10000.00	11500.00
Product D	300.00	1500.00	600.00	2200.00

In a row-oriented database, the records are stored like this:

[Table 3: Row-oriented database]

Product A,5000.00,5200.00,4700.00,7000.00;
Product B,1400.00,1100.00,900.00,1200.00;
Product C,10000.00,10000.00,10000.00,11500.00;
Product D,300.00,1500.00,600.00,2200.00;

To answer typical business intelligence queries, such as "What are the total Q3 sales?" or "What product has the highest sales figure in Q4?" you need to access all rows in order to get the relevant quarterly sales figures. Alternatively, row-oriented databases can build and maintain multiple indexes to answer this type of query, but the indexes add the overhead of additional disk space and memory.

In contrast, in a columnar database, the records are stored by grouping all values for a given field together. For example, all of the product names are together, then all of Q1 sales, then all of Q2 sales, and so forth:

[Table 4: Column-oriented database]

```
Product A, Product B, Product C, Product D;
5000.00,1400.00,10000.00,300.00;
5200.00,1100.00,10000.00,1500.00;
4700.00,900.00,10000.00,600.00;
7000.00,1200.00,11500.00,2200.00;
```

The beauty of this design is how quickly you can answer the business intelligence queries posed earlier. With the self-indexing columnar design, you don't need to access all records for all products to get the quarterly figures. You don't even need to access the sales figures for quarters that you're not interested in. With the columnar database, you only access data elements that are relevant to the query. With databases containing millions of records, you can see how quickly this savings adds up.

As shown in [Table 5, tests with customer data compare Spectre to the Model engine. Input to the builds were 46 million rows, 215 columns, and 102 GB of text. The Spectre build time was faster, the resultant database size smaller, and the time to open a nine-dimension MultiTab more than one thousand times faster.

[Table 5: Compare classic Model engine with Spectre]

Task	Classic [On server hardware]	Spectre
Build time	3 hours (in parallel) (13 Dimensions)	24 Minutes
Model/cBases size	227 GB (51 Models)	20 GB (1 cBase)
Marker Open	**4200 Seconds (70min)**	**3 Seconds**

Speed

Spectre is built for speed, both for builds and for calculations, significantly boosting productivity of IT staff and response time for clients.

Build times

Spectre can process 500 million – 1 billion rows of data in significantly reduced build times relative to Model builds. Because Spectre performs query-time instead of build-time calculations and does not need to pre-summarize to compute dives in advance, Spectre build windows can be reduced as much as 80%. With the smaller build windows, current data can be available to users quicker and more frequently.

Run-time performance

Spectre and the columnar database design make run-time performance extremely fast. This is because the Spectre engine algorithms optimize run-time performance for some of the most commonly used Diver computations. In particular, Spectre outperforms the Model engine on these specific functions:

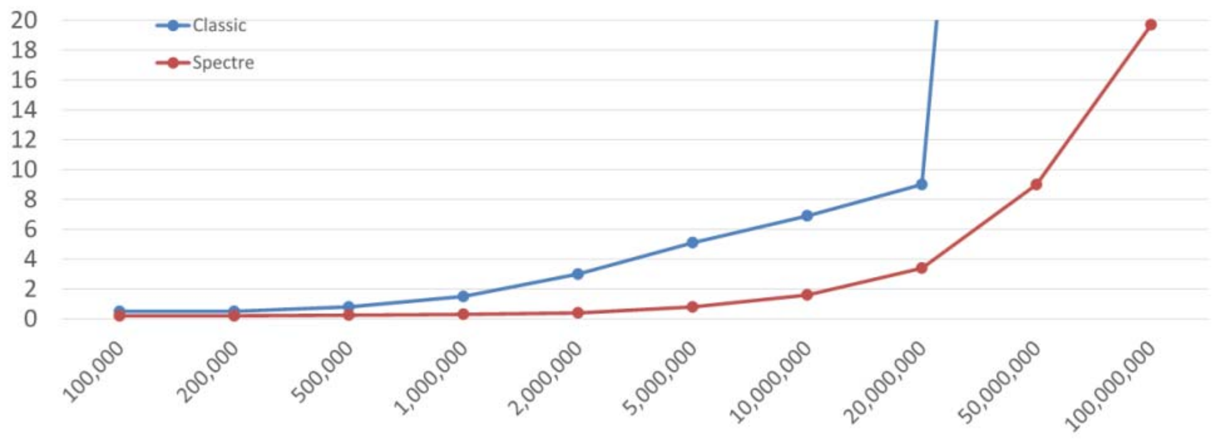
- Dimension Counts (DimCounts): Calculation of the number of unique values for a selected dimension.
- Groups: Functionality to combine multiple diveable entities into a single unit for diving.
- MultiTab: A tabular display option that presents multiple dimensions vertically.
- Multimodel: A combination of several models that contain the same or similar sets of dimensions, summaries, and info fields.
- Time Series: Functionality to present columns that are limited by time-based periods.

