Migration of a GSAM Data Base

This chapter covers the following topics:

- Introduction

- Restrictions

- Conversion of a GSAM Data Base

- Features of a Converted GSAM Data Base

Introduction

A GSAM database uses the Generalized Sequential Access Method (`GSAM`) under DL/I. It is recognized by the `'ACCESS=GSAM'` keyword for the DBD statement in the DBD definition. A `GSAM DBD` does not specify any segment or field definitions. The corresponding PCB statement in the PSB definition uses `'TYPE=GSAM'`.

For more information regarding the definition of a `GSAM DBD` refer to the IMS Utilities Reference Manual of IBM.

Restrictions

`GSAM` databases are supported by ADL with some restrictions regarding the DBD definition, the JCL and the `OPEN` command.

In particular, the following restrictions are valid for a `GSAM DBD`:

- The records must have fixed length, i.e. the `RECFM` keyword of the `DATASET` statement in the DBD definition must have a value of `'F'` or `'FB'`.

- The record length (`RECORD` keyword) must be at least 12 bytes.

If these conditions are met, the `GSAM DBD` can be converted and the data can be loaded into Adabas, as described in the next section.

Like for any other converted database, the data set corresponding to a converted GSAM database cannot longer be referenced in the JCL, because the data is in Adabas. Consequently, the record format (`RECFM`) parameter cannot be overwritten in the JCL. With DL/I the disposition (`DISP`) of the output dataset defines, whether a new dataset is created or whether the data is added to the end of the dataset. With ADL the data is always added to the end. If you want to insert data from the beginning of a data set, which does already contain data, you must refresh the Adabas file before your program starts. This can be performed by either using the `REFRESH` function of the Adabas `ADADBS` utility or manually with the Adabas Online System.

The DL/I `OPEN` and `CLSE` calls are ignored by ADL. Under DL/I you can add an option to the `OPEN` call, which specifies the kind of control character in the first byte of each record in the output data set. With ADL there is always no control character in the first byte of each output record.

With DL/I an application can use the `CLSE` and `OPEN` commands to read a `GSAM` database after it was loaded, without terminating in between. This works in the same way after the migration, because ADL does not need these calls for a re-positioning.

Conversion of a GSAM Data Base

Run the `GSAM DBD` through the step 1 (assembly), step 2 (CBC utility) and step 3 (create input decks) of the ADL control block conversion (CBC). For a more detailed description of the ADL CBC, see the section *ADL Conversion Utilities for DBDs and PSBs* in this documentation. Step 4 (assemble and link-edit Adabas user 6 extension) is not required for a `GSAM` database.

ADL adds a segment 'GSAMROOT' and a sequence field 'GSAMFLD' to the DBD definition. The length of the segment is the same as specified with the 'RECORD' keyword in the `DATASET` statement. The field always has a length of 12 bytes. Additionally, ADL adds a sensitive segment 'GSAMROOT' to the `GSAM PCB`.

The `ADACMP` cards which are generated by step 3 of the ADL conversion utility is used as input for the `ADACMP COMPRESS` utility. Note, that the `USERISN=YES` option must not be used for any Adabas utility run against a file of a converted `GSAM DBD`.

There is no need to run the ADL unload utility when unloading the DL/I `GSAM` database. Instead, the sequential file can directly be used as input for the Adabas compression utility, by specifying the `GSAM` dataset as `DDEBAND` for `ADACMP`. You must not use the ADL user exit 06 for the compression.

Use the following parameter settings for `ADALOD` when loading the data into Adabas:

```
DATAPFAC=1
DSREUSE=NO
ISNREUSE=NO
USERISN=NO
```

Features of a Converted GSAM Data Base

The `GN`, `ISRT` and `GU` calls are supported in the same way as under DL/I. GN and ISRT can be specified with or without a record search argument (`RSA`). With ADL the first 4 bytes of the `RSA` contain the Adabas ISN of the record and the last 4 bytes contain hexadecimal zeros. After a successful call, the `RSA` is also placed into the Key Feedback Area of the corresponding PCB. Further to these database calls, the system service calls `CHKP` and `XRST` are supported.

With the `LOAD` parameter of ADL, you can specify where the data of an `ISRT` call against a PCB with `PROCOPT=L` or `LS` is to be written. See the section *ADL Parameter Module* in the *ADL Installations* documentation for more information about the `LOAD` parameter. When you specify 'LOAD=UTILITY' the data is written to a variable blocked sequential file. For a `GSAM` database, this file can be loaded back into Adabas in the same way as described above. Especially you must set `USERISN=NO`, and you must not use the ADL user exit 6.

When 'LOAD=DIRECT' is specified as ADL parameter, you can insert and read data simultaneously in one application by using different PCBs in the PSB. In particular you do not need to issue a `CLSE` and an `OPEN` call, when you want to retrieve data after inserting it. You can also mix the `ISRT` and `GET` calls as you like, and the position is kept for each PCB.

Further to inserting and retrieving data, you can also replace and delete records, by using the `REPL` (replace) and `DLET` (delete) calls. The corresponding PCB must specify `PROCOPT=R` or `=D`, respectively. The data storage of a deleted record is not released, so that a new inserted record is put to the end of the database, as expected for `GSAM` databases.

Note:
An Adabas backout after a delete request acts like a re-insert, i. e. the record is inserted to the end of the database. After such a delete/backout the data will be therefore in another sequence than before.

After the data conversion you can manipulate it with Natural. This includes update and delete requests. Because there is no ADL internal pointer field stored with the data, there is no need to use the ADL Consistency.

ADL translates a get request against a `GSAM DBD` to an Adabas `L2` (read physical) call. For a high performance it is recommended to use the Adabas Multifetch feature for a `GN` sequence against a converted `GSAM` database. Refer to *Using the Adabas Multifetch Feature*, section *Performance Considerations* in the *ADL Interfaces* documentation for more information about the usage of the Adabas Multifetch feature against migrated data. In particular, you should make sure, that the correct Adabas command ID is prefetched.