DB2 LUW V11.1 CERTIFICATION TRAINING PART #2

Kent Collins @rcollins963
Mohankumar Saraswatipura @mohankumarsp

Agenda

1. BLU Acceleration Best Practices
2. New Monitoring Table Function
3. DB2 pureScale Enhancements
4. Sample Questions

DB2 BLU Acceleration

Data Load Best Practices

Column-Organized Table	Row-Organized Table
CREATE TABLE "MS"."T3" ("ID" INTEGER NOT NULL, "NAME" VARCHAR(20 OCTETS) , "ADDRESS" VARCHAR(50 OCTETS) , "SALARY" INT) DISTRIBUTE BY HASH("ID") IN "USERSPACE1" ORGANIZE BY COLUMN;	CREATE TABLE "MS"."T4" ("ID" INTEGER NOT NULL, "NAME" VARCHAR(20 OCTETS) , "ADDRESS" VARCHAR(50 OCTETS) , "SALARY" INT) COMPRESS YES ADAPTIVE DISTRIBUTE BY HASH("ID") IN "USERSPACE1" ORGANIZE BY ROW;
Records Loaded: 62,914,560	Records Loaded: 62,914,560

DB2 BLU Acceleration… Cont.

Synopsis Table

```
db2 "describe table SYSIBM.SYN170323092546645613_T3"
```

Column name	Data type schema	Data type name	Column Length	Scale	Nulls
IDMIN	SYSIBM	INTEGER	4	0	NO
IDMAX	SYSIBM	INTEGER	4	0	NO
NAMEMIN	SYSIBM	VARCHAR	20	0	Yes
NAMEMAX	SYSIBM	VARCHAR	20	0	Yes
ADDRESSMIN	SYSIBM	VARCHAR	50	0	Yes
ADDRESSMAX	SYSIBM	VARCHAR	50	0	Yes
SALARYMIN	SYSIBM	INT	8	0	Yes
SALARYMAX	SYSIBM	INT	8	0	Yes
TSNMIN	**SYSIBM**	**BIGINT**	**8**	**0**	**No**
TSNMAX	**SYSIBM**	**BIGINT**	**8**	**0**	**No**

```
  10 record(s) selected.
```

DB2 BLU Acceleration… Cont.

Synopsis Table

This is a meta-data describing which ranges of values exist in which portion of the user table.

- This enables data skipping when scanning data to fulfill a query request
- Pre-sorted data
- It is managed and maintained by DB2 automatically and transparently to the users
- It is a column-organized table and is compressed
- Generally 0.1 or 0.2% of the base table

For a table having 62,914,560 records, what will the synopsis table records be around?

61K ~ (62M/1024)

db2 "SELECT COUNT(*) FROM SYSIBM.SYN170323092546645613_T3"

```
1
-----------
      61512
```

DB2 BLU Acceleration… Cont.

ALTER TABLE MS.T3 ADD CONSTRAINT PK_T3 PRIMARY KEY (ID) ENFORCED;

db2 "SELECT SUBSTR(TABNAME,1,30) as TABNAME, SUBSTR(INDNAME,1,30) as INDNAME, INDEXTYPE, COMPRESSION from SYSCAT.INDEXES where TABNAME like '%T3%'"

```
TABNAME                        INDNAME                        INDEXTYPE COMPRESSION
------------------------------ ------------------------------ --------- ----
T3                             SQL170324061319724733          CPMA      Y
SYN170324061319264040_T3       SQL170324061319951469          CPMA      Y
T3                             PK_T3                          REG       Y

  3 record(s) selected.
```

CPMA: Column Page Map Index

Data Load Best Practices for BLU

- Batch up the data at least 1024 records per insert
- Set *UTIL_HEAP_SZ* to *AUTOMATIC* and over 1M pages
- Consider presorting data on column(s) that will frequently appear in query predicates to improve compression ratios and synopsis effectiveness
- Reduce total time required for the first load of the table by reducing the duration of the ANALYZE phase. This can be achieved by *cdeanalyzefrequency* and/or *maxanalyzesize* file type modifier in the LOAD command.

    - *cdeanalyzefrequency*: How frequently sample needs to be collected during ANALYZE phase?

