Troubleshooting Db2 Slowdown

Bharat Goyal

Db2 Support

bharat.goyal@ibm.com

CCDUG 2024

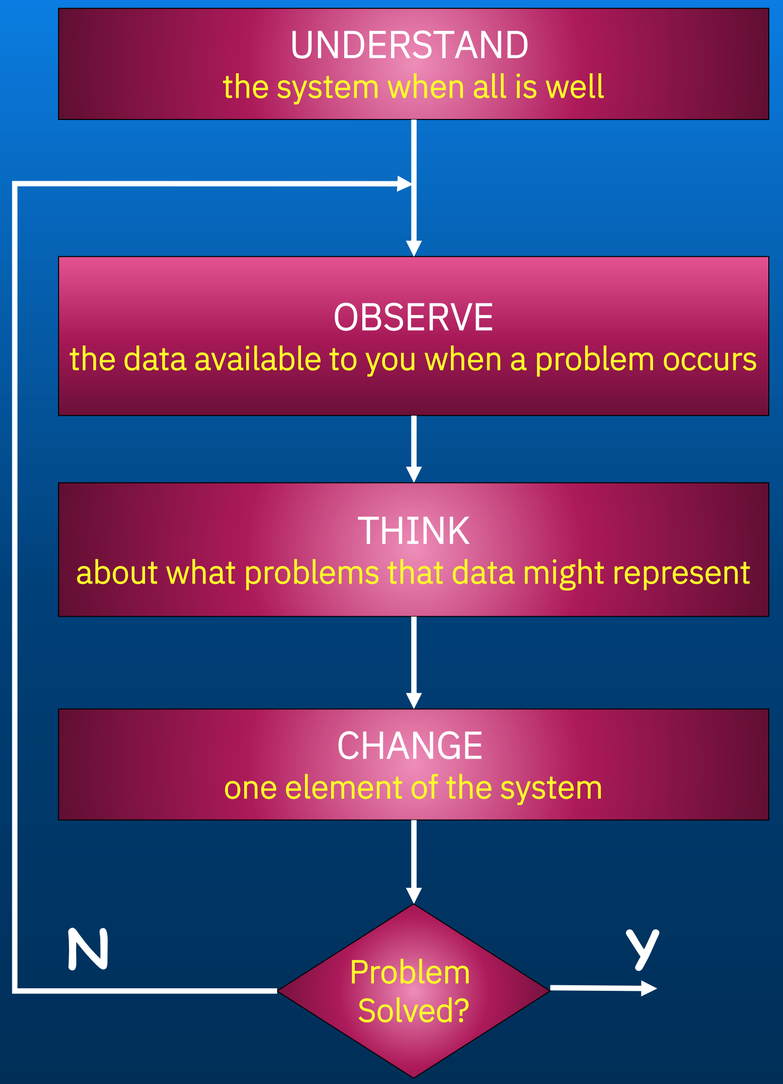
IBM

Agenda

- Introduction
- Data Collection Tools
- Common Bufferpool Latches
- Case Study - Select workload scalability

What is Performance?

- Performance is often evaluated by comparing its speed or efficiency against a previously tested or benchmarked runtime reference.
- We call something slow when we know it's running faster in another place or in another context.
- Tuning Technique: Methodical ‘decision tree’ approach



Data Collection Tools

- **DB2MON** - Uses Db2 ***lightweight*** *in-memory monitoring interfaces (*"MON_GET_*"*)* to collect monitoring data for a specific period.

```
~/sqllib/samples/perf/db2mon.sh DBNAME <Interval> >> db2mon.out
```

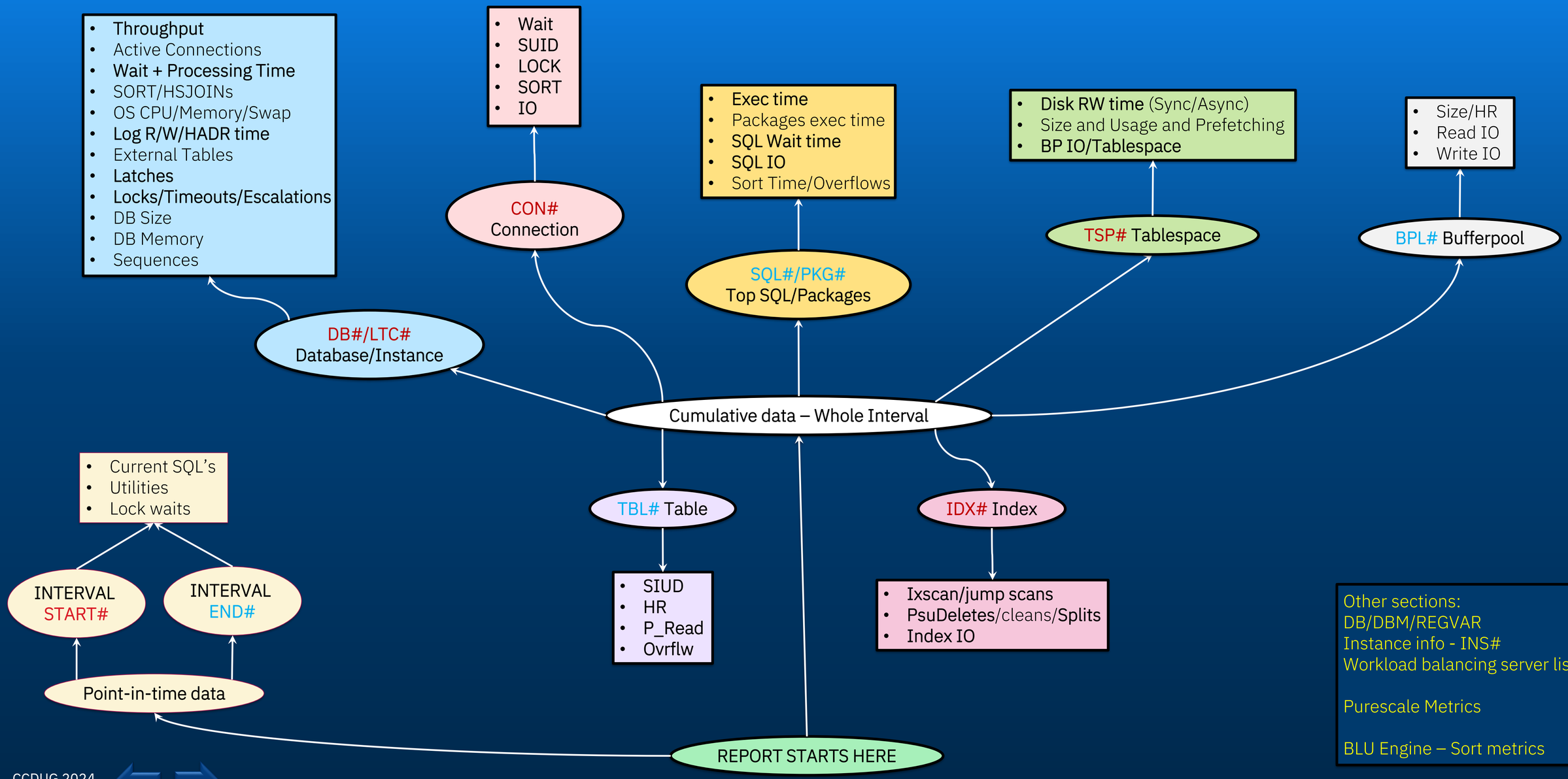
- **DMCTOP** - It works in a text-only environment. Monitoring is accomplished by using MON_GET_* table functions.

