Syncsort Optimize DB2

(EZ-DB2)

V10 What's New

Release Summary

Version 10.0.GA1

March 2019

Syncsort Optimize DB2 V10 What's New

NOTICE

Syncsort Optimize DB2, formerly called EZ-DB2 is designed to be used by licensed users of IBM Db2 for z/OS, as a supplement to this product.

Contents

THIS PAGE IS INTENTIONALLY LEFT BLANK (if necessary)

Audience

This document provides a Summary of the changes being provided in EZ-DB2 V10 for existing users of EZ-Tracer and EZ-Cache – it assumes the reader is familiar with EZ-Tracer/EZ-Cache and EZ-DB2 Workload concepts.

Note – not all features will be available at GA. Additional features will be rolled out via agile type development. Please refer to the current EZ-DB2 V10 User Guide for details of implemented features.

Introduction

EZ-DB2 provides a range of components to aid SQL statement level monitoring and tuning, with the primary building blocks being EZ-Tracer and EZ-Cache.

EZ-Tracer and EZ-Cache are proven Db2 tracing tools, in use at many large Db2 production sites, providing a deep insight into SQL workload metrics.

Despite the continuous rollout of new functional changes – many of which have been user driven, the underlying EZ-Tracer and EZ-Cache architecture has remained largely the same.

Version 10 of EZ-DB2 significantly changes and improves the tracing architecture, positioning EZ-Tracer and EZ-Cache ready for a more dynamic and flexible future!

Summary of New Features and Benefits

- *NEW* – Master EZ-DB2 Started Task, providing the ability to control and monitor EZ-Tracer and EZ-Cache workloads.
- *NEW* – Run traces (EZ-Tracer and EZ-Cache) as true started tasks – either standalone or via the Master Started Task.
- *NEW* – Replace flat log datasets with log stream (MVS System Logger - DASD only or CF log streams).
- *NEW* – Replace interval/log dataset post process batch jobs, with a continuous post processing started task.
- *NEW* – Introduction of workload 'templates' to allow new workloads to be created dynamically on-the-fly.
- *NEW* – EZ-Cache enhanced to capture flushed cache entries.
- *NEW* – EZ-Cache enhanced to allow it to run continuously.

- *BENEFIT* – Workloads can be created dynamically at runtime, without having to predefine all the trace parameters in advance.
- *BENEFIT* – Trace JCL does not have to be regenerated for each new trace – improves usability by allowing more spontaneous tracing.
- *BENEFIT* – Easier to control, set up and run traces in production environments. Started tasks are often easier to set the necessary high priority for EZ-Tracer.
- *BENEFIT* – EZ-Cache can be set to run continuously – can be a default trace for collecting SQL execution metrics with minimal overhead on Db2.

- *BENEFIT* - If using Syncsort's Ironstream to send EZ-Tracer events to Splunk, the individual SQL execution events can be sent in near realtime to Splunk, by the continuous post processing task.

The Reason for Change

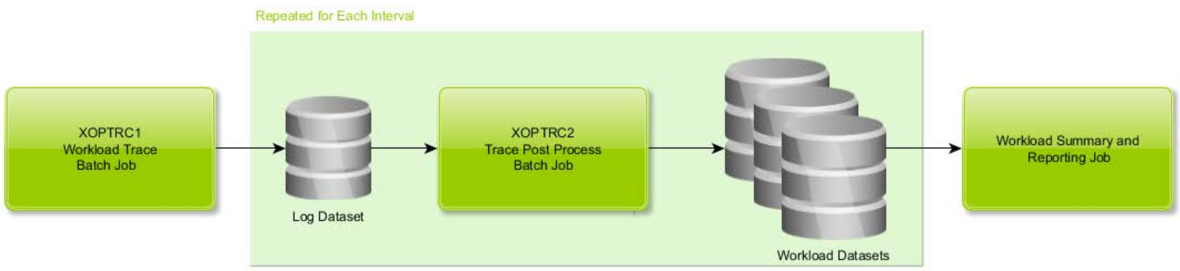
The architecture underpinning EZ-Tracer and EZ-Cache, whilst having been refined over the years, has largely remained the same:

- The workload and trace options must always be predefined in advance, via the EZ-DB2 panels.
- Trace JCL must be regenerated every time, for each run from the EZ-DB2 panels.
- Trace data is written to intermediate flat log datasets, which are then postprocessed at set intervals.
- Reports are generated after the trace completes.

