Enhanced Limits and ANSI AS Clause in ALLBASE/SQL

Kumaran N S

Hewlett-Packard India Software Operations Ltd.
29, Cunningham Road,
Bangalore, INDIA

Phone: 91-80-2251554 extn-1204
FAX: 91-80-2264107

nskumar@india.hp.com

Enhanced Limits

When the potential of a computing platform like MPE/iX increases with the advances in hardware technology and software enhancements, it is but natural that the applications become more demanding in terms of performance and scalability. The ability to meet these demands are, to a large extent, dependent on the back-end databases' ability to support a large number of user sessions and handle huge volumes of data. When applications push a database to its design limits, it is a clear indication that the limits are no longer sufficient to meet the ever-increasing demand for performance.

Therefore, it is not a surprise that some business applications supporting a large number of users from a single MPE/iX server have found the design limits of ALLBASE/SQL. It has been observed that database sessions run out of memory for Runtime Control Blocks (RCBs) in such environments, causing existing transactions to rollback or preventing new sessions from getting started. Hence, ALLBASE/SQL Limits Enhancement aims at increasing the memory space for RCBS.

With the increase in the number of pages for RCBs, more sessions will be able to connect to a DBE. Increased number of sessions would cause an increase in the number of concurrent transactions. The current limit of 240 transactions is a potential performance bottleneck when the number of sessions increases. Therefore, the limit on the number of concurrent transactions is also increased.

Another scenario where lack of free space for RCBs become evident is when a few sessions performing complex transactions use up all the memory space allocated for RCBs. When this happens, a new session trying to connect will not have space for RCBs. To overcome this situation, a facility has been added in the `ALTDBE` command of `SQLUTIL` to allow the DBA to pre-allocate RCBs for an estimated number of sessions based on the peak load. This will ensure that we can have as many sessions as predicted, though there is a limit on the maximum number of sessions for which the RCBs can be pre-allocated.

This enhancement has been carried out only for ALLBASE/SQL on MPE and not for ALLBASE/SQL on HP-UX.

New Limits in ALLBASE/SQL

1. The number of pages for Runtime Control Blocks (RCBs)

The maximum number of pages for RCBs that can be allocated has been increased from 2000 to 6000. The default value and the minimum value remain the same which is 37 and 17 pages respectively. This increase will enable ALLBASE/SQL to handle greater number of sessions. This parameter is referred to as `RUN BLOCK` = *ControlBlockPages* in the `START DBE NEW` command and the `START DBE` command. In the `ALTDBE` command of `SQLUTIL`, this parameter is referred to as "No. of Runtime Control Block Pages (opt):"

2. The Number of concurrent transactions

The maximum number of concurrent transactions has been increased from 240 to 750. The default value is retained at 2. This increase will improve the performance of ALLBASE/SQL on high end servers as it can support greater number of concurrent transactions. This parameter is referred to as `TRANSACTION` = *MaxTransactions* in the `START DBE NEW` command and the `START DBE` command. In the `ALTDBE` command of `SQLUTIL`, this parameter is referred to as "Max. Transactions (opt):"

The above two parameters which are a part of the startup parameters can be specified when a DBE is created and the values specified will be stored in the DBECon file and will be used every time the DBE is started. If the values for these parameters are not specified, the default value will be stored in the DBECon file.

When the `START DBE` command is specified, it establishes the values of these parameters for this and all subsequent connections until all connections to the DBEnvironment have been terminated. Hence, this command provides a way of altering these parameters temporarily. The value of any startup parameters not specified in this command is read from the DBECon file.

These two parameters can also be altered permanently, though not irreversible using the `ALTDBE` command of `SQLUTIL` and in this case, the new values are stored in the DBECon file over-writing the old values. The new value will be used in all the subsequent connections unless overridden.

Pre-allocation of RCBs based on peak load

To overcome the problem of most of the space for RCBs being used up by a few sessions and hence preventing new sessions from being started, a set of RCBs can be pre-allocated based on an estimated peak number of sessions. RCBs can be pre-allocated for sessions using the option "No. of Sessions for which CBs to be Pre-allocated (opt):" of the `ALTDBE` command of `SQLUTIL`. This will pre-allocate the required number of Session Blocks and Activation Blocks for the number of sessions specified. However, the number of sessions for which these blocks can be pre-allocated is limited by the “50% rule”. Upto 50% of the total space for RCBs can be used for pre-allocating these blocks.

Any file that is being used as input to the `ALTDBE` command of `SQLUTIL` has to be changed to add a new field for the new option included in this command.

SQLMIG & SQLMON

Since there are changes to internal structures of ALLBASE/SQL, this enhancement will be released as ALLBASE/SQL version H0.00. Therefore, SQLMIG has been enhanced to migrate databases between older versions and the new version. SQLMON has been enhanced to reflect the new limits in LOAD & OVERVIEW subsystems.