With the Model engine, these functions might have limitations when working with large models. DimCounts are computation-intensive and could be slow for very large data sets with many unique values in the dimension. DimCounts, Groups, MultiTab, and Time Series all require extra computations such as merging tables.

In Spectre, the traditional file size, column count, and dimension limits have been removed, the need for consolidated multimodels has been eliminated, and overall file size has been drastically reduced. The Spectre calculation engine is based on the LLVM compiler, which compiles formulas into machine code that is optimized for the processor on which it is running. The machine code runs raw as opposed to being run through an interpreter. All these design changes combined with the columnar database remove limitations, shave off computation processing time, and deliver fast run-time performance.

[Figure 1 shows the exponentially improved response time of Spectre over the classic Model engine when diving into a 100 million row data set with a three-dimension MultiTab. Spectre took merely 20 seconds, while Model was still running after 20 minutes.

[Figure 1: 3-Dimensional MultiTab time (seconds) over number of records]



Scalability

The Spectre design is robust enough for challenging enterprise-level business intelligence analysis and delivers fast performance without taxing resources.

Database size

Because Spectre does not need to pre-summarize or maintain separate indexes, the on-disk size of the columnar cBase format is smaller than Model database size. Spectre handles larger data volumes in a single cBase where previously multimodels were used.

Cache

Cached dives deliver fast response while avoiding stale results. As an in-memory data engine, Spectre answers most queries without needing to access the hard drive. The results are cached for reuse the next time you make the same dive request. Spectre can share the cache among multiple users with the same access when they do the same dive. People with different access levels use a different cache, so as not to compromise security.

When you rebuild a Spectre cBase, you can refresh the cache entries of previous dives. With this option, Spectre refreshes the cache of actual dives so that when you dive into previous dives, Spectre returns refreshed, not stale, results.

Memory

When Spectre loads parts of a cBase into memory, it does so in a way that is shared across multiple Spectre processes. If two different dives are running at the same time, the dives do not use twice as much memory for the database. Frequently, the database required for a dive is already in memory, so Spectre doesn't need to load it from disk. Users with different access share this memory, and Spectre processes ensure that each user gets the right results based on his or her access.

Low per-user overhead

Spectre delivers fast user performance with a low idle-connection cost per user. This supports more simultaneous users without a linear increase in memory and processor usage. Spectre operations are optimized for run time, so that user connections are closed when the operation completes, and resources are not devoted to idle user sessions. Resources are devoted to active users and concurrent dives.

Manageability

Users need rapid information access and IT needs to make sure they can manage and support user requirements. The Diver Platform with Spectre does both with Workbench for developers and DiveTab™ to keep your mobile workforce connected on the go.

Workbench

Workbench, an integrated development environment (IDE), helps developers manage the entire back-end process, from data source to portal. In version 7.0, Workbench is included with Diver, whether you opt for Spectre or not. With Workbench, developers can create and manage integrated, streamlined ETL processing from within a single application. The Workbench IDE combines a host of tools in a visual environment for efficient, centralized project management. Workbench integrates the following tools in a modular design with a familiar look and feel of individual components:

- Spectre: Define, build, and access cBases.
- DiveTab: Build data presentation portals for mobile users.
- DiveLine: Configure and control access at the server level in cBases and Models.
- Production: Configure and automate scheduled tasks.
- Visual Integrator: Define steps to extract, transform, and load raw data into cBases and Models.
- Visual Builder: Define and build Models.
- DiveMaster: Organize cBase and Model data by categorizing columns.
- DIAL: Prepare and schedule email delivery of reports for targeted users.

Spectre configuration and scripts use a single text-based scripting language, which developers access and edit with the robust Workbench editor. The scripting language is simpler and more powerful for builds and dives. Earlier script version calculations are supported under the covers and translated on the fly to take advantage of the Spectre engine. Workbench speeds development with highlights for important parts of the script, code suggestions, and descriptive help for syntax errors. Developers will appreciate how straightforward certain tasks can become in Spectre, such as setting up a new build script or how they can view a copy of the build script and log within every cBase.

DiveTab Client

Powered by Spectre, the DiveTab client is a tablet-based mobile technology for self-service reporting and analysis that drives data-driven decision making and information delivery using dashboards.

DiveTab uses the speed of Spectre's in-memory and columnar data management technologies for rapid and secure access to your data and other resources, such as presentations and documents, from one central location.

With DiveTab, you can connect to and synchronize (download) your data from up to 10 different server hosts for the most current information. During a work session, you can stay connected to a host or use the application in a disconnected mode with offline usage and local caching.

