DB2 Performance & Tuning (it’s still all about the SQL!)

MICHAEL WINGFIELD
STROBE SOLUTION CONSULTANT

February 8-10, 2016

Austin, Houston, Dallas DB2 User Groups
Mainframe Relevance

220 Billion Lines of COBOL Code (5 Billion Lines/Year)
Over 1 Million CICS Transaction per second per day
30 Billion Business Transactions per day on Mainframe
6 Trillion $ in Credit Card Payments per year on Mainframe
80% of Worldwide Corporate Data is on the Mainframe
50 Billion Devices generating transactions by 2020
43 Billion Banking Transactions in 2013 (360M in 2011)

The Mainframe is ideally suited to handle these transactions!
Moore’s Law

In 1965 one of the co-founders of Intel – Gordon Moore – described a trend that still holds true today – namely that processor speed doubles about every 18 months. However, that trend is nearing it’s end.

The Mainframe System z processor – zEC12 – is currently the fastest microprocessor in the industry at 5.5 GHz – but CMOS technology is not expected to ever exceed 7 GHz.

Application Performance Management is becoming more and more imperative to keep Operating Costs at a minimum since we can’t keep throwing hardware at performance issues.
The z13 is a Hybrid Solution that can run both the current “Sequential Programming” plus the new workloads that use “Vector Technologies” and “Parallel Programming”.

The immediate value in the z13 is that Vector Technologies will be used in the z13 to accelerate COBOL, PL/1 and Java code -- automatically by just using the new compilers.

The long term value however, is “Parallel Programming”, and requires not only a new compiler, but a re-design of applications to fully exploit this new functionality.
Rolling 4 Hour Average

We need to be selective on what we tune. Choose to tune SQL that is running during the four hours prior to the peaks identified by R4HA SMF data.

There may be many peaks during the month and days of the month and unless you can lower the peak MSUs during the month, no *real* money will be saved. CPU may be saved but not money.

Also, consider moving workloads around into some of the R4HA valleys to help lower the peak MSUs. Lowering the peak helps lower the cost of all the IBM software running on the LPAR.
Rolling 4 Hour Average Report

The Rolling 4 Hour Average Peak is used to determine the Monthly License Cost

60% of Processing Peaks are driven by Batch Processing
CICS/Batch/DDF
DB2 Version 10

The following screen shots are captured from iStrobe
Measurement Session Data - CICS

5 Minute Measurement of a CICS Region
CA Detector Module Names start with PSA and PDT and represent over 40% of the CPU

>200 Hours of CPU/Yr Per Region!!!
DB2 Activity by Query

### DB2 Activity by Query

<table>
<thead>
<tr>
<th>Target Statement</th>
<th>Statement Number</th>
<th>Type</th>
<th>Statement Count</th>
<th>Elapsed Time (sec)</th>
<th>Parallel Task CPU Distribution (sec)</th>
<th>CPU Time (sec)</th>
<th>CPU %</th>
<th>Wait %</th>
</tr>
</thead>
<tbody>
<tr>
<td>QUEREO</td>
<td>24</td>
<td>ND</td>
<td>32,147</td>
<td>0.39</td>
<td>6.172303</td>
<td>0.00</td>
<td>0.26</td>
<td>0.06</td>
</tr>
</tbody>
</table>

### SQL Statement Text

```
DECLARE CQ1 CURSOR FOR SELECT * FROM TABLE; 
DECLARE CQ2 CURSOR FOR SELECT * FROM TABLE2;
```

### Run Time Statistics for Scan Execution

<table>
<thead>
<tr>
<th>Type</th>
<th>Rows Processed</th>
<th>Rows Examined</th>
<th>Rows-Q Stage1</th>
<th>Rows-Q Stage2</th>
<th>Rows Inserted</th>
<th>Rows Deleted</th>
<th>Rows Updated</th>
<th>Calls' Page Requests</th>
<th>RI Page Scans</th>
<th>RI Page Deletions</th>
<th>LOG Page Scans</th>
<th>LOG Page Updates</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDX</td>
<td>328</td>
<td>13,002</td>
<td>12,118</td>
<td>2,039</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SBD3</td>
<td>14,358</td>
<td>14,313</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Parallel Processing

<table>
<thead>
<tr>
<th>Operation</th>
<th>CPU %</th>
<th>Wait %</th>
</tr>
</thead>
<tbody>
<tr>
<td>FETCH</td>
<td>0.17</td>
<td>N/A</td>
</tr>
<tr>
<td>CLOSE</td>
<td>0.15</td>
<td>N/A</td>
</tr>
<tr>
<td>DECLARE</td>
<td>0.31</td>
<td>N/A</td>
</tr>
</tbody>
</table>

The Mainframe Software Partner for the Next 50 Years
### SQL Analysis – DB2 Explain

#### Plan Table

<table>
<thead>
<tr>
<th>QBLOCK NO</th>
<th>PLAN NO</th>
<th>METHOD</th>
<th>INDEX ONLY</th>
<th>CREATOR</th>
<th>TNAME</th>
<th>ACCESS CREATOR</th>
<th>ACCESS NAME</th>
<th>ACCESS TYPE</th>
<th>MATCH COLS</th>
<th>SORTN UJOG</th>
<th>SOROTO UJOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>MIXOPSEQ</td>
<td>N</td>
<td>TNO</td>
<td>ACCESS DEGREE</td>
<td>JOIN DEGREE</td>
<td>ACCESS PGROUP_ID</td>
<td>JOIN PGROUP_ID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>MIXOPSEQ</td>
<td>N</td>
<td>TNO</td>
<td>ACCESS DEGREE</td>
<td>JOIN DEGREE</td>
<td>ACCESS PGROUP_ID</td>
<td>JOIN PGROUP_ID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>MIXOPSEQ</td>
<td>N</td>
<td>TNO</td>
<td>ACCESS DEGREE</td>
<td>JOIN DEGREE</td>
<td>ACCESS PGROUP_ID</td>
<td>JOIN PGROUP_ID</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### SQL Statement Text

**Profile Name:** [Redacted] **Job Name:** [Redacted] **Initiated On:** 2014-08-02 09:38:53

**SQL Statement Analysis:**

- **Translation:**
- **Observations:**
- **Catalog Statistics:**

---

**The Mainframe Software Partner for the Next 50 Years**
### SQL Analysis - DB2 Catalog Statistics

#### Table: Catalog Statistics

<table>
<thead>
<tr>
<th>DBNAME</th>
<th>TSNNAME</th>
<th>CARDF</th>
<th>NUM DEP MQTS</th>
<th>SPLIT ROWS</th>
<th>NPAGES</th>
<th>PCT PAGES</th>
<th>PCT CO</th>
<th>PCT ROW COMP</th>
<th>KEY NO</th>
<th>AVG ROW LCN</th>
<th>STATETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>7,249,072</td>
<td>0</td>
<td>59,628</td>
<td>71</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>35</td>
<td></td>
<td>2014-09-07 21:02:10</td>
</tr>
</tbody>
</table>

#### Columns (SYSCOLUMNS)

<table>
<thead>
<tr>
<th>NAME</th>
<th>CREATOR</th>
<th>CLUSTER ACT</th>
<th>CLUSTER SPC</th>
<th>INDEX SPACE</th>
<th>FIRST KEYCARD</th>
<th>FULL KEYCARD</th>
<th>NLEAF</th>
<th>NLEVELS</th>
<th>CLUSTER RATIO</th>
<th>BPPOOL</th>
<th>PSIZE</th>
<th>UNIQ RULE</th>
<th>IDX TYPE</th>
<th>AVG KEYLEN</th>
<th>PAD</th>
<th>GPM PRES</th>
<th>IDX EXT TYP</th>
<th>STATETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
<td>7,249,072</td>
<td>7,249,072</td>
<td>27,753</td>
<td>3</td>
<td>100.00%</td>
<td>BP3</td>
<td>4.096</td>
<td>P</td>
<td>7</td>
<td>N</td>
<td></td>
<td>N</td>
<td></td>
<td>2014-09-07 21:02:10</td>
</tr>
</tbody>
</table>

#### Key Columns (SYSKEYS)

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLCARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TP_ID</td>
<td>2</td>
<td>A</td>
<td>7,249,072</td>
</tr>
</tbody>
</table>

### Table: Catalog Statistics (continued)

<table>
<thead>
<tr>
<th>DBNAME</th>
<th>TSNNAME</th>
<th>CARDF</th>
<th>NUM DEP MQTS</th>
<th>SPLIT ROWS</th>
<th>NPAGES</th>
<th>PCT PAGES</th>
<th>PCT CO</th>
<th>PCT ROW COMP</th>
<th>KEY NO</th>
<th>AVG ROW LCN</th>
<th>STATETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1,324,194</td>
<td>0</td>
<td>14,004</td>
<td>55</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>60</td>
<td></td>
<td>2014-09-07 21:02:10</td>
</tr>
</tbody>
</table>

#### Columns (SYSCOLUMNS)

<table>
<thead>
<tr>
<th>NAME</th>
<th>CREATOR</th>
<th>CLUSTER ACT</th>
<th>CLUSTER SPC</th>
<th>INDEX SPACE</th>
<th>FIRST KEYCARD</th>
<th>FULL KEYCARD</th>
<th>NLEAF</th>
<th>NLEVELS</th>
<th>CLUSTER RATIO</th>
<th>BPPOOL</th>
<th>PSIZE</th>
<th>UNIQ RULE</th>
<th>IDX TYPE</th>
<th>AVG KEYLEN</th>
<th>PAD</th>
<th>GPM PRES</th>
<th>IDX EXT TYP</th>
<th>STATETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>N</td>
<td>N</td>
<td></td>
<td>1,324,194</td>
<td>1,324,194</td>
<td>5,049</td>
<td>3</td>
<td>64.30%</td>
<td>BP2</td>
<td>4,006</td>
<td>D</td>
<td>2</td>
<td>10</td>
<td>N</td>
<td>N</td>
<td></td>
<td>2014-09-07 21:02:10</td>
</tr>
</tbody>
</table>

#### Key Columns (SYSKEYS)

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLCARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TP_ID</td>
<td>2</td>
<td>A</td>
<td>1,324,194</td>
</tr>
</tbody>
</table>

### Table: Catalog Statistics (continued)

<table>
<thead>
<tr>
<th>DBNAME</th>
<th>TSNNAME</th>
<th>CARDF</th>
<th>NUM DEP MQTS</th>
<th>SPLIT ROWS</th>
<th>NPAGES</th>
<th>PCT PAGES</th>
<th>PCT CO</th>
<th>PCT ROW COMP</th>
<th>KEY NO</th>
<th>AVG ROW LCN</th>
<th>STATETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>7,249,072</td>
<td>0</td>
<td>59,628</td>
<td>71</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>35</td>
<td></td>
<td>2014-09-07 21:02:10</td>
</tr>
</tbody>
</table>

#### Columns (SYSCOLUMNS)

<table>
<thead>
<tr>
<th>NAME</th>
<th>CREATOR</th>
<th>CLUSTER ACT</th>
<th>CLUSTER SPC</th>
<th>INDEX SPACE</th>
<th>FIRST KEYCARD</th>
<th>FULL KEYCARD</th>
<th>NLEAF</th>
<th>NLEVELS</th>
<th>CLUSTER RATIO</th>
<th>BPPOOL</th>
<th>PSIZE</th>
<th>UNIQ RULE</th>
<th>IDX TYPE</th>
<th>AVG KEYLEN</th>
<th>PAD</th>
<th>GPM PRES</th>
<th>IDX EXT TYP</th>
<th>STATETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
<td>7,249,072</td>
<td>7,249,072</td>
<td>26,049</td>
<td>3</td>
<td>7.73%</td>
<td>BP3</td>
<td>4.096</td>
<td>U</td>
<td>7</td>
<td>N</td>
<td></td>
<td>N</td>
<td></td>
<td>2014-09-07 21:02:10</td>
</tr>
</tbody>
</table>

#### Key Columns (SYSKEYS)

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLCARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TP_NTRI_ID</td>
<td>2</td>
<td>A</td>
<td>1,324,194</td>
</tr>
<tr>
<td>2</td>
<td>TAX_ID</td>
<td>3</td>
<td>A</td>
<td>1,324,194</td>
</tr>
<tr>
<td>3</td>
<td>TAX_SUB_TY_ID</td>
<td>5</td>
<td>A</td>
<td>1,324,194</td>
</tr>
<tr>
<td>4</td>
<td>FIL_METH_CD</td>
<td>6</td>
<td>A</td>
<td>1,324,194</td>
</tr>
<tr>
<td>5</td>
<td>FIL_METH_SUB_CD</td>
<td>9</td>
<td>A</td>
<td>1,324,194</td>
</tr>
</tbody>
</table>

#### Index Package Dependencies (SYSPACKDEPEND)

<table>
<thead>
<tr>
<th>TBLNAME</th>
<th>PARTKEY COLNO</th>
<th>AVG ROW LCN</th>
<th>STATETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPFLCH1</td>
<td>7</td>
<td>N</td>
<td>2014-09-07 20:29:25</td>
</tr>
<tr>
<td>TPFLCH2</td>
<td>7</td>
<td>N</td>
<td>2014-09-07 20:29:25</td>
</tr>
</tbody>
</table>

