Most DBAs are familiar with the basic use of explains to improve performance by looking at access path details. This presentation will explore some of the less well-known features of explains including the new in Db2 11.5.9, EXPLAIN_FORMAT stored procedure and the EXPLAIN_FORMAT_STATS function. There are a number of other explain routines that will be explored. The various methods to capture explain data will also be covered. Ways to combine explains with db2batch and index usage data will also be covered.

Agenda

- Exploring ways to capture EXPLAIN data
- Using EXPLAIN_FORMAT stored procedure
- Using EXPLAIN_FORMAT_STATS function
- Using other EXPLAIN functions
- How to combine data from tools including db2batch and index usage data

Explain Facility

- The Db2 explain facility provides detailed information about the access path that the optimizer chooses for SQL or Xquery statement.

EXPLAIN Facility
https://www.ibm.com/docs/en/db2/11.5?topic=optimization-explain-facility

The explain facility is invoked by issuing the EXPLAIN statement, which captures information about the access plan chosen for a specific explainable statement and writes this information to explain. You can also set CURRENT EXPLAIN MODE or CURRENT EXPLAIN SNAPSHOT (special registers that control the behavior of the explain facility).

Capturing Explain Data

- **Execute SQL after setting current explain mode special register**
- **Issue EXPLAIN statement with a FOR SNAPSHOT or a WITH SNAPSHOT clause**
- **Execute db2batch with explain option**
- **Insert explain information in package cache into explain tables**

https://www.ibm.com/docs/en/db2/11.5?topic=facility-guidelines-capturing-explain-information

https://www.ibm.com/support/pages/db2-how-extract-explain-information-package-cache

Capturing Explain Data

- Section Explains
  - EXPLAIN_FROM_ACTIVITY
  - EXPLAIN_FROM_CATALOG
  - EXPLAIN_FROM_DATA
  - EXPLAIN_FROM_SECTION
    - From Package Cache
    - From Package Cache Event Monitor
- \•

https://www.ibm.com/support/pages/db2-how-extract-explain-information-package-cache

https://www.ibm.com/docs/en/db2/11.5?topic=facility-guidelines-capturing-section-explain-information

https://www.ibm.com/docs/en/ias?topic=er-explain-from-section-procedure-explain-statement-using-package-cache-package-cache-event-monitor-information

Capturing Explain Data

- Executable_id can be found in following sources:
  - Activity event monitor
  - MON_GET_ACTIVITY_DETAILS table function
  - Package cache event monitor
  - MON_GET_PKG_CACHE_STMT table function
  - MON_GET_PKG_CACHE_STMT_DETAILS table function
  - MONREPORT.PKGCACHE
  - MON_GET_APPL_LOCKWAIT table function
  - WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES table function
  - WLM_GET_SERVICE_CLASS_AGENTS table function

https://www.ibm.com/support/pages/db2-how-extract-explain-information-package-cache

https://www.ibm.com/docs/en/db2/11.5?topic=facility-guidelines-capturing-section-explain-information

https://www.ibm.com/docs/en/ias?topic=er-explain-from-section-procedure-explain-statement-using-package-cache-package-cache-event-monitor-information

What to Explain

Some one reported bad performing query

Or

"mine" for bad queries

MON_GET_PKG_CACHE Get Package Cache statement metrics

```
SELECT EXECUTABLE_ID, TOTAL_CPU_TIME, VARCHAR(STMT_TEXT, 100)
FROM TABLE(MON_GET_PKG_CACHE_STMT (NULL, NULL, NULL, -1)) AS T
ORDER BY TOTAL_CPU_TIME DESC FETCH FIRST 10 ROWS ONLY
```

https://www.ibm.com/docs/en/db2/11.5?topic=mpf-mon-get-pkg-cache-stmt-table-function-get-package-cache-statement-metrics

What Are Bad Queries

No exact answer but one with the following can be cause for investigation:
High TOTAL_CPU_TIME
High ROWS_READ
High ROWS_RETURNED
High ROWS_MODIFIED
High STMT_EXECUTION_TIME
High TOTAL_SORTS
High NUM_EXECUTIONS
High averages of any of the above are important

Many others in MON_GET_PKG_CACHE_STMT could be looked at.

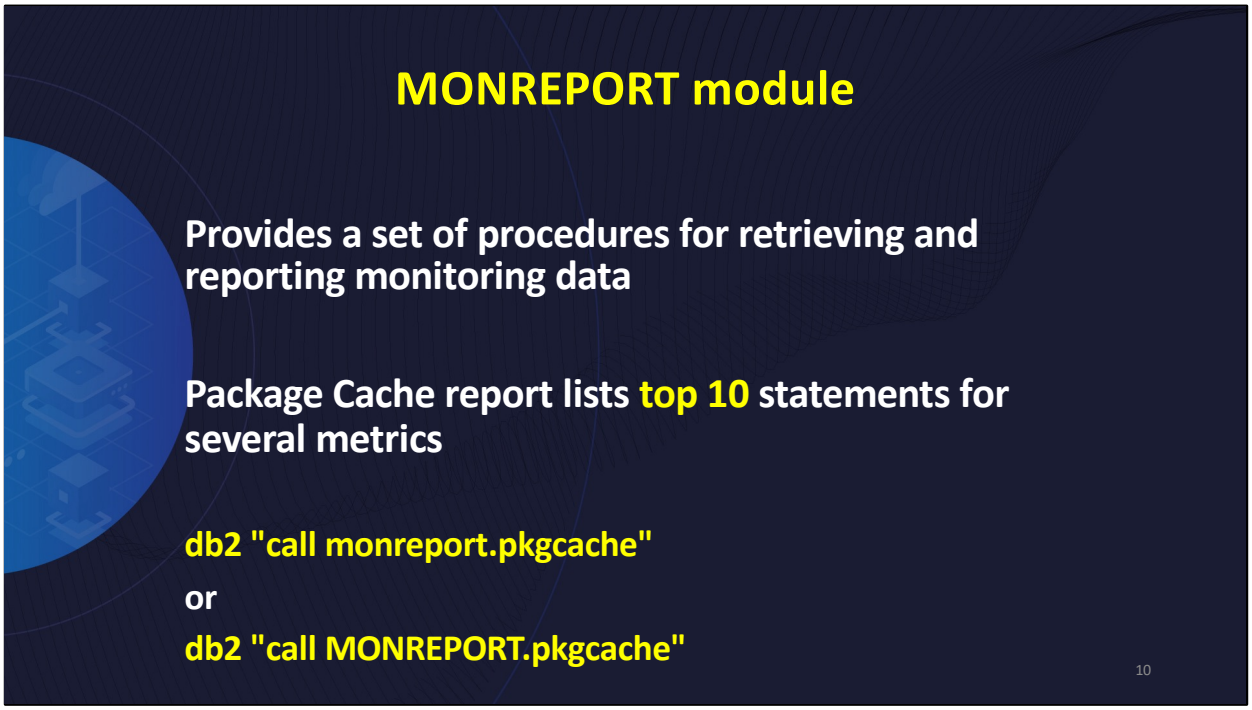
https://www.ibm.com/docs/en/db2/11.5?topic=mpf-mon-get-pkg-cache-stmt-table-function-get-package-cache-statement-metrics

What to Explain

```
SELECT EXECUTABLE_ID, TOTAL_CPU_TIME, VARCHAR(STMT_TEXT, 100)
FROM TABLE(MON_GET_PKG_CACHE_STMT (NULL, NULL, NULL, -1)) AS T
ORDER BY ROWS_READ DESC FETCH FIRST 10 ROWS ONLY
```

```
SELECT EXECUTABLE_ID, TOTAL_CPU_TIME, VARCHAR(STMT_TEXT, 100)
FROM TABLE(MON_GET_PKG_CACHE_STMT (NULL, NULL, NULL, -1)) AS T
ORDER BY ROWS_RETURNED DESC FETCH FIRST 10 ROWS ONLY
```

https://www.ibm.com/docs/en/db2/11.5?topic=mpf-mon-get-pkg-cache-stmt-table-function-get-package-cache-statement-metrics



Excellent way to get a large amount of performance information very quickly. DBSUMMARY especially good for that.

