External Pervasive SQL (Btrieve) Access

1. Introduction

This document describes the guidelines that **MUST** be followed when writing 3rd party, external applications that update one, or more, Pervasive SQL (a.k.a. Btrieve) format Speedbase databases.

For the purposes of this document an "external application" is defined as one that is developed using a development system other than the Global (Speedbase) Development System. The database access is performed using a direct interface (ODBC for example) that by-passes the Speedbase Btrieve Gateway.

Disclaimer: Updating Speedbase databases from external applications must be performed with the utmost care. TIS Software accept no responsibility if database corruption occurs and a rebuild or restore becomes necessary.

1.1 General Recommendations

If it is required to write new records to a Pervasive SQL format Speedbase database, you are strongly advised to write new records to an “import” record type reserved for just that purpose; and arrange for a Speedbase application to transfer records from the “import” record type to the various “live” record types.

The Speedbase Dictionary utility ($SDM for “restricted” non-DBX databases; $DXM for “unrestricted” DBX databases) must be used to analyse the Speedbase database format to determine the Global, GVF and GVA fields (see below).

2. Locking

Pervasive SQL (Btrieve) supports Exclusive locking. All Global locks (whether exclusive or non-exclusive) are converted to an explicit lock on the Btrieve database. An attempt to read and lock an externally locked record will result in the normal "record Locked exception" in Global.

Therefore, the only locking issues affect read, write and delete operations.

The Speedbase Gateway currently ensures that all master records required to perform an update have a Btrieve lock applied before the update proceeds. However, if a lock has been placed on a master record by a third party product, then the Gateway cannot do the database I/O. It immediately signals an exception back to Global. The BA$DXX DLM simply retries the operation until the external lock disappears. This can lead to a dead-lock situation if the Global task has a lock on a record that the external task is waiting on. However, PROVIDED that the external task performs updates in the following manner, there will be no problems:

1. The external task must exclusive lock a record before attempting to rewrite or delete it;

2. Where an update affects several records, the external task must lock all records in the transaction before performing the first update. In the event a lock cannot be obtained for any one record, the task MUST relinquish all locks, and start again from the beginning.

Alternatively, the external task could use optimistic locking for its updates and (using Btrieve transaction processing) roll back the transaction in the event of an optimistic update failure.

3. Recommendations for Special Field Types

There are two classes of special field types in a Speedbase Btrieve database:

- Non-relational special fields;
- Special fields that are present in records that have relations;

3.1 Non-relational Special Fields

The fields described in this section must be considered when the record has no relationship with other record sets in the database.

3.1.1 Identity Fields

Identity Fields only appear in databases created with the PervasiveIncludeIdentity registry setting enabled (i.e. PervasiveIncludeIdentity=On). See section 2.19 of Technical Note IN274 for more details of this setting. The Identity Field is the first field in the record and is defined as a 4-Byte "AutoIncrement" Field called S_ID.

The Identity Field must be set to 0 (binary zero) when writing the record. Pervasive SQL (Btrieve) will automatically set it to a value of 1 greater than the largest occurrence of that field in the record set.

The Identity Field must **NOT** be changed on a re-write.

3.1.2 Status Fields

The two Status Fields are labeled S_*rt*ST1 and S_*rt*ST2. The first Status Field is either the first field in the record (if there is no Identity Field, see section 3.1.1) or the second field in the record (if there is an Identity Field, see section 3.1.1). The second Status Field is always the last field in the record.

The Status Fields are mainly a hangover from the native (Global format) database internal structure days and only appear in “restricted” non-DBX databases. Status Fields do not appear in DBX databases. The Status Fields were originally used by the $BASAV native database backup utility.

The first Status Field (S_*rt*ST1) should be set to 2 when writing the record. The second Status Field (S_*rt*ST2) should be set to 0 (binary-zero) when writing the record.

The first Status Field (S_*rt*ST1) should not be changed on a re-write. The second Status Field (S_*rt*ST2) should be changed to 2 on a re-write.

3.2 Relational Special Fields

When a record in a non-DBX database is not linked to any master or servant records, the only special processing required by the external program is to set the SGC (Sub Group Count) field to zero when writing the record. When a record in a DBX database is not linked to any master or servant records, no special processing is required by the external program.

The SGC Field is always present on a non-DBX Database. It is only present on Master Records in a DBX Database.

If the record that is being written, updated or deleted is linked to other records there are 3 issues to consider:

- Setting Uplink Fields (see section 3.2.1);
- Honouring the SGC Count (see section 3.2.2);
- Updating GVA values (see section 3.2.3).