- **DB2PD** - It returns quick and immediate information from Db2 memory sets. It gathers information without acquiring latches or using engine resources.

- **EXPLAIN PLAN** - Optimizer query execution plan

```
SELECT COORD_PARTITION_NUM, EXECUTABLE_ID, STMT_TEXT FROM TABLE(MON_GET_ACTIVITY(NULL,-2))
WHERE MEMBER=COORD_PARTITION_NUM
AND SUBSTR(STMT_TEXT,1,50) LIKE '%INSERT INTO DETAIL_SUPPORT%'
OR APPLICATION_HANDLE = 1234

CALL EXPLAIN_FROM_SECTION(EXECUTABLE_ID,'M', NULL,COORD_PARTITION_NUM,NULL,?,?,?,?,?)

db2exfmt -d DBNAME  -1 -o query.exfmt
```

Db2mon



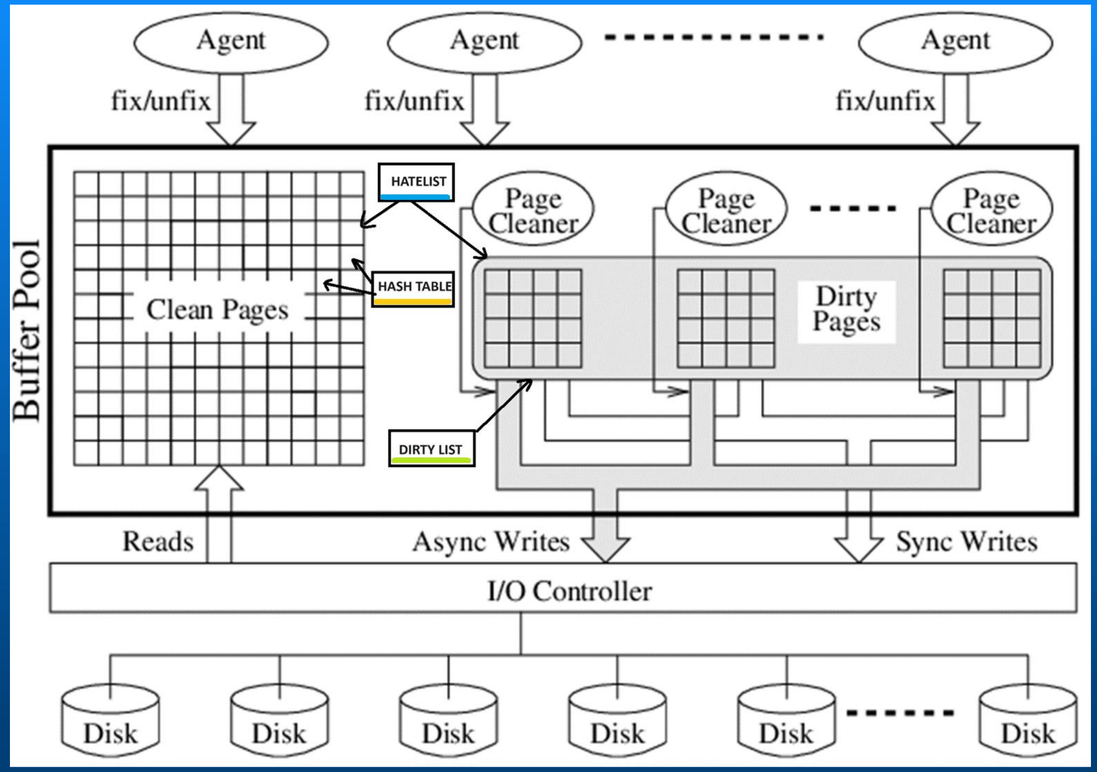
Other sections:
DB/DBM/REGVAR
Instance info - INS#
Workload balancing server list: WLB#

Purescale Metrics

BLU Engine – Sort metrics

Common Bufferpool Latches

- Internal DB2 locks used to serialize access to shared data structures.
- They ensure only one EDU can modify a data structure at a time.
- Frequent heavy usage of a shared resource leads to latch contention, possibly increasing CPU usage.



Read/Write pages

- SQLO_LT_SQLB_HASH_BUCKET_GROUP_HEADER__grouplatch
  Latch is held While searching the hash bucket for a specific page.

- SQLO_LT_SQLB_BPD__bpdLatch_SX
  While doing physical read/write of page in/from bufferpool and for fix/unfix.

Dirty List

- SQLO_LT_SQLB_DIRTY_LIST_SET__appendLatch
  Add/Remove pages to/from dirty list. Ex – Delete pages from dirty list when object is dropped example temp table.

- SQLO_LT_SQLB_DIRTY_LIST_SET__walkLatch
  Contention on this latch suggests many dirty pages. This could be due to obstacles preventing the pages from being written out, or repeated operations requiring bufferpool scans to evict pages (Often caused by creating/dropping DGTTs, requiring page eviction from DIRTYLIST).