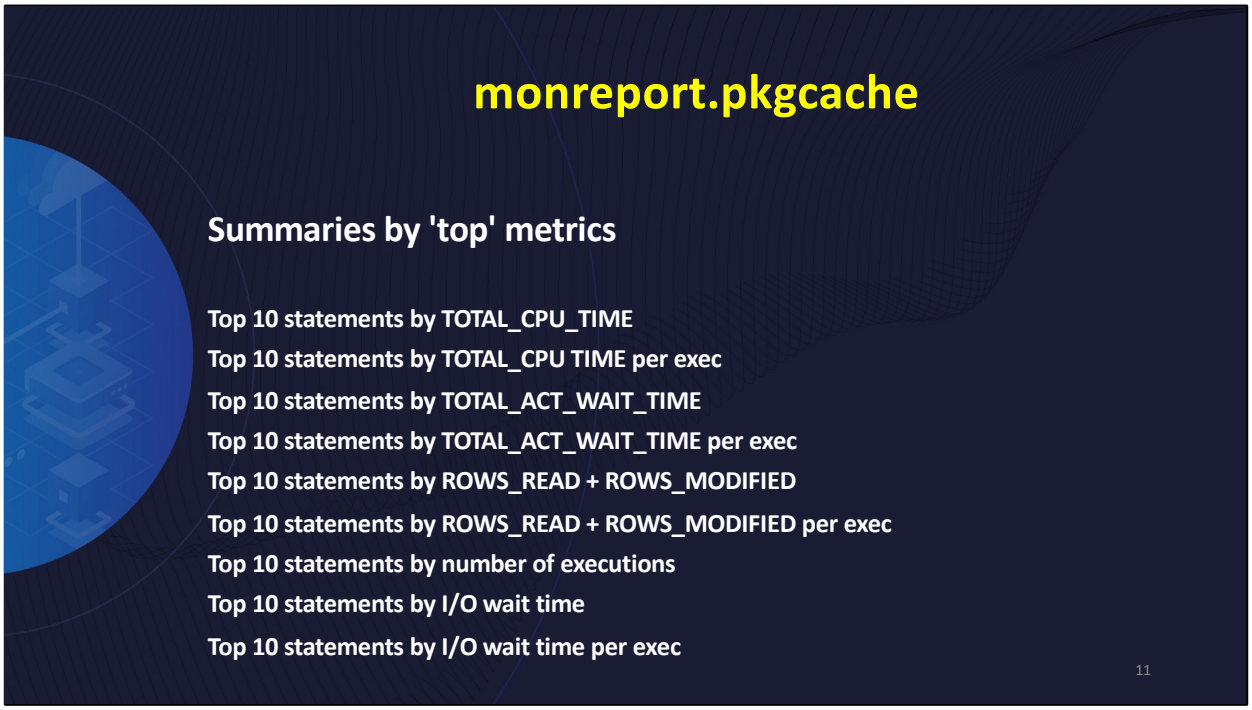
https://www.ibm.com/docs/en/db2/11.5?topic=modules-monreport-module

The other routines are:
CONNECTION
CURRENTAPPS
CURRENTSQL
DBSUMMARY
LOCKWAIT

https://www.ibm.com/docs/en/db2/11.5?topic=interfaces-reports-generated-using-monreport-module

It is possible to customize MONREPORT to make it Top 20 or Top N

https://datageek.blog/en/2014/12/02/db2-basics-monreport/



https://www.ibm.com/docs/en/db2/11.5?topic=mm-pkgcache-procedure-generate-summary-report-package-cache-metrics

monreport.pkgcache Details

```
Part 1 - Summaries by 'top' metrics

Top 10 statements by TOTAL_CPU_TIME
-------------------------------------------------------------------------------
#   TOTAL_          STMT_TEXT
    CPU_TIME
--  ------------  ------------------------------------------------------------
1   97207         CALL SYSPROC.SYSINSTALLOBJECTS('POLICY','V','','')
2   50900         CALL SYSIBM.SQLCAMESSAGECCSID (:HV00010  :HI00010 , 80, NULL, N
3   41290         SELECT ARRAY_AGG(VALUE) INTO :HV00029  :HI00029  FROM SYSIBMADM
4   25659         CALL SYSINSTALLOBJECTS( 'DB2AC', 'V', NULL, NULL )

Top 10 statements by TOTAL_CPU_TIME per exec
-------------------------------------------------------------------------------
#   TOTAL_          STMT_TEXT
    CPU_TIME
--  ------------  ------------------------------------------------------------
3   41290         SELECT ARRAY_AGG(VALUE) INTO :HV00029  :HI00029  FROM SYSIBMADM
11  23493         SELECT ARRAY_AGG(P.EXECUTABLE_ID ORDER BY M.CPU_TIME DESC), ARR
6   4820          VALUES (CURRENT DATE || SPACE(1) || CURRENT TIME) INTO :HV00018
```

The circles are around the statement number that was seen on the previous detail screen seen in the previous slide.

This shows the Top 10 statements and two important ones are by TOTAL_CPU_TIME and by TOTAL_CPU_TIME per execution. TOTAL_CPU_TIME is good for showing the total amount of CPU used by that single SQL. TOTAL_CPU_TIME per execution is good for showing how bad the CPU consumption is per execution.

The reports start with the hardest hitters and go in decreasing order.

monreport.pkgcache Report Details

```
Part 2 - EXECUTABLE_IDs for statements in Part 1

#   EXECUTABLE_ID
--  --------------------------------------------------------------------------------
1   x'00000001000000000000000000000002000000000002201107082024015964 83'
2   x'00000001000000000000000000000E6000000010001201107071850326992 31'
3   x'00000001000000000000000000000001D000000000002201107082040116776 49'
4   x'00000001000000000000000000000006000000000002201107082024042423 30'
5   x'0000000100000000000000000000000A000000000002201107082024048258 56'
6   x'00000001000000000000000000000EE0000000F0001201107071850517026 89'
7   x'00000001000000000000000000000EE0000000A0001201107071850517026 89'
8   x'00000001000000000000000000000010000000000002201107082024056607 80'
9   x'00000001000000000000000000000D000000000002201107082024054985 94'
10  x'00000001000000000000000000000004000000000002201107082024035096 75'
11  x'00000001000000000000000000000EE0000002200012011070718505170268 9'
12  x'00000001000000000000000000000EE0000002500012011070718505170268 9'
13  x'00000001000000000000000000000EE0000000D0001201107071850517026 89'
14  x'00000001000000000000000000000EE0000001300012011070718505170268 9'
15  x'00000001000000000000000000000EE0000002600012011070718505170268 9'
16  x'00000001000000000000000000000EE0000002400012011070718505170268 9'
17  x'0000000100000000000000000000001C000000000002201107082040097222 64'
18  x'00000001000000000000000000000EE0000001400012011070718505170268 9'
```

13

This shows the SQL details and assigns a number to each statement. Those assigned numbrs will be seen in the reports that follow this slide.

You are able to query Package Cache using EXECUTABLE_ID to get many details about a particular statement.

