



Database query performance is always toward the top of the list of priorities for administrators and users of data-driven applications. Tracking down a performance related query issue can be a time consuming and often frustrating process, especially when it involves only a small set of particular database queries. Since we at BASIS use all of our own products for our in-house accounting system, we often think of new ideas for improving the BBj® database in the areas of performance and data validation and as a result, we improved Table Analysis and the Query Analysis tool.

Unlike other commercial RDBMS databases that have an almost infinite number of constraints on what and how the data can be entered into the database, the BASIS database comes with a history of allowing developers to have full programmatic control of what and how they store their data. This flexibility means that BASIS has to offer extra functionality to help the developers shape or enhance their data and descriptions into a more SQL friendly format to achieve optimal access via both native and SQL

access respectively. Table and Query Analysis are two of the features that now give you more information about how to enhance your query performance with very little effort.

Table Analysis

Table Analysis performs an analysis of the tables in a database and gathers information about the tables' contents and structure that allows the SQL optimizer to more accurately determine how each query can be optimized when applied to a table. While the way the SQL engine uses this information has not changed, we made significant improvements in two areas: efficiency in gathering the information, and the addition of data and dictionary validation or "database alerts" during the process.

Improved Processing Speed

Version 12 of BBj and above, includes an improved process for gathering the table analysis information. With this change, customers see the time it takes to analyze large databases cut by several hours. For example, an extremely large database from our partner Audev (www.audev.com) took 11 hours to analyze in version 11.11 of BBj. In the improved version 12, the analysis took only 3 hours, a tremendous gain drastically reducing the load on the machine. This runs in the background and would generally only run once and would only require a re-analysis if there were significant changes to the structure such as the addition or deletion of indexes, or changes to the statistical makeup of a table like doubling the number of records.

Database Alerts


Since Table Analysis iterates over the table data inside the database to gather its statistical data, it made sense to add some data and dictionary validation to this process as well. Further, the Enterprise Manager now indicates the state of each table with an icon next to the table name in the table list. The four states are:

	Table analysis complete - no alerts.
	Informational alert - something BBj-specific about the definition of the table, i.e. how a particular index is limited in use for SQL optimization due to the type of data in the field.
	Warning - a possible problem depending on how the table is used, i.e. a variable length character column defined with one length but contains data longer than the defined length can cause problems when a third party application accesses it.
	Critical alert - a problem in the dictionary definition or the data in the table does not match the definition in the dictionary.



By Jeff Ash
Software Engineer

Figure 1 shows the newly improved Enterprise Manager interface which now displays the status of each table with one of the four icons, as well as an example of a warning alert message. Note that the alert also gives instructions on resolving the issue.

Not only does the Enterprise Manager display alert messages, but it can also automatically fix certain types of alerts. To see this in action, select the tables to auto-fix and click the  button at the bottom of the window. This displays a dialog with a list of the tables and alerts that Enterprise Manager may be able to fix automatically, depending on the underlying issues.

Query Analysis

Query Analysis displays information about the types of WHERE clauses used in queries executed on the database to aid administrators in improving indexing of their tables. It displays the columns included in the WHERE clause as well as whether those columns are indexed or partially indexed. The Query Analysis interface in the Enterprise Manager also has a new look, making it much easier to locate potential performance issues and then resolve those issues quickly and easily.



Note the warning icon  on the items in **Figure 2**. The Column List column shows the list of columns used in the WHERE clause. Since the Optimizable Index column is empty, this indicates there is no index that utilizes that list (or partial list) of columns. The Score indicates the likely number of records that the SQL engine would need to iterate over to return a single value matching the WHERE clause, while the Record Count column shows the total number of records in the file. The Pct. column indicates the percentage of the records in the file that would likely be examined in order to return a single value based on a query using the displayed columns.