3.2.1 Uplink Fields

Uplink fields do not exist in DBX databases. In pre-DBX databases Uplink Fields contain a link to each Master Record ID, in the order specified in the Dictionary. The Uplink Field is set to the unique ID of the related Master record (when the PervasiveIncludeIdentity option is enabled) or to a value which is effectively a hash of the Btrieve Record Position (when the PervasiveIncludeIdentity option is disabled). This hash value cannot be determined externally to the Speedbase Gateway so in order for external updates to be permissible the database must be created with the PervasiveIncludeIdentity option enabled.

When writing a record that has Masters, the updating task must set each Uplink field to the relevant master record's unique ID as identified by the Master Access key (i.e. the foreign key) present on the target record. The Uplink field order is critical and must match the order given by the current dictionary definition.

Uplink Fields are labeled S_*rt*LNK*nn* if they are present in the database. Uplink Fields only appear in “restricted” non-DBX databases and indicate that the record is a Servant type with Linked Masters.

You are strongly recommended **not** to attempt external writes on records that contain Link Fields. If you do write records that contain Link Fields you must know how these fields are calculated. This can be obtained by reading the Master Record by the Key in the Servant record and extracting the S_ID field if the Database was created with

"PervasiveIncludeIdentity" set to ON. Otherwise an algorithm that involves many factors, depending how the database was written would be needed and this is only provided in the Gateway.

When accessing an "unrestricted" DBX database you must analyse the dictionary to determine if the record type has Linked Masters as the Link Fields do not exist in DBX Databases.

If records containing Link Fields are re-written these fields should be left unchanged. Furthermore, you must ensure that no Global or GVF Fields in the record are changed otherwise you must also be prepared to consider the relational changes to the various Master Records.

3.2.2 SGC Fields

In addition to setting the Uplink fields on the target record (see section 3.2.1) the external task, when performing a write operation, must Increment the SGC of each master record pointed at. A Delete operation must decrement the SGC of the Master records. A rewrite operation must examine whether any Master Access Keys have changed, and if so alter the Uplink field to point to the new master record. This process requires the task to decrement SGC of the removed master record and increment the SGC of it's replacement.

The SGC count on each record indicates the number of servant records linked to that record. If the count is non-zero, the record must **not** be deleted. It is therefore a further requirement that an external task does NOT delete a data-record if the SGC contains any value other than zero.

SGC Fields are labeled S_*rt*SGC. They always exist in "restricted" non-DBX Databases but may not exist on all records in "unrestricted " DBX databases.

SGC Fields must be set to 0 (binary-zero) when writing a record.

SGC Fields must not be changed on a re-write. SGC Fields are only changed by relational updates by the Speedbase Gateway/NLM.

Important Note: A record that has a non-zero value in an SGC Field must **not** be deleted.

3.2.3 GVA Fields

GVA Fields don't have any special names and can't be differentiated from other "normal" fields just by examining the Pervasive description of the Record. The Speedbase dictionary must be analysed to determine the GVA fields.

A GVA Field can be either a floating point or integer field. GVA Fields occur at the end of a record immediately before the SGC Field (see section 3.2.2).

When writing a record that contains a GVA Field the GVA Fields must be set to 0 (binary zero).

GVA fields must **not** be changed during a re-write. GVA fields must only changed by relational updates by the Speedbase Gateway/NLM.

WARNING: GVA updates are extremely complex (especially with the advent of cascading GVF's in DBX databases). External updates to GVA field are NOT permitted. It is therefore not permissible to perform external updates to any table that contains GVF/GVA relations.

3.2.4 GVF Fields

GVF Fields, which can occur anywhere in the body of a record, don't have any special names and can't be differentiated from other "normal" fields just by examining the Pervasive description of the Record. The Speedbase dictionary must be analysed to determine the GVF fields.

A GVF Field can be either a floating point or integer field. In an "unrestricted" DBX database, if the cascading GVF technique has been used, a GVA Field can also be a GVF Field.

Do **NOT** attempt to write records that contain GVF Fields. If you do write a record that contains a GVF Field you will also have to apply all the relational updates in the other record types that are implied.

Do **NOT** attempt to modify GVF fields when re-writing a record. If you do re-write a record that contains a modified GVF Field you will also have to apply all the relational updates in the other record types that are implied.

3.3 Global Fields

Global Fields, which can occur anywhere in the body of a record, don't have any special names and can't be differentiated from other "normal" fields just by examining the Pervasive description of the Record. Unlike GVA and GVF fields, Global fields can be any field format.

Do **NOT** attempt to write Servant Records that contain Global Fields. If you do write a record that contains one, or more, Global Fields you must understand and apply the relational changes to other records.

Global fields must **NOT** be changed during a re-write.

4. Version Control

Version control is absolutely essential. If you convert a database (move to a new generation), you MUST ensure that your program is also updated. To ensure that you do not accidentally perform updates with a mismatching version, your program should at minimum contain the expected record length and check that against the record length returned by Btrieve. This technique is not foolproof, but it will catch most version problems.