XM4DB2

DYNAMIC AND STATIC SQL

Dynamic SQL

More and more dynamic SQL finds its way into applications. Dynamic SQL is much more flexible and makes it easier for programmers to build applications. On the other side of the coin however this flexibility makes it very hard to optimize the statements and DB2 system.

Almost every corporation has methods to tune static SQL statements. Lots of quality criteria is used to find bad statements and guarantee the performance and availability of the DB2 systems.

For dynamic SQL these methods and criteria do not work, because the statements do not exist at bind time. They are generated at run time and only exist a very short time.

XM4DB2 continuously reads the Dynamic Statement Cache (DSC) of the DB2 system and analyzes the dynamic SQL statements. It checks the statements against several rules and user defined thresholds to find statements making problems. The following list gives an overview of the problems being detected:

- DSC trashing – parameter markers not used
- Optimizer problems – estimation runstats
- Inefficient search – typical getpage causer
- Top consumer on user level
- Top CPU consumer within time frame
- Missing/inappropriate indexes, indexes used
- Top getpage causer
- Lock problems
- Utilization of tables by statements, user, packages

Static SQL

As mentioned above most corporations have their methods to keep static SQL statements performing well. The access paths are optimized and indices are created. But what if the access path tips at the next bind, for example if a PTF changes the behavior of the optimizer? XM4DB2 checks the access path after each bind and compares the current access path with older ones. If the performance decreases XM4DB2 alerts the DBAs.

Graphical Analyzer Interface

As an additional component XM4DB2 provides a graphical interface for the analysis of SQL statements. The GUI supports dynamic and static SQL. It can be used to obtain detailed information about the statements analyzed by XM4DB2. Diagrams showing the progress of the statements over time enable trends to be recognized. Another useful feature is the table based SQL analysis. With this the SQL statements fired against certain tables can be analyzed. Most corporations have hot spots on special tables, so with this function it is easy to find the SQL statements being fired against these tables.

The GUI also provides user defined reports. So for example you can create a report collecting all SQL statements which have been fired by user “FRANK” and which have more than 1000 IO per execution. The reports generate PDF files that can be stored for further use.