EXPLAIN_FROM_ACTIVITY

SELECT APPL_ID, UOW_ID, ACTIVITY_ID, USER_CPU_TIME
FROM ACTIVITY_A ORDER BY USER_CPU_TIME DESC

The following example shows output from this query. The application with an ID of N2.DB2INST1.0B5A12222841 is using a large amount of CPU time.

```
APPL_ID                  UOW_ID     ACTIVITY_ID    USER_CPU_TIME
-----------------------  -------- --------------  --------------
*N2.DB2INST1.0B5A12222841   1              1      92782334234
*N2.DB2INST1.0B5A12725841   2              7              326

  2 record(s) selected.
```

From:
https://www.ibm.com/docs/en/ias?topic=er-explain-from-activity-procedure-explain-statement-using-activity-event-monitor-information

https://www.ibm.com/docs/en/ias?topic=er-explain-from-activity-procedure-explain-statement-using-activity-event-monitor-information

https://www.ibm.com/docs/en/db2/11.5?topic=facility-guidelines-capturing-explain-information

https://www.ibm.com/support/pages/db2-how-extract-explain-information-package-cache

CREATE EXPLAIN TABLES

list tables for all | grep -i explain

If no result, then EXPLAIN tables need to be created

```
db2 CALL SYSPROC.SYSINSTALLOBJECTS('EXPLAIN', 'C',
CAST (NULL AS VARCHAR(128)), CAST (NULL AS
VARCHAR(128)))   Default schema is SYSTOOLS
```

```
db2 -tf EXPLAIN.DDL
```

<instance owner home directory>/sqllib/misc
Can customize to be in a certain tablespace

https://www.ibm.com/docs/en/db2/11.5?topic=information-creating-explain-tables

EXPLAIN_FROM_ACTIVITY

```
CALL EXPLAIN_FROM_ACTIVITY
('*N2.DB2INST1.0B5A12222841',   1 , 1, 'TESTEVMON', 'SYSTOOLS', ?, ?, ?, ?, ?)

appl_id, uow_id,
activity_id,
activity_evmon_name
explain_schema,
explain_requestor, explain_time
source_name, source_schema, source_version
```

https://www.ibm.com/docs/en/ias?topic=er-explain-from-activity-procedure-explain-statement-using-activity-event-monitor-information

https://www.ibm.com/docs/en/db2/11.5?topic=facility-guidelines-capturing-explain-information

https://www.ibm.com/support/pages/db2-how-extract-explain-information-package-cache

EXPLAIN_FROM_CATALOG

Use for static statements

```
CALL EXPLAIN_FROM_CATALOG
('NULLID', 'SQLE2G0S', ' ',  1 , 'SYSTOOLS', ?, ?, ?, ?, ?)
```

pkgschema, pkgname, pkgversion
sectno, explain_schema,
explain_requestor, explain_time
source_name, source_schema, source_version

Can find section number by querying syscat.statements

https://www.ibm.com/docs/en/db2/11.5?topic=facility-guidelines-capturing-explain-information

https://www.ibm.com/support/pages/db2-how-extract-explain-information-package-cache

EXPLAIN_FROM_DATA

- The EXPLAIN_FROM_DATA procedure explains a statement using the contents of the input section.
- Input section can come from:
  - Activity event monitor
  - Package cache event monitor
  - Catalog tables
- Can use this after using a package cache event monitor and extracting the event monitor data (EVMON_FORMAT_UE_TO_TABLES stored procedure)

https://www.ibm.com/docs/en/db2/11.5?topic=facility-guidelines-capturing-explain-information

https://www.ibm.com/support/pages/db2-how-extract-explain-information-package-cache

Package Cache Event Monitor

CREATE EVENT MONITOR PKG_CACHE_MONITOR FOR PACKAGE CACHE WRITE to UNFORMATTED EVENT TABLE (TABLE DB2INST1.PKG_CACHE IN TABSPC32K) AUTOSTART

Turn on

SET EVENT MONITOR PKG_CACHE_MONITOR STATE = 1

Turn off

SET EVENT MONITOR PKG_CACHE_MONITOR STATE = 0

```
https://www.ibm.com/docs/en/db2/11.5?topic=monitors-event-that-write-unformatted-event-tables
```

```
https://www.ibm.com/docs/en/db2/11.5?topic=tables-creating-event-monitors-that-write
```

```
CREATE EVENT MONITOR MY_LOCKS FOR LOCKING WRITE TO UNFORMATTED EVENT TABLE (TABLE DBA.MY_LOCKS IN TAB32K) AUTOSTART;
```

```
SET EVENT MONITOR MY_LOCKS STATE = 1;
```

https://virtual-dba.com/blog/investigating-locking-in-db2/
Shows how to create a locking event monitor

EVMON_FORMAT_UE_TO_TABLES

Used to extract data from unformatted event (UE) table. This puts the data into multiple relational tables.

EVMON_FORMAT_UE_TO_TABLES (evmon_type, xsrschema, xsrobjectname, xmlschemafile,tabschema, tbsp_name, options, commit_count, fullselect)

db2 "CALL EVMON_FORMAT_UE_TO_TABLES ('PKGCACHE', NULL, NULL, 'Paul', NULL, NULL, -1, 'SELECT * FROM PKGTBLE ORDER BY event_timestamp)"

https://www.ibm.com/docs/en/db2oc?topic=mpf-evmon-format-ue-tables-procedure-move-xml-document-relational-tables
Event monitors can write data to relational table or unformatted event (UE) tables, which is event data in a binary format.
There are two procedures to extract data from UE tables.

EVMON_FORMAT_UE_TO_TABLE. After using this, use PRUNE_UE_TABLE option to remove data that is no longer needed.
Run the EVMON_FORMAT_UE_TO_TABLES procedure, using the FORCE option. This option causes the old tables to be dropped, and a new set of tables to be produced.

db2evmonfmt, produces a readable flat-text output (text version) or a formatted XML output from the data generated by an event monitor that writes its output to a regular table or that uses the unformatted event table.

EVMON_FORMAT_UE_TO_TABLES(evmon_type,xsrschema,xsrobjectname,xmlschemafile,tabschema,tbsp_name,options,commit_count,fullselect)
evmon_type = LOCKING, PKGCACHE, UOW
not a one-to-one mapping between the records written to the UE table and the output of the EVMON_FORMAT_UE_TO_TABLES procedure.
https://xcoolwinds.wordpress.com/2012/06/28/how-to-speed-up-the-formatting-of-data-captured-by-the-db2-event-monitors/

EXPLAIN_FROM_DATA

Parameters for EXPLAIN_FROM_DATA

section, stmt_text, executable_id

explain_schema,
explain_requestor, explain_time
source_name, source_schema, source_version

https://www.ibm.com/docs/en/db2/11.5?topic=facility-guidelines-capturing-explain-information

https://www.ibm.com/support/pages/db2-how-extract-explain-information-package-cache

EXPLAIN_FROM_DATA

Source:
https://www.ibm.com/docs/en/db2/11.5?topic=er-explain-from-data-procedure-explain-statement-using-input-section

```
SET SERVEROUTPUT ON;

BEGIN
  DECLARE EXECUTABLE_ID VARCHAR(32) FOR BIT DATA; --
  DECLARE SECTION BLOB(134M); --
  DECLARE STMT_TEXT CLOB(2M); --
  DECLARE EXPLAIN_SCHEMA VARCHAR(128); --

  DECLARE EXPLAIN_REQUESTER VARCHAR(128); --
  DECLARE EXPLAIN_TIME TIMESTAMP; --
  DECLARE SOURCE_NAME VARCHAR(128); --
  DECLARE SOURCE_SCHEMA VARCHAR(128); --
  DECLARE SOURCE_VERSION VARCHAR(128); --

  SET EXPLAIN_SCHEMA = 'MYSCHEMA'; --

  SELECT P.SECTION, P.STMT_TEXT, P.EXECUTABLE_ID INTO
         SECTION, STMT_TEXT, EXECUTABLE_ID
  FROM PKGCACHE WHERE EXECUTABLE_ID =
   x'0100000000000000070000000000000000000000000200200811261904103698'; --
```

https://www.ibm.com/docs/en/db2/11.5?topic=facility-guidelines-capturing-explain-information

https://www.ibm.com/support/pages/db2-how-extract-explain-information-package-cache

