

Analytic Applications With PHP and a Columnar Database

No matter where you look these days, PHP continues to gain strong use both inside and outside of the enterprise. Although developers have many choices when it comes to building applications, PHP is so ubiquitous that it is difficult not to consider it for your analytic application. If you are not yet versed in PHP, you will find a large number of available resources including training, this website, books, forums, etc. to help you build world-class applications using PHP. Clearly the LAMP stack is here to stay.

PHP also serves as the foundation for many commercial and open source analytic applications. Analytic applications are applications that are used to analyze the performance of a wide variety of business operations and are typically associated with the use of business intelligence software. Analysis of both historical and real-time data is a big part of an analytic applications job. Functions of the analytic software also include interfacing to a variety of sources of data and most importantly, the company's data warehouse or data mart. An analytic application will also have a rich set of reports, scorecards and dashboards including the ability to "slice and dice" the data. Finally some analytic applications will also handle the what-if analysis and trend analysis.

Some of the most popular forms of analytic applications in use today include web analytics, mobile analytics and even financial analytics solutions. PHP is frequently used to build these types of analytical applications. For example, one very popular analytic solution in the market today is called Urchin, acquired by Google and now called Google Urchin. Urchin is written in PHP and is one of the most popular on-premise solutions for conducting website traffic analysis. Urchin is also the basis for the very popular Google Analytics, a hosted solution also using PHP as its foundation. Another rising star in the web analytics space is an open source project called PiWik. PiWik has grown in popularity primarily due to the void left by Urchin in the market once Google acquired the company. This open source project is written primarily in PHP and the source is available under GPL. One of the major differences between PiWik and Google Analytics is that you own the data with PiWik. In addition, PiWik is built as an on-premise solution, however it is still possible to put it in a hosted mode since it is based on the LAMP stack.

Infobright

403-47 Colborne St
Toronto, Ontario M5E 1P8 Canada
416.896.2483
www.infobright.com
www.infobright.org

Additional web analytic solutions that are open source and also written in full PHP or touch PHP include; AWstats, phpMyVisits and Open Web Analytics. Commercial vendors include Omniture, Unica, Birst, and PivotLink. Options in mobile analytics include Bango and Flurry. Not all the commercial vendors listed here use PHP but it seems that PHP is the popular choice for the open source vendors developing analytic applications.

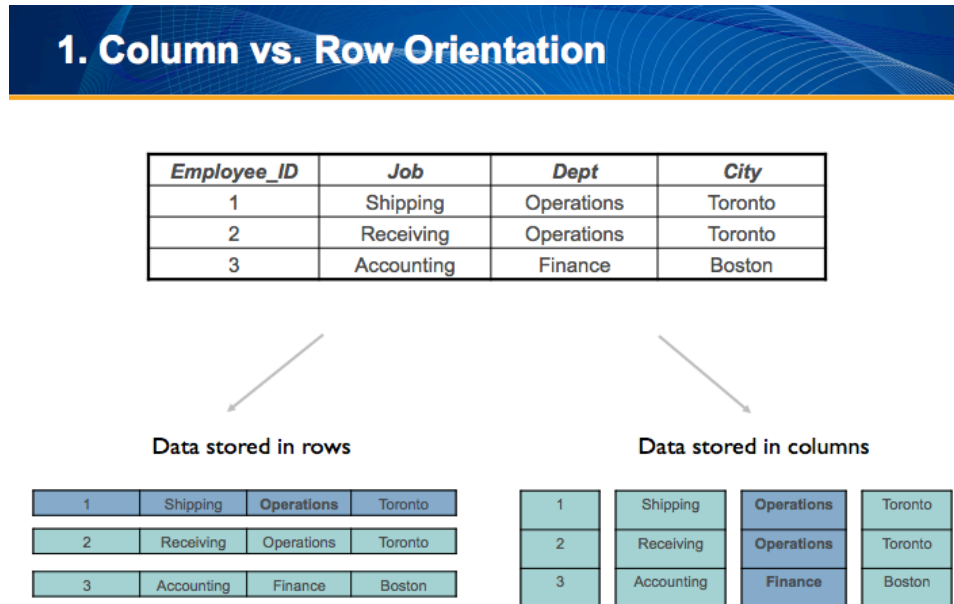
One of the most critical infrastructure components of a PHP based analytic application is the database. Seeing that the M of LAMP stands for MySQL, it goes without saying that MySQL has become pretty much the standard for the database behind open source analytic applications. This is certainly the case for PiWik. Building PHP based applications for MySQL is a fairly simple task and there is plenty of sample code out there on the Internet including the source code from the web analytics applications listed above under the open source category.

