Database Query Analysis

As computers have progressed in their power and speed, so too have the expectations of software users for instant access to their data. While application tuning and hardware improvements can help, it is often necessary to find ways to retrieve information from the data files or database used by the application in a more efficient manner. Most experienced BBx® developers understand that keys or indexes play a major role in gaining quick access to required data. However, they can have difficulty determining user requirements for indexing, especially when using SQL and third party data access or reporting applications.

Powerful New Analysis Tool

BBj® version 9.0 introduced a powerful new tool to aid the developer or database administrator in discovering the types of queries that users are performing on the database. This new feature, "Query Analysis" (not to be confused with "Table Analysis" that determines the uniqueness of index segments in a table), analyzes the combinations of columns used in the WHERE clause of all SELECT, UPDATE, and DELETE statements. The database engine stores this information in a new data dictionary table called DD_INDEX_USAGE. Each time an application executes a query, the database engine adds information about the columns included in the WHERE clause. When the application closes the connection, it writes the collected information to the DD_INDEX_USAGE table for permanent storage.

By Jeff Ash
Software Engineer

How Does This Help?

Using Enterprise Manager (EM), the developer or administrator can see all of this query usage data and make decisions regarding adding useful indexes on tables in the database. **Figure 1** shows an example of the information shown in the Query Analysis tab for a database.

The grid of information on the Query Analysis tab shows a list of query data consisting of a table name, combination of columns used in the query, score, and whether there is currently an index on that combination of columns.

Look at the first row. This shows that a WHERE clause containing both COST_ACCOUNT and PROD_CAT was used seven times by users. Note that it is positioned at the top of the list, which means it is quite common and that it is not indexed. Row 2 shows that COST_ACCOUNT was used seven times as well (every permutation of the columns in a WHERE clause will get its own row in the table) and that it is not indexed either. Analyzing this information, the developer might seriously consider adding an index on the COST_ACCOUNT column because of its common use in queries. While this example is very simple, it shows the power and possibilities for improving query performance for end users.

Real World Example

Here at BASIS, we used this new tool to improve our own intranet site and include such information as scheduled meetings and which employees are out on leave, vacation, or travel. This internal Web page generates its content dynamically based upon data stored in a BBj database. As more and more entries filled the "Out Today" and "Meetings" tables, there was a steady decline in performance. Using Query Analysis, we easily determined that the queries that the Web page ran were not using indexes in an optimal manner. Furthermore, we identified the necessary indexes and added them to the database using EM's built-in table definition capabilities, thereby greatly improving performance.

Summary

BBj 9.0 provides a powerful new feature to assist developers and administrators in further optimizing and fine tuning the performance of their databases. Using the Query Analysis tab in Enterprise Manager helps to quickly determine the index requirements and make the necessary changes to the database. Once again, BASIS provides another powerful tool to significantly improve the end user experience without changing a single line of BBj or other application code. ■

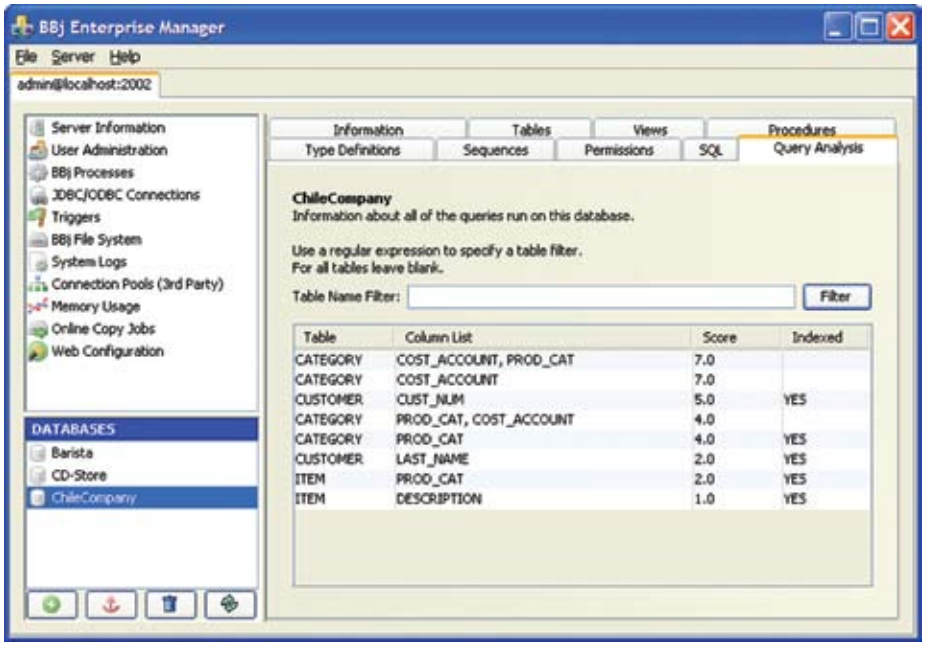

Figure 1. The Query Analysis panel in EM