



Brief Description of XM4DB2 Exceptions

1. Problems related to DB2 Objects

Restricted States: XM4DB2 recognizes restricted states on tablespaces and indexspaces. Utility executions cause spaces to be in certain restricted states. Therefore XM4DB2 further checks whether utilities are running on these spaces. XM4DB2 reports restricted states together with the involved utility and especially those caused by stopped utilities. Whenever an exception is detected, XM4DB2 will call skeletons, which can be used for automatic problem resolution, to:

- Terminate Reorgs that failed in unload Phase P cleanup of datasets, term util
- Run automatic Reorgs in case of AREO status
- Take Imagecopies when Copy Pending is detected
- Execute DB2 a Command if LPL, GRECP occur
- Start Rebuild in case of REBP

XM4DB2 detects any restricted state and can prepare a skeleton for each purpose. XM4DB2 can execute the job automatically, or write it into a library for further inspection by a DBA, or can pass it to another component or scheduler.

Conceivable Lack of Space: Many physical limits restrict the maximum size of tablespaces and indexspaces. DBAs are familiar with the problem of reaching a size limit. Substantial efforts are necessary for repartitioning or even reorg/re-balance a tablespace and might cause the DBA to work over night. A filled catalog table-space is an extra ugly issue. XM4DB2 prevents these things. The user specifies the maximum fill level and the number of allowed extents. XM4DB2 raises an exception as soon a threshold is exceeded. To do so it considers every limitation, be it VSAM dependent or DB2 dependent.

XM4DB2 can automatically invoke settlements, for example, submit reorg job.

The same is possible with over-allocated objects, space can be released.



2. Recovery

Recoverability/SLA Problems: XM4DB2 checks for recoverability, that is, the existence of valid imagecopies and associated log datasets as well as the nonexistence of LOG(NO) events. In case recoverability is not given, XM4DB2 can use job skeletons to take imagecopies or to invoke other components. XM4DB2 calculates the time to restore and compares it with the set point of Service Level Agreement. For example, the service level agreement might require that the tablespaces are to be recovered within 4 hours. XM4DB2 raises an exception as soon as this limit has exceeded. XM4DB2 checks for each object

- Time of last imagecopy
- Number of updates on the tablespace
- Number of Log/Archive Datasets to scan
- Location of required Archive Datasets (DASD, CART)
- Number of indexes to be rebuild

Based on this information the restore time for each object is calculated and compared with the SLA. If the SLA is not satisfied, skeletons are used to generate jobs to take copies.

BSDS Log Problems: DB2 stores a list of all log datasets in its BSDS datasets. XM4DB2 checks if this list has any gaps or if the datasets mentioned in this list are not available.

Log Offload Problems: Log data is a critical resource. XM4DB2 recognizes whether there are problems with this process, repetitive delays raise an exception. XM4DB2's history function displays how often the problem occurred in the past.

3. Performance Related Issues

Quality Assurance & Performance with Dynamic SQL

The traditional QS for SQL with Bind/Explain is more and more frequently by passed by dynamic SQL. These statements are generated right before they are executed, inappropriate and inefficient statements can flood the DSC and/or impair overall performance. The component DSC Analyzer (DSCA) of XM identifies problem SQL. DSCA audits the statement cache, checks if predefined quality criteria are exceeded and notifies DBA staff about evolving problems and performance degradation.



XM4DB2 deploys a unique technique to detect and display inefficient statements, beginning with the most inefficient: Statements without parameter markers that flood the DSC, optimizer issues (run time is much higher than explain time, normally dated RUNSTATS), bad erow/prow ratio (missing index), statements that cause too long lock waits, top consumers, etc.

XM's advantage over standard monitors is that it goes well beyond simply reporting the raw data, thus making it a valuable complement. It provides operational advice to resolve the issue at hand.