ANSI AS CLAUSE

The objective of the enhancement is to allow users to specify an alias name for the items in the SELECT statement using AS clause. Without the AS clause feature, when there is an expression in the select List of a SELECT statement, the result is returned with a heading "(EXPR)" and when there is a constant, the heading is "(CONST)," and so on. The presence of AS clause will allow the user to give a more meaningful name (an alias name) to the column headings. The alias name specified will be returned as the column heading in the query result.

There are 2 types of queries; dynamic & non-dynamic.

In case of dynamic queries, the column names are also returned in the query result. Hence, if an alias name is specified, it will be returned as the column name in the query result.

In case of non-dynamic queries, the column names are not returned. Hence, even if the alias name is specified it will not be returned. However, an error will NOT be flagged even if the alias name is specified in case of non-dynamic queries and the alias name will be ignored.

Usage of AS CLAUSE

If the alias name is specified as an identifier it should conform to the following rules which are the rules defined for "Basic Names" in ALLBASE/ SQL

- The name can be up to 20 characters in length.
- The name can be made of any combination of letters (a to z, A to Z), decimal digits (0 to 9), $, #, @, or underscore (_). The first character cannot be an underscore or a decimal digit.
- However, unlike in the case of Basic Names, lower case letters are not automatically changed to corresponding uppercase letters. Therefore, the alias name will be in the same case as specified by the user.

When the alias name is specified as a single-quoted string or as a double-quoted string, it can contain spaces and special characters in addition to the characters allowed in the "Basic Names" as defined above and the name can be up to 20 characters in length. The alias name cannot be only spaces and it should not be of zero length, i.e., it cannot be specified as '' or as "".

All queries processed through ISQL are dynamic queries and hence the alias name specified in the select statement will be returned as the column heading in the query result.

Dynamic queries can also be issued through application programs (For more information, please refer ALLBASE/SQL Application Programming Guide). Dynamic queries are handled in general as follows (though some specific details differ depending on the query type) :-

- Define a host variable (or a string) to hold the SELECT statement to be used by the PREPARE

command.

- The PREPARE command dynamically preprocesses the query.
- The DESCRIBE command makes available to your program information about each column in a query result. It is the describe statement that fills the format array with the information about the columns. The alias name will be returned in the sqlname field of the format array.
- The DECLARE CURSOR command maps the temporary section to a cursor so that other cursor manipulation commands can be used.
- The OPEN command allocates ALLBASE/SQL buffer space for holding qualifying rows and defines the active set.
- The FETCH command evaluates any predicates in the query and transfers rows from the ALLBASE/SQL buffer into host variables.
- The CLOSE command closes the cursor and frees any previously allocated buffer space.

The syntax of Select Statement with AS Clause is :

```
SELECT [ALL       ] SelectList  [INTO HostVariableSpecification]
       [DISTINCT]

FROM FromSpec [,...] [WHERE SearchCondition1]  [GROUP BY GroupColumnList]

[HAVING SearchCondition2]
```

where *SelectList* =

```
  {*                                              }
  { [Owner. ] Table.*                             }
  {CorrelationName.*                               } [,...]
  {Expression AS Alias_name                     }
  {[[Owner.]Table.]ColumnName AS Alias_name     }
  {CorrelationName.ColumnName AS Alias_name  }
```

where *Alias_name* =

```
  {identifier                }
  {single-quoted string      }
  {double-quoted string       }
```

Example (through ISQL) -

```
SELECT PartNumber AS "Part Number" , AVG(UnitPrice) AS avg_price,
AVG(deliverydays) AS 'avg days' FROM PurchDB.SupplyPrice GROUP BY
partnumber;
```

The above query gives the following result -

```
--------------+------------------+-----------
Part Number   | avg_price        |avg days
--------------+------------------+-----------
1123-P-01     |            495.83|         20
1133-P-01     |            198.75|         20
1143-P-01     |            180.00|         23
1153-P-01     |            210.00|         20
1223-MU-01    |             80.00|         20
1233-MU-01    |            295.00|         20
1243-MU-01    |            100.00|         16
1323-D-01     |            198.75|         20
-------------------------------------------
```

Constraints :

- The alias name cannot be used in other clauses of the query to refer to a column
- AS Clause cannot be used in sub-queries because the sub-query result is not returned to the user and hence the alias name doesn't have any significance there.
- AS Clause cannot be used in the SELECT statement that's part of a CREATE VIEW, Type 2 INSERT or GENPLAN because it doesn't have any significance in those situations.

Conclusion

The limit enhancements are targeted towards the scalability of ALLBASE/SQL on the new high performance HP e3000 platforms. The support for AS clause adds to the usability and opens up new possibilities for the developer community. These enhancements help users continue to derive maximum value of their investments in 3000 product line.

-----o-----