Not only is DiveTab available for the iPad, but Dimensional Insight also offers DiveTab for the PC with a similar look and feel to the iPad app. The DiveTab functionality is the same on both platforms for a smooth transition when on the road or in the office.

Migration

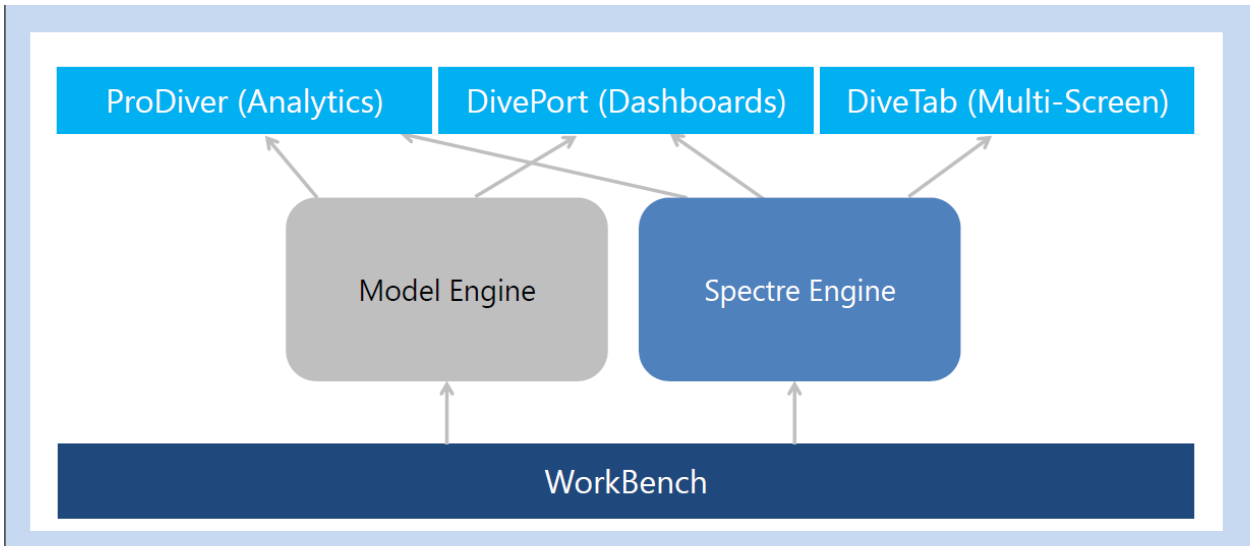
Migration of a large deployment might seem daunting, but Dimensional Insight supports both the Spectre and Model engines in the same installation, putting you in control of your migration path. For example, markers created on Models work with cBases by changing the target database. Spectre has a new calc language with slightly different syntax, but ProDiver converts the syntax to take immediate advantage of the efficiencies of the Spectre engine. Additionally, all Dimensional Insight clients are fully Spectre-enabled, and Spectre-enhanced back-end processes are fully supported in Workbench.

How do you take advantage of Spectre? As shown in [Figure 2, the Diver Platform 7.0 Architecture supports the Spectre and Model engines side-by-side, so you can start small. For example, to increase performance of a particular DivePort page, you can make the following changes:

1. In the integrator script, replace the supporting models with cBases
2. Recreate or edit the markers
3. Adjust the DivePort page to access the new objects

Using similar steps, portal pages, markers, and reports can be converted one at a time. New cBases developed with Spectre can take advantage of DiveTab™, a tablet-based mobile client.

[Figure 2: Diver Platform 7.0 architecture]



Migrate to Spectre using the approach that suits your environment and the resources:

- Create new Spectre cBases side-by-side with existing Models
- Replace Models with cBase

For new installations of Diver Platform 7.0 and Spectre, the consolidated installation package optionally includes required third-party software to handle installation of pre-requisites. Workbench gives you centralized project management to manage the entire back-end process from a single tool, and combines tools previously available as separate tools.

Summary

To recap, Dimensional Insight's Spectre engine is designed from the ground up to optimize powerful business intelligence. Whether your business is healthcare, supply chain, goods and services, or something else, Spectre engine delivers fast, scalable, and manageable business intelligence for enterprise customers.

- Speed
    - Fast build times keeps information current
    - Fast run-time performance for most frequently used functions boosts productivity
- Scalability
    - Columnar databases reduce build times and use less disk space
    - Refresh of cached dives and reuse across users with the same access keep results fresh and minimize per-user overhead
- Manageability
    - Spectre with the Workbench IDE simplify back-end processes with centralized project management
    - Capture the power of Spectre in DiveTab for business intelligence on-the-go