PDUG

Gdańsk, 2-3 czerwca 2026

MAKING YOUR COLUMNAR DB MORE ROBUST

monitoring, thresholds and access plan analysis

Kamil Kuduk

IBM Polska
kamil.kuduk@pl.ibm.com

POLISH DB2 USERS GROUP

AGENDA

PDUG

1. DB2 COLUMNAR DATA ENGINE (CDE/BLU): OVERVIEW
2. CDE: DATA ORGANIZATION
3. DB2 WLM: OVERVIEW
4. WLM: WORKLOADS
5. WLM: SERVICE CLASSES
6. WLM: THRESHOLDS
7. ADAPTIVE WLM
8. MONITORING: MON_GET_ACTIVITY
9. EVENT MONITOR FOR ACTIVITIES
10. ACCESS PLAN ANALYSIS
11. SECTION ACTUALS

IBM DB2

DB2 COLUMNAR DATA ENGINE (CDE/BLU): OVERVIEW

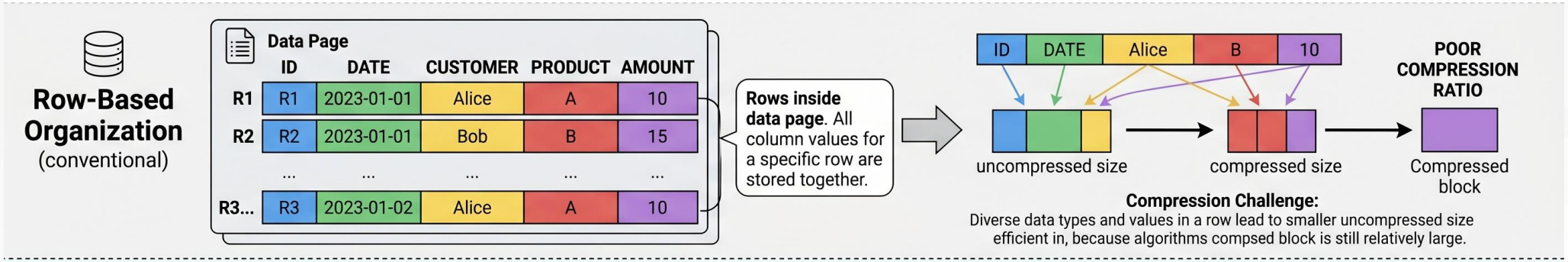
- Designed for analytics queries
- Massively intra-parallel
- CPU/memory optimized (SIMD)
- Column-organized
- Use the same storage (table spaces)
- Can coexist and be joined with row-organized tables