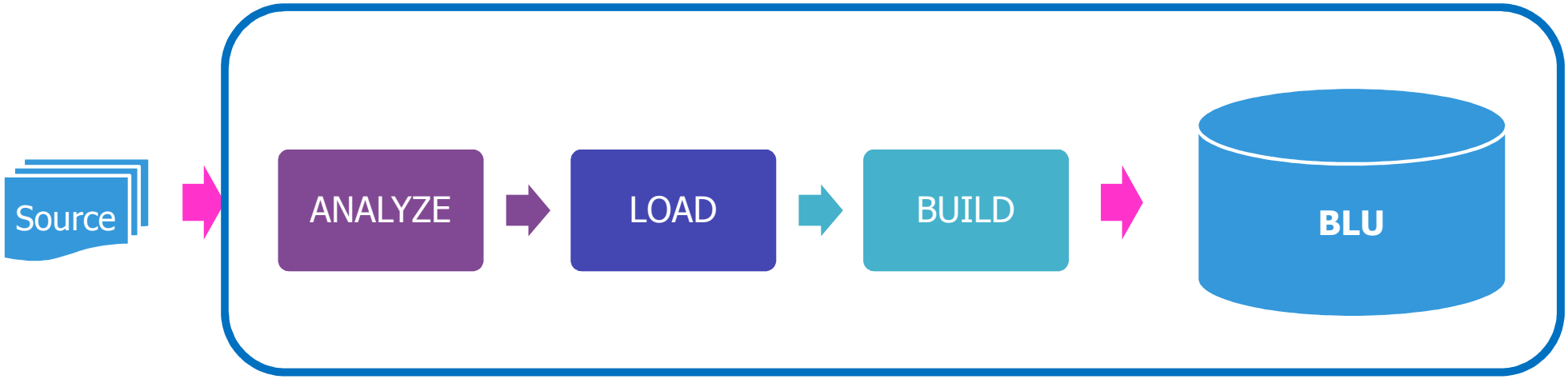
    - *maxanalyzesize*: How much data is sampled in the ANALYZE phase? Default is 128 GB and can be modified. Value ZERO means unlimited.

Data Load Best Practices for BLU… Cont.

LOAD	INGEST	IMPORT
Offline, parallel, bulk and non-logged operation	Online, parallel, transactional logged operation	Online, single threaded logged operation
SETUP, ANALYZE, LOAD and BUILD	Process a continuous data stream and perform an array insert	Serialized INSERT operation and pretty slow
Fast and yields best compression	Fast, however it is recommended to use LOAD for the initial data load to have best compression results and subsequent data load can use INGEST	Generally not a recommended way to load the data for BLU

Data Load Best Practices for BLU

LOAD Phases for BLU



- Analyze: This occurs only if dictionary creation is necessary. When necessary it builds histogram to track value frequency and column compression dictionaries.
- Load: Compress data pages, update synopsis table
- Build: Builds key for UNIQUE indexes

Data Load Comparison

Column-Organized	LOAD	INGEST	IMPORT
1ST Batch (12 M)	27 seconds (Analyze 16 sec, Load 10 sec, Build 1 sec)	21 seconds	14 minutes
2nd Batch (12 M)	14 seconds (Load 13 sec, build 1 sec)	21 seconds	14 minutes

Row-Organized	LOAD	INGEST	IMPORT
1ST Batch (12 M)	45 seconds	22 seconds	14 minutes
2nd Batch (12 M)	43 seconds	20 seconds	14 minutes

Table Size Comparison

```
SELECT
    SUBSTR(A.TABNAME,1,10 ) as TABNAME,
    ROWCOMPMODE,
    (DATA_OBJECT_P_SIZE + INDEX_OBJECT_P_SIZE +
COL_OBJECT_P_SIZE)/1024 as TOTAL_SIZE_MB
from SYSCAT.TABLES a, SYSIBMADM.ADMINTABINFO b
where
        A.TABNAME='T3' AND
        A.TABSCHEMA='MS' AND
        A.TABSCHEMA = B.TABSCHEMA AND
        A.TABNAME=B.TABNAME WITH UR;
```

Table	Records	Size
MS.T3 (Column-organized)	62,914,560	47 MB
MS.T4 (Row-organized)	62,914,560	843 MB

