DB2 11 Expanded RBA/LRSN – Does this bit make my tail look big?

Jim Dee
BMC Software

Wednesday September 10 9:00-10:15 | Platform: DB2 for z/OS

Acknowledgements / Disclaimers

Agenda

- Expanded RBA/LRSN
  - What does it mean and how do I get there?
- RBAs/LRSNs in the DB2 Catalog and Directory
- Table and Index Space pages that have RBAs/LRSNs
- Logging
- And more?

EXPANDED RBA/LRSN

Expanded RBA/LRSN – History – Early 1980's

- DB2 Version 1 – 6 byte RBA
  - RBA – Relative Byte Address
  - From the beginning of logging of this subsystem
  - Central to DB2 Recovery design
  - Ever increasing
  - Max x'FFFFFFFFFFFF' – 281 Trillion bytes
- We'll never approach that, right?

Oops… we got there.

- Machines got faster
- Concurrent Threads on a single DB2 increased
- The LRSN spin has been removed in many cases
  - Introduced multiple records at the same LRSN and increased density
- More log records
  - Mobile Phone and digital access to data exploded – more transactions
  - Diagnostic, Trace, and NO OP log records

- There is a process to reset the RBA to zero
  - Requires Cold Start
  - On Non-Data Sharing, all spaces must be copied after Cold Start
  - Not quite as painful on Data Sharing

Expanded RBA/LRSN – History – 1995

- DB2 Version 4 – 6 byte LRSN
  - LRSN – Log Record Sequence Number
  - Based upon the z/OS Store Clock
    - High order 6 bytes of the 8 byte Store Clock
  - Ever increasing
  - Central to DB2 Recovery Design in Data Sharing
    - Used to synchronize log records across a group
  - Low order bit ~ 16 microseconds
  - RBAs in the CATALOG are now LRSNs if Data Sharing
  - Store Clock x'FFFFFFFFFFFF' = September 17th 2042

- We'll never approach that, right?
  - Hopefully I'll be retired by then…

Oops… DELTA

- Design of Data Sharing
  - If Non-Data Sharing RBA > LRSN when convert to Data Sharing
  - The DELTA is the difference between the two
  - DELTA is added to the z/OS Store Clock to derive LRSN
  - DELTA is recorded in the BSDS
- Data Sharing Disablement
  - Conditional Restart to x'C4C9E2C1C2D3' (c'DISABL')
    - LRSNs began with x'A' when Version 4 came out
  - Led to DELTA when re-enabling to Data Sharing
- We have seen some large Deltas
  - Already approaching 2041 when added to STCK

DB2 11 Expanded RBA/LRSN

- Expanded RBA/LRSN OPTIONALLY available in DB2 11

- RBA increases from 6 to 10 bytes
  - A FULLWORD added in front (left) of the existing RBA
  - Range increases from $2^{(48)}$ to $2^{(80)}$
  - From 281 Trillion Bytes - 281,474,976,701,656
    - Terabytes
  - To 1 Septillion - 1,208,925,819,614,629,174,706,176
    - Yottabyte

X'00000000rrrrrrrrrrrr'

DB2 11 Expanded RBA/LRSN

- Will also increase LRSN from 6 to 10 bytes
  - One byte added before existing LRSN
    - My math – ~36,352 years.
    - Take that September 2042!
  - Existing 6 byte LRSN
  - Three bytes precision added at the end
    - Current LRSN has ~16 microsecond precision (10^{-6})
    - Current z/OS STCK has precision to ~244 picosecond (10^{-12})
    - STCKE and Expanded LRSN to ~953 femtosecond (10^{-15})

X'00111111111111000000'

LRSN / Extended LRSN / Timestamp Comparison

LRSN	EXTENDED LRSN	TIMESTAMP
CC7F1D54A740	00CC7F1D54A740000000	2014-01-01-00.00.00.000000000000000
CC7F1D54A740	00CC7F1D54A740000001	2014-01-01-00.00.00.000000000000953
CC7F1D54A740	00CC7F1D54A740000200	2014-01-01-00.00.00.000000000488281
CC7F1D54A741	00CC7F1D54A741000000	2014-01-01-00.00.00.000016000000000

- base for comparison
- lowest precision supported by Extended LRSN
- my CPU speed at the time I created this presentation
  - Other CPUs can be faster or slower
- lowest precision supported by basic LRSN

- DB2 11 is running internally using Extended LRSNs
  - Some normalization has to occur when using basic logic or logging

Okay… How do I get to Expanded RBA/LRSNs?