EXPLAIN_FROM_DATA

```
  CALL EXPLAIN_FROM_DATA( SECTION,
                          STMT_TEXT,
                          EXECUTABLE_ID,
                          EXPLAIN_SCHEMA,
                          EXPLAIN_REQUESTER,
                          EXPLAIN_TIME,
                          SOURCE_NAME,
                          SOURCE_SCHEMA,
                          SOURCE_VERSION ); --

  CALL DBMS_OUTPUT.PUT( 'EXPLAIN_REQUESTER = ' ); --
  CALL DBMS_OUTPUT.PUT_LINE( EXPLAIN_REQUESTER ); --
  CALL DBMS_OUTPUT.PUT( 'EXPLAIN_TIME = ' ); --
  CALL DBMS_OUTPUT.PUT_LINE( EXPLAIN_TIME ); --
  CALL DBMS_OUTPUT.PUT( 'SOURCE_NAME = ' ); --
  CALL DBMS_OUTPUT.PUT_LINE( SOURCE_NAME ); --
  CALL DBMS_OUTPUT.PUT( 'SOURCE_SCHEMA = ' ); --
  CALL DBMS_OUTPUT.PUT_LINE( SOURCE_SCHEMA ); --
  CALL DBMS_OUTPUT.PUT( 'SOURCE_VERSION = ' ); --
  CALL DBMS_OUTPUT.PUT_LINE( SOURCE_VERSION ); --
END;

SET SERVEROUTPUT OFF;
```

https://www.ibm.com/docs/en/db2/11.5?topic=facility-guidelines-capturing-explain-information

https://www.ibm.com/support/pages/db2-how-extract-explain-information-package-cache

EXPLAIN_FROM_SECTION

```
CALL EXPLAIN_FROM_SECTION
( x'00000001000000000000000000003D3000000000002201108311718149624 75',
'M', NULL, 0, 'SYSTOOLS', ?, ?, ?, ?, ?)
```

Executable id
source of section [M = from in-memory package cache, P = package cache event monitor]
Section source name (if source of section is P)
Member
Schema for EXPLAIN table
Explain requestor
Explain time
Name of package
Source schema
Source version

https://www.ibm.com/docs/en/db2/11.5?topic=SSEPGG_11.5.0/com.ibm.db2.luw.admin.explain.doc/doc/c0021365.htm

https://www.ibm.com/docs/en/db2/11.5?topic=statements-set-current-explain-snapshot

CURRENT EXPLAIN SNAPSHOT special register values

YES Enables the Explain snapshot facility, creating a snapshot of the internal representation for each eligible dynamic SQL statement. This information is inserted in the SNAPSHOT column of the EXPLAIN_STATEMENT table.

NO Disable

EXPLAIN Like YES except statement does not execute

Capturing Explain Data

db2 set current explain mode explain

Once this is set, every SQL statement executed in this session will be explained rather than executing

db2 -tvf query1.sql

db2 set current explain mode no

- **Turns off the explain facility**

db2 -tf EXPLAIN.DDL **SQL0219N**

- **Creates explain tables**

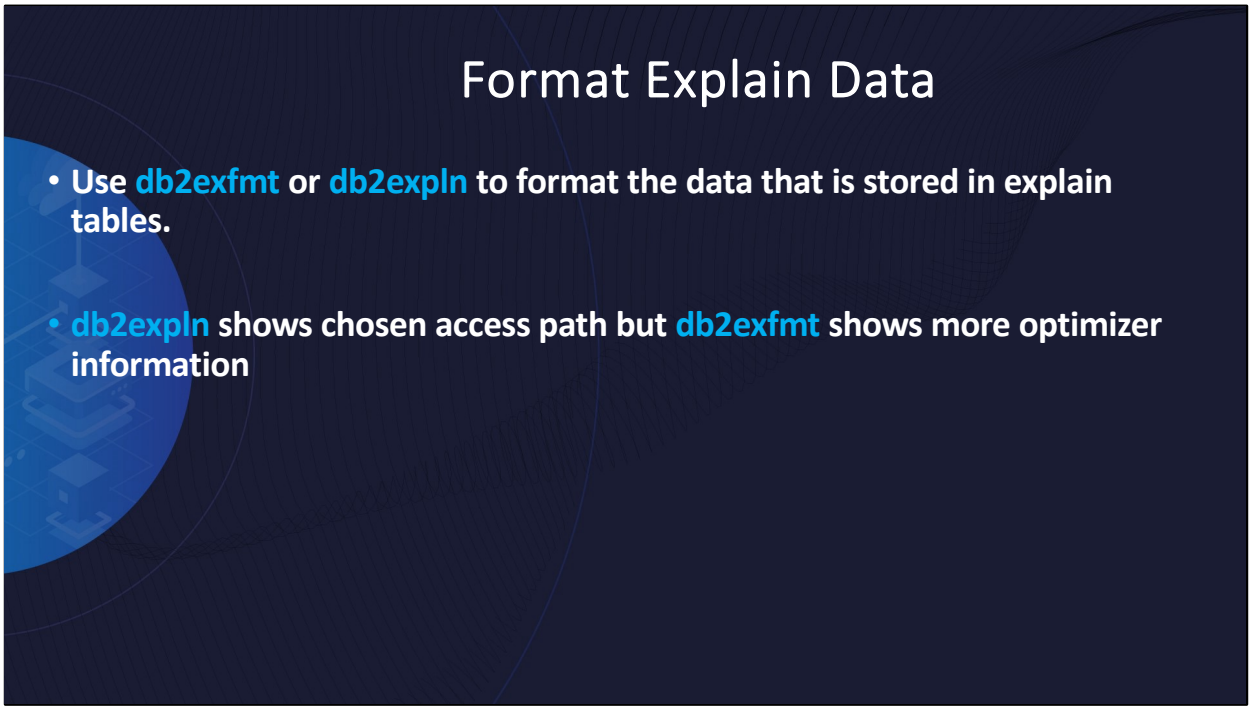
https://www.ibm.com/docs/en/db2/11.5?topic=registers-current-explain-mode

EXPLAIN Modes

NO	Disables the explain facility
YES	executes the statement and explains the statement
RECOMMEND INDEXES	Recommends indexes for the statement and populates ADVISE_INDEX table
EVALUATE INDEXES	Queries executed in this explain mode will be compiled and optimized using fabricated statistics based on the virtual indexes.