Victimization

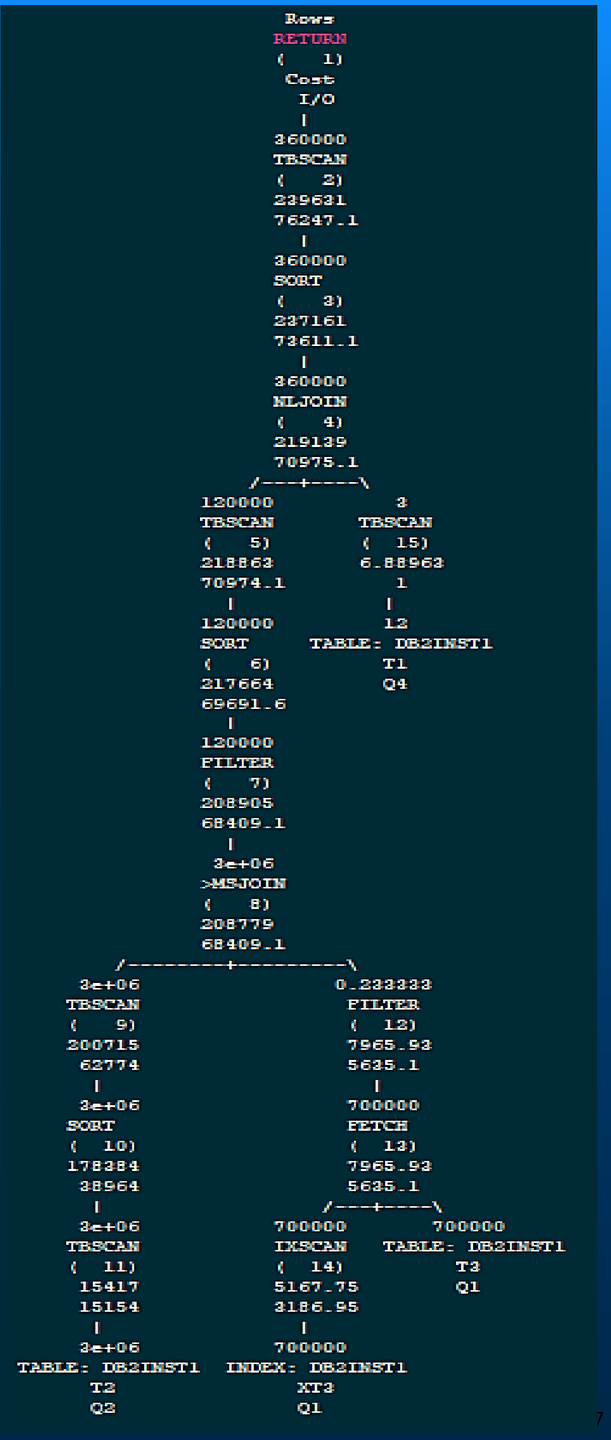
- SQLO_LT_SQLB_HATELIST_SET__hateLatch
  Hatelist is updated based on fix/unfix and when objects are dropped pages needs to be removed from hate list. Ex drop temp tables.

- SQLO_LT_SQLB_BP_AREA_INFO_pageArea__clockLatch
  Contention on this latch suggest a high victim demand (new pages being brought into the bufferpool), which suggests a combination of too-small-bufferpool and/or bad workload (e.g. tablescans, Sort spill → Temp tables).

Case Study - Select workload scalability

- Workload: Simple query with join with 3 tables.
- Java program to scale the workload with X threads
- 1 Thread – Query completes in 3-4 seconds (within SLA)

```
TABLE T1(IND CHAR, IS_ON BOOLEAN, EXP_DATE DATE)
TABLE T2(PRODUCER_ID CHAR(20), MBR_ID CHAR(20), PHONENO CHAR(20), IND CHAR, IS_ON BOOLEAN, EXP_DATE DATE)
TABLE T3(PRODUCER_ID CHAR(20), MBR_ID CHAR(20), EXP_DATE DATE)

SELECT *
FROM
( SELECT T2.*, T3.PRODUCER_ID AS PID
  FROM T1 T1, T2 T2
  LEFT OUTER JOIN T3 T3
  ON  T3.MBR_ID   = T2.MBR_ID
  AND T3.EXP_DATE = T2.EXP_DATE
  WHERE T1.IS_ON  = 1
    AND T2.IND    = COALESCE(T1.IND,'')
  ORDER BY T2.EXP_DATE, T3.PRODUCER_ID, T2.PRODUCER_ID
) AS T1
WHERE COALESCE(T1.PRODUCER_ID,T1.PID) = ?
ORDER BY 1,2
```



Case Study - Select workload scalability

```
^Cdb2inst1@IBM-PW03R8VZ:~/CCDUG$ java stmtwatcher
SQL: /*STMTWATCHER-2024-03-08-17.52.54.918*/ select * from ( select t2.*,T3.PRODUCER_ID as PID from t1 t1, t2 t2 LEFT OUTER JOIN t3 t3 ON t3.MBR_ID = T2.MBR_ID and T3.EXP_DATE = T2.EXP_DATE where t1.IS_ON = 1 AND  (t2.IND = COALESCE
(t1.IND, '')) ORDER BY T2.EXP_DATE, T3.PRODUCER_ID, T2.PRODUCER_ID ) as T1   where (COALESCE(T1.PRODUCER_ID, T1.PID ) = ? ) ORDER BY 1,2
num of parameters = 1
param[0]: PRODUCER_ID1001047
Number of threads = 1
Serial execution time 3809 millisecs
Plan id = -7263397770068094435

db2 "call explain_from_section(x'0100000000000000050000000000000000000000020020240308175255104389', 'M', null, -1, '', ?,?,?,?,? )" ; db2exfmt -d CCDUG -l > STMTWATCHER-2024-03-08-17.52.54.918.exfmt
```

msec	#execs	#ltchwaits	ltchwaittime	Cord_msec/exec	Act_msec/exec	waittm/exec	%processing
7	1	0	0	3785.000	3785.000	0.225	77.464
10007	2	3	3	4030.000	4030.000	0.222	77.792
5002	1	0	0	3898.000	3898.000	0.225	77.527
5002	1	0	0	3856.000	3856.000	0.213	78.734
5002	2	0	0	4045.500	4045.500	0.216	78.408
5003	1	0	0	4019.000	4019.000	0.242	75.765
5003	1	0	0	4072.000	4072.000	0.201	79.912
5003	1	0	0	4223.000	4223.000	0.228	77.220
5002	1	0	0	4024.000	4024.000	0.229	77.137