- Must be in NFM
- Job to convert each BSDS
  - DSNJCNVT
  - Can be done one member at a time in a data sharing group
  - Member must be down
- DSNZPARM / DSN6SPRM
  - OBJECT_CREATE_FORMAT = EXTENDED
  - UTILITY_OBJECT_CONVERSION = EXTENDED
- REORG, REBUILD, LOAD REPLACE
  - RBALRSN_CONVERSION EXTENDED
  - Can be executed partition by partition
  - Conversion back to BASIC format is possible
    - But, why? Copy Migration purposes? Can't be exploiting Extended…

Order of conversion?

- You have flexibility here
  - You can do spaces and BSDS conversions in any order
  - Can have some members BSDS converted and others not
- The real trick is getting converted to EXPANDED BEFORE you reach the basic limit on any member of a group or on a non-data sharing system.
- *WARNING* When you convert the BSDS, you start expanded logging
  - If you're tight on RBA range, you might want to convert the spaces first

Order of Conversion Resources

- SAP Performance Report asserts 22% increase in logging when converted to extended (report URL in notes).

- There is overhead related to normalization of the 10 byte internal RBA/LRSN to fit on the 6 byte pages and write 6 byte log records (IDUG DB2 11 for z/OS Technical White Paper – URL in notes)

- Redbook – IBM DB2 11 for z/OS Technical Overview (URL in notes) – Chapter 3.1 "Although you should run the BSDS conversion first, to improve performance, you can complete the other tasks in any order."

Order of Conversion Resources

- Net of my opinion
- If Non-Data Sharing and nearing end of log (RBA), Converting BSDS first will shorten the time to reach the soft and hard limits.
- If Data Sharing and converting because of LRSN Delta and you don't have an RBA shortage issue, Do BSDS first.
- If converting DS or Non-DS for performance reasons and to position for the future, do BSDS first.

When BASIC 6 byte RBA/LRSNs are reached?

- Convert SYSUTILX, SYSTSCPY, and SYSLGRNX
  - long before limits to ensure utilities can run when limits reached
  - **DSNTIJCV** is the job that converts the catalog and directory to extended format
- Warning ➔ "Soft Limit" ➔ "Hard Limit" (Critical)
- Warning is the first wakeup call
  - At RBA X'F00000000000' (16TB left) or at LRSN about one year before end of time
  - DSNJ032I (non data sharing) or DSNJ034I message repeated at every startup and log switch
  - Time to start running reorgs!

When BASIC 6 byte RBA/LRSNs are reached?

- Soft Limit
  - RBA x'FFF800000000' (< 32GB) or LRSN about two months before EOT
  - Objects in Basic Format go into Read Only
    - -904 with reason code 00C2026D
    - Can only be updated by utilities to convert format

- Hard Limit
  - RBA x'FFFF00000000' (<4GB) or LRSN about two weeks before EOT
  - DSNJ033E (non data sharing) or DSNJ035E message
  - DB2 terminates with 00D10251
  - If BSDS has not been converted, it is required to go forward
    - DB2 will not start until converted to Extended RBA/LRSN
    - Or if Non-Datasharing RBA hit versus LRSN, you can follow the RESET process

RBA/LRSNS IN THE CATALOG AND DIRECTORY

RBAs/LRSNs in the DB2 Catalog and Directory

- 14 columns across 8 Catalog / Directory Tables have RBAs/LRSNs in DB2 10 (one TB/COL added in DB2 11)

- SYSIBM.SYSCHECKS
  - RBA – The log RBA or LRSN when the constraint was created

- SYSIBM.SYSCOPY
  - START_RBA
  - PIT_RBA
    - Values differ based upon ICTYPE
    - See the manual