Ember Crooks Explain blog entrys
https://datageek.blog/2013/06/04/explain-part-1-explain-and-the-db2-optimizer/

https://datageek.blog/2013/06/11/explain-part-2-command-line-explain-plans-using-db2exfmt/



new in Db2 11.5.9

EXPLAIN_FORMAT stored procedure
https://www.ibm.com/docs/en/db2/11.5?topic=information-explain-format-stored-procedure&mhsrc=ibmsearch_a&mhq=EXPLAIN%26lowbar%3BFORMAT%20

Use EXPLAIN_FORMAT in Watson query
https://www.ibm.com/docs/en/cloud-paks/cp-data/4.7.x?topic=procedures-explain-format-stored-procedure&mhsrc=ibmsearch_a&mhq=EXPLAIN%26lowbar%3BFORMAT%20

EXPLAIN Facility
https://www.ibm.com/docs/en/db2/11.5?topic=optimization-explain-facility

EXPLAIN_FORMAT stored procedure

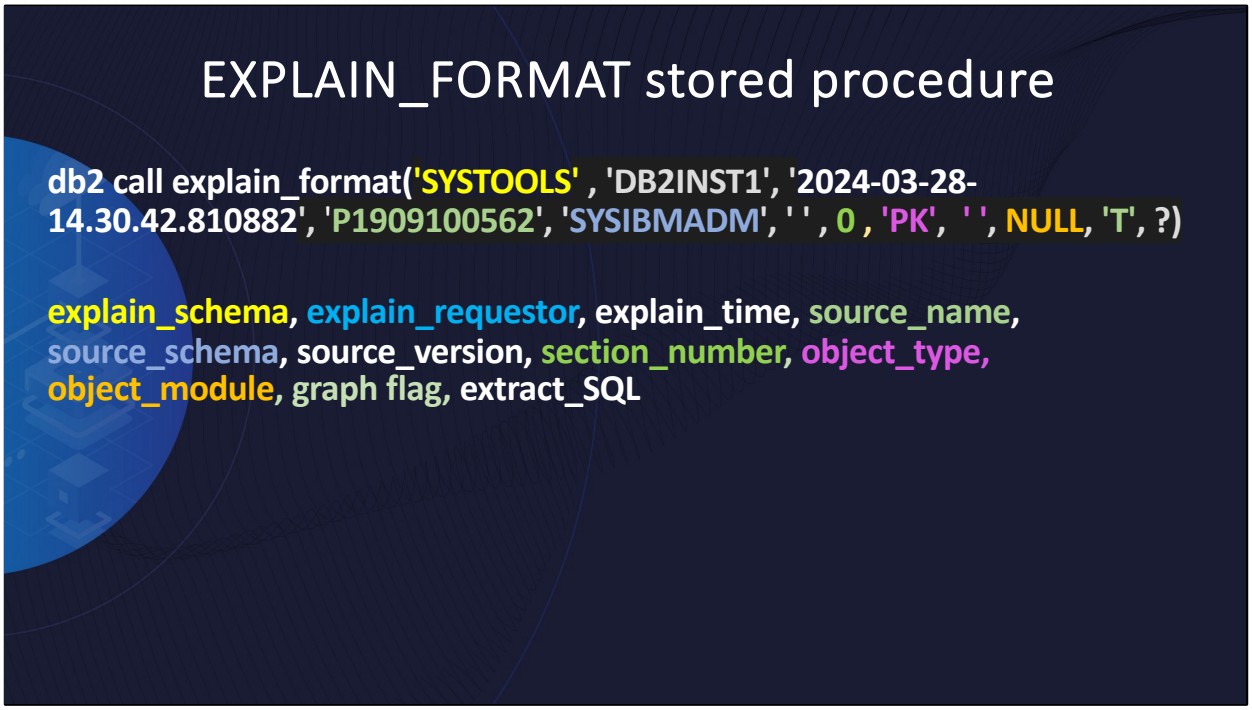
- Using EXPLAIN_FORMAT stored procedure
- Formats the contents of EXPLAIN_STATEMENT table
- Stores formatted explain data in EXPLAIN_FORMAT_TEXT column [CLOB(2M)]
- Procedure returns an SQL statement that can fetch the formatted data from the EXPLAIN_STATEMENT table

new in Db2 11.5.9

EXPLAIN_FORMAT stored procedure
https://www.ibm.com/docs/en/db2/11.5?topic=information-explain-format-stored-procedure&mhsrc=ibmsearch_a&mhq=EXPLAIN%26lowbar%3BFORMAT%20

Use EXPLAIN_FORMAT in Watson query
https://www.ibm.com/docs/en/cloud-paks/cp-data/4.7.x?topic=procedures-explain-format-stored-procedure&mhsrc=ibmsearch_a&mhq=EXPLAIN%26lowbar%3BFORMAT%20

EXPLAIN Facility
https://www.ibm.com/docs/en/db2/11.5?topic=optimization-explain-facility



new in Db2 11.5.9 (Fix pack 9)

EXPLAIN_FORMAT stored procedure
https://www.ibm.com/docs/en/db2/11.5?topic=information-explain-format-stored-procedure&mhsrc=ibmsearch_a&mhq=EXPLAIN%26lowbar%3BFORMAT%20

Use EXPLAIN_FORMAT in Watson query
https://www.ibm.com/docs/en/cloud-paks/cp-data/4.7.x?topic=procedures-explain-format-stored-

procedure&mhsrc=ibmsearch_a&mhq=EXPLAIN%26lowbar%3BFORMAT%20

•object_typeType of the object specified. The default type is package.PK: Package name
•P: SQL procedure name
•SP: A specific SQL procedure name
•F: Compiled function
•SF: A specific compiled function name
•T: Compiled trigger

Graph Flag
•Turn OFF options (default is to turn them ON). If empty, a graph, followed by formatted information for all of the tables, is generated. Otherwise, any combination of the following valid values can be specified:O: Generate a graph only. Do not format the table contents.
•T: Include total cost under each operator in the graph.
•F: Include first tuple cost in graph.
•I: Include I/O cost under each operator in the graph.
•C: Include the expected output cardinality (number of tuples) of each operator in the graph.

EXPLAIN_FORMAT stored procedure

```
select EXPLAIN_FORMAT_TEXT from SYSTOOLS.EXPLAIN_STATEMENT
where EXPLAIN_REQUESTER='DB2INST1'
and EXPLAIN_TIME='2022-11-08-02.28.42.810882'
and SOURCE_NAME='SQLC2P31'
and SOURCE_SCHEMA='NULLID'
--and SOURCE_VERSION= ''
and SECTION_NUMBER=0
and EXPLAIN_LEVEL='O' FOR READ ONLY;
```

new in Db2 11.5.9

EXPLAIN_FORMAT stored procedure
https://www.ibm.com/docs/en/db2/11.5?topic=information-explain-format-stored-procedure&mhsrc=ibmsearch_a&mhq=EXPLAIN%26lowbar%3BFORMAT%20

Use EXPLAIN_FORMAT in Watson query
https://www.ibm.com/docs/en/cloud-paks/cp-data/4.7.x?topic=procedures-explain-format-stored-procedure&mhsrc=ibmsearch_a&mhq=EXPLAIN%26lowbar%3BFORMAT%20

EXPLAIN_FORMAT_STATS function

Display formatted statistics information that is extracted from explain snapshot

```
SELECT EXPLAIN_FORMAT_STATS (SNAPSHOT)
   FROM SYSTOOLS.EXPLAIN_STATEMENT
WHERE EXPLAIN_REQUESTER = 'DB2USER1' AND
EXPLAIN_TIME = timestamp('2024-04-27-15.30.01.010123') AND
SOURCE_NAME = 'SQLC2F0A' AND SOURCE_SCHEMA = 'MYSCHEMA' AND
SOURCE_VERSION = ' ' AND EXPLAIN_LEVEL = 0 AND STMTNO = 1
AND SECTNO = 201
```

Obtain Explain Data with db2batch

- Explains are good at giving theoretical access paths
- db2batch runs the query and captures real-time access path (explain) and other usage data

db2batch -d TEST -f query1.sql -r query1.sql.out -o f 100 p 4 e yes -msw table on -mss dbase_tables

- The "e yes" option of the Control options (-o) will cause the Explain tables to be populated

https://www.ibm.com/docs/en/db2/11.5?topic=commands-db2batch-benchmark-tool

e *explain_mode*

No Run query only (default).
explain Populate explain tables ONLY. This option populates the explain tables and causes explain snapshots to be taken.
yes Populate explain tables and run query. This option populates the explain tables and causes explain snapshots to be taken.