### Index Package Dependencies (SYSPACKINDEX)

<table>
<thead>
<tr>
<th>TBLNAME</th>
<th>PARTKEY COLNO</th>
<th>AVG ROW LCN</th>
<th>STATETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPFLCH1</td>
<td>7</td>
<td>N</td>
<td>2014-09-07 20:29:25</td>
</tr>
<tr>
<td>TPFLCH2</td>
<td>7</td>
<td>N</td>
<td>2014-09-07 20:29:25</td>
</tr>
</tbody>
</table>
DECLARE C0011970 CURSOR FOR
SELECT A.FIL_METH_BEG_DT,
    A.FIL_METH_END_DT,
    A.TP_FIL_CHAR_ID,
    A.MAND_FIL_LVL_CD
FROM A,
    TAXPYR.VTAXPAYER01 B
WHERE A.TP_NTRL_ID = B.TP_NTRL_ID
    AND B.TP_ID = :H
    AND A.TAX_ID = :H
    AND A.TAX_SUB_TY_ID = :H
    AND A.FIL_METH_CD = 'B'
    AND A.FIL_METH_SUB_CD = 'M'
    AND A.FIL_METH_BEG_DT <= :H
    AND ( A.FIL_METH_END_DT >= :H
         OR A.FIL_METH_END_DT IS NULL)
    AND A.SETU_ERR_CD <> 'Y'
ORDER BY A.FIL_METH_BEG_DT DESC
FOR FETCH ONLY

Consider using INCLUDE for Column TP_NTRL_ID in Unique Index TAXPAY2 to change the access to Index Only. Consider adding Column FIL_METH_BEG_DT to the end of Index TPFLCH2 to eliminate the Order By Sort. Index TPFLCH3 has an overall cardinality of only 74.
DB2 Activity by Query
**SQL Analysis – DB2 Explain**

Access Path – Non-Matching Index Scan W/Page Reference – Sort for Uniqueness and Order By

Note – All Leaf Pages in the Index are Scanned as well as Table Pages of Rows that Qualify.
### SQL Analysis - DB2 Catalog Statistics

**Note** – The First Two Columns of Index XIE1RULE have a Cardinality of only 1.

**Index XIE1RULE is 2148 Leaf Pages**
DECLARE PBC-LOCAL-CSR CURSOR FOR
SELECT DISTINCT CHAR(CR.RCPT_INCUR_STRT_DT,ISO) AS LOCAL_EFF_DT,
    CHAR(CR.RCPT_INCUR_END_DT,ISO) AS LOCAL_EFF_TO_DT,
    CHAR((CR.RCPT_INCUR_END_DT - 1 DAY),ISO) AS LOCAL_EFF_THRU_DT
FROM XIE1RULE(0)S
    INNER JOIN CR
    ON R.PLAN_PROFL_UID = CR.PLAN_PROFL_UID
WHERE R.PRFX_CD = :H
    AND R.LOC_PLAN_CD ≠ 'STX'
    AND R.STS_CD = '3'
    AND R.STS_QUAL_CD = '02'
    AND CR.PGM_CD = 'A'
    AND CR.DLVY_METH_CD = '2'
    AND CR.RCPT_INCUR_STRT_DT ≠ VALUE(CR.RCPT_INCUR_END_DT, DATE(:H))
    AND VALUE(CR.RCPT_INCUR_END_DT,DATE(:H)) > DATE (:H)
ORDER BY LOCAL_EFF_DT ASC,
    LOCAL_EFF_TO_DT ASC

Consider adding an index to Table ITS.RULE to change the Non-Matching Index Scan to a Matching Index Only Scan as follows:

Column 1 - PRFX_CD
  2 - STS_QUAL_CD
  3 - STS_CD
  4 - LOC_PLAN_CD
  5 - PLAN_PROFL_UID
DB2 Activity by Query

### DB2 Activity by DBRMPackage

<table>
<thead>
<tr>
<th>DBRMPackage</th>
<th>Bind Timestamp</th>
<th>Statement Count</th>
<th>Elapsed Time (sec)</th>
<th>Parallel Task CPU Distribution (sec)</th>
<th>CPU Time (sec)</th>
<th>CPU %</th>
<th>Wait %</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Target Statement

<table>
<thead>
<tr>
<th>Statement Number</th>
<th>Statement Type</th>
<th>Statement Count</th>
<th>Elapsed Time (sec)</th>
<th>Parallel Task CPU Distribution (sec)</th>
<th>CPU Time (sec)</th>
<th>CPU %</th>
<th>Wait %</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECLARE 1718</td>
<td>NA, SC</td>
<td>0</td>
<td>1.52/475</td>
<td>9.16/4845</td>
<td>0.00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### SQL Statement Text

```
DECLARE C0011280 CURSOR FOR SELECT * FROM TAK ... OR C00182AP.
```

#### Executing Statement

<table>
<thead>
<tr>
<th>Statement Number</th>
<th>Statement Count</th>
<th>Elapsed Time (sec)</th>
<th>Parallel Task CPU Distribution (sec)</th>
<th>CPU Time (sec)</th>
<th>CPU %</th>
<th>Wait %</th>
</tr>
</thead>
<tbody>
<tr>
<td>FETCH 478</td>
<td>2</td>
<td>4.59/372</td>
<td>0.00/00000</td>
<td>0.00</td>
<td>1.20</td>
<td>0.03</td>
</tr>
</tbody>
</table>

#### Run Time Statistics for Scans Executed by the SQL Statement

<table>
<thead>
<tr>
<th>Type</th>
<th>Rows Processed</th>
<th>Rows Examined</th>
<th>Rows-Q Stage 1</th>
<th>Rows-Q Stage 2</th>
<th>Rows Inserted</th>
<th>Rows Deleted</th>
<th>Rows Updated</th>
<th>Get Page Requests</th>
<th>RI Page Scans</th>
<th>RI Page Deletes</th>
<th>LOB Page Scans</th>
<th>LOB Page Updates</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDEX</td>
<td>31,169,322</td>
<td>4</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>257,671</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SINDEX</td>
<td>24</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

#### Services Invoked

<table>
<thead>
<tr>
<th>Parallel Processing</th>
<th>DB2 Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLOSE</td>
<td>DB2Location</td>
</tr>
<tr>
<td>OPEN</td>
<td>DB2Location</td>
</tr>
</tbody>
</table>

### Compuware

The Mainframe Software Partner for the Next 50 Years
### SQL Analysis - DB2 Explain

<table>
<thead>
<tr>
<th>QBLOCK NO</th>
<th>PLAN NO</th>
<th>METHOD</th>
<th>INDEX ONLY</th>
<th>CREATOR</th>
<th>TNAME</th>
<th>ACCESS CREATED</th>
<th>ACCESS NAME</th>
<th>ACCESS TYPE</th>
<th>MATCH COLS</th>
<th>SORTN UJOC</th>
<th>SORTC UJOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>N</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>PROJ_S1</td>
<td>1</td>
<td>I</td>
<td>3</td>
<td>NNNN</td>
<td>NNNN</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>N</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>SORDER</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>NNNN</td>
<td>NNNN</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>N</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>SORDER</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>NNNN</td>
<td>NNNN</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>N</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>SORDER</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>NNNN</td>
<td>NNNN</td>
</tr>
</tbody>
</table>

**Access Path – Non-Matching Index Scan – Index Only**

---

*The Mainframe Software Partner for the Next 50 Years*
### SQL Analysis - DB2 Catalog Statistics

#### Catalog Statistics

<table>
<thead>
<tr>
<th>Table</th>
<th>DBNAME</th>
<th>TSNAMEN</th>
<th>CARDF</th>
<th>NUM DEP MQTS</th>
<th>SPLIT ROWS</th>
<th>NPAGES</th>
<th>PCT PAGES</th>
<th>PCT ROW COMP</th>
<th>PARTKEY GOL NO</th>
<th>AVG ROW LEN</th>
<th>STATTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>107,117,536</td>
<td>0</td>
<td>2,460,514</td>
<td>95</td>
<td>100</td>
<td></td>
<td>0</td>
<td>33</td>
<td>2012-10-14 05:39:58</td>
</tr>
</tbody>
</table>

#### Indexes (SYNSINDEXES)

<table>
<thead>
<tr>
<th>NAME</th>
<th>CREATOR</th>
<th>CLUSTER ACT</th>
<th>CLUSTER SPO</th>
<th>INDEX SPACEN</th>
<th>FIRST KEYWORD</th>
<th>FULL KEYWORD</th>
<th>NLEVELS</th>
<th>CLUSTER RATIO</th>
<th>BPOOL</th>
<th>PSIZE</th>
<th>UNIQUE RULE</th>
<th>IDX TYPE</th>
<th>AVG KEYLEN</th>
<th>PAD</th>
<th>CPM PRES</th>
<th>IDX EXT TYPEN</th>
<th>STATTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRDGB1</td>
<td>Y</td>
<td>N</td>
<td>5,526,750</td>
<td>107,117,536</td>
<td>836,399</td>
<td>4</td>
<td>99.53%</td>
<td>BP3</td>
<td>4,396</td>
<td>U</td>
<td>2</td>
<td>21</td>
<td>N</td>
<td>2012-10-14 05:20:56</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Key Columns (SYSKEYS)

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLDARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>POG_TP_ID</td>
<td>1</td>
<td>A</td>
<td>5,526,750</td>
</tr>
<tr>
<td>2</td>
<td>POG_TAX_ID</td>
<td>2</td>
<td>A</td>
<td>52</td>
</tr>
<tr>
<td>3</td>
<td>POG_SUB_TY</td>
<td>7</td>
<td>A</td>
<td>80</td>
</tr>
<tr>
<td>4</td>
<td>POG_END_DT</td>
<td>4</td>
<td>A</td>
<td>678</td>
</tr>
<tr>
<td>5</td>
<td>POG_PRO_TY</td>
<td>5</td>
<td>A</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>POG_OBJ_TY</td>
<td>6</td>
<td>A</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>POG_TAX_ID</td>
<td>3</td>
<td>A</td>
<td>52</td>
</tr>
<tr>
<td>8</td>
<td>POG_SQ</td>
<td>8</td>
<td>D</td>
<td>73</td>
</tr>
</tbody>
</table>

#### Index Package Dependencies (SYSPACKINDEX)

<table>
<thead>
<tr>
<th>NAME</th>
<th>CREATOR</th>
<th>CLUSTER ACT</th>
<th>CLUSTER SPO</th>
<th>INDEX SPACEN</th>
<th>FIRST KEYWORD</th>
<th>FULL KEYWORD</th>
<th>NLEVELS</th>
<th>CLUSTER RATIO</th>
<th>BPOOL</th>
<th>PSIZE</th>
<th>UNIQUE RULE</th>
<th>IDX TYPE</th>
<th>AVG KEYLEN</th>
<th>PAD</th>
<th>CPM PRES</th>
<th>IDX EXT TYPEN</th>
<th>STATTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAXAL1</td>
<td>N</td>
<td>N</td>
<td>AFFLGBL</td>
<td>1,294,479</td>
<td>1,294,479</td>
<td>4,863</td>
<td>3</td>
<td>47.88%</td>
<td>BP3</td>
<td>4,900</td>
<td>P</td>
<td>2</td>
<td>22</td>
<td>N</td>
<td>2014-09-14 19:32:44</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Key Columns (SYSKEYS)

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLDARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AFFLGBL</td>
<td>2</td>
<td>A</td>
<td>413,804</td>
</tr>
<tr>
<td>2</td>
<td>AFFLGBL</td>
<td>10</td>
<td>A</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>AFFLGBL</td>
<td>3</td>
<td>A</td>
<td>1,244,333</td>
</tr>
</tbody>
</table>

#### Indexes (SYNSINDEXES)

<table>
<thead>
<tr>
<th>NAME</th>
<th>CREATOR</th>
<th>CLUSTER ACT</th>
<th>CLUSTER SPO</th>
<th>INDEX SPACEN</th>
<th>FIRST KEYWORD</th>
<th>FULL KEYWORD</th>
<th>NLEVELS</th>
<th>CLUSTER RATIO</th>
<th>BPOOL</th>
<th>PSIZE</th>
<th>UNIQUE RULE</th>
<th>IDX TYPE</th>
<th>AVG KEYLEN</th>
<th>PAD</th>
<th>CPM PRES</th>
<th>IDX EXT TYPEN</th>
<th>STATTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFFLGBL</td>
<td>N</td>
<td>N</td>
<td>AFFLGBL</td>
<td>1,294,479</td>
<td>1,294,479</td>
<td>12,588</td>
<td>3</td>
<td>82.71%</td>
<td>BP3</td>
<td>4,098</td>
<td>D</td>
<td>2</td>
<td>22</td>
<td>N</td>
<td>2014-09-14 19:52:44</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The Mainframe Software Partner for the Next 50 Years
SQL Statement Text and Performance Recommendations

DECLARE C0011260 CURSOR FOR

SELECT 1 FROM TAX.VPRDOBLIG01, PRDOBLG1(3) 107M Rows/2.5M Pg
TAX.VAFFL_OBLG01 AFFLOBL3(0) YS 1.3M Rows/28.9K Pg
WHERE PDO_TP_ID = :H  Colcard - 5.5M
AND PDO_TAX_ID = :H  Colcard - 52
AND PDO_TXC_ID = :H  Colcard - 52
AND PDO_AUDTRL_REC_NO > 0  Colcard - 8.9M
AND PDO_SUB_TY = 10  Colcard - 60
AND ( PDO_AUDTRL_REC_NO = AFFL_PDO_ID  Colcard - 21
OR (PDO_AUDTRL_REC_NO = RPTG_ENT_PDO_ID))
AND PDO_CNCLSN_RSN_CD <> 'K'  Colcard - 3
AND SETU_ERR_CD = ''  Colcard - 3

FETCH FIRST 1 ROW ONLY
FOR FETCH ONLY

Perhaps re-write as a Union or two separate queries. Table AFFL_OBLG is a candidate for DB2 Compression. Consider adding an index to Table PRDOBLG to change the access to Index Only as follows:

Column 1 - PDO_TP_ID
2 - PDO_TAX_ID
3 - PDO_TXC_ID
4 - PDO_SUB_TY
5 - PDO_AUDTRL_REC_NO
6 - PDO_CNCLSN_RSN_CD
DB2 Activity by Query
### SQL Analysis - DB2 Explain

#### Access Path – Multiple Index Access

The Mainframe Software Partner for the Next 50 Years
### SQL Analysis - DB2 Catalog Statistics

- **Table**: XBP001A
- **Creator**: N
- **Cluster Act**: N
- **Cluster SPC**: N
- **Index SPACE**: XBP001A
- **First KEYCARD**: 7
- **Full KEYCARD**: 200,000
- **NLEAF**: 3
- **NLEVELS**: 3
- **CLUSTER RATIO**: 72.00%
- **BPOOL**: 1
- **PSIZE**: 64
- **UNIQUE RULE**: 2
- **IDX TYPE**: U
- **AVG KEYLEN**: 25
- **PAD**: 0
- **CPM PRES**: 0
- **IDX EXT TYP**: 0
- **STATTIME**: 2015-05-07 03:02:28

**Key Columns (SYSKEYS)**

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLCARD</th>
<th>NO</th>
<th>ORDERING</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WP_COMPANY</td>
<td>1</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>WP_BRANCH</td>
<td>24</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>WP_PCLREF_NUMBER</td>
<td>105,549</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>WP_PCLREF_CLUST_NO</td>
<td>11</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>WP_PCLREF_DESC</td>
<td>5</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>WP_TRAN_NO</td>
<td>54</td>
<td>A</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLCARD</th>
<th>NO</th>
<th>ORDERING</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WP_EFFECT_DATE</td>
<td>3,312</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>WP_AGENT_NUMBER</td>
<td>0,011</td>
<td>A</td>
<td></td>
</tr>
</tbody>
</table>

**Turn on DB2 Compression**

The Mainframe Software Partner for the Next 50 Years
DECLARE SEARCH7 CURSOR FOR
SELECT WIP.Company, WIP.Branch, WIP.PolRefNumber, WIP.PolRefQuotNo, WIP.PolRefDesc, 
    WIP.TabStatus, WIP.DecPrintInd, WIP.AssociateInd, WIP.QuoteInd, WIP.PolicyProv, WIP.GridInd, WIP.CrdAuthNo, WIP.CrdPaymentTy
FROM WIP
WHERE WIP.Company BETWEEN:H AND:H
AND WIP.Branch BETWEEN:H AND:H
AND WIP.PolRefNumber =:H
AND WIP.PolRefQuotNo =:H
AND WIP.PolRefDesc =:H
AND WIP.TerminalId =:H
AND ((WIP.Company || WIP.MasterBroker IN (:H,:H,:H,:H,:H,:H,:H,:H,:H,:H,:H):H))
OR WIP.MasterBroker >:H)
AND WIP.EffectDate BETWEEN:H AND:H
AND DATE (WIP.Timestamp) BETWEEN:H AND:H
AND (WIP.TransacnType IN (:H,:H,:H,:H,:H,:H,:H,:H,:H,:H,:H)
    AND WIP.ProductCode IN (:H,:H,:H,:H,:H,:H,:H,:H,:H,:H,:H)
    AND WIP.QuoteInd IN (:H,:H,:H)
ORDER BY DATE(WIP.Timestamp),
    WIP.EffectDate,
    WIP.TransacnType,
    WIP.UserID,
    WIP.PolRefNumber,
    WIP.PolRefQuotNo,
    WIP.PolRefDesc

Consider adding an index to Table WIP to improve the access as follows:

Column 1 - WIP.PolRefNumber
2 - WIP.PolRefQuotNo
DB2 Activity by Query

No Index Access
### SQL Analysis - DB2 Explain

#### Access Path – Tablespace Scan

![SQL Analysis Diagram](image)

- **Profile Name:** [Redacted]
- **Job Name:** [Redacted]
- **Initiated On:** 2015-06-09 18:08:37

#### SQL Statement Analysis

**SQL Statement Text**

**Plan Table Rows**

<table>
<thead>
<tr>
<th>QBLOCK NO</th>
<th>PLAN NO</th>
<th>METHOD</th>
<th>INDEX ONLY</th>
<th>CREATOR</th>
<th>TNAME</th>
<th>ACCESS CREATOR</th>
<th>ACCESS NAME</th>
<th>ACCESS TYPE</th>
<th>MATCH COLS</th>
<th>SORTN UJOC</th>
<th>SORTC UJOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>N</td>
<td></td>
<td>BCTSCAP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TSOLOCK MODE</th>
<th>PRE FETCH</th>
<th>MIXOPSEQ</th>
<th>TABNO</th>
<th>ACCESS DEGREE</th>
<th>JOIN DEGREE</th>
<th>ACCESS PGROUP_ID</th>
<th>JOIN PGROUP_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>IS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SORTNO PGROUP_ID</th>
<th>SORTN PGROUP_ID</th>
<th>COLUMNFN_EVAL</th>
<th>PAR MODE</th>
<th>COLLID</th>
<th>VERSION</th>
<th>PARENT QGROUPNO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PAGE RANGE</th>
<th>JOIN TYPE</th>
<th>MERGE JOIN COLS</th>
<th>CORRELATION NAME</th>
<th>GROUP MEMBER</th>
<th>PRIMARY ACCESSTYPE</th>
<th>TABLE TYPE</th>
<th>WHEN OPTIMIZE</th>
<th>HINT USED</th>
<th>OPTIMIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PROLICASA TABLE ROWS</th>
<th>TYPE</th>
<th>LEFT HAND SIDE</th>
<th>LEFT HAND PRO</th>
<th>LHS TABNO</th>
<th>LHS QENO</th>
<th>RIGHT HAND SIDE</th>
<th>RIGHT HAND PRO</th>
<th>RHS TABNO</th>
<th>RHS QENO</th>
<th>SEARCH Charge</th>
<th>KEY FIELD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AND</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>2</td>
<td>OR</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>3</td>
<td>EQUAL</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>VALUE</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>4</td>
<td>EQUAL</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>5</td>
<td>OR</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>6</td>
<td>EQUAL</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>VALUE</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>7</td>
<td>OR</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>8</td>
<td>OR</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>9</td>
<td>OR</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>10</td>
<td>OR</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

---

**The Mainframe Software Partner for the Next 50 Years**
Valid Index Available

The Mainframe Software Partner for the Next 50 Years
SQL Statement Text and Performance Recommendations

Unnecessary OR Predicates

The OR Clauses are causing DB2 to do a Tablespace Scan instead of using the existing index BCTBACCPK. Consider removing the OR Clauses or rewriting the SQL. Why are the OR Clauses in this SQL? Also, consider checking for spaces prior to executing SQL.
### DB2 Activity by Query

![DB2 Activity by Query Image](image)

The Mainframe Software Partner for the Next 50 Years
## SQL Analysis - DB2 Explain

**Access Path – Multiple Index Access (Union)**

![SQL Analysis - DB2 Explain](image-url)
### SQL Analysis - DB2 Catalog Statistics

#### Available Indexes

The Mainframe Software Partner for the Next 50 Years
### SQL Statement Text and Performance Recommendations

```sql
DECLARE CURSOR-GRVS-40 CURSOR FOR
  SELECT SUS_NO,SUS_SEQ,STS,APP_TYP,SUS_FRM,SUS_CHL,MCS_LCS_FLG,BUS_SYS_NO,BUS_NO,
  BUS_PSS_NO,OLD_BUS_PSS_NO,BUS_OPR_CDE,AP_MMO,MFM_SRV_CDE,AC,CCY,CTF_FLG,
  SUS_BR,OPR_TLR,ACO_TLR,SUS_DTE,VCH_NO,SUS_TME,JRNL_NO,CHO_DUE_DTE,CHO_CNT,
  SUS_AMT,SUS_BAL,APP_AMT,BAL_DIR,OPT_PRD_CDE,ECM_ID,FRE_TDE_AC_FLG,RMK_100,
  OWN_BR,OWN_BK,BAT_CTL_FLD,GEMB_STS,GEMS_SUS_BAL,
  GEMS_LAST_UPD_D,GEMS_LAST_UPD_D_BAT
FROM OCTGRVS
  OCTGRVS_I5(2)/ OCTGRVS_PK(2,1) ← 1.77M Rows/63.5K Pg
WHERE SUS_BR = :H
  AND (( SUS_NO > :H )
    OR (SUS_NO = :H
        AND SUS_SEQ > :H )
  AND SUS_DTE >= :H
  AND SUS_DTE <= :H
  AND STS = '0'
ORDER BY SUS_NO,
    SUS_SEQ
```

Consider adding an index to Table OCTGRVS to change the Multiple Index Access to a single index access and eliminate the Order By Sort as follows:

```
<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SUS_BR</td>
</tr>
<tr>
<td>2</td>
<td>STS</td>
</tr>
<tr>
<td>3</td>
<td>SUS_DTE</td>
</tr>
<tr>
<td>4</td>
<td>SUS_NO</td>
</tr>
<tr>
<td>5</td>
<td>SUS_SEQ</td>
</tr>
</tbody>
</table>
```

Add Index That is a Combination of the Two Indexes

---

The Mainframe Software Partner for the Next 50 Years
DB2 Activity by Query

>90 Hours of CPU/Yr Per Region!!!
### SQL Analysis - DB2 Explain

**Plan Table Rows**

<table>
<thead>
<tr>
<th>QBLOCK NO</th>
<th>PLAN NO</th>
<th>METHOD</th>
<th>INDEX ONLY</th>
<th>CREATOR</th>
<th>TNAME</th>
<th>ACCESS CREATOR</th>
<th>ACCESS NAME</th>
<th>ACCESS TYPE</th>
<th>MATCH COLS</th>
<th>SORTN UJOG</th>
<th>SORTC UJOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TSLOCK MODE** | **PRE FETCH** | **MIXOPSEQ** | **TABNO** | **ACCESS DEGREE** | **JOIN DEGREE** | **ACCESS PGROUP_ID** | **JOIN PGROUP_ID** |
| N          | 0       | -      | 1         | -          | -          | 0              | 0            |

**SORTC PGROUP_ID** | **SORTN PGROUP_ID** | **COLUMN FN_EVAL** | **PAR MODE** | **COLLID** | **VERSION** | **PARENT QBLOCKNO** |
| -          | -       | -      | -           | -          | 0          |                 |

**PAGE_RANGE** | **JOIN_TYPE** | **MERGE JOIN_COLS** | **CORRELATION_NAME** | **GROUP_MEMBER** | **PRIMARY ACCESTYPE** | **TABLE TYPE** | **WHEN OPTIMIZE** | **HINT USED** | **OPTHINT** |
| -          | -       | -      | DB2         |              | -          | T              |              |                |            |

**TABLE_ENCODE** | **TABLE_SCCSID** | **TABLE_MCCSID** | **TABLE_CCSDK** | **ROUTINE_ID** | **CTEREF** | **STMTTOKEN** | **PARENT PLAN NO** |
| E          | 37      | -2     | -2          | 0            | 0          | -            | 0            |

**Access Path – Non-Matching Index Scan W/Page Reference**

**Note** – All Leaf Pages in the Index are Scanned as well as Table Pages of Rows that Qualify.
### SQL Analysis - DB2 Catalog Statistics

<table>
<thead>
<tr>
<th>NAME</th>
<th>CREATOR</th>
<th>CLUSTER ACT</th>
<th>CLUSTER SPG</th>
<th>INDEX SPACE</th>
<th>FIRST KEYCARD</th>
<th>FULL KEYCARD</th>
<th>NLEAF</th>
<th>NLEVELS</th>
<th>CLUSTER RATIO</th>
<th>SPACES</th>
<th>POT ROWS</th>
<th>POT ROW COMP</th>
<th>PARTKEY COL NO</th>
<th>AVG ROW LEN</th>
<th>STATISTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDIPRNP1</td>
<td>Y</td>
<td>Y</td>
<td>EDIPRNP1</td>
<td>42344</td>
<td>480004</td>
<td>3020</td>
<td>3</td>
<td>100.0%</td>
<td>BPS</td>
<td>10218</td>
<td>60</td>
<td>100</td>
<td>2</td>
<td>122</td>
<td>2011-12-03</td>
</tr>
<tr>
<td>EDIPRNP2</td>
<td>N</td>
<td>N</td>
<td>EDIPRNP2</td>
<td>152</td>
<td>480886</td>
<td>3063</td>
<td>3</td>
<td>78.0%</td>
<td>BPS</td>
<td>14</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Key Columns (SYSKEYS)**

- COLNAME: PROV_PLAN_CD
- COLNO: 2
- ORDERING: 152
- COLCARD: 21,189

**Index Package Dependencies (SYSINDEXP)**

- DNAME: EDIPRNP1
- QUALIFIER: 1
- TYPE: P
- DNAME (PACKAGE): BATCHPRD
- INDENTIF: EDIPRNP1
- PRED: EDIPRNP1
- DEPEND: EDIPRNP1

**Index space statistics (for Partitioned and Non-Partitioned Indexes) (SYSINDEXF)**

1. NAME: EDIPRNP1
   - IXNAME: 480004
   - IXCREATOR: 1
   - CARD: 0
   - LEAFDIST: 0
   - FAROFFPOSF: 0
   - NEAROFFPOSF: 0
   - PERC FREE: 10
   - FREE PAGE: 0
   - STATISTIME: 2011-12-03