RBAs/LRSNs in the DB2 Catalog and Directory

- SYSIBM.SYSINDEXES
  - COPYLRSN – RBA or LRSN when index space was created or altered to COPY YES
- SYSIBM.SYSINDEXESTATS
  - COPYUPDATELRSN – The RBA or LRSN of the first update after the most recent copy
- SYSIBM.SYSLGRNX (directory table)
  - LGRSRBA - Starting RBA of log range
  - LGRSPBA - Ending RBA of log range
  - LGRSLRSN - Starting LRSN of log range
  - LGRELRSN - Ending LRSN of log range

RBAs/LRSNs in the DB2 Catalog and Directory

- SYSIBM.SYSTABLES
  - RBA1 – CREATE RBA or LRSN of the table
  - RBA2 – ALTER RBA or LRSN of the last time it was altered
- SYSIBM.SYSTABLESPACESTATS
  - COPYUPDATELRSN – The RBA or LRSN of the first update after the most recent copy
- SYSIBM.SYSUTIL (directory)
  - USURLSN - Latest utility start LRSN
  - USURLSNO - Original utility start LRSN

Okay, so when is the Catalog/Directory converted?

- *hlq*.DSNSAMP(DSNTIJEN)
  - Standard ENFM/NFM Enabling job
  - Includes CATMAINT steps to convert 6 byte RBA/LRSNs to 10 byte
  - Catalog/Directory is converted whether you choose to use Extended RBA/LRSN or not.

  - *hlq*.DSNSAMP(DSNTIJCV**)** is the job that converts the physical file format of the catalog and directory to extended format

Now for the Geeky Bits…

- We'll look at some of the various internal structures to understand what impact having the expanded RBA/LRSNs will have on the physical data.

- I will not map entire structures, but highlight where the RBA/LRSNs live in the objects I am discussing.

RBA/LRSNS ON SPACE PAGES

RBAs/LRSNs in Table and Index Space Pages

- PGHEAD/PGTAIL
  - PGHEAD - First x'14' bytes of the page
    - Has PGLOGRBA – 6 byte RBA/LRSN of the last time the page was updated
  - PGTAIL – Last x'02' bytes of the page
    - Free RID pointer
    - Has a parity bit which corresponds to a bit in the PGHEAD
- *Change for EXPANDED RBA/LRSN Implies fewer bytes on each page*
- Space map implications
  - The size of the PGHEADER/PGTAIL increase
  - Number of pages covered by a Space Map may decrease slightly
- Dictionary Header has an RBA/LRSN too for when built

Basic - PGTAIL

```
***  BEGINNING OF PAGE NUMBER 00000128
     Page Header
     |- Parity flag bit b'00010000'
     | PGLOGRBA
0000 10 0007D4 D17DC300 00012800 00220FB4 00001401 0100C800 2E01C4C9 F0C5D7F0
...                                  Rid Map
0FC0 00000000 00000000 00000000 00000000 00000000 00000E24 001404C4 058C0654
                                                    PGTAIL
0FE0 03FC01A4 0EEC026C 033400DC 0B040C94 0D5C0BCC 071C0A3C 097408AC 07E400D5
```

```
PAGE: # 00000128 ------------------------------------------------------------------------
DATA PAGE: PGCOMB='10'X  PGLOGRBA='0007D4D17DC3'X  PGNUM='00000128'X  PGFLAGS='00'X  PGFREE=34
    PGFREE='0022'X  PGFREEP=4020  PGFREEP='0FB4'X  PGHOLE1='0000'X  PGMAXID='14'X  PGNANCH=1
PGTAIL:  PGIDFREE='00'X  PGEND='N'
```

- PGTAIL – Has parity bit that matches flag in PGHEADER
  - Historically a x'D5'/c'N' (b'00010000') or x'C5'/c'E' (b'00000000')