db2batch uses

- Good for strange problems like 6 billion rows read due to confused nested loop join
- Gives details about table usage
- Some more setup work for db2batch, especially with SQL with parameter markers, but excellent performance tool

https://www.ibm.com/docs/en/db2/11.5?topic=commands-db2batch-benchmark-tool

e ***explain_mode***

No **Run query only (default).**
explain **Populate explain tables ONLY. This option populates the explain tables and causes explain snapshots to be taken.**
yes **Populate explain tables and run query. This option populates the explain tables and causes explain snapshots to be taken.**

Format Explain Data from db2batch

- **Run this immediately after the db2batch command, to format the explain data**

-e % -n % -s % -v % -w -1 -# 0

db2exfmt -d sample -g TIC -w -1 -n % -s % -# 0 -o query1_June1.out

-g Graph plan

-w Explain timestamp. -1 is latest explain request

-1 Uses these defaults. -e % -n % -s % -v % -w -1 -# 0

And will need to answer 3 questions at prompt

-o Output file name

Use defaults -e % -n % -s % -v % -w -1 -# 0

https://www.ibm.com/docs/en/db2/11.5?topic=commands-db2exfmt-explain-table-format

https://www.ibm.com/docs/en/db2/11.5?topic=facility-understanding-db2exfmt-information

Format Explain Data from db2batch

- **Run this sometime after the db2batch command, to format the explain data. It specifies exact EXPLAIN TIME.**

-e % -n % -s % -v % -w -1 -# 0

db2exfmt -d sample -g TIC -w '2024-05-30-19.28.00.091070' -n % -s % -# 0 -o query1_June1.out

Use defaults -e % -n % -s % -v % -w -1 -# 0

https://www.ibm.com/docs/en/db2/11.5?topic=commands-db2exfmt-explain-table-format

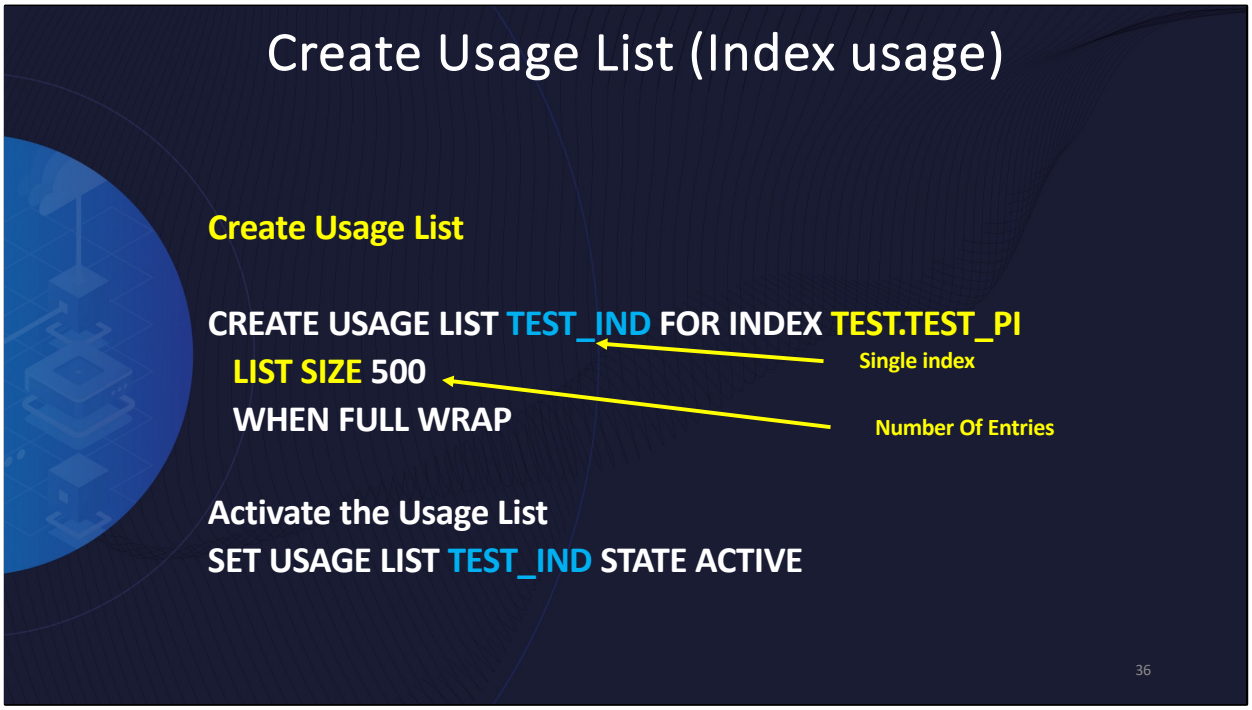
https://www.ibm.com/docs/en/db2/11.5?topic=facility-understanding-db2exfmt-information

Combine Explain Data with index usage data

- **Need to set up usage list for each index that needs to be tracked**
- **Create Index Usage List for each index**
- **Can find all indexes with catalog query or by using**
- **Describe relational data indexes for table <tabschema>.<tabname>**

https://www.ibm.com/docs/en/db2/11.5?topic=commands-describe

https://www.ibm.com/docs/en/db2/11.5?topic=statements-create-usage-list



https://www.ibm.com/docs/en/db2/11.5?topic=statements-create-usage-list

Usage list is a database object for monitoring all unique sections (DML statements) that have referenced a particular table or index during their execution.

The states if a Usage List are:
Active
Inactive
Released Memory for it has been released

Create Usage List
https://www.ibm.com/docs/en/db2/11.5?topic=statements-create-usage-list

Activate (set) the Usage List
https://www.ibm.com/docs/en/db2/11.5?topic=statements-set-usage-list-state

SYSCAT.USAGELISTS

SYSCAT.USAGELISTS can be queries to find all usage lists

syscat.indexes has column LASTUSED which contains a date of last use but no details about what used it

db2pd - d <dbname> -tcbstats

Gives indication that index was used to read a table but no details about what index was used

https://www.ibm.com/docs/en/db2/11.5?topic=statements-create-usage-list

Create Usage List (Index usage)

```
SELECT
SUBSTR(USAGELISTSCHEMA,1,20) as SCHEMA,
SUBSTR(USAGELISTNAME,1,25) as USAGELIST,
OBJECTTYPE,
SUBSTR(OBJECTSCHEMA,1,20) as OBJSCHEMA,
SUBSTR(OBJECTNAME,1,30) as OBJNAME,
STATUS, MAXLISTSIZE, WHENFULL, AUTOSTART
FROM SYSCAT.USAGELISTS
ORDER BY OBJECTTYPE, OBJSCHEMA, OBJNAME
```