2. NAME: EDIPRNP2
   - IXNAME: 480886
   - IXCREATOR: 1
   - CARD: 0
   - LEAFDIST: 1024
   - FAROFFPOSF: 0
   - NEAROFFPOSF: 0
   - PERC FREE: 10
   - FREE PAGE: 0
   - STATISTIME: 2011-12-03

**Relational Partitions (SYSTABLESPACE)**

- NAME: EDIPRNP1
  - IXNAME: 480004
  - IXCREATOR: 1
  - CARD: 0
  - FAR IND REF: 0
  - NEAR IND REF: 0
  - PERC ACTIVE: 0
  - PERC DROP: 0
  - PAGE SIZE: 40
  - PERC FREE: 20
  - FREE PAGE: 5
  - STATISTIME: 2011-12-03

**Index EDIPRNP2 is 3063 Leaf Pages**

The Mainframe Software Partner for the Next 50 Years
SQL Statement Text and Performance Recommendations

- CICS - DBRM - 612 Select (Stmt Count 819)

```
SELECT 'Y'
  INTO :H
FROM EDIPRNP2(0)
WHERE (( PROV_NPI = :H
          OR PYEE_NPI = :H)
       OR ( PROV_PLAN_CD = :H
            AND PROV_NR = :H))
AND SUBM_ID = :H
FETCH FIRST ROW ONLY
WITH UR
```

Consider adding an index to Table PROV.EDI_PROV_NPI to change the Non-Matching Index Scan to a Matching Index Only Scan as follows:

- Column 1 - SUBM_ID
- 2 - PROV_NR
- 3 - PROV_PLAN_CD
- 4 - PROV_NPI
- 5 - PYEE_NPI

The Size of this New Index would be a little more than Double the Size of Index EDIPRNP22
Measurement Session Data - CICS
DB2 Stored Procedure SQL Activity
DB2 Stored Procedure SQL Activity

>100 Hours of CPU/Yr Per Region!!!
SQL Analysis - DB2 Explain

<table>
<thead>
<tr>
<th>QBLOCK NO</th>
<th>PLAN NO</th>
<th>ME METHOD</th>
<th>INDEX ONLY</th>
<th>CREATOR</th>
<th>TNAME</th>
<th>ACCESS CREATOR</th>
<th>ACCESS NAME</th>
<th>ACCESS TYPE</th>
<th>MATCH COLS</th>
<th>SORTN UJOG</th>
<th>SORTC UJOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0 N</td>
<td></td>
<td>BSELVAL</td>
<td>BSELVAL</td>
<td>BSELVAL1</td>
<td>I</td>
<td>2</td>
<td>NNNN</td>
<td>NNNN</td>
<td>NNNN</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1 N</td>
<td></td>
<td>BSELVAL</td>
<td>SELTXTX</td>
<td>SELTXTX1</td>
<td>I</td>
<td>2</td>
<td>NNNN</td>
<td>NNNN</td>
<td>NNNN</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0 N</td>
<td></td>
<td>BSELVAL</td>
<td>BSELVAL</td>
<td>BSELVAL1</td>
<td>I</td>
<td>2</td>
<td>NNNN</td>
<td>NNNN</td>
<td>NNNN</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>_ N</td>
<td></td>
<td>BTXTPCE</td>
<td>BTXTPCE</td>
<td>BTXTPCE1</td>
<td>I</td>
<td>1</td>
<td>NNNN</td>
<td>NNNN</td>
<td>NNNN</td>
</tr>
</tbody>
</table>

Access Path Consists of a Join and Two SubSelects and No Index Only Access
Consider Turning DB2 Compression On for any Table with more than 40 to 50 Pages

Note – Use the DSN1COMP Utility to determine Compression Ratio – if less than 25%, don’t turn on
SQL Statement Text and Performance Recommendations

```
SELECT JSTX.BEN_SEL_ID,  
    JSTX.EFF_DT  
INTO :H,:H 
FROM BEN_SEL_TXT_XREF JSTX, 
    SELTXTX1(2)  
BSELVAL1(2)  
WHERE (JVAL.GRP_NR = :H)  
    AND (JVAL.DIV_FROM <= :H)  
    AND (JVAL.DIV_THRU >= :H)  
    AND (JVAL.INCL_VAL_SW = 'Y')  
    AND (NOT EXISTS (SELECT 'X'  
                    FROM BEN_SEL_VAL JVAL2  
                    BSELVAL1(2)  
                    WHERE (JVAL2.BEN_SEL_ID = JVAL.BEN_SEL_ID)  
                        AND (JVAL2.GRP_NR = :H)  
                        AND (JVAL2.DIV_FROM <= :H)  
                        AND (JVAL2.DIV_THRU > :H)  
                        AND (JVAL2.INCL_VAL_SW = 'N')))  
    AND (JSTX.BEN_SEL_ID = JVAL.BEN_SEL_ID)  
    AND (JSTX.PROD_TST_CD = 'P')  
    AND (JSTX.EFF_DT <= :H)  
    AND ((JSTX.EFF_THRU_DT IS NULL  
          OR (JSTX.EFF_THRU_DT >= :H))  
    AND (EXISTS (SELECT 'X'  
                    FROM BEN_TXT_PCE JPCE  
                    BTXTPCE1(1)  
                    WHERE (JSTX.BEN_TXT_PCE_ID = JPCE.BEN_TXT_PCE_ID)  
                        AND (JPCE.HLTH_DENT_CD = :H)))) 
FETCH FIRST 1 ROWS ONLY
```

Turn DB2 Compression on for Tables ACSIS.BEN_SEL_VAL and ACSIS.BEN_TXT_PCE. Also, Table ACSIS.BEN_SEL_TXT_XREF is only 83% Compressed - run a Reorg followed by Runstats. INCLUDE Column HLTH_DENT_CD in Index BTXTPCE1 for Index Only Access in the Second SubSelect. Consider adding an index to Table ACSIS BEN_SEL_VAL to change the access of the First SubSelect to Index Only as follows:

```
Column 1 - GRP_NR  
2 - DIV_FROM  
3 - DIV_THRU  
4 - INCL_VAL_SW  
5 - BEN_SEL_ID
```

Note – INCLUDE (BSELVAL1) could be used for Index Only Access
DB2 SQL Activity

>10 Hours of CPU per Year per Region
SQL Analysis

DB2 Explain

Non-Matching Index Scan

DB2 Catalog Statistics

The Mainframe Software Partner for the Next 50 Years
SQL Statement Text and Performance Recommendations

- CICS DBRM - 2740 Select (Stmt Count 76)

SELECT INT(CDE_DSC)
INTO :H
FROM CAD_CDE_VAL CIP0XA71(0) ← 6,379 Rows/63 Pg
WHERE SUBSTR(CDE_SET_ID,1,5) = 'CIO7H'
AND CDE_VAL = 'VALUE1' ← Colcard 3228

Change the SUBSTR to WHERE CDE_SET_ID LIKE 'CIO7H%' to change the access from a Non-Matching Index Scan to a Matching Index Scan with a MATCHCOLS of 1.

Another option would be to also create an index to increase the MATCHCOLS as follows:

Column 1 - CDE_VAL
2 - CDE_SET_ID

In DB2 V11, SUBSTR (from position 1) becomes indexable – similar to LIKE
DB2 SQL Activity

>100 Hours of CPU/Yr Per Region!!!
### SQL Analysis

#### DB2 Explain

- **Profile Name:** [Redacted]
- **Job Name:** [Redacted]
- **Initialized On:** 2013-03-11 09:10:21

#### Plan Table Rows

<table>
<thead>
<tr>
<th>GLOCK NO</th>
<th>PLAN NO</th>
<th>METHOD</th>
<th>INDEX ONLY</th>
<th>CREATOR</th>
<th>THNAME</th>
<th>ACCESS CREATOR</th>
<th>ACCESS NAME</th>
<th>ADDRESS TYPE</th>
<th>MATCH COLS</th>
<th>SORTN_UGID</th>
<th>SORTU_UGID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Table Enqueue:**
- **Table SCOSID:**
- **Table MOOSID:**
- **Table DCOSID:**
- **Routine ID:**
- **CTREF:**
- **STMTTOKEN:**
- **PARENT PLAN NO:**

#### DB2 Catalog Statistics

<table>
<thead>
<tr>
<th>Table</th>
<th>DNAME</th>
<th>TNAME</th>
<th>CARDF</th>
<th>NUM BCP METS</th>
<th>SPLIT ROWS</th>
<th>NPAGES</th>
<th>POT PAGES</th>
<th>POT ROW COMP</th>
<th>PART NO</th>
<th>PARTKEY COL NO</th>
<th>AVG ROW LEN</th>
<th>STATSTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>XYPK1</td>
<td>0</td>
<td>20774142</td>
<td>0</td>
<td>303210</td>
<td></td>
<td>80</td>
<td>80</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>2013-03-03</td>
</tr>
</tbody>
</table>

#### Indexes (SYSPINDEXES)

- **XYPK1**
- **XYPK2**

#### PK Columns (SYSPKYS)

<table>
<thead>
<tr>
<th>GCOL</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLCARD</th>
<th>GCOLCARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PLCY_SEQ_NUM</td>
<td>1</td>
<td>A</td>
<td>169</td>
<td>364</td>
</tr>
<tr>
<td>2</td>
<td>ROWEFF_DATE</td>
<td>2</td>
<td>A</td>
<td>253</td>
<td>535</td>
</tr>
<tr>
<td>3</td>
<td>VBR_SEQ_NUM</td>
<td>3</td>
<td>A</td>
<td>140</td>
<td>2165</td>
</tr>
<tr>
<td>4</td>
<td>RISK_SEQ_NUM</td>
<td>4</td>
<td>A</td>
<td>140</td>
<td>2165</td>
</tr>
<tr>
<td>5</td>
<td>XPRG2_SEQ_NUM</td>
<td>5</td>
<td>A</td>
<td>140</td>
<td>2165</td>
</tr>
</tbody>
</table>

#### Index Proc Log Dependencies (SYSPACKINDEX)

- **YPRK1**
- **YPRK2**

#### Index space statistics (SYSPKYSINDEXSPACESTATS)

<table>
<thead>
<tr>
<th>Name</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLCARD</th>
<th>GCOLCARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>YPRK1</td>
<td>YPRK2</td>
<td>364169</td>
<td>2412514</td>
<td>20130801.4</td>
<td>1240748</td>
</tr>
</tbody>
</table>

- **Index space statistics for Partitioned and Non-Partitioned Indexes (SYSPKYSINDEXPART)
- **Tablespaces (SYSPALTSPACES)
- **Tablespaces Partitions (SYSPALTSPACESTPART)**
SQL Statement Text and Performance Recommendations

DECLARE CVRG_CURSOR CURSOR FOR
    SELECT A.ROW_EXPR_DATE,
        B.CVG_DSPLY_CODE,
        B.CVG_DSPLY_SEQ_NUM,
        B.DESC_CODE
    FROM RISK_CVG A,
        XPKRISK_CVG(1)
    CVG_DESC B
    WHERE A.PLCY_SEQ_NUM = :H
        AND :H BETWEEN A.ROW_EFF_DATE AND A.ROW_EXPR_DATE
        AND A.VER_SEQ_NUM = 0
        AND A.RISK_SEQ_NUM = :H
        AND A.ROW_EXPR_DATE > :H
        AND A.CLASS_CODE <> '009460'
        AND A.CVG_CODE = B.CVG_CODE
        AND A.CVG_TYPE_CODE = B.CVG_TYPE_CODE
        AND A.CVG_SUB_TYPE_CODE = B.CVG_SUB_TYPE_CODE
    ORDER BY B.CVG_DSPLY_SEQ_NUM
    WITH UR

Consider re-writing the BETWEEN predicate to change from a Stage 2 Predicate to a Stage 1 Predicate as follows:

    AND A.ROW_EFF_DATE <= :H
    AND A.ROW_EXPR_DATE >= :H

In DB2 V11, this is re-written to be indexable
Another consideration would be to add an index to Table RISK_CVG to increase the MATCHCOLS from 1 to 4 and change the access to Index Only as follows:

Column 1 - PLCY_SEQ_NUM
2 - RISK_SEQ_NUM
3 - VER_SEQ_NUM
4 - ROW_EXPR_DATE
5 - ROW_EFF_DATE
6 - CLASS_CODE
7 - CVG_CODE
8 - CVG_TYPE_CODE
9 - CVG_SUB_TYPE_CODE

Run Time Statistics after Index was added

<table>
<thead>
<tr>
<th>Type</th>
<th>Rows Processed</th>
<th>Rows Examined</th>
<th>Rows-Q Stage1</th>
<th>Rows-Q Stage2</th>
<th>Rows Inserted</th>
<th>Rows Deleted</th>
<th>Rows Updated</th>
<th>Get Page Requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDX</td>
<td>127</td>
<td>10,969</td>
<td>19,716</td>
<td>9,857</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8,129</td>
</tr>
<tr>
<td>SEQD</td>
<td>9,859</td>
<td>9,859</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4,970</td>
</tr>
</tbody>
</table>

Over 90% Reduction in Getpages

Higher Throughput & Faster Response Time
DB2 SQL Activity

Poor Performing Delete Statement
An Average of Over 19,000 Getpages per Delete
SQL Analysis

DB2 Explain

Non-Matching Index-Only Scan With Sequential Prefetch

DB2 Catalog Statistics

The Mainframe Software Partner for the Next 50 Years
DELETE FROM SHIP_ITEMS XSHPITM1(0) S ← 2.9M Rows/103K Pg
WHERE AS_ORD_DOC_NBR = :H AND SHIPMENT_TYPE = :H

Consider adding Column BRANCH as a Predicate or add an index to Table SHIP_ITEMS to change the Non-Matching Index Scan of over 19K Index Pages to a Matching Index Scan as follows:

Column 1 - AS_ORD_DOC_NBR
2 - SHIPMENT_TYPE

Probably save most of the 45 seconds on this one. 300 hours/yr

Change the SQL to Include the First Column (BRANCH) of Index XSHPITM1 or Add an Index to Eliminate the Non-Matching Index Scan
DB2 SQL Activity

Update Statement

The Mainframe Software Partner for the Next 50 Years
### SQL Analysis - DB2 Explain

The table below shows the SQL statement analysis generated by DB2 Explain:

<table>
<thead>
<tr>
<th>QBLOCK NO</th>
<th>PLAN NO</th>
<th>METHOD</th>
<th>INDEX ONLY</th>
<th>CREATOR</th>
<th>TNAME</th>
<th>ACCESS CREATOR</th>
<th>ACCESS NAME</th>
<th>ACCESS TYPE</th>
<th>MATCH COLS</th>
<th>SORTN UJOG</th>
<th>SORTG UJOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td>0</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Profile Name:** [Redacted]  
**Job Name:** [Redacted]  
**Initiated On:** 2015-04-03 11:55:34

**SQL Statement Analysis**

<table>
<thead>
<tr>
<th>Method</th>
<th>Index Only</th>
<th>TNAME</th>
<th>Access Creator</th>
<th>Access Name</th>
<th>Access Type</th>
<th>Match COLS</th>
<th>Sortn UJOG</th>
<th>Sortg UJOG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Details:**

- **TNAME:** [Redacted]  
- **Access Creator:** [Redacted]

**Other Columns:**

- **TSLOCK MODE:** [Redacted]  
- **PRE FETCH:** [Redacted]  
- **MIXGSEQ:** [Redacted]  
- **TABNO:** [Redacted]  
- **ACCESS DEGREE:** [Redacted]  
- **JOIN DEGREE:** [Redacted]  
- **ACCESS PGROUP ID:** [Redacted]  
- **JOIN PGROUP ID:** [Redacted]  

**More Columns:**

- **SORTC PGROUP ID:** [Redacted]  
- **SORTN PGROUP ID:** [Redacted]  
- **COLUMN FN_EVAL:** [Redacted]  
- **PAR MODE:** [Redacted]  
- **COLLID:** [Redacted]  
- **VERSION:** [Redacted]  
- **PARENT QBLOCK NO:** [Redacted]  
- **PAGE RANGE:** [Redacted]  
- **JOIN TYPE:** [Redacted]  
- **MERGE JOIN GOLS:** [Redacted]  
- **CORRELATION NAME:** [Redacted]  
- **GROUP MEMBER:** [Redacted]  
- **PRIMARY ACCESTYPE:** [Redacted]  
- **TABLE TYPE:** [Redacted]  
- **WHEN OPTIMIZE:** [Redacted]  
- **HINT USED:** [Redacted]  
- **OPTHINT:** [Redacted]  

**Additional Details:**

- **TABLE ENCODE:** [Redacted]  
- **TABLE SGCSID:** [Redacted]  
- **TABLE MGCSID:** [Redacted]  
- **TABLE DGCSID:** [Redacted]  
- **ROUTINE ID:** [Redacted]  
- **CTEREF:** [Redacted]  
- **STMNTOKEN:** [Redacted]  
- **PARENT PLAN NO:** [Redacted]
SQL Analysis - DB2 Catalog Statistics

Available Indexes

Primary Key

The Mainframe Software Partner for the Next 50 Years
### SQL Analysis - DB2 Catalog Statistics

#### Table 1

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLCARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>UPDATE_DATE</td>
<td>65</td>
<td>D</td>
<td>915</td>
</tr>
<tr>
<td>2</td>
<td>QTY_ALLOCATED</td>
<td>22</td>
<td>A</td>
<td>976</td>
</tr>
<tr>
<td>3</td>
<td>QTY_SHIPCAND</td>
<td>55</td>
<td>A</td>
<td>607</td>
</tr>
<tr>
<td>4</td>
<td>QTY_BACK_ORDER</td>
<td>34</td>
<td>A</td>
<td>400</td>
</tr>
<tr>
<td>5</td>
<td>ORDER_TYPE</td>
<td>20</td>
<td>A</td>
<td>0</td>
</tr>
</tbody>
</table>

#### Table 2

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLCARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CUST_NBR</td>
<td>2</td>
<td>A</td>
<td>21,000</td>
</tr>
<tr>
<td>2</td>
<td>ENTRY_NBR</td>
<td>1</td>
<td>D</td>
<td>951,565</td>
</tr>
<tr>
<td>3</td>
<td>ENTRY_LINE_NBR</td>
<td>3</td>
<td>A</td>
<td>5,246</td>
</tr>
<tr>
<td>4</td>
<td>BRANCH</td>
<td>6</td>
<td>A</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>QTY_BACK_ORDER</td>
<td>24</td>
<td>A</td>
<td>406</td>
</tr>
<tr>
<td>6</td>
<td>QTY_ORDER</td>
<td>26</td>
<td>A</td>
<td>948</td>
</tr>
<tr>
<td>7</td>
<td>QTY_PREFBOOK</td>
<td>37</td>
<td>A</td>
<td>300</td>
</tr>
<tr>
<td>8</td>
<td>QTY_SHIPED</td>
<td>30</td>
<td>A</td>
<td>500</td>
</tr>
</tbody>
</table>

#### Table 3

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLCARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ORDLINE_STATUS_CODE</td>
<td>44</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>CUST_NBR</td>
<td>2</td>
<td>A</td>
<td>21,000</td>
</tr>
<tr>
<td>3</td>
<td>ENTRY_NBR</td>
<td>1</td>
<td>D</td>
<td>951,565</td>
</tr>
<tr>
<td>4</td>
<td>CREATE_DATE</td>
<td>51</td>
<td>A</td>
<td>1,130</td>
</tr>
<tr>
<td>5</td>
<td>CREATE_USER</td>
<td>40</td>
<td>A</td>
<td>182</td>
</tr>
<tr>
<td>6</td>
<td>UPDATE_DATE</td>
<td>55</td>
<td>A</td>
<td>915</td>
</tr>
<tr>
<td>7</td>
<td>UPDATE_USER</td>
<td>60</td>
<td>A</td>
<td>232</td>
</tr>
</tbody>
</table>

#### Key Columns (SYSKEYS)

- XORDLN4
- XORDLN5
- XORDLN6
- XORDLN7
- XORDLN8
- XORDLN9
- XORDLN10

---

The Mainframe Software Partner for the Next 50 Years
SQL Statement Text and Performance Recommendations

UPDATE ORDLINE

SET 
QTY_ORDER = :H,
QTY_ALLOCATED = :H,
QTY_BACK_ORDER = :H,
QTY_PREBOOK = :H,
UPDATE_USER = :H,
UPDATE_PROG = :H,
UPDATE_DATE = CURRENT_DATE,
UPDATE_TIME = CURRENT_TIME

WHERE CUST_NBR = :H
AND ENTRY_NBR = :H
AND ENTRY_LINE_NBR = :H
AND UPDATE_DATE = :H
AND UPDATE_TIME = :H

Column QTY_PREBOOK is in Index XORDLN3. Columns QTY_ALLOCATED and QTY_BACK_ORDER are in Index XORDLN4. Columns QTY_ORDER, QTY_BACK_ORDER, and QTY_PREBOOK are in Index XORDLN6. Columns UPDATE_USER and UPDATE_DATE are in Index XORDLN7. Index XORDLN8 has an overall cardinality of only 50 – consider adding columns. Also, consider turning on DB2 Compression. Consider using Extended Variable Indicators.
Batch
Measurement Session Data - Batch

Daily Batch Job Consuming Over 6 Hours of CPU per Day

Nearly 1700 Hours of CPU per Year!

The Mainframe Software Partner for the Next 50 Years
DB2 Activity by Query

Over 92% of the CPU on One SQL Statement

The Mainframe Software Partner for the Next 50 Years
### SQL Analysis - DB2 Explain

**Many Tables Being Joined; Only a Few Index Only Accesses**
Seven of the Thirteen Tables Don’t Have DB2 Compression Turned On.
DECLARE FROMMV-PCM CURSOR FOR
SELECT DISTINCT 'STK' STK_SOS,T2580.T617_FNC_TYP_CD,T2580.T616_VBU_NBR,T2594.T024_ITM_NBR,
CASE T2580.MOD_IDF_TXT WHEN ' ' THEN T2605.SLL_ITM_IDF ELSE T2580.MOD_IDF_TXT END,
T937.BRN_NME
FROM .T2591_GIN_BAR_CD T2591,
.T2580_GIN_VBU T2580,
.T024_ITM T024,
.T010_BAR_CD T010,
.T937_BRN T937,
.T679_VBU_ITM_MSR T679,
.T4482_GIN_PRD_CGY T4482,
.T4480_PRD_CGY T4480
WHERE T024.T024_ITM_NBR = T2594.T024_ITM_NBR
AND T024.T028_ITM_TYP_CD > 1
AND T2594.T1989_TGT_CRY_CD = 1
AND T679.T617_FNC_TYP_CD = T2594.T617_FNC_TYP_CD
AND T679.T679_BGN_EFC_DT <= CURRENT DATE
AND T249A.ITM_NBR = T2594.T249A_ITM_NBR
AND T2580.T616_VBU_NBR = T2594.T616_VBU_NBR
AND T2580.T2580_GIN_ID = T2594.T2580_GIN_ID
AND T2580.T616_VBU_NBR = T2591.T616_VBU_NBR
AND T2580.T2580_GIN_ID = T2591.T2580_GIN_ID
AND T2580.T616_VBU_NBR = T2591.T616_VBU_NBR
AND T2580.T2580_GIN_ID = T2591.T2580_GIN_ID
AND T2580.T616_VBU_NBR = T2591.T616_VBU_NBR
AND T2580.T2580_GIN_ID = T2591.T2580_GIN_ID
AND T2580.T616_VBU_NBR = T2591.T616_VBU_NBR
AND T2580.T2580_GIN_ID = T2591.T2580_GIN_ID
AND T2580.T616_VBU_NBR = T2591.T616_VBU_NBR
AND T2580.T2580_GIN_ID = T2591.T2580_GIN_ID
AND T2580.LWS_CSM_UNT_IDC = 'Y'
AND T2591.T1989_TGT_CRY_CD = 2
AND T4482.T4480_PRD_CGY_NBR = T4482.T4480_PRD_CGY_NBR
AND ( T2580.MOD_IDF_TXT = :H
OR T2605.SLL_ITM_IDF = :H)
AND T2594.T616_VBU_NBR = :H
UNION ALL

SQL Statement Text and Performance Recommendations
FROM LOWES.T2591_GIN_BAR_CD T2591, T2591I1(1)
    T2580_GIN_VBU T2580,
    T2594_GIN_VBU_ITM T2594,
    T4482_GIN_PRD_CGY T4482,
    T024_ITM T024,
    T1677_ITM_PRL T1677,
    T331_CTG_ITM T331,
    T249A_WK_ITM_HRC T249A,
    T4480_PRD_CGY T4480
ORDER BY 2,
    3,
    4,
    5
WITH UR

WHERE T331.SOS_ITM_NBR = T024.T024_ITM_NBR
AND T024.T028_ITM_TYP_CD = 1
AND T331.T331_ITM_STS_CD <> 8
AND T331.T617_FNC_TYP_CD = 2
AND T331.SOS_ITM_NBR = T249A.ITM_NBR
AND T331.T616_VBU_NBR = T2591.T616_VBU_NBR
AND T331.UPC_ID = T2591.T2591_BAR_CD_ID
AND T2580.T616_VBU_NBR = T2591.T616_VBU_NBR
AND T2580.T2580_GIN_ID = T2591.T2580_GIN_ID
AND T2580.T617_FNC_TYP_CD = T2594.T617_FNC_TYP_CD
AND T2580.T2580_GIN_ID = T2594.T2580_GIN_ID
AND T2580.T617_FNC_TYP_CD = T2594.T617_FNC_TYP_CD
AND T2580.T2580_GIN_ID = T2594.T617_FNC_TYP_CD
AND T2580.T617_FNC_TYP_CD = 2
AND T4482.T4480_PRD_CGY_NBR = T4480.T4480_PRD_CGY_NBR
AND T331.T331_MOD_ID = :H

AND T024.T231_ITM_STS_CD <> 8
AND T331.SOS_ITM_NBR = T1677.T024_ITM_NBR
AND T2591.T1989_TGT_CRY_CD = 1
AND T331.SOS_ITM_NBR = T2594.T024_ITM_NBR
AND T331.T617_FNC_TYP_CD = T2591.T617_FNC_TYP_CD
AND T2580.T2580_GIN_ID = T2591.T2580_GIN_ID
AND T2580.T617_FNC_TYP_CD = T2591.T617_FNC_TYP_CD
AND T2580.T617_FNC_TYP_CD = T2594.T617_FNC_TYP_CD
AND T2580.LWS_CSM_UNT_IDC = 'Y'
AND T2580.T616_VBU_NBR = T2594.T616_VBU_NBR
AND T2580.T617_FNC_TYP_CD = T2594.T617_FNC_TYP_CD
AND T2580.T617_FNC_TYP_CD = 2
AND T4482.T4480_PRD_CGY_NBR = T4480.T4480_PRD_CGY_NBR
AND T331.T331_MOD_ID = :H

ORDER BY 2,
    3,
    4,
    5
WITH UR

Thank Goodness For Correlation Names!!!
SQL Statement Text and Performance Recommendations

Consider turning on DB2 Compression for Tables T937_BRN, T679_VBU_ITM_MSR, T4482_GIN_PRD_CGY, T4480_PRD_CGY, T1677_ITM_PRL, T024_ITM, T010_BAR_CD.