Extended - PGBIGTAIL

*** BEGINNING OF PAGE NUMBER 00000128
Page Header
|- Parity flag bit b'00010000'
| PGLOGRBA
0000 18**000000 00000000** 00012800 09E60FAA 00001A17 0200D200 0007D9D4 C4C4C2F6
... Rid Map
0FA0 40404080 01D7D540 00CCD9C4 C1E2E9E2 40408000 80000000 0ED80E06 0D340C62
0FC0 0B900ABE 09EC891A 877686A4 85D28500 842E8000 8000835C 828A81B8 80E68014
PGBIGTAIL(unused) PGBIGRBA
0FE0 80008000 80008000 80008848 00000000 00000000 **00CB5227 FDB88F7D C600**00**52**

DATA PAGE: PGCOMB='18'X **PGBIGRBA='00CB5227FDB88F7DC600'X** PGNUM='00000128'X PGFLAGS='00'X
PGFREE=2534 PGFREE='09E6'X PGFREEP=4010 PGFREEP='0FAA'X PGHOLE1='0000'X
PGMAXID='1A'X PGNANCH=23
PGBIGTAIL: PGPRETAIL='0000000000000000**00CB5227FDB88F7DC600**'X PGIDFREE='00'X PGEND='**52**'X

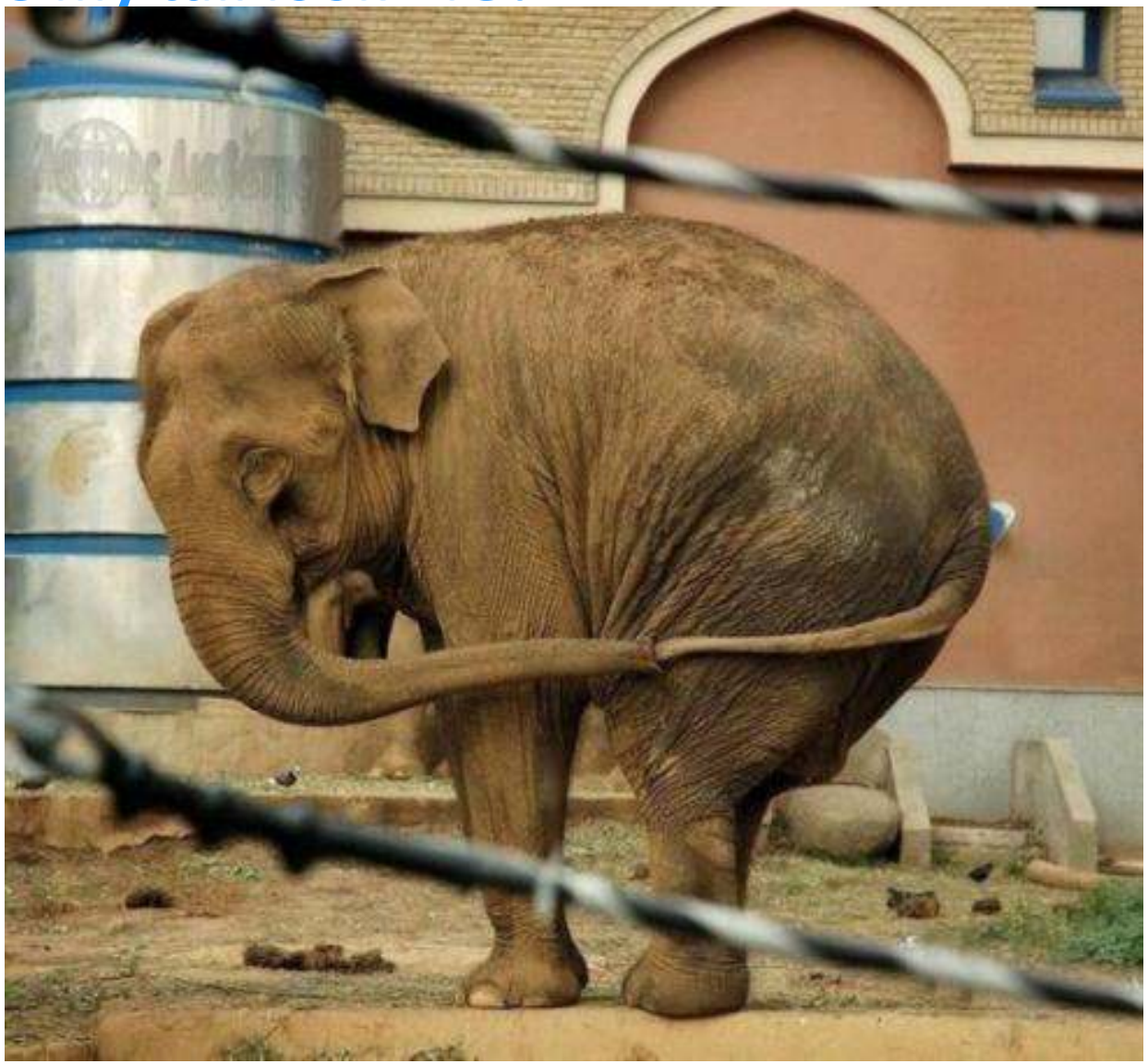
- PGTAIL/PGBIGTAIL – Has a bit that designates extended format
  - b'00000**0**0**0**' – Basic Format
  - b'00000**0**1**0**' – Extended Format