38

https://www.ibm.com/docs/en/db2/11.5?topic=statements-create-usage-list

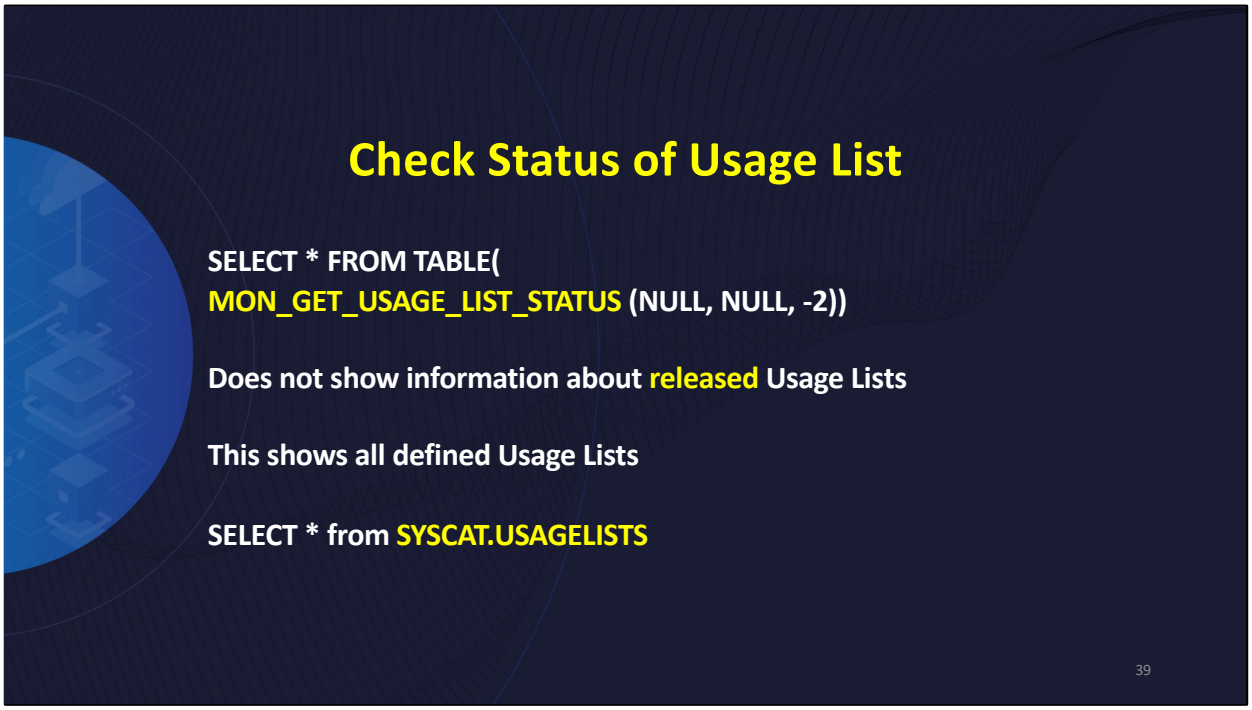
Usage list is a database object for monitoring all unique sections (DML statements) that have referenced a particular table or index during their execution.

Usage list can be for table or index

The states if a Usage List are:
Active
Inactive
Released Memory for it has been released

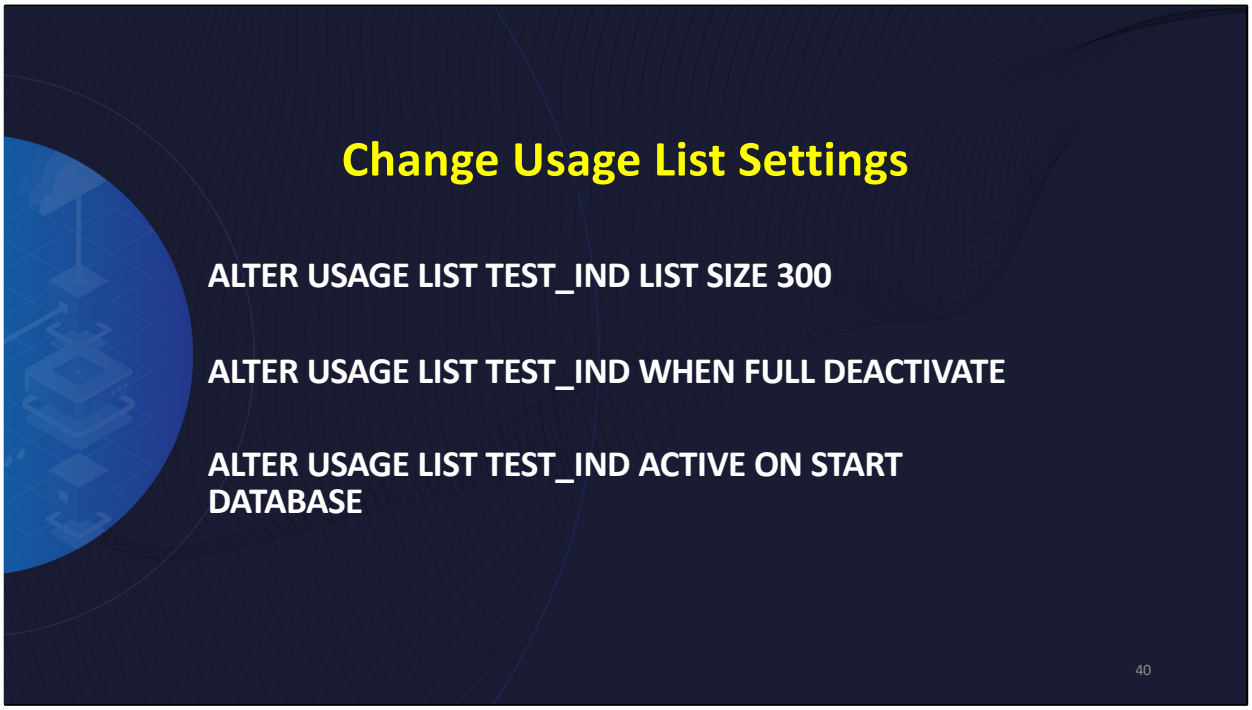
Create Usage List
https://www.ibm.com/docs/en/db2/11.5?topic=statements-create-usage-list

Activate (set) the Usage List
https://www.ibm.com/docs/en/db2/11.5?topic=statements-set-usage-list-state



The three parameters for **MON_GET_INDEX_USAGE_LIST** table function are:
usagelistschema - valid schema name. NULL (or null) return data from all schemas
usagelistname - valid usage list name. NULL (or null) return data for all Usage Lists
member - specify a valid member or -1 for current member or -2 for all members. If parameter is passed as NULL then -1 is used

https://www.ibm.com/docs/en/db2/11.5?topic=mpf-mon-get-index-usage-list-table-function-returns-information-from-index-usage-list



Can change
LIST SIZE
WHEN FULL
INACTIVE ON START DATABASE
ACTIVE ON START DATABASE

https://www.ibm.com/docs/en/db2/11.5?topic=statements-alter-usage-list

MON_GET_INDEX_USAGE_LIST

MON_GET_INDEX_USAGE_LIST (NULL, 'TEST_IND', -2)

usagelistschema
usagelistname
Member

This gives details about index usage

41

https://www.ibm.com/docs/en/db2/11.5?topic=mpf-mon-get-index-usage-list-table-function-returns-information-from-index-usage-list

MON_GET_INDEX_USAGE_LIST

Source: https://www.ibm.com/docs/en/db2/11.5?topic=mpf-mon-get-index-usage-list-table-function-returns-information-from-index-usage-list

```
SELECT * FROM TABLE(
MON_GET_INDEX_USAGE_LIST(NULL, 'USL_MON_IND1', -2))

USAGELISTSCHEMA    USAGELISTNAME    INDSCHEMA      INDNAME         MEMBER
------------------ ---------------- -------------- --------------- ------
ISAYYID            USL_MON_IND1     ISAYYID        I1              0
ISAYYID            USL_MON_IND1     ISAYYID        I1              0


DATA_PARTITION_ID
-----------------
                -
                -


EXECUTABLE_ID
--------------------------------------------------------------------------
x'0100000000000000490000000000000000000000200201107060938025770 65'
x'0100000000000000 4B0000000000000000000000200201107060938259815 48'


MON_INTERVAL_ID     LAST_UPDATED               NUM_REFERENCES
----------------------------------------------- --------------------
                    02011-07-06-09.38.15.881668                    1
                    02011-07-06-09.38.25.984147                    1
```

EXECUTABLE_ID which can be associated with particular SQL statements

42

https://www.ibm.com/docs/en/db2/11.5?topic=mpf-mon-get-index-usage-list-table-function-returns-information-from-index-usage-list

MON_GET_INDEX_USAGE_LIST

```
NUM_REF_WITH_METRICS OBJECT_INDEX_L_READSOBJECT_INDEX_P_READS
-------------------- ----------------------------------------
                   1                    1                   0
                   1                    1                   0


 OBJECT_INDEX_GBP_L_READS OBJECT_INDEX_GBP_P_READS
 ------------------------ ------------------------
                        0                        0
                        0                        0

OBJECT_INDEX_GBP_INVALID_PAGES OBJECT_INDEX_LBP_PAGES_FOUND
------------------------------ ---------------------------
                             0                           0
                             0                           0


2 record(s) selected.
```


https://www.ibm.com/docs/en/db2/11.5?topic=mpf-mon-get-index-usage-list-table-function-returns-information-from-index-usage-list

What SQL is using a particular index?