Consider making the following index changes:

**INCLUDE** Columns LWS_CSM_UNT_IDC and T937_BRN_CD in Unique Index T2580I0.
Add Columns T2580_GIN_ID, T616_VBU_NBR, T617_FNC_TYP_CD, and T1989_TGT_CRY_CD to the end of Non-Unique Index T2591I1 for Index Only access.
**INCLUDE** Columns T028_ITM_TYP_CD and T231_ITM_STS_CD in Unique Index T024I0 for Index Only access.
**INCLUDE** Column END_EFC_DT in Unique Index T678I0 for Index Only access.
**INCLUDE** Columns T2582_GIN_STS_CD and T4480_PRD_CGY_NBR in Unique Index T4482I0 for Index Only access.
**INCLUDE** Column CTG_MGT_ETR_IDC in Unique Index T4480I0 for Index Only access.
**INCLUDE** Columns SOS_ITM_NBR, UPC_ID, T231_ITM_STS_CD, and BRN_NME in Unique Index T331I0 for Index Only access.

Add the following index to Table T2594_GIN_VBU_ITM to change the access to Index Only:

Column 1 - T616_VBU_NBR
2 - T2580_GIN_ID
3 - T617_FNC_TYP_CD
4 - T1989_TGT_CRY_CD
5 - T024_ITM_NBR

Most of the Access Paths Are Changed to Index Only

Also, Indexes T024I4 and T024I5 have very low cardinality and should be investigated for possibly adding columns to increase overall cardinality and improve access efficiency.
Measurement Session Data - Batch

20 Hours of CPU per Year
Over 95% of the CPU on One SQL Statement

On Average, Over 1450 Getpages per Select
SQL Analysis - DB2 Explain

Index Only – But…

MATCHCOLS of 1 and Sequential Prefetch Turned On
### SQL Analysis - DB2 Catalog Statistics

#### Catalog Statistics

<table>
<thead>
<tr>
<th>NAME</th>
<th>CREATOR</th>
<th>CLUSTER ACT</th>
<th>CLUSTER SPO</th>
<th>INDEX SPACE</th>
<th>FIRST KEYCARD</th>
<th>FULL KEYCARD</th>
<th>NLEAF</th>
<th>NLEVELS</th>
<th>CLUSTER RATIO</th>
<th>DPOOL</th>
<th>PGSIZE</th>
<th>UNIQ RULE</th>
<th>AVG KEYLEN</th>
<th>PAD</th>
<th>CPM PRES</th>
<th>IDX EXT TYP</th>
<th>STATETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>XEDINV1</td>
<td>Y</td>
<td>Y</td>
<td>XEDINV1</td>
<td>22.571</td>
<td>10,230,110</td>
<td>236,797</td>
<td>4</td>
<td>100.00%</td>
<td>0%</td>
<td>573</td>
<td>4,099</td>
<td>U</td>
<td>2</td>
<td></td>
<td></td>
<td>N</td>
<td>2015-03-28 20:50:48</td>
</tr>
</tbody>
</table>

#### Key Columns (SYSKEYS)

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLCARD D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CUST_NBR</td>
<td>2</td>
<td>A</td>
<td>22.571</td>
</tr>
<tr>
<td>2</td>
<td># EIR_NBR</td>
<td>1</td>
<td>A</td>
<td>1,490,456</td>
</tr>
<tr>
<td>3</td>
<td># INVOICE_DATE</td>
<td>3</td>
<td>A</td>
<td>264</td>
</tr>
<tr>
<td>4</td>
<td># INVOICE_NBR</td>
<td>4</td>
<td>A</td>
<td>1,490,456</td>
</tr>
<tr>
<td>5</td>
<td># INVOICE_LINE_SEQ</td>
<td>5</td>
<td>A</td>
<td>1,728</td>
</tr>
<tr>
<td>6</td>
<td># EDI10 REC_ID</td>
<td>6</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td># EDI10 REC_ID_SEQ</td>
<td>7</td>
<td>A</td>
<td>5</td>
</tr>
</tbody>
</table>

#### Index Package Dependencies (SYSPACKDEP)

| XEDINV5 | OR | Y | N | XEDINV2 | 2 | 3 | 22,426 | 3 | 100.00% | 0% | 4,098 | D | 2 | 2 | N | 2015-03-28 23:00:18 |

#### Key Columns (SYSKEYS)

<table>
<thead>
<tr>
<th>COLSEQ</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLCARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CORRECTION_IND</td>
<td>12</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>APPROVAL_IND</td>
<td>8</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>TRANSMITTED_IND</td>
<td>19</td>
<td>A</td>
<td>2</td>
</tr>
</tbody>
</table>

#### Index Package Dependencies (SYSPACKDEP)

| XEDINV5 | N | N | XEDINV3 | 2 | 3 | 34 | 4 | 64.56% | 0% | 4,098 | D | 2 | 16 | N | 2015-03-28 20:50:48 |
| XEDINV4 | N | N | XEDINV4 | 132 | 83,840 | 23,002 | 4 | 84.73% | 0% | 4,098 | D | 2 | 17 | N | 2015-03-28 20:50:48 |
SQL Statement Text and Performance Recommendations

- Batch - DBRM - 332 Select (Stmt Count 4,270)

```
SELECT VALUE(COUNT(*),0)
  INTO :H
FROM EDI_INVOICE
WHERE CUST_NBR = :H
  AND INVOICE_DATE = :H
  AND INVOICE_NBR = :H
```

Consider adding an index to Table EDI_INVOICE to increase the MATCHCOLS from 1 to 3, still be Index Only, but much faster since now on average there are over 1450 Getpages per Select and Sequential Prefetch is turned on making the average elapsed time for this SQL around .07 seconds which is not very good - as follows:

- Column 1 - CUST_NBR
- Column 2 - INVOICE_NBR
- Column 3 - INVOICE_DATE

Also, consider adding WITH UR to the end of the SQL to eliminate any locking.

Also, Index XEDINV2 has an overall cardinality of only 3 - consider adding more columns based on the SQL using this index in Packages ORB614, ORC087, and ORC089.
Measurement Session Data - Batch

Daily Batch Job Responsible For Over 100 Hours of CPU per Year

Over 250 Hours of Stretch Time per Year

The Mainframe Software Partner for the Next 50 Years
Nearly 50% of the CPU on this One SQL
Good Access Path with a MATCHCOLS of 5
### SQL Analysis - DB2 Catalog Statistics

#### Key Columns (SYSKEYS)

<table>
<thead>
<tr>
<th>GOLSEQ</th>
<th>COLNAME</th>
<th>COLNO</th>
<th>ORDERING</th>
<th>COLGARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DOCUMENT_TYPE</td>
<td>14 A</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>DOCUMENT_NBR</td>
<td>2 A</td>
<td>311,128</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>DOCUMENT_BR</td>
<td>3 A</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>DOCUMENT_SEQ</td>
<td>9 A</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>LINE_SEQ</td>
<td>10 A</td>
<td>1,260</td>
<td></td>
</tr>
</tbody>
</table>

#### Index Package Dependencies (SYSPACKDEP)

Index package real time statistics (SYSPACKDEP)

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>COUNT</th>
<th>STATTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>XAPINRX1</td>
<td>76</td>
<td>Tables and Indexes Look Good</td>
<td></td>
</tr>
</tbody>
</table>

#### Catalog Statistics

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>COUNT</th>
<th>STATTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>XAPINRX1</td>
<td>76</td>
<td>Tables and Indexes Look Good</td>
<td></td>
</tr>
</tbody>
</table>
SQL Statement Text and Performance Recommendations

DECLARE INVOICE_CSR CURSOR FOR
SELECT INVOICE_NBR, INVOICED_QTY, DOCUMENT_NBR, DOCUMENT_BR,
   DOCUMENT_SEQ, LINE_SEQ, RECEIVING_NBR, SCAN_ID,
   INVOICE_TYPE, INVOICE_DTE
FROM AP_INV_RECV_XRF XAPINRX3(5) ← 7M Rows/89K Pg
WHERE DOCUMENT_TYPE = :H
   AND DOCUMENT_NBR = :H ← Colcard 311K
   AND DOCUMENT_BR = :H
   AND DOCUMENT_SEQ = :H
   AND LINE_SEQ = :H ← Colcard 440K
   AND RECEIVING_NBR = :H<br> Consider adding Column RECEIVING_NBR or Column SCAN_ID to the end of Index XAPINRX3 to increase the MATCHCOLS.
   AND SCAN_ID = :H ← Colcard 3.7M

It looks like overwhelmingly only one row is Fetched per Open. How necessary is the cursor instead of a SELECT...FETCH FIRST ROW ONLY?

Consider adding FOR FETCH ONLY to the end of this cursor for Lock Avoidance – as long as there is no UPDATE WHERE CURRENT OF CURSOR lingering in the program somewhere.

Ensure access to Table AP_INV_RECV_XREF is in the order of Index XAPINRX3 to invoke Dynamic Prefetch.
DB2 Activity by Query
SQL Analysis

DB2 Explain

DB2 Catalog Statistics
Consider turning on DB2 Compression for Table AP_INV_HDR. Why 4.3M Selects against a 1.4M Table? Perhaps check programmatically to see if the previous Select matches the current Select.

Also, consider INCLUDING Columns INVOICE_STATUS_CD and VENDOR_NBR with Unique Index XAPINVH1 to change the access to Index Only.
Just Your Typical 36 Hour Batch Job!!!

Over 1800 Hours of Runtime Per Year; Over 350 Hours of CPU Per Year; Over 500 Hours Waiting For CPU Per Year; Over 1000 Hours of DB2 Wait
Over 20 Million Getpages Associated with this one SQL Statement
SQL Analysis - DB2 Explain

Index Only Access on Two of the Three Tables – Nested Loop Joins – Overall Not Too Bad
**DB2 Catalog Statistics**

**Table TP6L000 with Over 19,000 Pages is a Candidate for DB2 Compression**
Using OR 0 = 1 Discourages DB2 From Using Index XP6L0002
Performance Recommendations

Turn DB2 Compression on for Table TP. Using OR 0=1 (which changes the predicate to a Stage 2 predicate) is done to keep DB2 from choosing a particular index (in this case probably XPL60002 – did this index not perform well). Consider using concatenate as follows: BUSGRTY_CD = :H||’’ which keeps it a Stage 1 predicate. Also, consider changing predicate AND BUSGRST_CD NOT IN ('NLV','CLSD','IPAR','FRZN','PEND') from NOT IN to IN and name the other four values for that column.

Using the Concatenate Instead of the OR 0 = 1, Keeps the Predicate Stage 1

Change From NOT IN to IN and Use the Other Four Values

Input File Sorted by Account Number Reduced the High Number of Synchronous DB2 I/O and Invoked Dynamic Prefetch.

All Told, Runtime Reduced From 36 Hours to 4 Hours.
Measurement Session Data - Batch

90 Minute Daily Batch Job

50 Hours of CPU per Year
330 Hours of DB2 Wait

Wait Time by Module

The Mainframe Software Partner for the Next 50 Years
### DB2 Activity by Query

#### SQL Statement Text
```
DECLARE CURSOR CURSOR FOR SELECT CPS_ID, CP ...
```

#### Run Time Statistics For Scans Executed By The SQL Statement

<table>
<thead>
<tr>
<th>Type</th>
<th>Rows Processed</th>
<th>Rows Examined</th>
<th>Rows-Q Stage1</th>
<th>Rows-Q Stage2</th>
<th>Rows Inserted</th>
<th>Rows Deleted</th>
<th>Rows Updated</th>
<th>Get Page Requests</th>
<th>RI Page Scans</th>
<th>RI Page Deletes</th>
<th>LOB Page Scans</th>
<th>LOB Page Updates</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDX</td>
<td>3,161,815</td>
<td>3,099,985</td>
<td>3,453</td>
<td>3,449</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>59,108,834</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SEQD</td>
<td>2,085,093</td>
<td>84,238,115</td>
<td>81,003,220</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>18,772,282</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SEQW</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4,088</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

#### Services Invoked
- FETCH: 50 rows, 6,457 scans, 0.000034 sec, N/A CPU
- CLOSE: 159 rows, 3,624 scans, 0.000016 sec, N/A CPU

#### Parallel Processing
- **DB2 Location**: CA0C8CDDB2P

---

**This Single SQL Statement is Responsible For Nearly All of the CPU Usage and DB2 Wait Time**
Low Number of Matching Columns for Index OTH2_05_CPOS, List Prefetch is On, and No Index Only Access
Both Index OTH2_05_CPOS and FK1_03_CPOS have Columns with 0 Cardinality Indicating that Runstats was Run at Some Time When the Tables were Empty
SQL Statement Text and Performance Recommendations

- Batch - DBRM - 17 Open (Stmt Count 3,624)
  50 Fetch - 6,457

DECLARE AGENT_CURSOR CURSOR FOR
  SELECT CPG_ID, CPG_CRT_DTE, CPG_CRT_TME, CP_ID,
          CP_AGNT_ID, CP_PYMT_MTHD_CDE, CP_AMT
  FROM CPO_PAYMENT_GROUP, OTH2_05_CPOS(1)
  JOIN CP_CPO_PAYMENT
    ON CP_CPG_ID = CPG_ID
    AND CP_AGNT_ID = :H
    AND CP_STATUS = 'U'
    AND CP_PYMT_MTHD_CDE = :H
    AND CP_AUTH_NUM = ''
    AND CPG_FLT_NUM IS NULL
    AND CPG_CRT_DTE = :H
    AND :H - CPG_CRT_TME <= :H
    AND CPG_CRT_TME - :H <= :H
  ORDER BY CPG_CRT_TME

After a Reorg, run a Full Runstats on all Columns and Indexes for both of these Tables - some of the columns on these tables have 0 for Column Cardinality which means that Runstats was run when the table was empty. Consider doing the arithmetic in the program rather than in the SQL - perhaps a Range Predicate could then be used and would be a Stage 1 Predicate as follows:

    AND CPG_CRT_TME BETWEEN :H AND :H

Consider Doing the Arithmetic in the Program Rather Than in the SQL
Consider adding indexes to these two Tables to increase the MATCHCOLS and change the access to Index Only for both Tables as follows:

**Column 1**
- AGNT_ID
- CP_STATUS
- CP_PYMT_MTHD_CDE
- CP_AUTH_NUM
- CP_CPG_ID
- CP_ID
- CP_AMT

**Column 2**
- CPG_CRT_DTE
- CPG_FLT_NUM
- CPG_CRT_TME
- CPG_ID

*Changed from 90 Minutes Down to 5 Minutes by Increasing the Number of Matching Columns and Changing the Access Path to Index Only*
DDF
DDF – Distributed Data Facility

Mainframe DB2 Subsystem A

DB2 Database

Mainframe DB2 Subsystem B

DB2 Database

SNA - TCP/IP

Neon Shadow DB2 Connect Oracle Gateway

DB2 Connect

Workstation

Early 1990s
Typical Transaction Flow from Browser to Mainframe

Today!