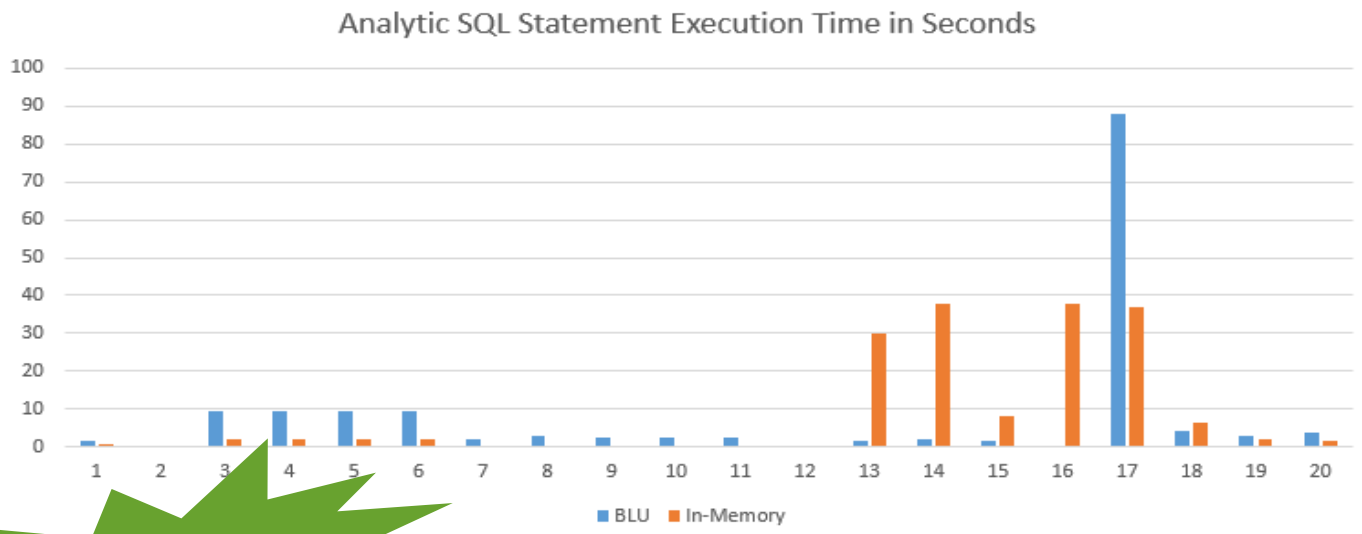
```
TABNAME    ROWCOMPMODE        TOTAL_SIZE_MB
---------- ----------- --------------------
T3                                       47
T4         A                        843.375
```

Execution Time Comparison

Table	SQL Statement	Time
MS.T3 (Column-organized)	SELECT COUNT (*)	0.033793 sec
MS.T4 (Row-organized)	SELECT COUNT (*)	16.973138 sec

Table	SQL Statement	Time
MS.T3 (Column-organized)	SELECT *	174.598066 sec
MS.T4 (Row-organized)	SELECT *	166.198395 sec

Analytical SQL Statement Performance Assessment



9% Better Performance with 1/5th the cores

	BLU	In-Memory
Overall Time:	155.60 Seconds	170.68 Seconds

New Monitoring Table Function

Lock	Latch
Locks assure logical consistency of the data in a table via record level or table level locks.	Latch is a lock at the internal memory structure consistency of the object in memory
Held till the logical completion of the transaction.	This will only be held at the page level for a short term.
Monitoring via SYSPROC.MON_GET_LOCKS	Monitoring via SYSPROC.MON_GET_LATCH

SYSPROC.MON_GET_LATCH

```
SELECT SUBSTR(LATCH_NAME,1,40) LATCH_NAME,
       SUBSTR(MEMORY_ADDRESS,1,20) MEMORY_ADDRESS,
       EDU_ID,
       SUBSTR(EDU_NAME,1,20) EDU_NAME,
       APPLICATION_HANDLE,
       LATCH_STATUS,
       LATCH_WAIT_TIME FROM TABLE ( MON_GET_LATCH( NULL, -2 ) )
ORDER BY LATCH_NAME, LATCH_STATUS;
```

