What's hidden in the ARCHIVES - DB2's answer to American Pickers ?

Steen Rasmussen, Sr. Engineering Services Architect
CA technologies

DISCLAIMER

Scenarios outlined performed on a DB2 11 NFM with maintenance as of June 2015 (RSU1506).

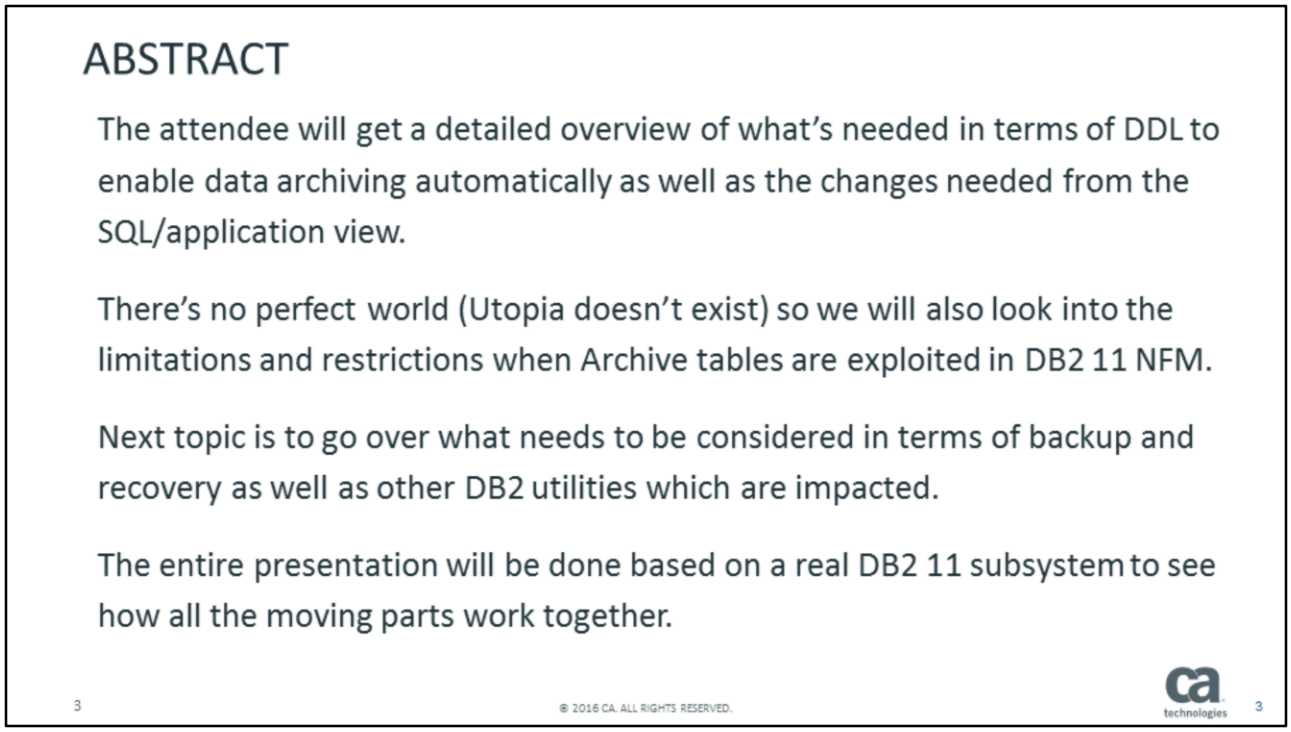
Discrepancies do exist between the documentation and some of the live scenarios.

Opinions and expressions are my own and do not reflect CA technologies or IBM.

2

All live scenarios have been performed on a DB2 11 NFM system with maintenance applied up to June 2015.

Once we get to the LIMITATIONS / RESTICTIONS section, the IBM documentation does describe limitations I did not find to be true, but maintenance can have changed this.



BIO:

Steen Rasmussen is a Sr. Engineering Services Architect currently instrumental in the ongoing development and support of the CA Technologies DB2 tools. In 1985 Steen started as an IMS/DB2 DBA at a major insurance company in Denmark working with all aspects of DB2 - like tuning, application design and implementation, education of developers, backup and recovery planning and automation of housekeeping processes. During this job, Steen also served as a member of the planning committee for DB2 GUIDE SHARE Nordic Region. In 1995 Steen became a technical manager at PLATINUM Technology managing technical support and presales for the DB2 products.
Steen has been working with DB2 for z/OS Release 1.0 since 1985 and is always looking for new opportunities in the CA DB2 solutions which can help customers manage DB2 more efficiently. Besides from providing support to the teams in the field as well as internal groups at CA technologies working with DB2, Steen is also a frequent speaker at IDUG in North America, EMEA and Australia as well as local DB2 User Groups around the world. Since 2014 Steen has been an IBM Information Management Champion. Since 2013 Steen has been the CA liaison for IDUG NA.