Although MySQL is seen as a popular choice for building PHP based general purpose web applications, over the past several years, a new category of database solution has arrived in the market that has proven to be a strong option for analytic solutions. This new database falls in the category of what is called a columnar-based database and one of its many unique capabilities is that it stores data in columns rather than rows as shown in Figure 1. This approach to storing data has resulted in significant performance improvements for query intensive analytic applications.

Infobright

403-47 Colborne St
Toronto, Ontario M5E 1P8 Canada
416.896.2483
www.infobright.com
www.infobright.org

Figure 1:



Starting with a generic table, there are two ways in which your database can store analytic data.

The first is by row, which means that all values from each record are stored as one entity. A single value from within the record, such as 'Department', can't be accessed without reading the entire row. This is the general way MySQL stores the data in the MyISAM storage engine.

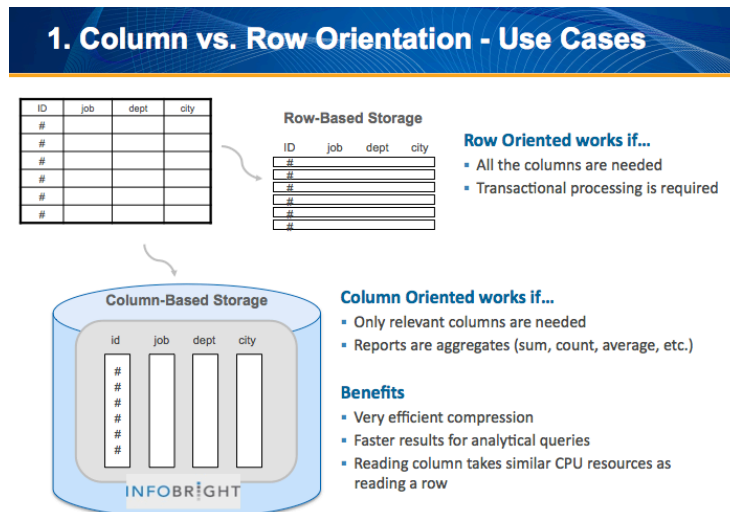
The second method, which a columnar database uses, is to store data by column so data from each column is stored together. This means that a particular attribute, such as 'Department', can be accessed without having to read the other attributes of this row. So now we can count the number of employees within each department without having to read and sort through the rest of the data in the database. Each method has its benefits depending on your use case.

Row oriented databases are better suited for transactional environments, such as a call center where a customer's entire record is required when their profile is retrieved.

Column oriented databases are better suited for analytics, where only portions of each record are required. By grouping the data together like this, the database only needs to retrieve columns that are relevant to the query, greatly reducing the overall I/O needed. By contrast, returning a specific 'record' would require retrieving information from each column store. See Figure 2.

Infobright is an example of an open source column oriented database built for high-speed reporting and analytical queries, especially against large volumes of data. Analytic queries typically ask questions about the data such as trends and aggregates, rather than questions that retrieve individual records from the data like MySQL. The good news is that Infobright's columnar database is built as a layer underneath MySQL - somewhat akin to a storage engine yet much more. This means that if you are comfortable building PHP based analytic applications using MySQL, but want the query performance and power of the columnar database, Infobright would be a good choice to consider due to its MySQL compatibility layer.