LATCH_NAME	MEMORY_ADDRESS	EDU_ID	EDU_NAME	APPLICATION_HANDLE	LATCH_STATUS	LATCH_WAIT_TIME
SQLO_LT_SQLE_KRCB__E	0x0000000020005E0D2	1372	db2agntp (DPF_DB) 1	4726	H	-
SQLO_LT_SQLE_KRCB__E	0x0000000020005E0D2	732	db2agent (DPF_DB) 0	4726	H	-
SQLO_LT_SQLP_CTSDAEM	0x00007F8059D6D530	654	db2ctsdaemon (DPF_DB	3685	H	-
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1459	db2agntcol (DPF_DB)	5012	H	-
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1460	db2agntcol (DPF_DB)	5012	W	1576
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1458	db2agntcol (DPF_DB)	5012	W	1576
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1398	db2agntcol (DPF_DB)	5012	W	1576
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1375	db2agntcol (DPF_DB)	5012	W	1576
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1371	db2agntcol (DPF_DB)	5012	W	1574
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1368	db2agntcol (DPF_DB)	5012	W	1576
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1364	db2agntcol (DPF_DB)	5012	W	1576
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1360	db2agntcol (DPF_DB)	5012	W	1576
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1356	db2agntcol (DPF_DB)	5012	W	1575
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1281	db2agntcol (DPF_DB)	5012	W	1576
SQLO_LT_ibm_cde__que	0x00007F893F24C1A8	1277	db2agntcol (DPF_DB)	5012	W	1576
SQLO_LT_sqeWLDispatc	0x000000024BCC0470	15	db2wlmt 0	-	H	-

SYSPROC.MON_GET_LATCH

MON_GET_LATCH (<search_args>,<member>)

- application_handle
- latch_name
- edu_id
- latch_status

PureScale Enhancements

- Packaging
  - New Direct Advanced Edition
  - Passive/Active Licensing
- Improvements in Installation and Deployment
  - Install Complexity Reduced
  - Smarter Defaults
  - GPFS Replication step reduction from 30 to 4
  - db2cluster command enhancements
- Install recovery/restart logic
- Improved Documentation
- Improved Buffer Management
  - Truncate, Drop Table, Load/Import and Ingest speed increases

PureScale Enhancements - Continues

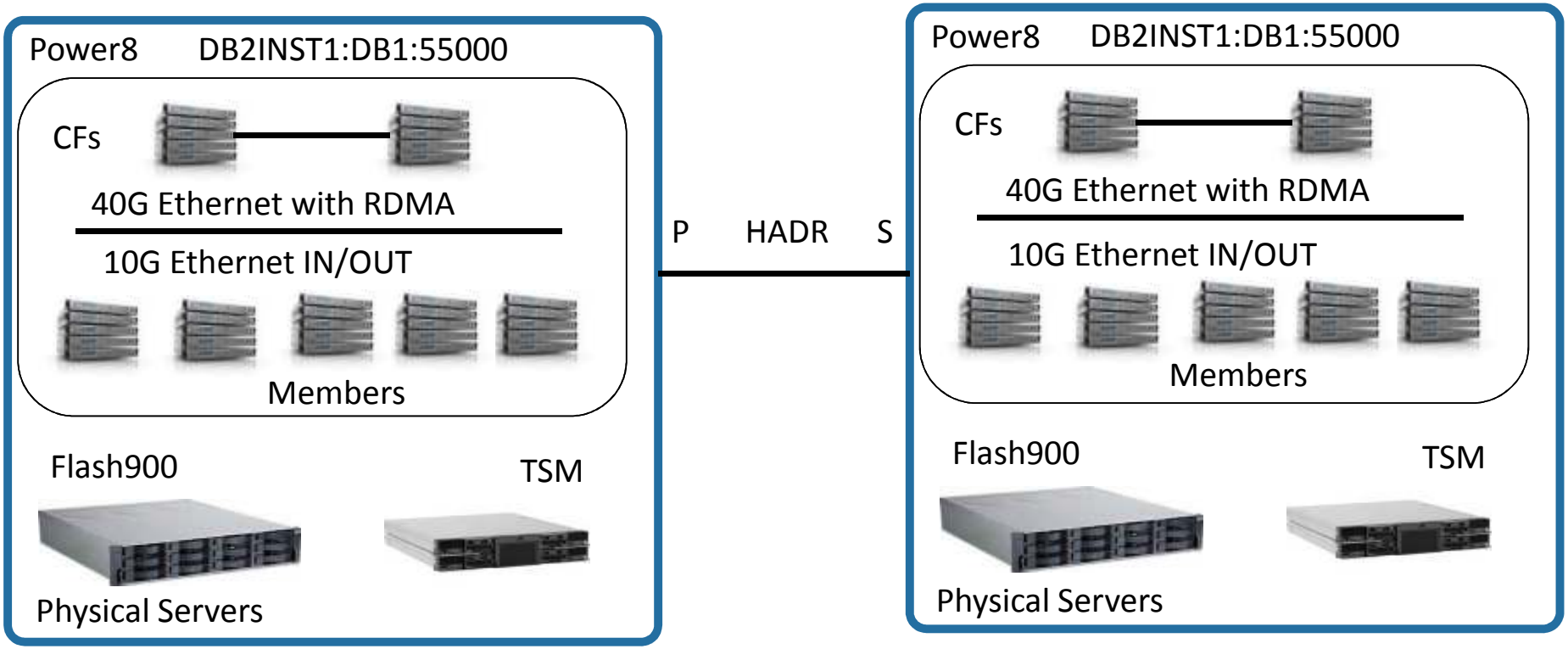
- Enhanced pre-checking script
- Improved Member Set Alternate Member Failover Priority
- Linux Virtual Enhancements
  - Single-Root I/O Virtualization
  - Shared Single Adapter across VMWare Partitions