Way to determine what SQL is using a particular index

Combine MON_GET_INDEX_USAGE_LIST and MON_GET_PKG_CACHE_STMT

Use EXECUTABLE_ID to join the two

44

https://www.ibm.com/docs/en/db2/11.5?topic=mpf-mon-get-index-usage-list-table-function-returns-information-from-index-usage-list

https://www.ibm.com/docs/en/db2/11.5?topic=reference-i#r0058970

Get SQL Using Particular Index(es)

```
SELECT
   t.EXECUTABLE_ID,
   SUBSTR(t.STMT_TEXT,1,80),
   t.NUM_EXECUTIONS,  t.STMT_EXEC_TIME
 FROM
TABLE (MON_GET_PKG_CACHE_STMT (null,null,null,-2 )) as t,
TABLE(MON_GET_INDEX_USAGE_LIST (NULL, NULL, -2)) as i
WHERE t.EXECUTABLE_ID  = i.EXECUTABLE_ID
  ORDER by NUM_EXECUTIONS DESC;
```

45

Joins **MON_GET_PKG_CACHE_STMT** and **MON_GET_INDEX_USAGE_LIST** on the **EXECUTABLE_ID** column

This shows what SQL is using a particular index which can be very helpful when analyzing index usage.

Get SQL Using a Particular Index

```
SELECT *
  FROM TABLE(MON_GET_INDEX_USAGE_LIST (null,null,-2 ))
WHERE EXECUTABLE_ID =
x'00000001000000000000000000000038900000000000220120420073357611215'
;
```

If you already know EXECUTABLE_ID

46

The three parameters for **MON_GET_INDEX_USAGE_LIST** table function are:
usagelistschema - valid schema name. NULL (or null) return data from all schemas
usagelistname - valid usage list name. NULL (or null) return data for all Usage Lists
member - specify a valid member
or -1 for current member or -2 for all members. If parameter is passed as NULL then -1 is used

One way to get EXECUTABLE_ID is from MONREPORT.pkgcache details list

Real Example using SAMPLE database

Create SAMPLE database with db2sampl

```
select substr(INDSCHEMA,1,8) as INDSCHEMA, substr(INDNAME,1,40) as
INDNAME,
substr(colnames,1,70) as COLNAMES from syscat.indexes
where tabname in ('DEPARTMENT') order by INDNAME ASC ;

INDSCHEMA  INDNAME                                 COLNAMES
---------  --------------------------------------  ----------
DB2INST1   PK_DEPARTMENT                           +DEPTNO
DB2INST1   XDEPT2                                  +MGRNO
DB2INST1   XDEPT3                                  +ADMRDEPT
```

47

Real Example using DEPARTMENT Table

DEPTNO	DEPTNAME	MGRNO	ADMRDEPT	LOCATION
A00	SPIFFY COMPUTER SERVICE DIV.	000010	A00	-
B01	PLANNING	000020	A00	-
C01	INFORMATION CENTER	000030.	A00	-
D01	DEVELOPMENT CENTER	-	A00	-
D11	MANUFACTURING SYSTEMS	000060	D01	-
D21	ADMINISTRATION SYSTEMS	000070	D01	-
E01	SUPPORT SERVICES	000050	A00	-
E11	OPERATIONS	000090	E01	-
E21	SOFTWARE SUPPORT	000100	E01	-
F22	BRANCH OFFICE F2	-	E01	-
G22	BRANCH OFFICE G2	-	E01	-
H22	BRANCH OFFICE H2	-	E01	-
I22	BRANCH OFFICE I2	-	E01	-
J22	BRANCH OFFICE J2	-	E01	-

Indexes for the Department Table

INDEX NAME	Columns in index
PK_DEPARTMENT	+DEPTNO
XDEPT2	+MGRNO
XDEPT3	+ADMRDEPT

Usage Lists for tracking index usage

```
CREATE USAGE LIST PK_DEPARTMENT FOR INDEX DB2INST1.PK_DEPARTMENT LIST SIZE 500
WHEN FULL WRAP;
CREATE USAGE LIST XDEPT2 FOR INDEX DB2INST1.XDEPT2  LIST SIZE 500 WHEN FULL WRAP;
CREATE USAGE LIST XDEPT3 FOR INDEX DB2INST1.XDEPT3  LIST SIZE 500 WHEN FULL WRAP;
SET USAGE LIST PK_DEPARTMENT STATE ACTIVE;
SET USAGE LIST XDEPT2 STATE ACTIVE;
SET USAGE LIST XDEPT3 STATE ACTIVE;
```

Indexes for the Department Table

Queries to put data in Package Cache and into Usage lists

```
select * from DEPARTMENT WHERE DEPTNO = 'A00';
select * from DEPARTMENT WHERE MGRNO = '000090';
select * from DEPARTMENT WHERE ADMRDEPT = 'E01';
```

50

Query to see Index Usage Tied to SQL

```
SELECT
SUBSTR(t.STMT_TEXT,1,80) AS STMT,
SUBSTR(i.indschema,1,20) as INDSCHEMA,
   SUBSTR(i.indname,1,30) as INDNAME
FROM
TABLE (MON_GET_PKG_CACHE_STMT(null,null,null,-2 )) as t,
TABLE(MON_GET_INDEX_USAGE_LIST(NULL, NULL, -2)) as i
WHERE t.EXECUTABLE_ID  = i.EXECUTABLE_ID
  ORDER by NUM_EXECUTIONS DESC;
```

51

SQL Tied to Indexes

STMT	INDSCHEMA	INDNAME
select * from DEPARTMENT WHERE MGRNO = '000090'	DB2INST1	XDEPT2
select * from DEPARTMENT WHERE ADMRDEPT = 'E01'	DB2INST1	XDEPT3
select * from DEPARTMENT WHERE DEPTNO = 'A00'	DB2INST1	PK_DEPARTMENT

3 record(s) selected.

Agenda

- Exploring ways to capture EXPLAIN data
- Using EXPLAIN_FORMAT stored procedure
- Using EXPLAIN_FORMAT_STATS function
- Using other EXPLAIN functions
- How to combine data from tools including db2batch and index usage data

Paul Turpin manages a database engineering team for a financial services company. He specializes in Db2 for LUW on large systems, along with other DBMS like PostgreSQL, Neo4j and MongoDB. He enjoys exploring new features and functionality in Db2. He has spoken at IDUG North America, IDUG Europe, SHARE, IBM's Information on Demand conference, and several RUGs. Paul served on the IDUG Board of Directors and is a Past President of IDUG.

Db2 11.5 Manuals

https://www.ibm.com/support/pages/db2-version-115-linux-unix-and-windows-english-manuals