Figure 2:



Infobright

403-47 Colborne St
Toronto, Ontario M5E 1P8 Canada
416.896.2483
www.infobright.com
www.infobright.org

Some of the key database requirements for building read intensive PHP based analytic applications include the following:

- Many ad hoc queries
- Near real-time response
- Fast data load speeds
- Fast queries, especially against large data volumes
- Wide tables
- Aggregates: count, sum, etc.
- Deep compression to reduce storage and server hardware
- Rapid deployment and ease of use

Query types typically involved in analytic applications are:

- Analytic-intensive queries
- Use standard data types
- Limited number of joins

Example queries might be:

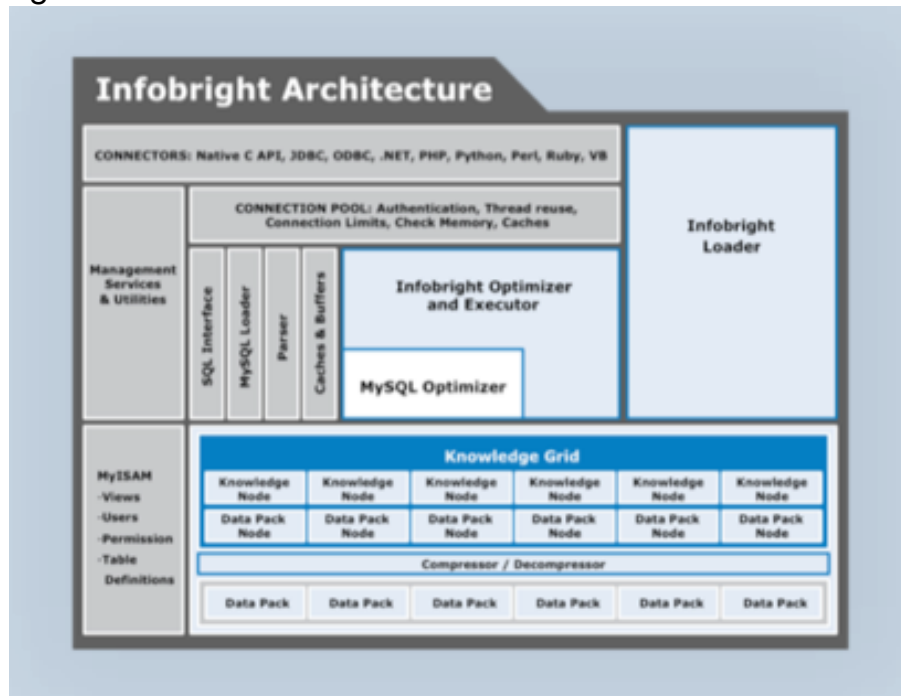
- Average clicks per visit
- Total number of visits
- Total visit time
- Total bounce rate
- Unique values
- Largest sales made
- Number of customers in a region

Columnar databases are best suited to meet these requirements. So let's investigate further how one columnar database, Infobright, works. (Figure 3)

Infobright

403-47 Colborne St
Toronto, Ontario M5E 1P8 Canada
416.896.2483
www.infobright.com
www.infobright.org

Figure 3:



With Infobright, as each column is read in from a source of data, it is divided vertically into groups of 65,536 row elements, referred to as Data Packs. See Figure 4. So each column of data is defined in terms of a stack of Data Packs. As each Data Pack is loaded into an Infobright table, Infobright extracts statistical information about this data, which is kept in this meta data layer called the Knowledge Grid. The goal of the Knowledge Grid is to minimize the need to access data in order to resolve a query – as it knows from its metadata which data packs are relevant to a particular query and which are not.