- HADR support for Sync and NearSync Mode
- HADR Streamline Upgrade Support
- Crash Recovery Time improvements
  - Allow logs written during crash recovery to be replayed

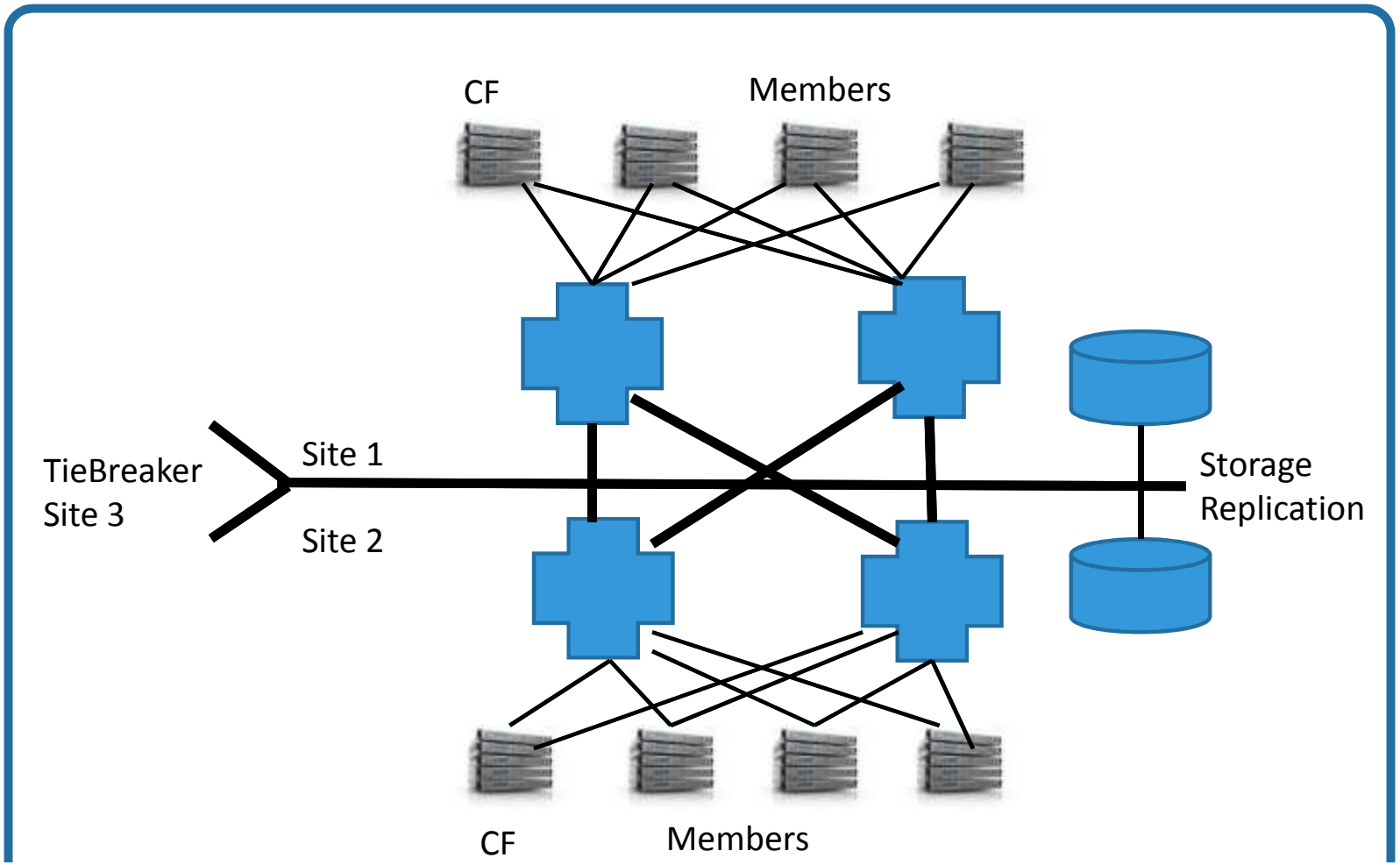
- GDPC Enhancements
  - Multiple adapter ports per member
  - Dual Switches