Scaling to 30 Threads

```
db2inst1@IBM-PW03R8VZ:~/CCDUG$ java stmtwatcher
SQL: /*STMTWATCHER-2024-03-08-17.58.36.909*/ select * from ( select t2.*,T3.PRODUCER_ID as PID from t1 t1, t2 t2 LEFT OUTER JOIN t3 t3 ON t3.MBR_ID = T2.MBR_ID and T3.EXP_DATE = T2.EXP_DATE where t1.IS_ON = 1 AND  (t2.IND = COALESCE
(t1.IND, '')) ORDER BY T2.EXP_DATE, T3.PRODUCER_ID, T2.PRODUCER_ID ) as T1   where (COALESCE(T1.PRODUCER_ID, T1.PID ) = ? ) ORDER BY 1,2
num of parameters = 1
param[0]: PRODUCER_ID1001047
Number of threads = 30
Serial execution time 3841 millisecs
Plan id = -7263397770068094435

db2 "call explain_from_section(x'0100000000000005E0000000000000000000000020020240308175837099321', 'M', null, -1, '', ?,?,?,?,? )" ; db2exfmt -d CCDUG -l > STMTWATCHER-2024-03-08-17.58.36.909.exfmt
```

msec	#execs	#ltchwaits	ltchwaittime	Cord_msec/exec	Act_msec/exec	waittm/exec	%processing
4	14	3	3	3874.428	3874.428	0.217	78.284
20039	1	326	2658	18651.000	18651.000	0.419	58.147
10010	2	1499	12009	26734.000	26734.000	0.497	50.275
10011	25	37878	300426	36834.559	36834.559	0.425	57.458
5019	2	2279	24499	41923.000	41923.000	0.471	52.921
15030	3	2476	16712	32411.334	32411.334	0.419	58.094

~/sqllib/samples/perf/db2mon.sh CCDUG 30 >> db2mon.out

Scaling to 30 Threads - Analysis

```
DB#THRUP
TS_DELTA ACT_PER_S ROWS_MOD_P_S P_RD_PER_S
-------- --------- ------------ ----------
      31       2.3    3021999.2    36049.4

DB#CLACT
ACTIVE_CLIENTS ACTIVE_RQ_PER_S ACTIVE_CIWT_RQ_RATIO
-------------- --------------- --------------------
            32           10.87                 0.04

DB#TIMEB
TOTAL_RQST_TM PCT_SECTION PCT_SORT
------------- ----------- --------
       909516       99.77    75.04

DB#WAITT
TOTAL_RQST_TM TOTAL_WAIT_TM PCT_RQST_WAIT PCT_LTCH PCT_PFTCH PCT_POOL_R
------------- ------------- ------------- -------- --------- ----------
       909516        474607         52.18    33.76      6.68       9.11

DB#SORT
TOTAL_SORTS SORT_OVERFLOWS SORT_SHRHEAP_ALLOCATED
----------- -------------- ----------------------
         75             15                   4846
```

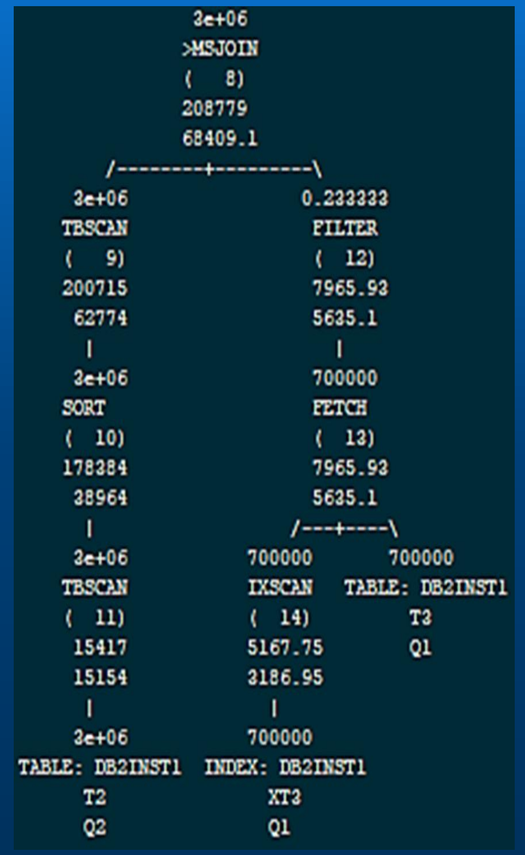
```
SQL#TOPEXECT
NUM_EXEC    AVG_COORD_EXEC_TIME PCT_WAIT_TIME   STMT_TEXT
----------- ------------------- --------------- ------------------
         14            47258.07           49.12 /*STMTWATCHER-2024

SQL#TOPWAITT
PCT_WAIT PCT_LTCH PCT_PFTCH PCT_POOL_R STMT_TEXT
-------- -------- --------- ---------- ------------------
   49.12    28.27      7.97       8.87 /*STMTWATCHER-2024

SQL#TOPIOSTA
AVG_D_PRD    AVG_I_PRD    AVG_TD_PRD   STMT_TEXT
------------ ------------ ------------ ---------
     1970.0        952.8      43866.0 /*STMTWAT

SQL#TOPROWS
AVG_ROWS_MOD   AVG_ROWS_READ  AVG_ROWS_RET   STMT_TEXT
-------------- -------------- -------------- -------------
    5357142.8      9057154.8            2.0 /*STMTWATCHER

SQL#TOPSORT
PCT_SORT_TIME   AVG_TOT_SORTS AVG_SORT_OVFLWS ACTIVE_SORTS_TOP STMT_TEXT
--------------- ------------- --------------- ---------------- -----------
          71.82           3.0             1.0                2 /*STMTWATCH
```

Scaling to 30 Threads – Analysis cont.

```
TBL#ROWMC
TABNAME  TBSP_ID ROWS_READ OBJ_DATA_P_RDS
-------- ------- --------- --------------
T2             3 24029976           1809
T3             3 16527048          32703

IDX#READS
TABNAME  INDNAME L_READS  P_READS
-------- ------- -------- -------
T3       XT3     81038    19908
```

```
BPL#STATS
TBSP_NAME  DATA_L_READS DATA_P_READS SYNC_DATA_P_READS INDEX_L_READS INDEX_P_READS
---------- ------------ ------------ ----------------- ------------ --------------
XTS16K          12843887        74323             49633       152296          79708
TEMPSPACE1       9650146      1826218           1124845            0              0

BP_NAME      ROW_DATA_LBP_HITRATIO INDEX_LBP_HITRATIO
------------ --------------------- ------------------
XBP16K                       99.61              78.96
IBMDEFAULTBP                 88.35              92.26
```

```
TSP#DSKIOSYNC
TBSP_NAME  NUM_READS AVG_SYNC_READ_TIME NUM_WRITES AVG_SYNC_WRITE_TIME
---------- --------- ------------------ ---------- -------------------
TEMPSPACE1   1124845               0.00      49749                0.37
XTS16K         81758               1.27          0                   -

LTC#WAITT
LATCH_NAME                                        TOT_EXT_LATCH_WAIT_TIME_MS TOT_EXT_LATCH_WAITS  TIME_PER_LATCH_WAIT_MS
------------------------------------------------- -------------------------- -------------------- ----------------------
SQLO_LT_SQLB_HASH_BUCKET_GROUP_HEADER__groupLatch                      270905                 4655                  58.19
SQLO_LT_SQLB_BP_AREA_INFO_pageArea__clockLatch                          71065                35106                   2.02
SQLO_LT_SQLB_BPD__bpdLatch_SX                                            8884                 3847                   2.30
SQLO_LT_SQLB_DIRTY_LIST_SET__walkLatch                                   8122                 3374                   2.40
SQLO_LT_SQLB_DIRTY_LIST_SET__appendLatch                                 5082                 2690                   1.88
SQLO_LT_SQLB_HATELIST_SET__hateLatch                                      398                  174                   2.28
```