For EZ-Cache only:

- EZ-Cache reads the Db2 cache statistics (dynamic and static) at a user defined interval and calculates the 'delta' metrics, for all SQL statements in the cache at the time of the cache read.
- If Db2 flushes a statement from the cache, then the SQL metrics for that statement since the last read of the cache, are lost.

The entire trace logic is very much predefined and single threaded:



From an operational viewpoint, having to run the trace as a batch job can also have several issues:

- Local production restrictions can limit users' ability to run ad-hoc batch jobs in production environments, sometimes requiring change controls and job scheduling, simply to submit an EZ-DB2 trace job.
- The main batch trace job has, to run at a high priority (at least as high as Db2), to prevent Db2 trace data from being lost. Running batch jobs at such a high priority can again require change controls and pre-planning.

One of the most requested features has always been for the ability to run EZ-Tracer or EZ-Cache as a true started task - while options were added to create 'started task' trace jobs, these were not true started tasks – the workload still had to be predefined and a batch JCL stream generated, which could then be saved and run as a started task – each workload needing its own started task proc.

In designing the EZ-DB2 V10 changes, we wanted to provide the user with a significant number of improvements:

- Ability to run traces as true started tasks.
- Flexibility in creating and modifying trace workloads.
- Control and monitoring of EZ-Tracer and EZ-Cache traces via a Master EZ-DB2 started task.
- The ability to run the trace post processing continuously, while the trace data is still being captured.
- Capture of flushed cache statement metrics in EZ-Cache.
- Simpler setup of EZ-Cache to allow it to run as a 'default' continuous trace.

Architecture Changes

To implement these changes, we have made significant architecture changes to the EZ-Tracer and EZ-Cache products.

Running as Started Tasks

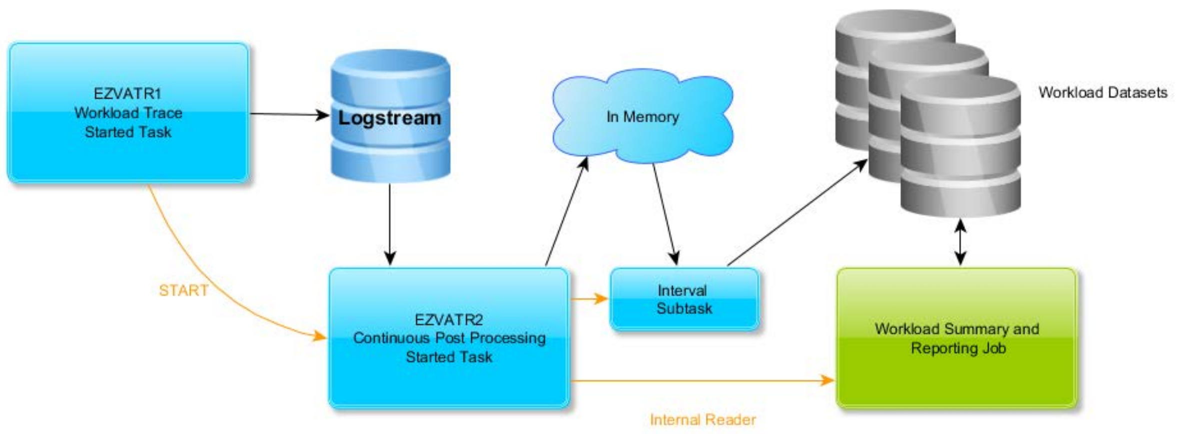
While you can **still** elect to use flat log datasets and run traces as batch jobs, there are now options to START trace workloads as true started tasks, either standalone or from a master started task, using common started task procedures.

Additionally, if running the trace as a started task, you can now elect to use an (MVS System Logger) logstream in place of the flat log datasets. The use of a logstream allows a separate post processing started task to run continuously while the main trace task continues to run. As the main trace task writes trace records to the logstream, the post processing task continuously reads the logstream in FIFO sequence, deleting records as they are post-processed. The logstream is therefore used as a transient queue of trace data.

When processing the logstream, SQL metrics are retained in memory for the current interval – the interval time will still be controlled by the user. The continuous post processing task now runs the interval processing under a separate subtask, completely invisible to the user, which updates the workload while allowing the post processing task to continue.