Does this bit make my tail look BIG?

In PGTAIL

b’0000000**0**0’ –
Basic Format

b’0000000**1**0’ –
Extended Format



Space Maps

- Diagnosis Guide And Reference has a Table of pages covered by a single space map page
  - Based upon Segmented (and SEGSIZE) or Not Segmented
  - PAGESIZE
  - MEMBER CLUSTER
  - LOB Space Maps have different calculations/behavior
- New Diagnosis Manual has Two Tables
  - For 20 (x'14) byte PGBIGTAIL
  - Or 2 byte basic PGTAIL
- Impact from 0 pages to factors of 4 of fewer pages covered

Dictionary Page - Basic Format

```
 ***  BEGINNING OF PAGE NUMBER 00000002 ***
0000 00000000 00000000 00000220 00000003 0FE0D3E9 C4C9C3D2 DE410045 80C3C1C2
*.................LZDICK.....CAB*
...
...
0FE0 00200029 00000000 4001FF3A 86000000 00200011 00000000 D3E9C4C9 C3D200C5
*........ ..............LZDICK.E*
```

- Unformatted DSN1PRNT of a Dictionary page
  - Big Tail bit is not on
  - Each dictionary page has LZDICK eye catcher at beginning and end of page.

Dictionary Page - Extended Format

```
 ***  BEGINNING OF PAGE NUMBER 00000002 ***
0000 00000000 00000000 00000220 00000003 0FE0DFC1 00488000 4055D004 5D730087
*.....................A.... ...)...*
...
...
                                                  PGBIGTAIL
                                                  PGBIGRBA
0FE0 00000000 00000000 00006000 3F280664 2C030000 00000000 00000000 00000042
*..........-.....................*
```

- Unformatted DSN1PRNT of a Dictionary page
  - Big Tail bit is on (last byte x'02')
  - Dictionary page header and trailer updated to remove LZDICK and reformatted dictionary page header/tail to still fit dictionary on single page (well, up to 16 4K pages, really)

PGTAIL versus PGBIGTAIL implications

- 18 Fewer bytes on every DB2 page for data
  - 18 Byte bigtail (plus unused old pglogrba, but those aren't new bytes)
  - Minor reformatting of every page in table and index spaces
  - Slightly smaller space per page for actual data
    - Regular spaces, XML, LOBs, indexes, etc.
- Fewer pages potentially covered by a space map page
  - More space map pages per partition
- Reformatting of Dictionary pages to accommodate

- *If you've used most every byte of every page of a space and are close to the DSSIZE limit, the REORG to extended format could encounter space issues.*
  - Expectation is existing FREESPACE will accommodate conversion

RBA/LRSNS IN THE LOG

RBAs/LRSNs in logging

- BSDS
  - Chock full o' RBA/LRSNs (no examples included)
- Log Page Control Interval
  - Every physical Active VSAM LOG page has the CI
- Log Record Header
  - Every Log Record written
- Data Manager Log Record Subheader
  - Every Insert/Update/Delete/Page Format/Index Update/etc.
- SAVEPOINT records
- 'Special' Space SYSCOPY records - Mirrors SYSCOPY row