Change_1: Increase Sortheap

```
CFG#DB:
NAME             CURRENT_VALUE
---------------- --------------
sheapthres_shr   5000
sortheap         500
```

```
db2 UPDATE DB CFG FOR CCDUG USING SHEAPTHRES_SHR 50000 SORTHEAP 5000 IMMEDIATE
```

```
db2inst1@IBM-PW03R8VZ:~/CCDUG$ java stmtwatcher
SQL: /*STMTWATCHER-2024-03-08-18.04.08.333*/ select * from ( select t2.*,T3.PRODUCER_ID as PID from t1 t1, t2 t2 LEFT OUTER JOIN t3 t3 ON t3.MBR_ID = T2.MBR_ID and T3.EXP_DATE = T2.EXP_DATE where t1.IS_ON = 1 AND  (t2.IND = COALESCE
(t1.IND, '')) ORDER BY T2.EXP_DATE, T3.PRODUCER_ID, T2.PRODUCER_ID ) as T1   where (COALESCE(T1.PRODUCER_ID, T1.PID ) = ? ) ORDER BY 1,2
num of parameters = 1
param[0]: PRODUCER_ID1001047
Number of threads = 30
Serial execution time 952 millisecs
Plan id = -7722274454911209402

db2 "call explain_from_section(x'0100000000000000010000000000000000000000020020240308180408539079', 'M', null, -1, '', ?,?,?,?,? )" ; db2exfmt -d CCDUG -1 > STMTWATCHER-2024-03-08-18.04.08.333.exfmt

          msec    #execs     #ltchwaits   ltchwaittime Cord_msec/exec  Act_msec/exec    waittm/exec    %processing
             6         1              0              0        922.000        922.000          0.059         94.143
         15009        30            454           5901       9382.700       9382.700          0.043         95.690
          5003         2             28            106       7706.000       7706.000          0.030         97.002
          5003        28            366            959       8571.107       8571.107          0.023         97.652
          5004        11            194            246       8159.455       8159.455          0.040         95.996
          5005        30           1345           2203       7438.067       7438.067          0.051         94.905
          5005        26           3350           7905       6269.077       6269.077          0.274         72.618
          5005        25           2663           6437       5564.600       5564.600          0.202         79.787
          5006        30           3023          11191       5264.667       5264.667          0.159         84.112
```

```
                  Rows
                 RETURN
                 (   1)
                  Cost
                   I/O
                   |
                360000
                TBSCAN
                (   2)
                155897
                  NA
                   |
                360000
                SORT
                (   3)
                155893
                  NA
                   |
                360000
                NLJOIN
                (   4)
                155810
                  NA
              /---+---\
           120000          3
           TBSCAN       TBSCAN
           (   5)       (  11)
           155534       6.88963
             NA           NA
              |            |
           120000          12
           SORT     TABLE: DB2INST1
           (   6)          T1
           155533          Q4
             NA
              |
           120000
           FILTER
           (   7)
           155505
             NA
              |
            3e+06
           >HSJOIN
           (   8)
           155379
             NA
        /-----+------\
      3e+06          700000
     TBSCAN          TBSCAN
     (   9)          (  10)
      15417          2507.18
       NA             NA
        |              |
      3e+06          700000
TABLE: DB2INST1  TABLE: DB2INST1
      T2               T3
      Q2               Q1
```

Change_1_Sortheap: Analysis

```
DB#THRUP
ACT_PER_S    ROWS_MOD_P_S P_RD_PER_S
------------ ------------ ------------
         5.2      50883.9      33332.7

DB#PROCT
PCT_SECT_PROC PCT_SECT_SORT_PROC
------------- ------------------
        69.77               0.00

DB#WAITT
PCT_RQST_WAIT PCT_LTCH PCT_PFTCH
------------- -------- ---------
        30.19    16.53     12.03

DB#SORT
TOTAL_SORTS SORT_OVERFLOWS SORT_SHRHEAP_ALLOCATED TOTAL_HSJN  HSJN_OVFL
----------- -------------- ---------------------- ----------- -----------
        246              0                  48273         133         118

SQL#TOPEXECT
AVG_COORD_EXEC_TIME PCT_WAIT_TIME   STMT_TEXT
------------------- -------------- -------------
            8360.45          30.00 /*STMTWATCHER
```

```
SQL#TOPWAITT
PCT_WAIT PCT_LTCH PCT_PFTCH STMT_TEXT
-------- -------- --------- -------------
   30.00    16.42     11.99 /*STMTWATCHER

SQL#TOPIOSTA
AVG_D_PRD    AVG_TD_PRD   STMT_TEXT
------------ ------------ -------------
         1.3       8755.6 /*STMTWATCHER

SQL#TOPROWS
AVG_ROWS_MOD   AVG_ROWS_READ  AVG_ROWS_RET   STMT_TEXT
-------------- -------------- -------------- -------------
       13270.4      3713282.4            2.0 /*STMTWATCHER

TSP#DSKIO
TBSP_NAME            NUM_READS            NUM_WRITES
-------------------- -------------------- --------------------
TEMPSPACE1                        2504690              1373270
XTS16K                             145876                    0
```


Change_1_Sortheap: Analysis cont.