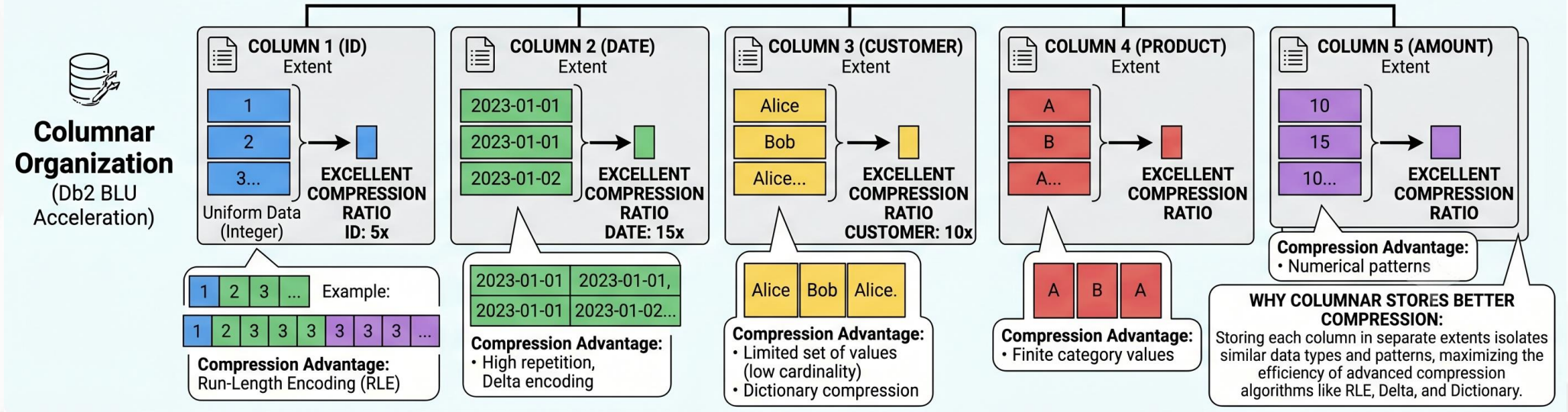
CDE: DATA ORGANIZATION

DB2 DATA ORGANIZATION: ROW VS. COLUMNAR

Logical Table

	ID	DATE	CUSTOMER	PRODUCT	AMOUNT
R1	R1	2023-01-01	Alice	A	10
R2	R2	2023-01-01	Bob	B	15
R3...	...	2023-01-02	Alice	A	10





DB2 WLM: OVERVIEW

Goals of Db2 WLM:

- Workload identification
- Governance and monitoring
- Prioritization / Isolation
- Ensure system stability and responsiveness

DB2 WLM: WORKLOADS

PDUG

- Workloads are the primary mechanism for **identifying and classifying** incoming database activities.
- You define a workload based on various attributes of the connection or the activity itself, such as:
  - Application name
  - Session user
  - Client workstation name/IP address
  - Client accounting string
- Once an activity matches a defined workload, it is assigned to a specific service class for execution.

DB2 WLM: WORKLOADS

PDUG

- Example:

```
$ db2 "CREATE SERVICE CLASS SC_CLP_BATCH"
$ db2 "CREATE WORKLOAD WL_CLP_BATCH CURRENT CLIENT_APPLNAME('CLP batch.db2') SERVICE
CLASS SC_CLP_BATCH"
$ db2 "GRANT USAGE ON WORKLOAD WL_CLP_BATCH TO PUBLIC"

$ db2 "CREATE SERVICE CLASS SC_CLP_ALL"
$ db2 "CREATE WORKLOAD WL_CLP_ALL APPLNAME('db2bp') SERVICE CLASS SC_CLP_ALL"
$ db2 "GRANT USAGE ON WORKLOAD WL_CLP_ALL TO PUBLIC"

$ db2 "select substr(workload_name,1,20) from
table(MON_GET_AGENT('','',mon_get_application_handle(),-1))"

1
--------------------
WL_CLP_ALL
```

DB2 WLM: SERVICE CLASSES

- **Service classes** define the **execution environment** and resource allocation for activities.
- They are hierarchical:

  - Service Superclasses: High-level buckets (e.g., SYSDEFAULTSYSTEMCLASS)

  - Service Subclasses: Smaller buckets within a superclass where activities run
- Within a service class, you can control, CPU limits/shares, prefetch priority, bufferpool priority (SORT only in AWLM)

DB2 WLM: SERVICE CLASSES

- An example of a service class. CPU shares can be SOFT or HARD, default SC has 10000 shares assigned

```
$ db2 "CREATE SERVICE CLASS SC_CLP_BATCH
    CPU SHARES 10000
    PREFETCH PRIORITY HIGH
    BUFFERPOOL PRIORITY HIGH"
$ db2 "CREATE WORKLOAD WL_CLP_BATCH CURRENT CLIENT_APPLNAME('CLP batch.db2') SERVICE
CLASS SC_CLP_BATCH"
$ db2 "GRANT USAGE ON WORKLOAD WL_CLP_BATCH TO PUBLIC"
```

DB2 WLM: THRESHOLDS

PDUG

- Define limits and actions that are taken when certain resource consumption or activity characteristics are exceeded
- Can be of 3 types:
  - predictive (SQL cost)
  - reactive (activity time, rows read, connection idle time)
  - concurrency
- Can be applied on different levels: database, workload, service class
- Threshold violation will enforce certain actions, e.g. stopping the activity, queuing, remap or collecting activity data