Log Page Control Interval – Basic Format

- Last 21/x'15' bytes of every log page
- Describes the page (e.g. free bytes) including page begin RBA
- 4075 bytes on every 4K page for log records
- The second to the last byte has the bit that indicates extended

```
000000  008B0000 06000019 0EC008A8 E6AA3E73   08A8E6AA 8DF00726 08A8E6AA 7000CB70
…
000F20  0000190E C008A8E6 AA3E7308 A8E6AA9E   27072608 A8E6AA55 20CB70A1 51B68C00
…
000FC0  0003B600 00000000 000001FF 474C4F42   414C2054 454D504F 52415259 20544142
                              Log Page Control Interval
                                                 page begin RBA/last LRSN on page
000FE0  4C452053 45535349 4F4E2000 00000000   0F1B08A8 E6AA9000 CB70A151 B68C0000
```

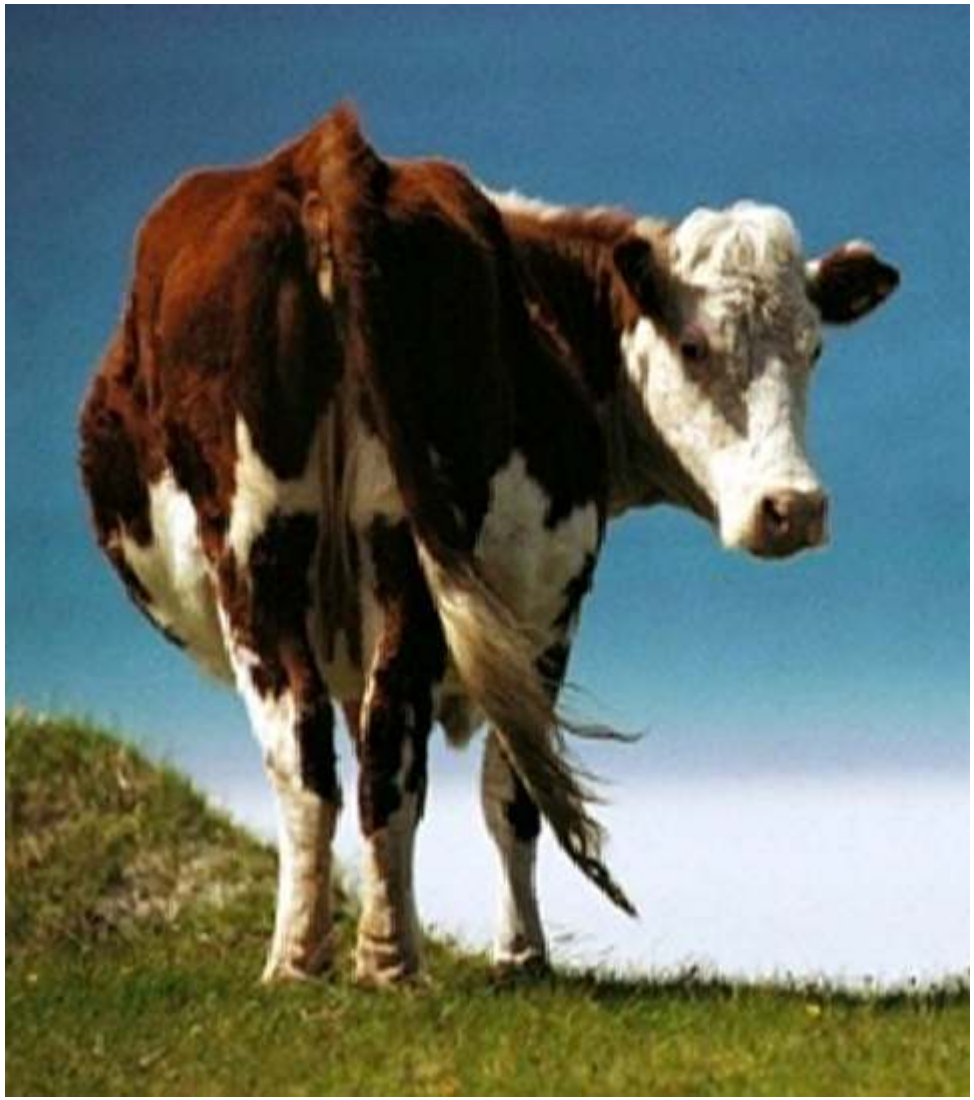
Log Page Control Interval – Extended Format

