3.2.3 EDM Pool

3.2.3.1 Introduction

The EDM Pool contains:

1. DBDs
2. Skeleton package and cursor tables
3. Authorization cache for plans
4. Skeletons of dynamic SQL if CACHEDYN=YES

There are 2 type of SQL cache:

- Global cache stores a copy of the prepared statement for use by everyone.
- Local cache is used only when KEEPDYNAMIC=YES and contains a copy of the prepared statement for the particular user, so every thread has a local cache. Needless to say, that can add up to a considerable amount of storage in DBM1, so the MAXKEEPD Zparm is used to limit the number of statements kept in local cache.

A prepare is the action of determining the appropriate access path for a SQL statement by examining different options and evaluating their cost using statistics from the DB2 catalog. The more complex the SQL statement is, the longer this takes.

An **Explicit prepare** occurs when the application issues a SQL PREPARE or EXECUTE IMMEDIATE.
An **Implicit prepare** occurs when the user copy of the prepared SQL statement no longer resides in the local cache and DB2 issues a PREPARE.
Both explicit prepares and implicit prepares can result in either a Full Prepare or a Short Prepare.
A **Full Prepare** occurs when the copy of the prepared SQL statement is not found in the Global SQL cache in the EDM pool.
A **Short Prepare** occurs when it is found in Global SQL cache, and is then copied to local cache.

Prepare avoidance occurs when the statement is found in local cache, and neither a Full Prepare nor a Short Prepare has to be done. A Short Prepare costs 1% or more over prepare avoidance. A Short Prepare (10K-20K instructions) is considerable less expensive that a Full Prepare (70K-100K instructions).

For large EDM pools, space utilization and fragmentation is less of an issue than for small buffer pools. So DB2 emphasizes performance at the expense of memory management for pools larger than 40MB. If you want to optimize storage allocation at the cost of performance in pools greater than 40MB, you can set EDMBFIT to YES

As SQL statements are executed, the control blocks (DBD sections etc) needed to process the SQL statements get loaded into the EDM pool (from DSNDB01 database via the buffer pool). These structures usually get released at de-allocate. Long running threads can hold on to a large amount of EDM pool storage, which can be released periodically (they're examined at commit time) using the CONTRACT THREAD STORAGE Zparm (CONTSTOR). Usually this should be set to NO for performance, but if you are running into DBM1 storage constraints, and considering that the PSE Application Server can be holding these threads connected for days or weeks, this Zparm can help to release storage. (MINSTOR may also help in this area too)

3.2.3.2 DB2 Version 7

In V7, if global dynamic statement caching is active (Zparm CACHEDYN=YES), the statements can be cached in a data space

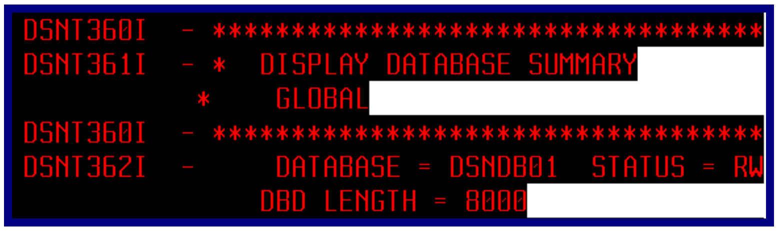
© Database Consulting Services 2006 - All rights reserved

(EDMDSPAC>0), or in the normal EDM pool if a data space is not defined

If you specify YES for CACHEDYN, the Zparm Clist generator creates a value for EDMDSPAC. If the EDM pool storage size (EDMPOOL) is <= 40MB, it will generate an EDM data space of 40MB. If it's larger than 40MB, it generates a size equal to the EDMPOOL size. To prevent use of a data space, you will have to change the EDMDSPAC Zparm to 0. Many customers are using data spaces for dynamic SQL cache without realizing it. That's not necessarily a bad thing, since it's purpose is to relieve DBM1 from additional storage requirements that would push it closer to it's 2GB virtual memory limit, but from a performance perspective, it doesn't buy you much unless you are on a z900 series box along with OS/390 2.10 or z/OS 1.1 64-bit real storage support. With this configuration, performance of data spaces are improved, but their main advantage is in using them in place of hiperpools for buffer pool storage. You can read more about this in the DB2 V7 Admin Guide Ch 27 "Choosing Backing storage: primary or data space".

There is also an EDMDSMAX Zparm that specifies the maximum size of the data space that can be used by the EDM Pool. This is to prevent you from using dynamic Zparm modification (SET SYSPARM) to bump up the EDMDSPAC Zparm too much.