4 out of 5 Consumers use Smartphones

The Mainframe Software Partner for the Next 50 Years
According to IBM, DDF is the #1 DB2 for z/OS Interface for Client-Server Applications – or - represents the fastest growing access into DB2.

These Client-Server Applications may be using CICS Transaction Gateway or Websphere MQ or Websphere Application Server utilizing JDBC or ODBC.

And, most of this traffic is Dynamic SQL!!!

Finding and Tuning this Dynamic SQL Traffic from DDF is critical to reduce MIPS associated with DDF and iStrobe provides everything needed to do Performance & Tuning in the DDF Address Space.
Most of the CPU shows up in Pre-Emptible SRBs (Service Request Blocks) called Enclaves under Control of Work Load Manager
### DB2 DDF SQL Activity

<table>
<thead>
<tr>
<th>DBRM/Package</th>
<th>Bind Timestamp</th>
<th>Statement Count</th>
<th>Elapsed Time (sec)</th>
<th>Parallel Task CPU Distribution (sec)</th>
<th>CPU Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Average</td>
<td>Max</td>
<td>Total</td>
</tr>
<tr>
<td>SYSNL300</td>
<td>08/29/2011 06:56:05 PM</td>
<td>1,463,150</td>
<td>0.005196</td>
<td>10.8666927</td>
<td>7601.999911</td>
</tr>
<tr>
<td>SYSLH200</td>
<td>08/29/2011 06:56:05 PM</td>
<td>54,643</td>
<td>0.010559</td>
<td>14.249172</td>
<td>576.649482</td>
</tr>
<tr>
<td>SYSSTAT</td>
<td>08/29/2011 06:56:10 PM</td>
<td>61</td>
<td>0.004060</td>
<td>23.933333</td>
<td>46.047444</td>
</tr>
<tr>
<td>CG22211A</td>
<td>08/29/2011 06:56:34 PM</td>
<td>99</td>
<td>0.004604</td>
<td>0.271971</td>
<td>1.556575</td>
</tr>
<tr>
<td>SYSNL200</td>
<td>08/29/2011 07:00:37 PM</td>
<td>199</td>
<td>0.03452</td>
<td>3.88138</td>
<td>6.069423</td>
</tr>
<tr>
<td>TR2111A</td>
<td>08/29/2011 07:00:19 PM</td>
<td>4,055</td>
<td>0.001267</td>
<td>0.057439</td>
<td>5.137725</td>
</tr>
<tr>
<td>ASNMONIT</td>
<td>08/29/2011 07:00:35 PM</td>
<td>101</td>
<td>0.003164</td>
<td>0.079246</td>
<td>0.316568</td>
</tr>
<tr>
<td>ASNMUFDT</td>
<td>08/29/2011 06:55:55 PM</td>
<td>92</td>
<td>0.007283</td>
<td>0.337413</td>
<td>0.670081</td>
</tr>
<tr>
<td>CA8921A</td>
<td>08/29/2011 06:56:06 PM</td>
<td>551</td>
<td>0.001239</td>
<td>0.188592</td>
<td>0.825888</td>
</tr>
<tr>
<td>TR2201A</td>
<td>08/29/2011 07:00:19 PM</td>
<td>482</td>
<td>0.005681</td>
<td>0.070245</td>
<td>2.738265</td>
</tr>
<tr>
<td>ASNMPROC</td>
<td>08/29/2011 06:56:36 PM</td>
<td>4</td>
<td>0.058255</td>
<td>0.073291</td>
<td>0.234219</td>
</tr>
<tr>
<td>SYSSH100</td>
<td>08/29/2011 07:00:00 PM</td>
<td>30</td>
<td>0.227348</td>
<td>6.653936</td>
<td>6.620425</td>
</tr>
<tr>
<td>DSNREXX</td>
<td>08/29/2011 07:00:41 PM</td>
<td>20</td>
<td>0.001689</td>
<td>0.01432</td>
<td>0.037784</td>
</tr>
<tr>
<td>RAARDB1</td>
<td>08/29/2011 06:57:36 PM</td>
<td>3</td>
<td>0.010831</td>
<td>0.015379</td>
<td>0.032493</td>
</tr>
<tr>
<td>RAARDB2</td>
<td>08/29/2011 06:56:38 PM</td>
<td>15</td>
<td>0.012958</td>
<td>0.150474</td>
<td>0.194363</td>
</tr>
<tr>
<td>RAARDB3</td>
<td>08/29/2011 06:56:39 PM</td>
<td>6</td>
<td>0.02029</td>
<td>0.062855</td>
<td>0.121737</td>
</tr>
<tr>
<td>TR9101A</td>
<td>08/29/2011 07:00:20 PM</td>
<td>15</td>
<td>0.000264</td>
<td>0.001955</td>
<td>0.004255</td>
</tr>
<tr>
<td>DSNAPCO8</td>
<td>08/29/2011 07:00:09 PM</td>
<td>2</td>
<td>0.011773</td>
<td>0.023845</td>
<td>0.023845</td>
</tr>
<tr>
<td>BIEST111A</td>
<td>08/29/2011 07:03:21 PM</td>
<td>3</td>
<td>0.160096</td>
<td>0.0279</td>
<td>0.48287</td>
</tr>
<tr>
<td>CG19111A</td>
<td>08/29/2011 06:56:09 PM</td>
<td>4</td>
<td>0.000185</td>
<td>0.000236</td>
<td>0.00074</td>
</tr>
</tbody>
</table>
DB2 DDF SQL Activity

The Mainframe Software Partner for the Next 50 Years
### SQL Analysis

<table>
<thead>
<tr>
<th>Plan Table Rows</th>
<th>Method</th>
<th>Index Only</th>
<th>Creator</th>
<th>TNAME</th>
<th>Address Creator</th>
<th>Address Name</th>
<th>Address Type</th>
<th>Match Cols</th>
<th>Softn</th>
<th>Sortn</th>
<th>UJOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>QBLOCK NO</td>
<td>PLAN NO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>N</td>
<td>TX131N</td>
<td>CND</td>
<td>0</td>
<td>TX131CB</td>
<td>I</td>
<td>0</td>
<td>NNNN</td>
<td>NNNN</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>N</td>
<td>TX131N</td>
<td>CND</td>
<td>0</td>
<td>TX131SU</td>
<td>I</td>
<td>0</td>
<td>NNNN</td>
<td>NNNN</td>
<td></td>
</tr>
</tbody>
</table>

### DB2 Explain

#### DB2 Catalog Statistics

<table>
<thead>
<tr>
<th>Table</th>
<th>DBNAME</th>
<th>TSNAME</th>
<th>CARDF</th>
<th>NUM DCP</th>
<th>MOTS</th>
<th>SPLIT ROWS</th>
<th>NPAGES</th>
<th>PAGES</th>
<th>EXT LEN</th>
<th>PARTKEY COL</th>
<th>AVG ROW LEN</th>
<th>STATETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>TX1310</td>
<td>2000530</td>
<td>2000530</td>
<td>0</td>
<td>80587</td>
<td>0</td>
<td>825784</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>383</td>
<td>2011-03-27 19:30:31</td>
</tr>
<tr>
<td>TX1311</td>
<td>2000530</td>
<td>2000530</td>
<td>0</td>
<td>80587</td>
<td>0</td>
<td>825784</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>383</td>
<td>2011-03-27 19:30:31</td>
</tr>
<tr>
<td>TX1312</td>
<td>2000530</td>
<td>2000530</td>
<td>0</td>
<td>80587</td>
<td>0</td>
<td>825784</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>383</td>
<td>2011-03-27 19:30:31</td>
</tr>
<tr>
<td>TX1313</td>
<td>2000530</td>
<td>2000530</td>
<td>0</td>
<td>80587</td>
<td>0</td>
<td>825784</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>383</td>
<td>2011-03-27 19:30:31</td>
</tr>
<tr>
<td>TX1314</td>
<td>2000530</td>
<td>2000530</td>
<td>0</td>
<td>80587</td>
<td>0</td>
<td>825784</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>383</td>
<td>2011-03-27 19:30:31</td>
</tr>
<tr>
<td>TX1315</td>
<td>2000530</td>
<td>2000530</td>
<td>0</td>
<td>80587</td>
<td>0</td>
<td>825784</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>383</td>
<td>2011-03-27 19:30:31</td>
</tr>
<tr>
<td>TX1316</td>
<td>2000530</td>
<td>2000530</td>
<td>0</td>
<td>80587</td>
<td>0</td>
<td>825784</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>383</td>
<td>2011-03-27 19:30:31</td>
</tr>
<tr>
<td>TX1317</td>
<td>2000530</td>
<td>2000530</td>
<td>0</td>
<td>80587</td>
<td>0</td>
<td>825784</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>383</td>
<td>2011-03-27 19:30:31</td>
</tr>
</tbody>
</table>

### Compuware

The Mainframe Software Partner for the Next 50 Years
SQL Statement Text and Performance Recommendations

**DIST - DDF - Package SYSLH200 - 60 Select (Stmt Count 69)**

```sql
SELECT REQUEST_DT_TM,
       CORR_BY_IND
FROM DBC4DIST.DDF_TXI13G0(0)
WHERE CORR_TYPE = ????
AND REQUEST_DT_TM = (SELECT MAX(A.REQUEST_DT_TM)
                      FROM DBC4DIST.DDF_TXI13G0(2) A,
                       DDF_TXI132U(2) B
                      WHERE B.FILE_ID = ????
                         AND B.SS_ROW_NUM = ????
                         AND A.CORR_TYPE = ????
                         AND A.CLAIM_NUM = B.CLAIM_NUM)
       WITH UR

Consider adding Column CLAIM_NUM to the end of Index TXI132U to change the access to Index Only. Also, consider adding an index to Table CORRESPOND2 to change the Non-Matching Index Scan to Matching Index Only as follows:

Column 1 - CORR_TYPE
2 - REQUEST_DT_TM
3 - CORR_BY_IND
```

Change the Non-Matching Index Scan of Nearly 16,000 Leaf Pages to a Matching Index Only Access – and Change the Access to Index Only of Table OLD_CLAIM in the SubSelect
The Access is a Non-Matching Index Scan of Over 14,000 Leaf Pages and only 1% of the Rows on Table MASS_EDIT are Compressed (Needs a Reorg to Compress all Rows)
SQL Statement Text and Performance Recommendations

```
SELECT * FROM DBC4TXD1.TXV137_MASS_EDIT
WHERE EDIT_IND = ????
AND UPD_DT_TM = ????
AND UPD_ID = ????
WITH UR
```

Table TXT137_MASS_EDIT is only 1% compressed – Reorg followed by Runstats. Select only the columns needed rather than Select *. Consider adding an index to Table TXT137_MASS_EDIT to change the Non-Matching Index Scan to a Matching Index Scan as follows:

```
Column 1 - UPD_ID
2 - UPD_DT_TM
3 - EDIT_IND
```

Change the Non-Matching Index Scan of Over 14,000 Leaf Pages to a Matching Index Access – and Avoid Using SELECT * (of all Columns) and just Select the Columns Required
SQL Analysis

DB2 Explain

A Tablespace Scan of Over 16,000 Data Pages
SQL Statement Text and Performance Recommendations

SELECT **MAX** (FILE_ID) AS FILE_ID
FROM DBC4TXD1.TXT134_SS_LOG
WHERE ORIG_FILE_ID = ????
AND SINGLE_CLAIM_IND = ????
WITH UR

608.6K Rows/16K Pg

Consider adding an index to Table **TXT134_SS_LOG** to change the Tablespace Scan to a Matching Index Only Scan as follows:

Column 1 - ORIG_FILE_ID
Column 2 - SINGLE_CLAIM_IND
Column 3 - FILE_ID

Change the Tablespace Scan to a Matching Index Only Access – in Fact, An I1 Fetch which is Wicked-Fast
DB2 DDF SQL Activity
**SQL Analysis**

**DB2 Explain & DB2 Catalog Statistics**

A Tablespace Scan of Nearly 280,000 Data Pages and DB2 Compression is Not Turned On
SQL Statement Text and Performance Recommendations

**DIST - DDF Package SYSLN200 - 80 Select (Stmt Count 6)**

```sql
SELECT OIMS_SID_NO AS SID,
      OIMS_TDCJID_NO AS TDCJ,
      OIMS_NAME AS NAME,
      OIMS_DOB AS DOB,
      OIMS_SEX AS GENDER,
      OIMS_RACE AS RACE,
      OIMS_SETUP_DTE AS SETUPDATE,
      OIMS_LEG_RESI_INC AS LCOR,
      OID_SSN_NO AS SSN
FROM TAMASTER
  LEFT OUTER JOIN
    TAOID_DOCUMENTS X1CU_SAOIDDOC(1)
  ON OID_SID_NO = OIMS_SID_NO
WHERE UPPER(OIMS_NAME) LIKE ?
WITH UR
```

Consider turning DB2 Compression on for Table TAMASTER. Using the UPPER Function on the predicate prevents DB2 from using the index that is on Column OIMS_NAME (Index ). Consider removing the UPPER Function.