DB2 WLM: THRESHOLDS

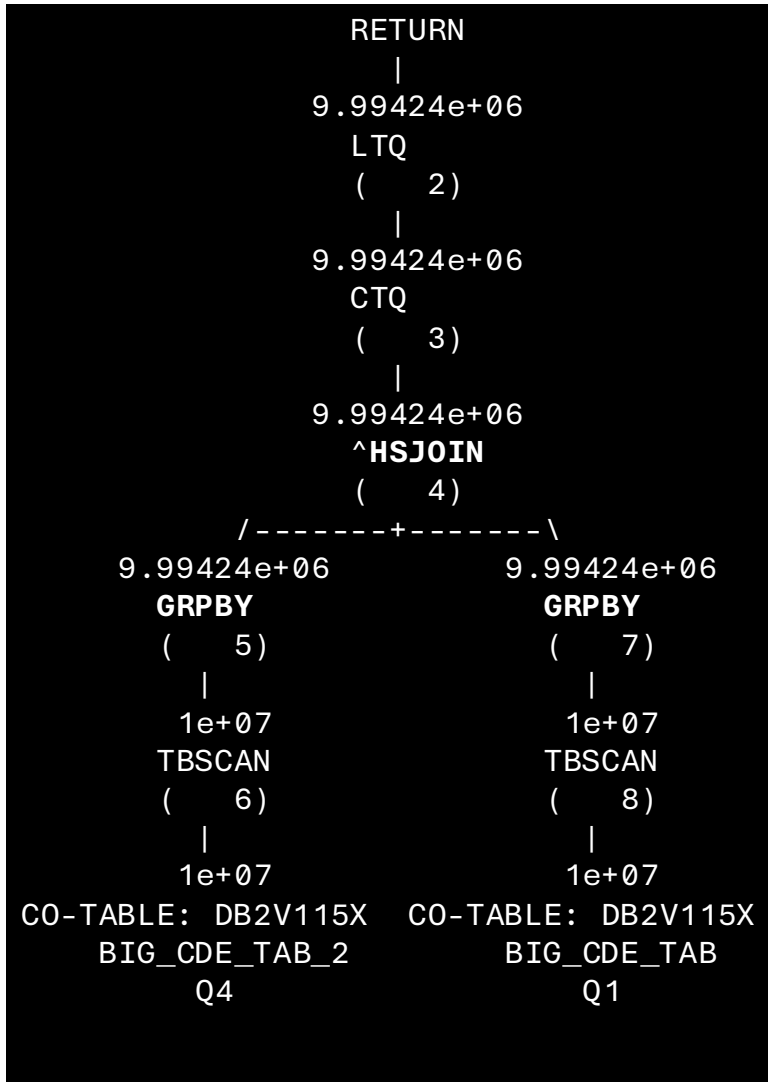
- Example usage:

```
db2 "CREATE THRESHOLD TH_COST_STOP FOR DATABASE ACTIVITIES ENFORCEMENT DATABASE WHEN
ESTIMATEDSQLCOST > 10000000 STOP EXECUTION"

db2 "CREATE THRESHOLD TH_LONG_ACT FOR DATABASE ACTIVITIES ENFORCEMENT DATABASE WHEN
ACTIVITYTOTALTIME > 2 HOURS COLLECT ACTIVITY DATA WITH DETAILS CONTINUE"

db2 "CREATE THRESHOLD TH_CONCURRENT FOR DATABASE ACTIVITIES ENFORCEMENT DATABASE
WHEN CONCURRENTDBCOORDACTIVITIES > 5 STOP EXECUTION"
```

DB2: ADAPTIVE WLM

- Adjusts concurrency implicitly based on workload
- Intelligent job scheduling to utilize resources well
- Resources controlled:
  - SORT
  - CPU / threads
  - UTILITY HEAP
- Incorporates feedback (exec time, actual resource usage) from earlier executions (based on the stmtid+planid)
- Enabled via db cfg WLM_ADMISSION_CTRL or WLM_ENABLE_ADMISSION_CTRL procedure (effective on the next activation)

DB2: ADAPTIVE WLM – EXAMPLE QUERY




```
select * from
  (select c1, max(c2)
    from big_cde_tab
      group by c1) a,
  (select c1, min(c2)
    from big_cde_tab_2
      group by c1) b
where a.c1 = b.c1
```

DB2: ADAPTIVE WLM

- Includes 3 pre-defined workload types that affect queue type
  - BATCH (longer jobs, FIFO queue)
  - INTERACTIVE (response sensitive jobs, latency queue)
  - MIXED (combination of the two above)
- Implementation:
  - Create service classes and assign resources (SOFT or HARD) to divide resources available, with the desired workload type
  - Create workloads for the applications/user and assign them to service classes