Figure 3 shows two other types of icons. The green circle with check mark  indicates there is a primary or unique key/index on the combination of columns, which means it only needs to iterate over a single record to return a value for the query. The other icon, a small grey bullet, indicates that there is an index available for optimization and using the value in the Score column

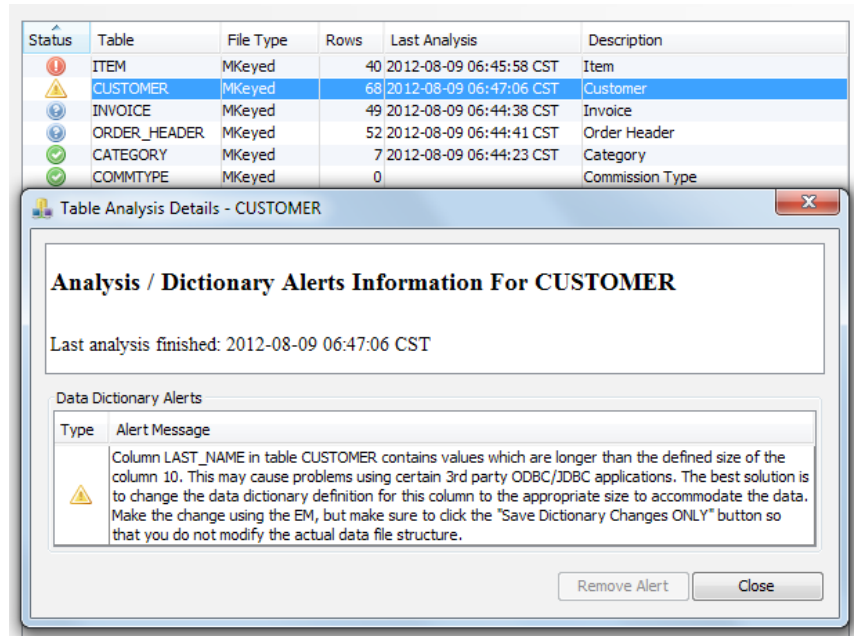


Figure 1. Example of the new Dictionary Alert








Status	Table	Column List	Count	Optimizable Index	Score	Rec. Count	Pct.
	ART13	LINE_CODE	8716		609319	609319	100.00%
	TRM03	END_USER_NBR	5890		10495	10495	100.00%
	ART73	TYPE_OF_SALE	5751		235904	235904	100.00%
	SNM01	PRODUCT_REV	1612		207960	207960	100.00%
	SNT01	PRODUCT_REV	1432		637980	637980	100.00%
	CANADIAN_EXCH	RECEIPT_DATE, AR_CHECK_NBR	1390		27	27	100.00%
	SNM1	ACTIVE_FLAG	1394		207960	207960	100.00%

Figure 2. Updated Query Analysis panel with alert status









Status	Table	Column List	Count	Optimizable Index	Score	Rec. Count	Pct.
	GLM_ACCT	FIRM_ID	14	PRIMARY (0)	24	141	17.02%
	GLM_ACCT	GL_ACCOUNT	15	AO_ACCT_F...	2	808	0.25%
	GLT_TRANSDetail	FIRM_ID, GL_ACCOUNT, TRNS...	134	INDEX0 (0)	1105	658821	0.17%
	ADC_COUNTRY	CNTRY_ID	30	COUNTRY (0)	1	11	9.09%
	ADC_LANGUAGE	LANGUAGE_ID	20	PRIMARY (0)	1	10	10.00%
	ADM_MODULES	ASC_COMP_ID, ASC_PROD_ID	9	COMP_PROD...	1	23	4.35%
	ADM_USER	USER_ID	2	PRIMARY (0)	1	23	4.35%
	ADM_SESSIONS	ED_TABLE_NAME, PER_COLUMN	4	PRIMARY (0)	1	55	1.82%

Figure 3. Updated Query Analysis panel with additional status types

in conjunction with the Record Count allows the administrator to see the optimization level of queries with this combination of columns.

Summary

Performance of an application is very important because it determines how efficiently users can enter and retrieve data, as well as contributing to the overall user experience. Sometimes BASIS can give customers improved performance by refactoring or enhancing our products. However, sometimes it requires the developer or administrator to make some changes to their database structure to gain improvements.

The Table Analysis performance improvements cuts down on the time necessary to analyze the tables while the improvements to the Query Analysis interface make it easier for administrators and developers to locate potential areas for improvement in SQL query performance by better indexing their tables based on user usage of those tables. If you are using Barista, our data dictionary driven RAD tool that makes heavy use of SQL, or you are running any third party SQL tools against your BASIS data, then dramatic performance improvement is just a click away! ■