BSSL – (Bold, Simple, Stable, Limited) - HADR

No SPOF: DB2 Version Upgrade Outage

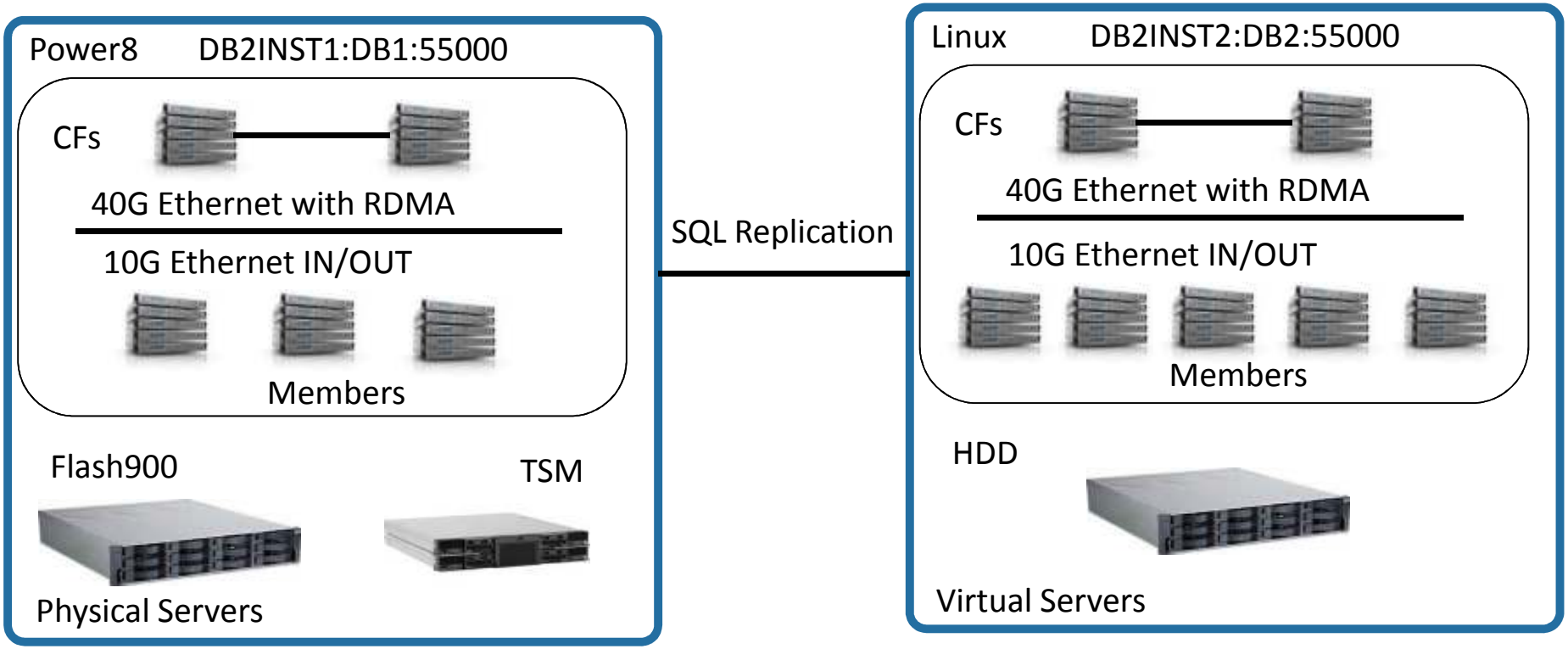


TCNL – (Tight, Complex, Network, Limited) - GDPC



SAAU(Simple, Always Available, Unlimited)- CDC/QRep

No SPOF: No Scheduled Outage Requirement



PureScale Monitoring

Three Ways to get Information

- ✓ Table Functions and Administrative Views

DB2_GET_CLUSTER_HOST_STATE	Table Function
DB2_CLUSTER_HOST_STATE	View
DB2_GET_INSTANCE_INFO	Table Function
DB2_MEMBER	View
DB2_CF	View
DB2_INSTANCE_ALERTS	View

- ✓ CLP Commands

LIST INSTANCE
LIST INSTANCE SHOW DETAIL
LIST APPLICATIONS AT MEMBER <node>
LIST APPLICATIONS GLOBAL

PureScale Monitoring

Three Ways to get Information

- ✓ Shell Commands

db2instance –list
db2cluster -list *options*
db2nps <node> db2_ps executes db2nps on each member
db2nkill <node> db2_kill Kills all engine processes on all member drastically
db2_all <command> Executes <command> on all logical nodes

db2cluster -list –alert
db2cluster -cm -verify –resources
db2cluster -cfs -verify -configuration

PureScale Index Space Reclaim

List reclaimable space for all indexes in schema XX

```
SELECT TRIM (indname) AS index_name,
       TRIM (tabname) AS TABLE_NAME,
       iid,
       index_object_l_size,
       index_object_p_size,
       reclaimable_space
  FROM TABLE (admin_get_index_info ('', 'XX', NULL)) AS T1