DB2: ADAPTIVE WLM – SERVICE CLASSES

PDUG

- Example implementation

```
db2 "create service class HIPRI soft resource shares 2000 for workload type
INTERACTIVE minimum resource share 50 percent"
db2 "create service class ETL soft resource shares 2000 for workload type BATCH"
db2 "create service class GENERAL soft resource shares 5000 for workload type MIXED"

db2 "create workload REPORTS session_user('EDW_REPORTS') service class HIPRI"
db2 "create workload ETLJOBS session_user('EDW_ETL_USER') service class ETL"
db2 "alter workload SYSDEFAULTUSERWORKLOAD service class GENERAL"
```

Total: 2000+2000+5000+1000 (default class) = 10000

WLM: SECURING YOUR SYSTEM

- Map your applications to workloads, even if you do not take any actions: it helps to monitor the activities
- If SLAs are not met, use service classes to ensure important workloads get required resources (via service classes with minimum resources)
- If queuing is an issue, even with service classes, use SESSION PRIORITY
- Block all high temp users (WHEN SQLTEMPSPACE > 100 G) in all non-essential workloads via thresholds
- Limit SORT for via Service Class (SORTMEM LIMIT <PCT_OF_SHEAPTHRES_SHR>) for ad-hoc queries
- Collect access plan and metrics for all heavy queries

MONITORING: MON_GET_ACTIVITY

MON_GET_ACTIVITY information about each query in the database

- APPLICATION_HANDLE: Application that submitted the query
- SESSION_AUTH_ID: Authorization ID of user that submitted the query
- SORT_SHRHEAP_ALLOCATED: Current amount of shared sort memory in use by the query
- SORT_SHRHEAP_TOP: Peak amount of shared sort memory in used by the query
- ESTIMATED_SORT_SHRHEAP_TOP: Estimated peak sort memory usage for the query
- EFFECTIVE_QUERY_DEGREE: Query degree; counted against agent load target
- QUERY_COST_ESTIMATED: Estimated cost
- ADM_RESOURCE_ACTUALS: Indicates whether or not the memory estimate is based on past executions
- ACTIVITY_STATE: State of the query; if the query is executing, queued or idle
- ADM_BYPASSED: Indicates whether the query bypassed admission control
- STMT_TEXT – Query statement text

MONITORING: EVENT MONITOR FOR ACTIVTIES

- Make sure you do not collect activity data for default workload

```
db2pd -db bludb -wlm

ALTER WORKLOAD "SYSDEFAULTUSERWORKLOAD"
ALLOW DB ACCESS
COLLECT ACTIVITY DATA NONE
COLLECT AGGREGATE ACTIVITY DATA NONE
COLLECT AGGREGATE UNIT OF WORK DATA NONE
COLLECT ACTIVITY METRICS NONE
COLLECT UNIT OF WORK DATA NONE
COLLECT LOCK TIMEOUT DATA WITHOUT HISTORY
COLLECT DEADLOCK DATA WITHOUT HISTORY
COLLECT LOCK WAIT DATA NONE
```

MONITORING: EVENT MONITOR FOR ACTIVTIES

- Create event monitor in table space on all partitions

```
db2 "create tablespace wlm_ts in ibmdefaultgroup"

db2 "create event monitor act_mon for activities write to table
    activity (table act_mon_activity in wlm_ts),
    activitymetrics (table act_mon_metrics in wlm_ts),
    activitystmt (table act_mon_stmt in wlm_ts),
    activityvals (table act_mon_vals in wlm_ts)
    manualstart"

db2 "set event monitor act_mon state 1"
```

MONITORING: EVENT MONITOR FOR ACTIVTIES

- Use it only when you need it (set state 0/1) and/or use workload

```
db2 "create workload perf_debug current client_acctng('perf_debug') collect
activity data with details,section include actuals base and values"
# run query
db2 "select count(*) from act_mon_activity"
1
-------------------------------
                            0.

db2 "CALL SYSPROC.WLM_SET_CLIENT_INFO(NULL, NULL, NULL, 'perf_debug', NULL)"
# run query and check monitor data (from a different connection)
db2 "select count(*) from act_mon_activity"

1
-------------------------------
                            1.
```