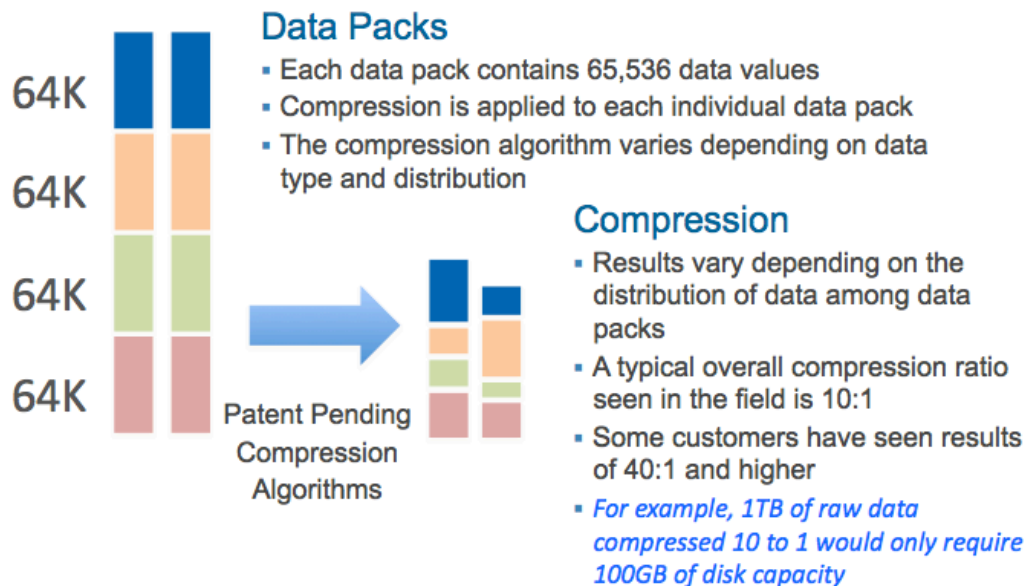
As Data Packs are created they are compressed individually using an iterative series of compression algorithms, giving average overall compression ratios of 10:1 to 40:1. Data is read in without having to alter a business's existing data model, and there are no indexes to build or maintain. The Knowledge Grid is built automatically during the load, and doesn't require any work on the part of the user. And because Knowledge Grid information is generated only relative to

Infobright

403-47 Colborne St
Toronto, Ontario M5E 1P8 Canada
416.896.2483
www.infobright.com
www.infobright.org

the data being loaded, the incremental load speeds are constant, regardless of the growing size of the database.

Figure 4:



The Knowledge Grid (see Figure 5) is a summary of statistical and aggregate information collected about each table as the data is loaded. It's information about the data. For each column and each Data Pack within that column, the Knowledge Grid information is collected automatically and up to 4 different types of Knowledge Nodes are built with no configuration or setup required in advance.

Three of the Knowledge Nodes, called Data Pack Nodes, Numerical Histograms, and Character Maps, are built for each Data Pack during the load; this eliminates the need for indexes

A 4th Knowledge Node, called a Pack-to-Pack Node, is built when a join query is run. This eliminates the need for keys.

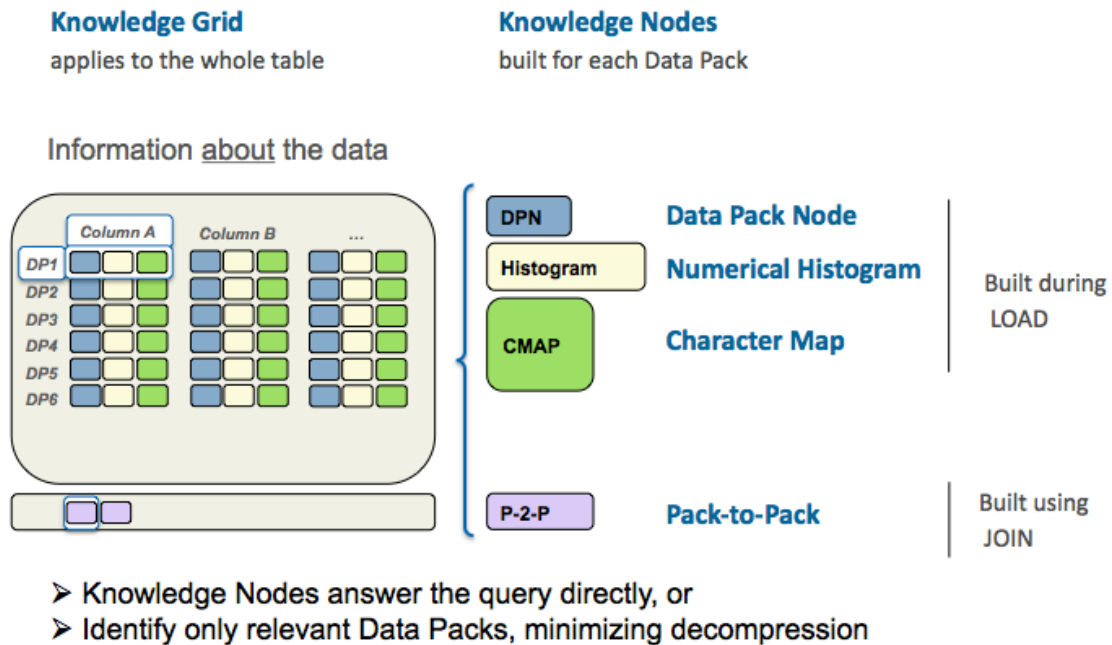
Because they contain summary information about the data within the table, the Knowledge Nodes are used as the first step in resolving queries quickly and