ORDER BY reclaimable_space DESC
  WITH UR@
```

INDEX_NAME	TABLE_NAME	INDEX_OBJECT_L_SIZE	INDEX_OBJECT_P_SIZE	RECLAIMABLE_SPACE
XPK	TB1	1262208	1262208	901824
XD1	TB1	266880	266880	179776

PureScale Index Space Reclaim

Reorg Index ME.XPK with CLEANUP ALL AND Reclaim Extents

REORG index XX.XPK for table TB1 allow write access
CLEANUP ALL RECLAIM EXTENTS @

INDEX_NAME	TABLE_NAME	INDEX_OBJECT_L_SIZE	INDEX_OBJECT_P_SIZE	RECLAIMABLE_SPACE
XPK	TB1	285184	285184	704

PureScale Index Space Reclaim

Run All Indexes for table TB6 CLEANUP ALL with Reclaim Extents

reorg indexes all for table TB6 allow write access
CLEANUP ALL RECLAIM EXTENTS @

INDEX_NAME	TABLE_NAME	INDEX_OBJECT_L_SIZE	INDEX_OBJECT_P_SIZE	RECLAIMABLE_SPACE
XD2	TB6	174464	174464	150272
XPK	TB6	174464	174464	150272

INDEX_NAME	TABLE_NAME	INDEX_OBJECT_L_SIZE	INDEX_OBJECT_P_SIZE	RECLAIMABLE_SPACE
XD2	TB6	16384	16384	768
XPK	TB6	16384	16384	768

Get all the space for a single index across data partitions

```
SELECT TRIM (indname) AS index_name,
      TRIM (tabname) AS TABLE_NAME,
      SUM(index_object_l_size) AS  index_object_l_size,
      SUM(index_object_p_size) AS index_object_p_size,
      SUM(reclaimable_space) AS reclaimable_space
  FROM TABLE (admin_get_index_info ('', 'XX', '')) AS T1
GROUP BY indname, tabname
HAVING SUM(reclaimable_space) > 0
ORDER BY reclaimable_space DESC
WITH UR@
```