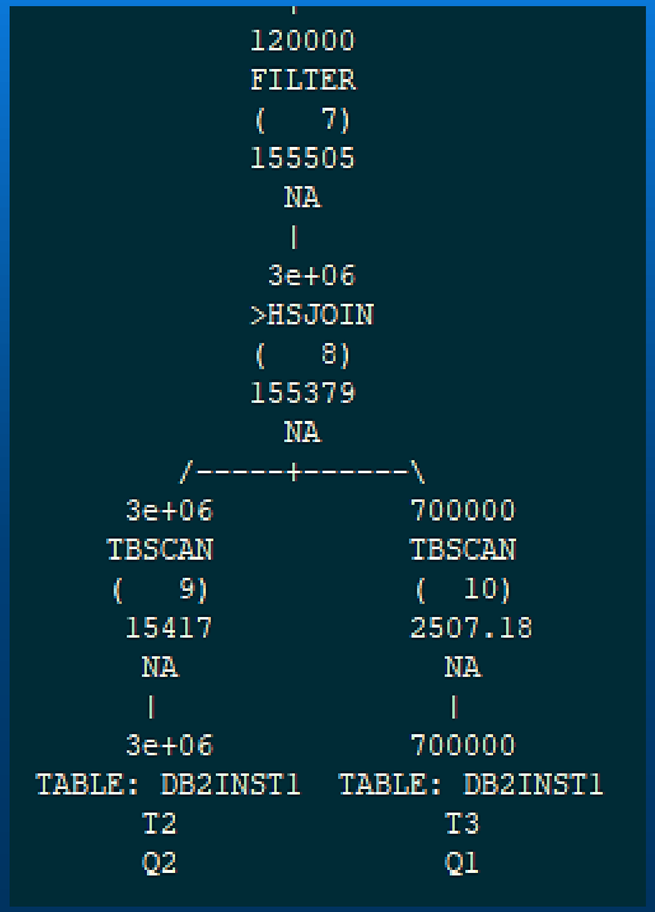
```
TBL#ROWMC
TABNAME  ROWS_READ  OBJ_DATA_P_RDS
-------  ---------  --------------
T2       316054742               2
T3        77825650             149

BPL#STATS
TBSP_NAME   DATA_P_READS     SYNC_DATA_P_READS   DATA_HR  DATA_LBP_HR
----------  -------------  --------------------  -------  -----------
XTS16K             145876                  2907    99.88        99.88
TEMPSPACE1        2504690                977527    77.68        77.68
```

```
TSP#BPMAP
TBSP_NAME       BPNAME
--------------  --------------------
TEMPSPACE1      IBMDEFAULTBP
XTS16K          XBP16K

BPL#SIZES
BP_NAME        SIZE_MB       AUTOMATIC
-------------  ------------  ---------
IBMDEFAULTBP          89.11          1
XBP16K                15.62          0

BPL#RDASYNC/BPL#WRITE
BP_NAME       POOL_ASYNC_DATA_READS  POOL_ASYNC_READ_TIME  AVG_ASYNC_READ_TIME  POOL_DATA_WRITES
------------  ---------------------  --------------------  -------------------  ----------------
IBMDEFAULTBP                1548260                  6745                 0.00           1366521
XBP16K                       142660                 23115                 0.16                 -
```

```
LTC#WAITT
LATCH_NAME                                                TOT_EXT_LATCH_WAIT_TIME_MS  TOT_EXT_LATCH_WAITS  TIME_PER_LATCH_WAIT_MS
--------------------------------------------------------  --------------------------  -------------------  ----------------------
SQLO_LT_SQLB_BP_AREA_INFO_pageArea__clockLatch                                 176023                89709                    1.96
SQLO_LT_SQLB_BPD__bpdLatch_SX                                                  19705                16503                    1.19
SQLO_LT_SQLB_HASH_BUCKET_GROUP_HEADER__groupLatch              10368                 1532                    6.76
SQLO_LT_SQLB_DIRTY_LIST_SET__appendLatch                        1586                  704                    2.25
SQLO_LT_SQLB_HATELIST_SET__hateLatch                            1209                  693                    1.74
SQLO_LT_SQLB_DIRTY_LIST_SET__walkLatch                           881                  355                    2.48
SQLO_LT_SQLD_TCB__datExist                                       559                  129                    4.33
```


Change_1_Sortheap: Access Plan Review

```
TBL#ROWMC
TABNAME ROWS_READ OBJ_DATA_P_RDS
------- --------- --------------
T2      316054742              2
T3       77825650            149


SQL#TOPROWS  → 9420.exfmt.txt
AVG_ROWS_MOD    AVG_ROWS_READ  AVG_ROWS_RET    STMT_TEXT
--------------- -------------- -------------- -------------
       13270.4      3713282.4            2.0 /*STMTWATCHER
```




```
8) HSJOIN: (Hash Join)
   Predicates:
   ----------
   6) Predicate used in Join,
      Filter Factor:         0.5

      Predicate Text:
      --------------
      (Q1.EXP_DATE = Q2.EXP_DATE)

   7) Predicate used in Join,
      Filter Factor:         3.33333e-07

      Predicate Text:
      --------------
      (Q1.MBR_ID = Q2.MBR_ID)

7) FILTER: (Filter)
   Predicates:
   ----------
   3) Residual Predicate,
      Filter Factor:         0.04
      Predicate Text:
      --------------
      (COALESCE(Q3.$C2, Q3.$C6) = ?)
```

Change_1_Sortheap: Access Plan Review cont.

```
8) HSJOIN: (Hash Join)
   Predicates:
   ----------
   6) Predicate used in Join,
      Filter Factor:         0.5

      Predicate Text:
      --------------
      (Q1.EXP_DATE = Q2.EXP_DATE)

   7) Predicate used in Join,
      Filter Factor:         3.33333e-07

      Predicate Text:
      --------------
      (Q1.MBR_ID = Q2.MBR_ID)

7) FILTER: (Filter)
   Predicates:
   ----------
   3) Residual Predicate,
      Filter Factor:         0.04
      Predicate Text:
      --------------
      (COALESCE(Q3.$C2, Q3.$C6) = ?)
```

```
Optimized Statement:
.
(SELECT
   Q2.IS_ON,
   Q2.PHONENO,
   Q2.PRODUCER_ID,
   Q2.IND,
   Q2.EXP_DATE,
   Q2.MBR_ID,
   Q1.PRODUCER_ID
 FROM
   DB2INST1.T3 AS Q1
   RIGHT OUTER JOIN DB2INST1.T2 AS Q2
   ON Q1.EXP_DATE = Q2.EXP_DATE AND
        Q1.MBR_ID = Q2.MBR_ID
) AS Q3,
DB2INST1.T1 AS Q4
WHERE Q4.IS_ON           = TRUE AND
COALESCE(Q3.$C2, Q3.$C6) = ?    AND
Q3.IND = COALESCE(Q4.IND, '')   AND
VARCHAR(Q3.IND, 1) = COALESCE( Q4.IND, '')
.
```

```
Original Statement:
SELECT *
FROM
(SELECT T2.*,
        T3.PRODUCER_ID AS PID
 FROM T1 T1, T2 T2
 LEFT OUTER JOIN T3 T3
   ON T3.MBR_ID   = T2.MBR_ID
  AND T3.EXP_DATE = T2.EXP_DATE
 WHERE T1.IS_ON   = 1
   AND T2.IND     = COALESCE(T1.IND, '')
 ORDER BY T2.EXP_DATE, T3.PRODUCER_ID, T2.PRODUCER_ID
) AS T1
WHERE COALESCE(T1.PRODUCER_ID,T1.PID) = ?
ORDER BY 1, 2
```