Database professionals realize pure or raw monitor data needs to be analyzed, understood, interpreted and used to render an effective solution. This is where XM really provides immediate value to the organization. XM's ongoing analysis delivers ready to use briefings and analyzed information:

- assessments like, 'obsolete Runstats', 'Index missing', 'DSC flushing because of usage of statements without parameter markers', 'too many lock waits', 'inefficient search', 'bad ratio of examined/processed', etc.
- builds unique hash keys to identify statements and statement groups to indicate when a problem initially occurred, how it developed, its behavioral trends and impacts.
- filters for runaways, noxious developments and dangerous behavior. For example, XM4DB2 rates the ratio between CLASST and Lock Wait to uncover those statements that are inefficient or wasting resources.

When issues occur, the usual questions is, "What happened, and why?" XM makes it easy for database professionals to see if this week's SQL is running worse than the week before and for what reasons. While XM's DSCA is able to deliver the same details that traditional monitors offer, its main focus is to provide results, assessments and operational briefings rather than just data. It filters out inefficient statements using a quality criteria expert engine. The values used in these criteria are controlled by the user. DSCA maintains a database and recognizes developments and trends with the help of data collected over time. The tool identifies the top consumers (programs, users, statements, statement groups) of CPU, IO, Wait, etc. XM's DSCA is designed for operations. It detects self-acting troublemakers and wasters, helps to maintain performance and availability. XM4DB2 and the DSCA, in particular, have a powerful notification feature to ensure that exceptions are detected and resolved in a timely manner. XM4DB2 raises exceptions if

- DSC Trashing occurs: Too many statements without parameter markers flood the Dynamic Statement Cache
- Optimizer problems are detected: Statement Runtime is for example 500% of the time estimated by the optimizer, typically wrong or missing RUNSTATS



- Access Path problems are detected: a huge volume of rows within the index and the tablespace are touched, but only a few rows are returned to the application. Typical for this are wrong or missing indexes or again runstats
- Top CPU Consumers are detected
- Top Getpage Requester are detected, mostly identical with Top CPU Consumers
- Statements with Lock-Problems are detected
- Statements with IO Problems are detected
- Top Consumer (CPU/Getpage/IO) on user-level are detected

The user can, as in general with XM4DB's exceptions, set/modify the quality thresholds of the exceptions to be raised.

4. Bufferpool Problems

In respect of system performance bufferpools are a crucial piece of DB2. There are many situations around bufferpools which can cause dramatic degradation of application response time or batch performance. Bufferpool related exceptions are

Re Read IO Rate < 10%: Re Read Rate (%) - this rate informs staff about the percentage of avoidable IO's, because it counts IO's which are done on page numbers, where an IO occurred a few seconds before. This type of IO is the best tuning opportunity overall. XM4DB2 and BPA4DB2 are the only products on the market, which have such a criteria available. Only ReRead IO is a reliable indicator for avoidable IO, and an indicator for tuning opportunities.

SPTH Hits < 1% of all Prefetch requests: Sequential Prefetch is a powerful feature within DB2. It can move up to 120 GB / Sec of data (DB2 V9) from the hard disc to the bufferpools. It is used mainly in sequential scans, but big joins of course benefit from this technique too (especially for new workloads like JAVA or modern workstation tools it becomes more and more important). Hits on the SPTH show us critical space problems in the bufferpool.

DMTH Hits = 0: This threshold is very critical. It is an indicator for heavy space problems in the bufferpool. If this threshold is hit DB2 changes the way it processes pages. Because of the space problem DB2 tries to get rid of the pages as soon as possible. Imagine, a cursor is going through a page - up to 255 rows can be located on one page - problems with this threshold leads to a Release Page immediately after the first row was fetched.



Then the next row is accessed, a new Getpage is required, the row is fetched and the Page Release follows immediately. In such a situation, we see a significant increase of CPU consumption and the application performance degrades dramatically.