- Last 37/x'25' bytes of every log page
- Describes the page (e.g. free bytes) including page begin RBA
- 4059 bytes on every 4K page for log records (16 fewer bytes)
- The second to the last byte has the bit that indicates extended

```
000000  00000087 00008008 62044E04 3A042604   1203FE03 EA03D603 C203AE03 9A038603
000020  72035E03 4A033603 22030E02 FA02E602   D202BE02 AA029602 82026E02 5A024602
…                                                                  Log Page
000FC0  00004B0C 000FD9D4 C44BC4D9 C7D9D7F0   F488DFC0 00004B0D 000FD9FF 00000000
        Control Interval
                      page begin RBA          / Last LRSN on page          Extended
000FE0  105A0000 105A0000 000000B6 C97A6000   00CB711A 7F8CD9ED 920004BC 00008000
```

- Bit b'**0**0000000' / x'00' - Basic Format
- Bit b'**1**0000000' / x'80' - Extended Format

Does this bit make my tail look BIG?



Physical Log Page Control Interval

b’ **0**0000000’ – Basic Format

b’ **1**0000000’ – Extended Format

Log Record Header – Basic Format

- On Every Log Record written
- 38 / x'26' bytes
- This is a LRH only data sharing heartbeat record

```
Log Record RBA – derived from log page CI + offset
01DAED14DE35  MEMBER(DIZ1     )
              LRSN(CB718654C3BE)  TYPE(SYSTEM EVENT)          13:52:07 13.151
              SUBTYPE(SUBSYSTEM NO LOG ACTIVITY RECORD)

                          Extended Flag not on (bit x'00100000')
                          | LRHURID       LRHLINK           LRHUNLSN     LRSN
*LRH* 00260026 00100006 03800000 00000000 01DAED14 DE0F0726 01DAED14 DE0FCB71
      8654C3BE 0001
```

Log Record Header – Extended Format

- On Every Log Record written
- 80 / x'50' bytes (42 / x'2A' bytes more)
- This is a LRH only data sharing heartbeat record

```
Log Record RBA – derived from log page CI + offset
00000000000FF3B6D3FA LRSN(00CB717F618510A75000) TYPE(SYSTEM EVENT) 13:21:01 13.151
                SUBTYPE(SUBSYSTEM NO LOG ACTIVITY RECORD)
                              Extended Flag   LRHURID_V2
*LRH* 00000050 00D00008 03A00000 00000000 00000000 00000000 00000000 00000000
      LRHLINK_V2                                  LRHUNLSN_V2               LRHLRSN_V2
      00000000 000FF3B6 9E005000 00100006 00000000 0000000F F3B69E00 000000CB
      717F6185 10A75000 00010000 00000000
```

- Bit b'00**0**00000' / x'00' - Basic Format
- Bit b'00**1**00000' / x'20' - Extended Format

Does this bit make my tail look BIG?

Log Record Header PGFLAGS

b’ 00**0**00000’ – Basic Format

b’ 00**1**00000’ – Extended Format

Data Management Header – Basic Format

- On Every Log Record written that updates an index or table space
- 18 / x'12' bytes

```
Log Record RBA – derived from log page CI + offset
0115A3BE6E9B  MEMBER(DIZ3    )  TYPE( UNDO  REDO )  URID(0115A3BE6B16)
     LRSN(CB7186543E3B)  DBID(0006)  OBID(0361)  PAGE(00000469) 13:52:06 13.151
               SUBTYPE(UPDATE IN-PLACE IN A DATA PAGE)  CLR(NO)
               PROCNAME(DSNIREPR)

*LRH* 0134002F 06000001 0E800115 A3BE6B16 0115A3BE 6E6C0726 0115A3BE 6E6CCB71
      86543E3B 0002
                              LGDPRBA
*LG** 8C000603 61000004 69003071 86543E39 2D08
```