Change_2: Query Rewrite

```
REWRITE:
(COALESCE(T1.PRODUCER_ID, T1.PID) = ?) ➔ T1.PRODUCER_ID = ? OR (T1.PRODUCER_ID IS NULL AND T1.PID = ?)

Similarly,
(T2.IND = COALESCE(T1.IND, ‘’))        ➔ (T2.IND = T1.IND OR (T1.IND IS NULL AND T2.IND = ‘’))
```

```
Original SQL:
SELECT *
FROM
(SELECT T2.*, T3.PRODUCER_ID AS PID
 FROM T1 T1, T2 T2
 LEFT OUTER JOIN T3 T3
   ON T3.MBR_ID   = T2.MBR_ID
  AND T3.EXP_DATE = T2.EXP_DATE
 WHERE T1.IS_ON   = 1
   AND T2.IND     = COALESCE(T1.IND, '')
 ORDER BY T2.EXP_DATE,T3.PRODUCER_ID,T2.PRODUCER_ID
) AS T1
WHERE COALESCE(T1.PRODUCER_ID,T1.PID) = ?
ORDER BY 1, 2
```



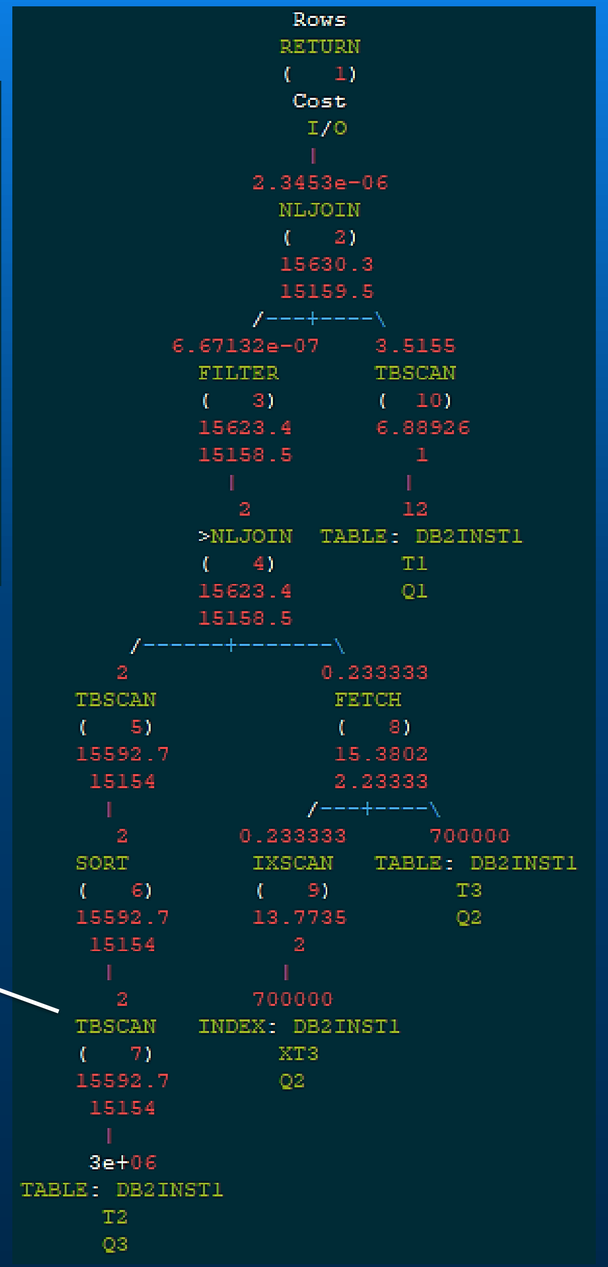
```
Rewrite:
SELECT *
FROM
(SELECT T2.*, T3.PRODUCER_ID AS PID
 FROM T1 T1, T2 T2
 LEFT OUTER JOIN T3 T3
   ON T3.MBR_ID   = T2.MBR_ID
  AND T3.EXP_DATE = T2.EXP_DATE
 WHERE T1.IS_ON   = 1
   AND (T2.IND = T1.IND OR (T1.IND IS NULL AND T2.IND = ‘’))

) AS T1
WHERE T1.PRODUCER_ID = ? OR (T1.PRODUCER_ID IS NULL AND T1.PID = ?)
ORDER BY 1, 2
```


Change_2_QueryRewrite: Review

```
db2inst1@IBM-PW03R8VZ:~/CCDUG$ java stmtwatcher
SQL: /*STMTWATCHER-2024-03-08-18.45.49.740*/ select   * from   (select      t2.*,     T3.PRODUCER_ID as PID    from      t1 t1,     t2 t2      LEFT OUTER JOIN t3 t3      ON t3.MBR_ID = T2.MBR_ID and      T3.EXP_DATE = T2.EXP_DATE
 where      t1.IS_ON = 1 AND      (t2.IND = t1.IND OR (t1.IND IS NULL AND t2.IND = ''))   ) as T1 where   (T1.PRODUCER_ID IS NULL AND T1.PID = ? )  OR T1.PRODUCER_ID = ? ORDER BY   1,  2   ;
num of parameters = 2
param[0]: PRODUCER_ID1001047
param[1]: PRODUCER_ID1001047
Number of threads = 30
Serial execution time 789 millisecs
Plan id = 6521863271411465852

db2 "call explain_from_section(x'0100000000000000320000000000000000000000002002024030818454996105 6', 'M', null, -1, '', ?,?,?,?,? )" ; db2exfmt -d CCDUG -1 > STMTWATCHER-2024-03-08-18.45.49.740.exfmt
```

msec	#execs	#ltchwaits	ltchwaittime	Cord_msec/exec	Act_msec/exec	waittm/exec	%processing
5	1	0	0	758.000	758.000	0.000	100.000
5008	44	60	462	424.023	424.023	0.025	97.524
5004	325	607	4174	459.905	459.905	0.028	97.207
5003	293	554	4101	501.747	501.747	0.028	97.210
5004	301	489	3631	497.193	497.193	0.024	97.574
5005	296	360	2626	486.432	486.432	0.018	98.176
5004	218	427	3455	677.899	677.899	0.023	97.662
5006	197	487	3682	736.041	736.041	0.025	97.461

```
7) TBSCAN: (Table Scan)
    Predicates:
    ----------
    15) Sargable Predicate,
        Filter Factor:  6.66667e-07
        Predicate Text:
        --------------
        ((Q3.PRODUCER_ID = ?) OR Q3.PRODUCER_ID IS NULL)
```