Cross Invalidation because of Directory Reclaims < 10% of all reclaims: This indicator is related to data sharing environments. The Groupbufferpool is split into Directory and Data (data is similar to the data pages we have in local bufferpools) If we face space problems with the directory, we generate more (Re-Read IO's). This happens, because when inter system read write interest exists, reading DB2 members have to register their pages in the directory of the group bufferpools. This is necessary, because if an update of a data page occurs, DB2 has to know, which member has a copy of this page. This information is kept in the directory. If the directory has insufficient space, readers can do no more registration and pages have to be reclaimed. If this happens valid pages in local bufferpools are invalidated, because they loose their connection to the Groupbufferpool. This might cause reread IO from Dasd when the member does the next getpage on this Page number.

Write failed in Groupbufferpool = 0: Write failed is a very critical counter in a data sharing environment. Everything > 0 indicates problems with the size of the data part of the group buffer pool (it is too small). Writes in the GBP are typically done at commit. If the space in the directory is not sufficient, write failed conditions occur. This is not a performance problem, but it might become a problem with availability P this is why the counter is critical.

Page IN for Read IO < 1% of all Getpage: This counter is very helpful to determine, if the z/OS memory is over allocated. If we see more than 1 % of all getpage ending in a Page in for read IO, the LPAR does not have enough real memory for the workload. DB2 is one of the top memory consumers on z/OS operating system. It might be, DB2's bufferpools are over allocated.

SCA Fill level < 10%: This is a data sharing problem again. The SCA is the security net of the group. Ensure there isn't more than 10% of the SCA allocated during normal processing. If so, you might not have enough memory available in critical situations. In a worst case scenario, you face a unavailability of the complete data sharing group. Ensure the INITSIZE of the SCA is 10 -20 % below the MAXSIZE of the SCA- structure size. If not, you risk dramatic problems in the case of a group restart after a crash.

GBPT: How often the group bufferpool threshold was hit. If too often, CLASST is probably set too low.



NOWRIENG (no write engine): The write thresholds are too high. DB2 has to write too much data in one fell swoop.

WRITE_SNAP: Fill level of the GBP with changed pages. XM4DB2 does checks against all these thresholds and counters. If hits are encountered, Exceptions are generated.

5. Thread Problems

Threads are connections from applications to DB2. XM4DB2 recognizes in-doubt and postponed threads and threads which don't commit over a long time. Especially long running threads can cause problems: End Users are generating reports with their fancy tools and start queries that might run for hours. XM4DB2 detects such threads and can be used for an automatic cancel or other appropriate actions. XM4DB2's skeleton allows for easy adaptation.

6. DDF Issues

DB2 uses DDF to communicate in a data sharing group. DDF can have a certain number of parallel accesses. XM4DB2 recognizes if DDF is not available, or the number of queued requests exceeds the threshold.

7. Stored Procedures

Stored Procedures are code routines stored in DB2 that can be called by applications. DB2 can run a certain number of procedures in parallel. The remaining requests are queued. XM4DB2 can detect if the number of queued requests exceeds a threshold or if any procedure fails.

8. Plans – Packages

Invalid/Inoperative Plans: Plans are used to describe applications and their programs from a DB2 view. XM4DB2 finds plans that are no longer valid or that are inoperative. It also finds plans that are affected by a restricted state, a stored procedure problem, or an invalid/inoperative package. All information about utilities, restricted states, plans, packages are combined in XM4DB2 to find out the influence on the application landscape, whenever one component has a problem.



Invalid/Inoperative Packages: Packages are used to find the best access path to the data in DB2. The packages must fit to the applications. If the application changes the package has to be rebound. XM4DB2 finds packages that are no longer valid or that are inoperative. It also finds packages that are affected by a restricted state or stored procedure problem.

The exception / skeleton technique that XM4DB2 offers, can be applied to run automated binds or rebinds if requested.

Note: XM4DB2's user interface combines these exceptions and displays them in a convenient manner. For example, a user can see all plans that are affected by a restricted state, or all restricted states caused by a utility.