MONITORING: EVENT MONITOR FOR ACTIVTIES

- Enable activity data collection ad-hoc for the current connection

```
db2 "CALL WLM_SET_CONN_ENV(NULL,'<collectactdata>WITH DETAILS,
SECTION</collectactdata>
<collectsectionactuals>BASE</collectsectionactuals>')"
```

MONITORING: EVENT MONITOR FOR ACTIVTIES

- Collect the data automatically for all long queries

```
db2 "CREATE THRESHOLD LONG_QUERY  FOR DATABASE
    WHEN ACTIVITYTOTALTIME  > 30 seconds
    COLLECT ACTIVITY DATA  ON ALL MEMBERS WITH DETAILS, SECTION AND VALUES
    CONTINUE"

db2 "CREATE THRESHOLD TOO_LONG_QUERY
    FOR WORKLOAD AD_HOC WHEN ACTIVITYTOTALTIME  > 30 minutes
    COLLECT ACTIVITY DATA  ON ALL MEMBERS WITH DETAILS, SECTION AND VALUES
    STOP EXECUTION"
```

MONITORING: EVENT MONITOR FOR ACTIVTIES

- Get the details and explain

```
db2 "select M.EVENT_TIMESTAMP, M.UOW_ID,
M.ACTIVITY_ID,  substr(M.APPL_ID,1,36) as appl_id, M.TOTAL_ACT_TIME,
length(SECTION_ENV) sec_len, substr(S.STMT_TEXT,1,20) STMT from act_mon_stmt
S, act_mon_metrics M WHERE S.ACTIVITY_ID = M.ACTIVITY_ID AND S.UOW_ID =
M.UOW_ID AND S.APPL_ID = M.APPL_ID ORDER BY  TOTAL_ACT_TIME DESC"


EVENT_TIMESTAMP: 2026-06-03-07.45.46.453789
UOW_ID: 1
ACTIVITY_ID: 8
APPL_ID: *N0.db2v121w.260603052607
TOTAL_ACT_TIME: 70939
SEC_LEN: 44056
STMT: SELECT ST.regio[...]
```

MONITORING: EVENT MONITOR FOR ACTIVTIES

- Get the details and explain

```
db2 "CALL EXPLAIN_FROM_ACTIVITY( '*N0.db2v121w.260603052607', 8, 1,
'ACT_MON', '', ?, ?, ?, ?, ? )"

  Value of output parameters
  --------------------------
  Parameter Name  : EXPLAIN_SCHEMA
  Parameter Value : SYSTOOLS

  Parameter Name  : EXPLAIN_REQUESTER
  Parameter Value : DB2V121W

  Parameter Name  : EXPLAIN_TIME
  Parameter Value : 2026-06-03-07.46.46.486734
```

MONITORING: THRESHOLD VIOLATIONS

```
db2 "create event monitor thresh_violation_mon for  threshold violations
write to table
    thresholdviolations (table thresh_violations in wlm_ts)
    autostart"
db2 "set event monitor thresh_violation_mon state 1"

db2 "select TIME_OF_VIOLATION, substr(APPLICATION_NAME,1,20) app_name,
substr(SESSION_AUTH_ID,1,20) as auth_id, THRESHOLDID, THRESHOLD_ACTION,
THRESHOLD_PREDICATE from thresh_violations"

TIME_OF_VIOLATION   APP_NAME             AUTH_ID
------------------- -------------------- ----------------
2026-06-03-10.01.12 db2bp                DB2V121W

THRESHOLDID THRESHOLD_ACTION THRESHOLD_PREDICATE
----------- --------------- ----------------------
          1 Continue        ActivityTotalTime
```