Loading DBDs into the EDM pool used to cause problems when PeopleSoft delivered a minimal number of physical databases, since a DBD could not (and can still not) occupy more than 25% of the EDM pool (and prior to DB2 V6, required contiguous storage). You can find out the size of your DBDs with the –DIS DB (*) ONLY command.



```
DSNT360I  - ***********************************
DSNT361I  - *  DISPLAY DATABASE SUMMARY
            *    GLOBAL
DSNT360I  - ***********************************
DSNT362I  -      DATABASE = DSNDB01  STATUS = RW
               DBD LENGTH = 8000
```

Delivering more databases pretty much eliminated that problem until the AE8 "multi-instanced temp table" generation delivered hundreds or thousands of additional tables and made some of the DBDs very large again. I have heard of customers running into the 25% rule again. Of course, large DBDs have drawbacks for several other reasons too:

- The entire DBD is logged at each commit.
- When objects in a DBD are dropped, storage is not automatically reclaimed. You may have to use REORG and MODIFY to reclaim space of OBIDs representing dropped objects, in the DBD
- The database descriptor table space DBD01 is locked when it needs to be loaded into the EDM pool, causing potential conflicts. Note that if the DBD is already in the EDM pool, dynamic DML statements (select, delete, insert, update) obtain S locks on a DBD while DDL (alter create drop) acquire X locks.

When pages in the EDM pool are required for loading objects, they are first obtained from free pages, then "stolen" from inactive SKCT, SKPT, DBD or dynamic SQL skeletons. If there is still not enough room, you get an SQL error.

If your EDM pool is too small you can incur increased I/O activity to DSNDBD01, where DBDs, SKCT and SKCT are stored, and increased response time due to having to load the above, and re-prepare SQL statements. If your subsystem is PeopleSoft only, then loading SKCT and SKPTs are minimal since there are only a handful of plans/packages needed.

3.2.3.3 DB2 Version 8

In V8 the EDM pool is always split into three parts:

- Storage for the global dynamic statement cache. Cached, dynamic SQL statements are always cached in the dynamic statement cache pool above the 2 GB bar (Zparm EDMSTMTC). This value is used at DB2 startup time as the minimum

© Database Consulting Services 2006 - All rights reserved

value. This value cannot be decreased below the value that is specified at DB2 startup. This is because the CACHEDYN Zparm is online changeable in V8, and DB2 needs an initial size to be allocated at startup time to allow the size to be changed online. The maximum value is 1 GB. SQL statements cached in the EDM Statement Pool Cache may be much larger in V8 than SQL statements cached in V7. This is primarily due to DB2 V8 support for larger names and Unicode

- Storage for DBDs. Almost all of the storage related to managing DBDs is allocated above the 2 GB bar. This gives the DBDs the needed space to grow and relieves contention with other objects in the EDM pool. The size of the DBD cache is governed by the EDMDBDC Zparm, with a size between 5 MB and 2 GB. This value is used at DB2 startup time as the minimum value and also cannot be decreased below the value that is specified at DB2 startup
  DBDs are also larger in DB2 V8 in new-function mode. There is little overhead in below the bar storage to support this new DBD pool above the bar. So, make sure you adequately size the EDMDBDC parameter in order to avoid continually loading DBDs from disk.
- Storage for plans and packages (SKCT, CT, SKPT, and PTs). Plans and packages (SKCT, CT, SKPT, and PT) and the authorization cache are the only control blocks that are stored in the EDM pool that remains below the bar. The Zparm is EDMPOOL as before. As other DB2 structures have moved above the bar in the DBM1 address space, more space may become available below the bar to accommodate more plans and packages in the EDM pool. PeopleSoft uses a minimal amount here, since there are only a few plans and packages

There is one Local Dynamic Statement Cache pool allocated for each thread in DB2 V7. This has changed in V8. Now the Local Dynamic Statement Cache is allocated in an array of 30 global pools. This change should reduce the storage fragmentation below the bar, and it also means that if you need to reduce storage below the bar, it may be possible to set MAXKEEPD to 0 so that no dynamic SQL statements are retained in this pool after commit, at the cost of having to do a short prepare for every statement

The installation Clist calculates values for all three parameters based on the input provided on the installation panels. The installation is aware of the fact that instead of a single storage structure (or potentially 2 storage areas if using statement caching in data spaces), V8 uses three different storage areas, governed by three Zparm.

If you use the current V7 size of the EDMPOOL Zparm in V8, you may be over allocated, as the DBDs (and potentially also the dynamic statement cache) no longer reside in that storage area in V8. However, the requirement for Skeleton cursor and package tables has increased, although PeopleSoft does not need these with its dynamic SQL. Therefore, it is important to provide accurate values as input to the installation Clist. This will allow the Clist to calculate accurate values for the EDMPOOL, EDMDBDC, and EDMSTMTC.

EDMDSPAC and EDMDSMAX no longer exist since the EDM pool cannot be put in a Data Space. CONTSTOR and MINSTOR still exist but probably won't be needed. EDMBFIT still exists.

3.2.3.4 Recommendations

Make sure the EDM pool is large enough to handle the PeopleSoft DBDs. Any DBD that is more than 25% the size of the EDM pool will fail to load.
Depending on the applications you are installing, the DBD cache should have a minimum size of 10 to 30 MB, the SQL cache should be in the 50-100MB region, and the plan cache can be kept at a minimum

© Database Consulting Services 2006 - All rights reserved