INDEX_NAME	TABLE_NAME	INDEX_OBJECT_L_SIZE	INDEX_OBJECT_P_SIZE	RECLAIMABLE_SPACE
XD1	TB7	149504	149504	103296
XPK	TB7	149504	149504	103296

INDEX_NAME	TABLE_NAME	INDEX_OBJECT_L_SIZE	INDEX_OBJECT_P_SIZE	RECLAIMABLE_SPACE
XD1	TB7	41472	41472	128
XPK	TB7	41472	41472	128

PureScale Tablespace Reclaimable Storage

```
SELECT varchar(tbsp_name, 30) as tbsp_name,
  reclaimable_space_enabled,
  tbsp_free_pages,
  tbsp_page_top,
  tbsp_usable_pages,
  ( tbsp_free_pages - (tbsp_usable_pages - tbsp_page_top)) as
freeSpaceBelowHWMinPages,
   tbsp_page_size * ( tbsp_free_pages - (tbsp_usable_pages - tbsp_page_top))
as freeSpaceBelowHWMinBytes
FROM TABLE(MON_GET_TABLESPACE('',-2)) AS t
where ( tbsp_free_pages - (tbsp_usable_pages - tbsp_page_top)) > 0
ORDER BY freeSpaceBelowHWMinPages desc @
```

TBSPNM	RECLAIMABLE_SPACE	TBSP_FREE_PAGES	TBSP_PAGE_TOP	TBSP_USABLE_PAGES	FREESPACEBELOWHWM	FREESPACEBELOWHWM
TS1	1	2195424	1644160	2555776	1283808	5258477568
TS2	1	2195424	1644160	2555776	1283808	5258477568

PureScale Table CF Potential Conflict Report

```
select substr(tabname,1,55) as name
     , substr(objtype,1,5) as type
     , page_reclaims_x as PGRCX
     , page_reclaims_s as PGRCS
     , spacemappage_page_reclaims_x as SMPPGRCX
     , spacemappage_page_reclaims_s as SMPPGRCS
from table( mon_get_page_access_info('XX', NULL,
NULL ) ) AS WAITMETRICS
order by page_reclaims_x DESC, page_reclaims_s DESC
with ur @
```

NAME	TYPE	PGRCX	PGRCS	SMPPGRCX	SMPPGRCS
TB1	INDEX	60244	17	69	0
TB1	INDEX	38477	4	44	0
TB1	TABLE	12945	258	26	1

Thank you!

Robert (Kent) Collins

@rollins963

Kent, founder of Shiloh Consulting, Inc., is currently a Data Solutions Architect with BNSF Railway. He is an IBM Champion (2010–2017) and a frequent speaker at the DB2Night Show and IDUG and IOD (IBM Insight) conferences. Kent has worked continually with DB2 from its introduction to the market in 1984, amassing a wealth of knowledge and experience. He graduated from the University of Texas at Dallas with majors in mathematics and computer science. He is an IBM Certified Solutions Expert and also holds certifications in DB2, AIX, Linux, .NET, Oracle, SQL Server, Windows, and z/OS. Kent is proficient in many programming languages and, as a Java Architect, specializes in Enterprise HA/Performance systems. He lives in Dallas with his wife, Vicki; together, they have three children and one grandchild.

Mohankumar Saraswatipura

@mohankumarsp

Mohan works as a Database Solutions Architect focusing on IBM DB2, Linux, UNIX, and Windows solutions. Prior to his current position, he worked as a Database Solutions Architect at Reckitt Benckiser Group, plc (UK) focusing on IBM Smart Analytics System 5600, Siebel, SQL Server, and SAP HANA solutions. He is an IBM Champion (2010–2017) and a *DB2's Got Talent* 2013 winner. Mohan has written dozens of technical papers for IBM developerWorks and *IBM Data* Magazine. He is an IBM Certified DB2 Advanced Database Administrator, DB2 Application Developer, and DB2 Problem Determination Master. Mohan holds a Master's of Technology (M Tech) degree in computer science and an Executive MBA (IT).

Please contact us if you have any questions @ www.EnterpriseDB2.com