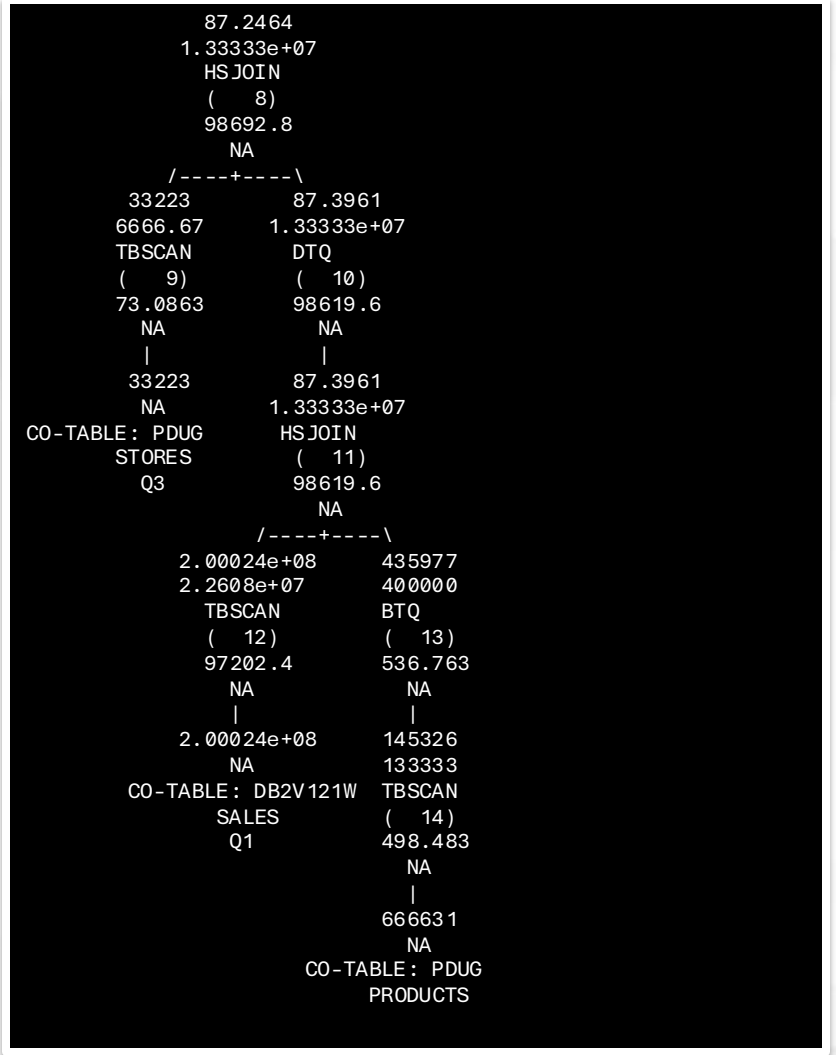
ACCESS PLAN ANALYSIS

- Get the details and explain

```
Explain level:    Explain from section
                       1.72458e+08
                         HSJOIN
                         (   8)
                          62697
                            NA
                  /-------+-------\
            1.66346e+08           100722
              CTQ                 BTQ
              (   9)              (  14)
               59325              103.413
                 NA                 NA
                 |                  |
            1.66346e+08            33574
              HSJOIN              LTQ
              (  10)              (  15)
              58933.5             94.4985
                NA                  NA
           /----+----\              |
      1.66631e+08      99669        33574
        TBSCAN         BTQ          TBSCAN
        (  11)         (  12)       (  16)
        58279.7        81.8437      92.3601
          NA             NA           NA
          |              |            |
      1.66631e+08      33223        33574
CO-TABLE: PDUG       TBSCAN    TABLE: PDUG
        SALES        (  13)      STORE_LEASES
          Q3         73.0863        Q2
                       NA
                       |
                     33223
                CO-TABLE: PDUG
                     STORES
                       Q
```

ACCESS PLAN ANALYSIS: ACTUALS

- actuals works now with CDE (12.1)
- can be collected using db2caem
- or via workload / WLM
- look for HSJOINs and aggregations
- `db2caem -d <db_name> -sf q.sql`



ACCESS PLAN ANALYSIS: ACTUALS

- Db2caem works now with CDE

```
db2caem

db2caem (DB2 Capture Activity Event Monitor data tool)
-------------------------------------------------------------------------
Syntax: db2caem [ -d <database_name> ][ -u <user_id>  -p <password> ]
        [ -tn <tenant_name> ]
        [ query-statement-options | event-monitor-options ]
        [ -o <output_path> ][ -h ]

query-statement-options:
  [ -st <query_statement> | -sf <query_statement_file> ]
  [[ -compenv <compilation_env_file> ] |
   [ -cs <current_schema> ][ -fp <function_path> ]]
  [ -tbspname <table_space_name> ]
  [ -terminator <termination_character> ]

event-monitor-options:
  [ -actevm <event_monitor_name> -appid <application_id> -uowid <uow_id>
    -actid <activity_id> ]
```