Change_3: Create Index

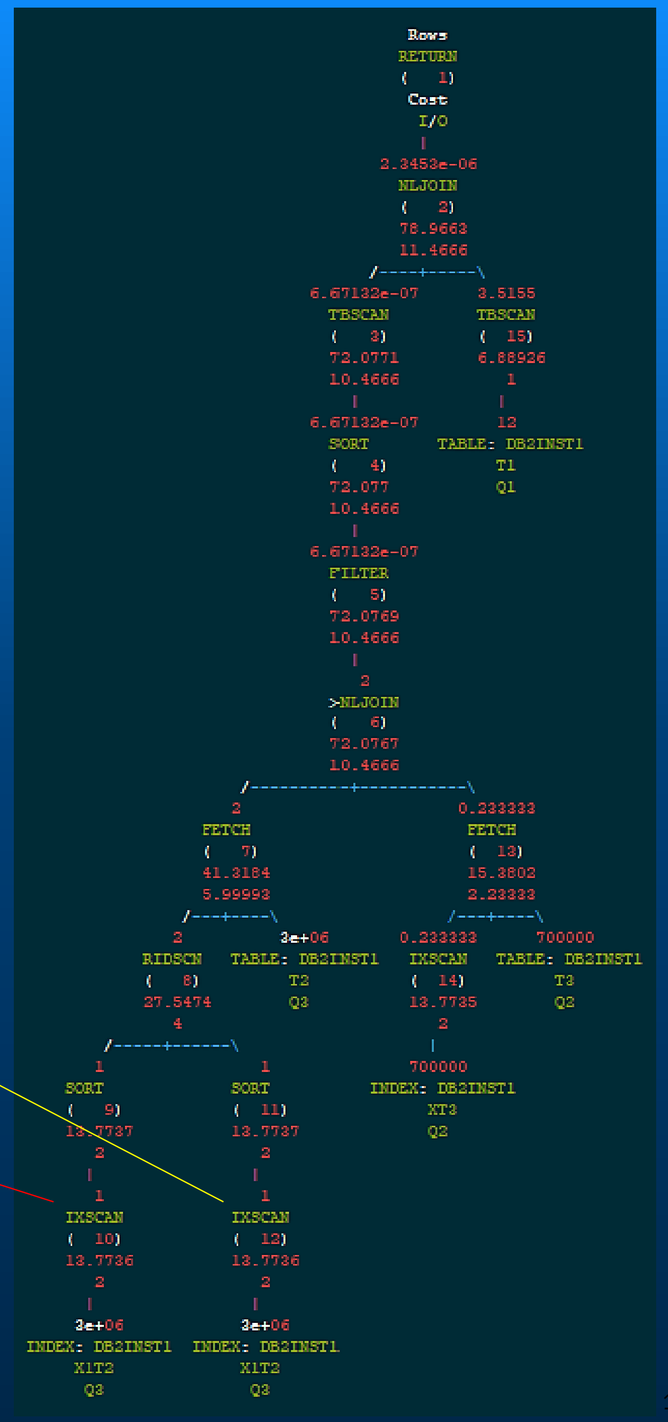
```
CREATE INDEX X1T2 ON T2 (PRODUCER_ID)
```

```
db2inst1@IBM-PW03R8VZ:~/CCDUG$ java stmtwatcher
SQL: /*STMTWATCHER-2024-03-08-18.52.01.110*/ select   * from   (select      t2.*,     T3.PRODUCER_ID as PID    from      t1 t1,      t2 t2      LEFT OUTER JOIN t3 t3      ON t3.MBR_ID = T2.MBR_ID and      T3.EXP_DATE = T2.EXP_DATE
 where      t1.IS_ON = 1 AND      (t2.IND = t1.IND OR (t1.IND IS NULL AND t2.IND = ''))   ) as T1 where   (T1.PRODUCER_ID IS NULL AND T1.PID = ? )  OR T1.PRODUCER_ID = ? ORDER BY   1,  2   ;
num of parameters = 2
param[0]: PRODUCER_ID1001047
param[1]: PRODUCER_ID1001047
Number of threads = 30
Serial execution time 17 millisecs
Plan id = 4496458349625396799

db2 "call explain_from_section(x'0100000000000000540000000000000000000000002002024030818520132179 4', 'M', null, -1, '', ?,?,?,?,? )" ; db2exfmt -d CCDUG -1 > STMTWATCHER-2024-03-08-18.52.01.110.exfmt
```

msec	#execs	#ltchwaits	ltchwaittime	Cord_msec/exec	Act_msec/exec	waittm/exec	%processing
5	1	0	0	1.000	1.000	1.000	0.000
5006	3354	44	5	0.098	0.098	0.015	98.476
5006	489055	16199	5997	0.086	0.086	0.143	85.727
5004	516978	18556	6167	0.084	0.084	0.142	85.849
5005	396729	14938	7039	0.112	0.112	0.158	84.162
5009	505602	16415	5545	0.086	0.086	0.127	87.262

```
10) IXSCAN: (Index Scan)

   Predicates:
   ----------
   16) Start Key Predicate,
    Filter Factor:          3.33333e-07
    Predicate Text:
    --------------
    Q3.PRODUCER_ID IS NULL

   16) Stop Key Predicate,
    Filter Factor:          3.33333e-07
    Predicate Text:
    --------------
    Q3.PRODUCER_ID IS NULL
```

```
12) IXSCAN: (Index Scan)

   Predicates:
   ----------
   17) Start Key Predicate,
    Filter Factor:          3.33333e-07
    Predicate Text:
    --------------
    (Q3.PRODUCER_ID = ?)

   17) Stop Key Predicate,
    Filter Factor:          3.33333e-07
    Predicate Text:
    --------------
    (Q3.PRODUCER_ID = ?)
```



Resources

1. DB2MON - https://www.ibm.com/docs/en/db2/11.5?topic=tuning-collecting-reporting-performance-monitor-data
2. EXPLAIN_FROM_SECTION - https://www.ibm.com/docs/en/db2/11.5?topic=gcsei-example-investigating-query-performance-using-explain-information-obtained-from-section
3. DB2PD - https://www.ibm.com/docs/en/db2/11.5?topic=commands-db2pd-monitor-troubleshoot-db2-engine-activities
4. DMCTOP - https://www.ibm.com/docs/en/db2-data-mgr-console/3.1.x?topic=monitoring-utility-dmctop
5. Monitoring and troubleshooting using db2pd - https://www.ibm.com/docs/en/db2/11.5?topic=tools-db2pd
6. Bufferpool Latch Contention - https://www.ibm.com/support/pages/identifying-bufferpool-latch-contention