There is still the requirement for a final report job to be submitted, to do the heavy report lifting and when running in started task mode, a separate report job is automatically submitted at the end of the trace. The report job still runs as a batch job, as it can run at a lower priority than the main trace and post processing tasks.

The figure below shows the processes when running as a started task using log stream:



EZ-Cache Improvements

IBM enhanced Db2 10 to externalize IFCIDs 316 and 401 when statements where flushed from the cache.

EZ-Cache prior to V10 did not take advantage of the flushed cache statement metrics, so could potentially 'miss', a small number of execution metrics. By capturing these flushed entries, EZ-Cache can maintain a complete set of SQL execution data, providing more accurate reports and providing the ability to maintain an accurate warehouse of SQL metrics, over an extended period of time, when using the optional Trend Analysis Database.

Implementation

The most noticeable changes that existing EZ-DB2 users will see, include new started task procedures – for master, trace and trace post-processing.

A number of trace programs must now run authorized, so it is recommended that the main EZ-DB2 load library is authorized at installation time – though a separate authorized library can be specified.

We have ensured that existing users who are familiar with running EZ-Tracer and EZ-Cache as batch jobs, using flat log datasets, can still continue to run in the same way, but can switch or try the started task options with no additional software changes.

EZ-Tracer and EZ-Cache have always interfaced with Db2, using only the IBM published API's to the Db2 Instrumentation Facility - this has always been important to ensure EZ-DB2 cannot adversely impact Db2 subsystems. You can be confident that V10 continues to interface with Db2, only via the published APIs.

Trace 'Modes'

There will be three distinct 'modes' for EZ-Tracer and EZ-Cache workloads:

- **Started Task + Logstream – "Full Started Task Mode"**
    - ❖ The user can define a new workload and trace controls via the EZ-DB2 panels, or use an existing template workload.
    - ❖ The trace is started by the user, with a single /START command using a common trace procedure or using the Master EZ-DB2 Started Task – various parameters and overrides can be passed with the Master START command.
    - ❖ The main trace started task writes the captured SQL trace details to a ***logstream***.
    - ❖ The main trace started task initiates a separate post processing started task to process the events written to the log stream. This post processing task runs continuously until the trace ends, monitoring the logstream for new events and deleting those it has already processed.
    - ❖ At the end of the trace period, a final batch JCL reporting job is submitted internally to do the main reports generation.

- **Started Task + Log Datasets – "Hybrid Mode"**
    - ❖ The user can define a new workload and trace controls via the EZ-DB2 panels or use an existing 'template' workload.
    - ❖ The trace is started by the user, with a single /START command or using the Master EZ-DB2 Started Task – various parameters and overrides can be passed with the Master START command.

- The main trace dynamically allocates the required flat log datasets and writes the captured SQL trace details to the next available log dataset.
- At the trace interval, or if the current log dataset is full, a batch post processing job is submitted internally to process the previous log dataset.
- At the end of the trace period, a final batch JCL reporting job is submitted internally to do the main reports generation.

- **Batch + Log Datasets – "Batch Mode"**
  - The user defines a new workload and trace controls in advance, via the EZ-DB2 panels.
  - Batch JCL is generated via the panels for each new trace.
  - The trace is started by submitting a single batch JCL job – all datasets are allocated in the JCL stream.
  - The main trace job writes the captured SQL trace details to the next log dataset.
  - At the trace interval, or when the current log dataset is full, a batch post processing job is submitted internally to process the last log dataset.
  - At the end of the trace period, a final batch JCL reporting job is submitted internally to do the main reports generation.

Option 1 above provides all the V10 benefits of dynamic workload creation and management, running as a true started task, with continuous post processing of the SQL data.

Option 2 above, is a 'hybrid' of the old flat log datasets and batch post processing, but with the benefits of the main trace being defined and run as a true started task.

Existing EZ-DB2 users will be familiar with *Option 3* above, as this is the equivalent to EZ-DB2 V9 trace processing.

Roll Out of Changes

The initial GA release of V10 concentrates on providing the underlying architecture changes to support running as started tasks (via Master or standalone), starting and monitoring workloads via the Master task and the ability to replace log datasets with a logstream.

The initial release will quickly be updated via 'agile' type delivery of maintenance and new features, to fully implement all the changes detailed above.

Version 10 positions EZ-DB2 with strong emphasis on the tracing tools, ready for continued improvements and future integration with other tools and products.

Please refer to the separate document covering the V10 installation pack, which details the features currently available, implemented and how to use them.