**Change the Tablespace Scan to a Matching Index Access by Removing the UPPER Function – or – Create an Index on Expression**

INCLUDING Column OID_SSN_NO with Index X1CU_SAOIDDOC will change the Access to Table TAOID_DOCUMENTS to Index Only.
Bufferpool Tuning – It’s Only Memory

DISPLAY BUFFERPOOL(ACTIVE) DETAIL(INTERVAL)

Consider doing this command weekly/monthly saving the output in a text file and use the Getpage Requests and I/O counts to determine the bufferpool hit ratios; increasing the bufferpool sizes and parameters to try and achieve hit ratios greater than 90%. These counts are reset after each time this Command is issued.

\[
\text{(GetPage Req} - (\text{Sync I/O} + \text{Async Pgs Rd})) / \text{GetPage Req} * 100
\]

Async Pgs Rd = Sequential Prefetch Pg + List Prefetch Pg + Dynamic Prefetch Pg
Remember!

Business Functions don’t exist to reduce costs; they exist to generate value!

So reducing Operating Costs through Application Performance Management helps the Business generate value.
Thank You
IDENTIFYING TABLES WITHOUT COMPRESSION

--CATALOG QUERY TO LIST TABLES WITHOUT COMPRESSION
--
SELECT SUBSTR(B.TSNAME,1,10) AS TSNAME,
       SUBSTR(B.CREATOR,1,10) AS CREATOR,
       SUBSTR(B.NAME,1,15) AS NAME,
       INT(B.NPAGESF) AS NPAGESF, INT(B.CARDF) AS CARDF,
       A.COMPRESS AS CMP, C.BPOOL,
       A.PCTFREE, A.FREEPAGE, C.PGSIZE,
       C.SEGSIZE, C.STATSTIME
FROM SYSIBM.SYSTABLEPART A,
     SYSIBM.SYSTABLES B,
     SYSIBM.SYSTABLESPACE C
WHERE A.COMPRESS = ' '  -- 'Y' WILL LIST TABLES WITH COMPRESSION
   AND B.CREATOR <> 'SYSIBM'  -- NO CATALOG TABLES
   AND A.TSNAME = B.TSNAME  -- JOIN PREDICATES
   AND B.TSNAME = C.NAME
   AND A.DBNAME = B.DBNAME
   AND B.DBNAME = C.DBNAME
   AND A.PARTITION IN (0,1)  -- DON'T COUNT PARTITIONS MORE THAN ONCE
--AND B.PCTROWCOMP < 98  -- THOSE TABLES NOT FULLY COMPRESSED
   AND B.CARD <> -1  -- ENSURE RUNSTATS
   AND A.CARD <> -1  -- ENSURE RUNSTATS
   AND B.TYPE = 'T'  -- ONLY TABLES
--AND A.FREEPAGE < 10  -- ADDITIONAL QUALIFIERS
--AND A.CARD < 50000  --
   AND B.NPAGES > 50  -- NUMBER OF PAGES THRESHOLD
ORDER BY B.TSNAME, B.NPAGES DESC WITH UR;
<table>
<thead>
<tr>
<th>TSNAME</th>
<th>CREATOR</th>
<th>NAME</th>
<th>NPAGESF</th>
<th>CARDF</th>
<th>CMP</th>
<th>BPOOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DASC</td>
<td>MFHCWW0</td>
<td>DASC</td>
<td>447</td>
<td>39261</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>DBMXSTS</td>
<td>DFHMXS0</td>
<td>DM66MODLIST</td>
<td>223</td>
<td>2548</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>DBMXSTS</td>
<td>DFHMXS0</td>
<td>DM66COLUMNS</td>
<td>176</td>
<td>2795</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>DBMXSTS</td>
<td>DFHMXS0</td>
<td>DM66MCTRL</td>
<td>116</td>
<td>1017</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>DBTJSTS</td>
<td>BUSTJS0</td>
<td>DBCOLDATA</td>
<td>5429</td>
<td>227113</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>DHOBJ</td>
<td>MFHCWW0</td>
<td>DHOBJ</td>
<td>210</td>
<td>2727</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>DPRP</td>
<td>MFHCWW0</td>
<td>DPRP</td>
<td>435</td>
<td>25617</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>DSUBEX</td>
<td>MFHCWW0</td>
<td>DSUBEX</td>
<td>195</td>
<td>21367</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>FDCI630</td>
<td>BFHRWA0</td>
<td>DBCOLDATA_NFM</td>
<td>16129</td>
<td>712579</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>SASC</td>
<td>MFHCWW0</td>
<td>SASC</td>
<td>577</td>
<td>25944</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>SPRP</td>
<td>MFHCWW0</td>
<td>SPRP</td>
<td>1255</td>
<td>41391</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>SUMABW1</td>
<td>FADB2</td>
<td>DBCOLDATA_N22</td>
<td>123</td>
<td>5332</td>
<td></td>
<td>BP0</td>
</tr>
<tr>
<td>SUMCOLN</td>
<td>FDDBA</td>
<td>DBCOLDATA</td>
<td>21446</td>
<td>949361</td>
<td></td>
<td>BP0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PCTFREE</th>
<th>FREEPAGE</th>
<th>PGSIZE</th>
<th>SEGSIZE</th>
<th>STATSTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0</td>
<td>4</td>
<td>64</td>
<td>2011-05-16-12.15.26.642782</td>
</tr>
<tr>
<td>10</td>
<td>15</td>
<td>4</td>
<td>16</td>
<td>2013-01-17-10.24.54.264222</td>
</tr>
<tr>
<td>10</td>
<td>15</td>
<td>4</td>
<td>16</td>
<td>2013-01-17-10.24.54.264222</td>
</tr>
<tr>
<td>10</td>
<td>15</td>
<td>4</td>
<td>16</td>
<td>2011-06-09-07.44.48.666261</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>4</td>
<td>64</td>
<td>2011-05-16-12.15.28.650474</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>4</td>
<td>64</td>
<td>2011-05-16-12.15.26.992491</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>4</td>
<td>64</td>
<td>2011-05-16-12.15.27.606492</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>4</td>
<td>64</td>
<td>2011-05-16-12.15.28.134038</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>4</td>
<td>64</td>
<td>2013-03-20-15.16.34.202069</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>4</td>
<td>64</td>
<td>2011-05-16-12.15.23.389421</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>4</td>
<td>64</td>
<td>2011-05-16-12.15.25.821831</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>4</td>
<td>64</td>
<td>2011-10-19-15.59.33.172783</td>
</tr>
</tbody>
</table>
Identifying Unused Non-Unique Indexes

--CATALOG QUERY TO IDENTIFY UNUSED NON-UNIQUE INDEXES
SELECT DISTINCT(SUBSTR(A.NAME,1,16)) AS INDEXNAME,
      SUBSTR(B.TSNAME,1,10) AS TSNAME,
      SUBSTR(B.CREATOR,1,10) AS CREATOR,
      SUBSTR(B.NAME,1,12) AS TABLENAME,
      C.NLEAF, LASTUSED
FROM SYSIBM.SYSINDEXES A,
     SYSIBM.SYSTABLES B,
     SYSIBM.SYSINDEXSPACESTATS C
WHERE C.LASTUSED IS NULL -- SET TO NULL IF NOT USED
  AND B.CREATOR <> 'SYSIBM' -- IGNORE CATALOG
  AND A.TBNAME = B.NAME -- JOIN PREDICATES
  AND A.NAME = C.NAME
  AND C.NLEAF > 100 -- SIZE THRESHOLD
  AND A.UNIQUERULE = 'D' -- NON-UNIQUE INDEXES
ORDER BY C.NLEAF DESC -- SHOW LARGEST FIRST
WITH UR;

<table>
<thead>
<tr>
<th>INDEXNAME</th>
<th>TSNAME</th>
<th>CREATOR</th>
<th>TABLENAME</th>
<th>NLEAF</th>
<th>LASTUSED</th>
</tr>
</thead>
<tbody>
<tr>
<td>PERF6INDEX_4</td>
<td>FRJGDF6</td>
<td>PFHJGD0</td>
<td>FRPERF6_EMP</td>
<td>7954</td>
<td></td>
</tr>
<tr>
<td>PERF1INDEX_5</td>
<td>FRJGDF1</td>
<td>PFHJGD0</td>
<td>FRPERF1_EMP</td>
<td>7172</td>
<td></td>
</tr>
<tr>
<td>INDEX_4</td>
<td>FDHA14SF</td>
<td>FDDBA2</td>
<td>FDHASH_EMP</td>
<td>5336</td>
<td></td>
</tr>
<tr>
<td>PERF8INDEX_6A</td>
<td>FRNEWF8</td>
<td>FRNEW</td>
<td>FRPERF8_EMP</td>
<td>3220</td>
<td></td>
</tr>
<tr>
<td>PERF8INDEX_6</td>
<td>FRPERF8</td>
<td>FRDBA</td>
<td>FRPERF8_EMP</td>
<td>3208</td>
<td></td>
</tr>
<tr>
<td>FACC21CT_IX1_NFM</td>
<td>SUMCOLN</td>
<td>MFHCWW0</td>
<td>FACC21CT_NFM</td>
<td>1893</td>
<td></td>
</tr>
<tr>
<td>DBCOLDATA_IX1_NF</td>
<td>FDC1630</td>
<td>BFHRWA0</td>
<td>DBCOLDATA_NF</td>
<td>1353</td>
<td></td>
</tr>
<tr>
<td>DBCOLDATA_IX_NFM</td>
<td>SUMCOLN</td>
<td>PMIASH0</td>
<td>DBCOLDATA_NF</td>
<td>1125</td>
<td></td>
</tr>
<tr>
<td>FD621CO_IX1_NFM</td>
<td>SUMCOLN</td>
<td>MFHCWW0</td>
<td>FD621CO_NFM</td>
<td>707</td>
<td></td>
</tr>
<tr>
<td>FD621CO_IX_NFM</td>
<td>SUMCOLN</td>
<td>MFHCWW0</td>
<td>FD621CO_NFM</td>
<td>682</td>
<td></td>
</tr>
<tr>
<td>CWAF_EVENT.UTC_I</td>
<td>AF123TS</td>
<td>AF123AS</td>
<td>CWAF_EVENT</td>
<td>404</td>
<td></td>
</tr>
</tbody>
</table>
Number of Pages and Tables per Bufferpool

```sql
-- Number of Pages and Tables defined to each Bufferpool
--
select bpool, sum(b.npages) as NPAGES, sum(ntables) as NTABLES
--
--lect substr(a.name,1,8) as TSNAME,
-- substr(a.creator,1,8) as CREATOR,
-- substr(a.dbname,1,8) as DBNAME,
-- char(b.npages) as NPAGES, bpool,
-- pgsz as PGSZ, segsize as SGSZ,
-- ntables as NTBLS, a.statstime
from sysibm.systablespace a,
     sysibm.systables b
where bpool = 'BP0'
  and a.creator not like 'SYS%'  -- Skip the Catalog
  and npages >  1               -- Size Threshold
  and a.name = b.tsname         -- Join Predicates
  and a.dbname = b.dbname
  and a.creator = b.creator
group by bpool with ur;       -- Group by Bufferpool
--
--order by a.creator, a.name with ur;
```

<table>
<thead>
<tr>
<th>BPOOL</th>
<th>NPAGES</th>
<th>NTABLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>BP0</td>
<td>246610</td>
<td>19138</td>
</tr>
<tr>
<td>BP1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>BP16K0</td>
<td>1460</td>
<td>20</td>
</tr>
<tr>
<td>BP32K</td>
<td>8651</td>
<td>1</td>
</tr>
</tbody>
</table>
Number of Leafs and Indexes per Bufferpool

--Number of Leafs and Indexes defined to each Bufferpool
--
select bpool, sum(a.nleaf) as NLEAFS, count(name) as INDEXES
--lect substr(a.name,1,18) as IXNAME,
--    substr(a.creator,1,8) as CREATOR,
--    substr(a.tbnname,1,12) as TBNAME,
--    a.nleaf as LEAFS, a.bpool,
--    reset(a.nlevels) as LVLS,
--    char(a.pgsize) as PGSZ, a.statstime
from sysibm.sysindexes a
--where bpool = 'BP0'
    where a.creator not like 'SYS%'     -- Skip the Catalog
    and a.nleaf > 1           -- Size Threshold
    group by bpool with ur;    -- Group by Bufferpool
--order by bpool with ur;
--order by a.creator, a.name with ur;

---------+-------------------+----------+
BPOOL   | NLEAFS | INDEXES |
---------+-------------------+----------+
BP0     | 395726 |   177   |
BP1     |    27  |    11   |
BP8K0   | 23481  |    28   |
BP16K0  | 1539   |    21   |
---------+-------------------+----------+