Infobright

403-47 Colborne St
Toronto, Ontario M5E 1P8 Canada
416.896.2483
www.infobright.com
www.infobright.org

efficiently by answering the query directly, or by identifying only relevant Data Packs within a table and minimizing decompression.

Figure 5:



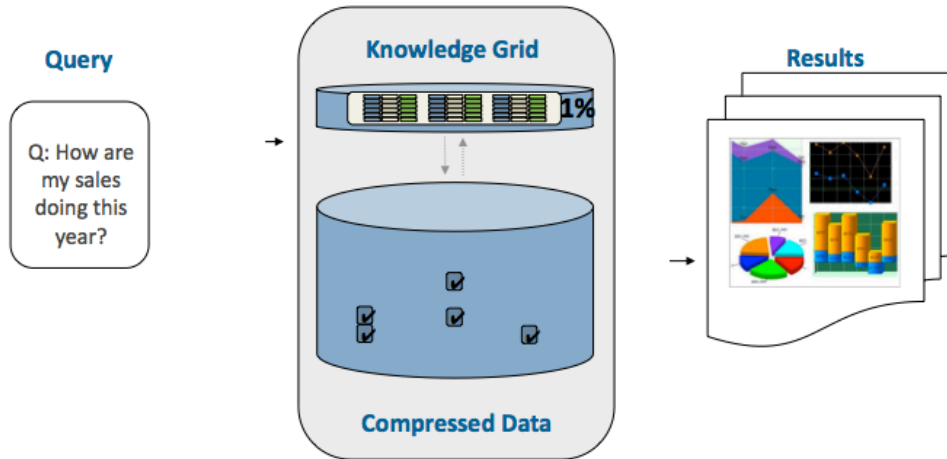
When a SQL query comes in from the PHP application, it's run through the Infobright optimizer, which looks at the Knowledge Grid first to resolve the query. (See Figure 6) Because the Knowledge Grid stores aggregate information about the data from the Data Packs, the query can often be answered using only the Knowledge Grid, without having to look at the data specifically. The Knowledge Grid also stores information about the range of values within each Data Pack, so in cases where more detail is required, the Knowledge Grid will narrow the field of search to only those Data Packs relevant to the query, and then only decompress and read these relevant Data Packs.

Infobright

403-47 Colborne St
Toronto, Ontario M5E 1P8 Canada
416.896.2483
www.infobright.com
www.infobright.org

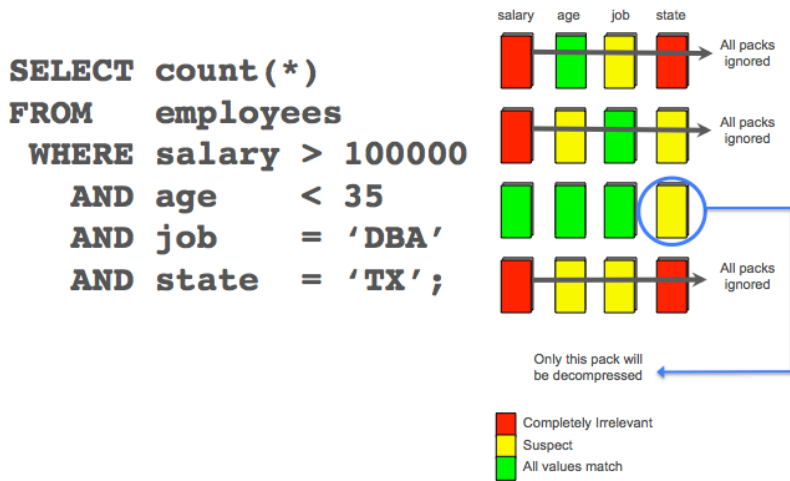
Figure 6:

1. Query received
2. Optimizer iterates on Knowledge Grid
3. Each pass eliminates Data Packs
4. If any Data Packs are needed to resolve query, only those are decompressed



Here in Figure 7, we show how the data is discovered and resolved

Figure 7:



So now that we have a basic orientation to the Infobright columnar database, we can show the basics of hooking up Infobright to PHP.

Once you have Infobright downloaded and installed, you are ready to establish a connection to the Infobright columnar database. It's simply a matter of executing the following PHP code as you would with MySQL. (See Figure 8). In the rudimentary example we will use root for the user and 'password' as the password. Use whatever you chose when you set up Infobright. We assume the database name is 'carsales'.

Figure 8:

```
<? Php
    $dbhost = 'localhost';
    $dbuser = 'root';
    $dbpass = 'password';

$conn = mysql_connect($dbhost, $dbuser, $dbpass) or die("Unable to Connect")
$dbname = 'carsales';
mysql_select_db($dbname) or die("Unable to select database");
$query = "SELECT dealerid, cartype, carname from Cars";
$result = mysql_query($query);
```

At this point, once you have the results set stored in \$result array you can build the PHP script to display the information by parsing out the information in the columns using the mysql_results command. As you can see, it is as easy as working with MySQL but with all the benefits of a columnar database.

Figure 9 shows how Infobright performs against other database systems for analytic applications.

Infobright

403-47 Colborne St
Toronto, Ontario M5E 1P8 Canada
416.896.2483
www.infobright.com
www.infobright.org

Figure 9:

Customer's Test	Alternative	Infobright
Analytic queries	2+ hours with MySQL	<10 seconds
1 Month Report (15MM events)	43 min with SQL Server	23 seconds
Oracle query set	10 seconds – 15 minutes	0.43 – 22 seconds
BI report	7 hours in Informix	17 seconds
Data load	11 hours in MySQL ISAM	11 minutes

Here are some results from a mobile analytic vendor who needed faster analytic performance when the volume of data increased:

Query	SQL Server	Infobright
1 Month Report (5MM events)	11 min	10 secs
1 Month Report (15MM events)	43 min	23 secs
Complex Filter (10MM events)	29 min	8 secs

Data Compression: Data that required 450GB of storage using SQL Server required only 10GB with Infobright, due to Infobright's massive compression and the elimination of all indexes.

Summary

Analytic applications are growing fast in the market and PHP is a great language for building these applications and for complementing apps written in PHP. The

growing types of critical information needed by businesses range from understanding and discovering customer usage patterns, customer churn, campaign management, website navigation and customer interactions. All of these sources of information likely land in a data warehouse. The ideal data warehouse for analytic application initiatives would be a columnar database because of the nature of the complex queries that professionals need to answer rapidly on a day-to-day basis. Any other solution would likely increase costs, create lag and friction in getting results to the business team and be less likely to see significant returns on investments, especially when open source columnar databases are available at much lower costs than their commercial counterparts.

To learn more about Infobright, visit our web site at www.infobright.com
To [download](#) Infobright Community Edition or join our open source community, go to infobright.org

About the Author: Bob Zurek is CTO and VP, Product Management for Infobright. Bob has over 25 years of proven success in software development, technology research, and product management, along with deep expertise in database management systems, business intelligence (BI) and open source technologies. Bob can be reached at bob.zurek@infobright.com

Infobright

403-47 Colborne St
Toronto, Ontario M5E 1P8 Canada
416.896.2483
www.infobright.com
www.infobright.org