Data Management Header – Extended Format

- On Every Log Record written that updates an index or table space
- 24 / x'18' bytes (6 more per record)

```
Log Record RBA – derived from log page CI + offset
00000000000DFB4F8E87  MEMBER(DJV2) TYPE( UNDO  REDO )  URID(00000000000DFB4F8D74)
  LRSN(00CB717F5EC0C3E82000) DBID(0006) OBID(08A2) PAGE(00000119) 13:20:58 13.151
       SUBTYPE(UPDATE IN-PLACE IN A DATA PAGE)  CLR(NO)  PROCNAME(DSNIREPR)

*LRH* 00000178 00594008 0EA00000 00000000 00000000 000DFB4F 8D740000 00000000
      00000000 000DFB4F 8E2E5000 06000001 00000000 0000000D FB4F8E2E 000000CB
      717F5EC0 C3E82000 00040000 00000000
                                LGDPRBA/LGDCRBA
*LG** 8C000608 A2000001 190035CB 717F5E7A EA636800 2D000000
```

Special Space SYSCOPY records

- SYSCOPY and DSNDB01 SYSCOPY events recorded in the log
- Increases parallel those of other logging
  - LRH is larger
- Actual SYSCOPY row slightly larger for the RBA size increases and a new CHAR(2) column added to the end.

Logging that I expected to increase, but did not

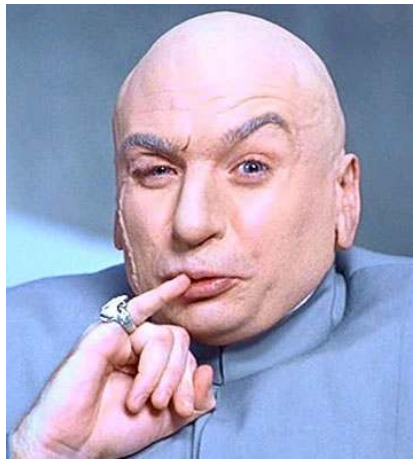
- URID records
  - Changes were limited to the LRH increases

Logging implications in Extended Format

- 16 fewer data bytes on every physical log page
  - Seemed to be a wash between the additional CI size versus Freespace
  - Was about 1.3% to 1.5% combined of every page
- 42 more bytes logged for EVERY record in the LRH
- 6 more bytes logged for every DM log record
- More bytes for Savepoint and Check Point log records

- But, hey! We have a Yottabyte of log now!

A Septillion Bytes



RBAs/LRSNs in other places?

- JES Messaging
- SYSPRINT and other outputs

- We start seeing 10 byte RBA/LRSNs in DB2 11 CM mode

- Implications for anything that scrapes/analyzes messages
  - Automated Operator tools
  - Post processing REXX

What does the future hold?

- I don't know.
- It's OPTIONAL now… but, there are many benefits
  - 1) Avoid End Of Log or End Of Time
- The more granular LRSN seems to be very beneficial
  - More LRSN spin avoidance
  - Faster machines revving LRSN faster
  - Why wait 16 milliseconds when you can wait 953 femtoseconds
- Overhead to normalize to 10 byte format for DB2 processing and extra code to write two styles of log records.
  - EMEA 2013 IDUG presentation asserted 3% to 4% overhead.

Other good references on Expanded RBA/LRSN

- EMEA IDUG 2013 Session B08 – Pimp My LRSN by Timm Zimmerman of IBM
- EMEA IDUG 2013 Session B07 – DB2 11 for z/OS - IDUG User Experiences by Julian Stuhler
- IBM Technote (FAQ) – The Expanded RBA and LRSN in DB2 11 for z/OS (url in notes)

Questions?

Jim Dee
BMC Software
jim_dee@bmc.com

DB2 11 Expanded RBA/LRSN – Does this bit make my tail look big?