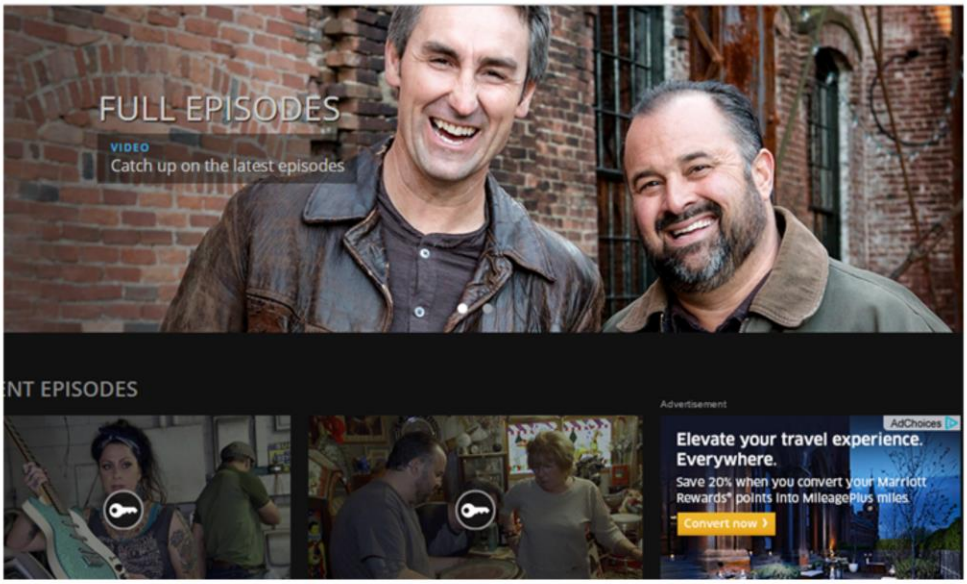
Agenda

- DDL changes to Enabled Archive Tables
- DML changes / add-on's to facilitate Archive tables
- Restrictions, limitations and gotcha's
- Utility considerations
- A case study using a real DB2 11 system – some ideas about implementation (used throughout the presentation).

First we will cover why archiving can be of interest and what we have done so far since this has been an issue since the early age of computing.
Next we will dive into the DB2 schema changes needed to enable archiving before moving into the application / SQL changes introduced.
We will spend quite some time dealing with the limitations introduced once archive tables are in place – talking about what to be aware of, how to deal with these challenges and some interesting gotcha's I discovered while "playing" with archive tables.
Next is what impacts DB2 utilities and what you need to be aware of.

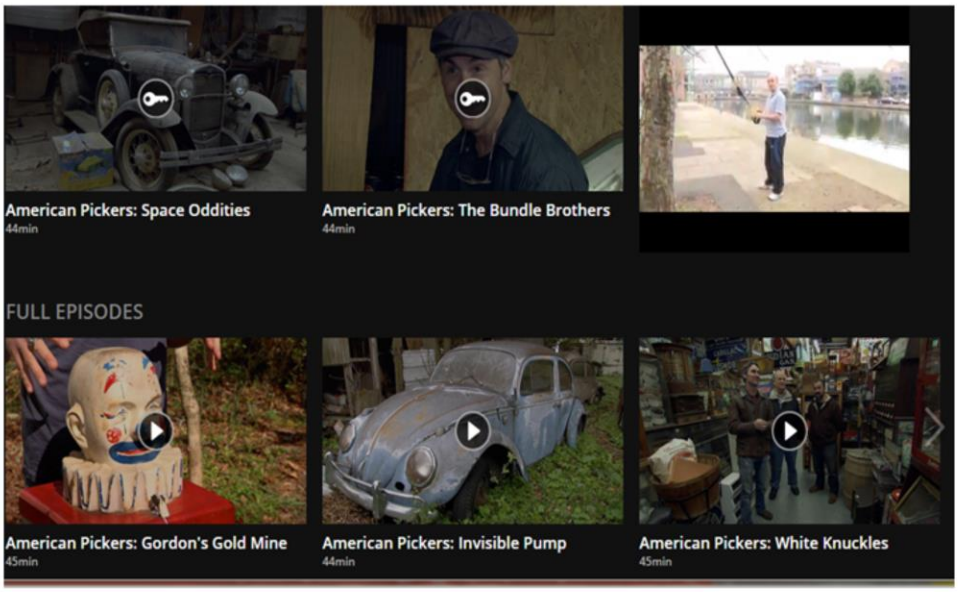
Througout the presentation, a live DB2 11 NFM system has been used to illustrate the power of archive tables and how all the pieces really work.

Mike Wolfe and Frank Fritz are on a mission to recycle America, scouring the country for hidden gems in junkyards, basements, garages and barns, meeting ...



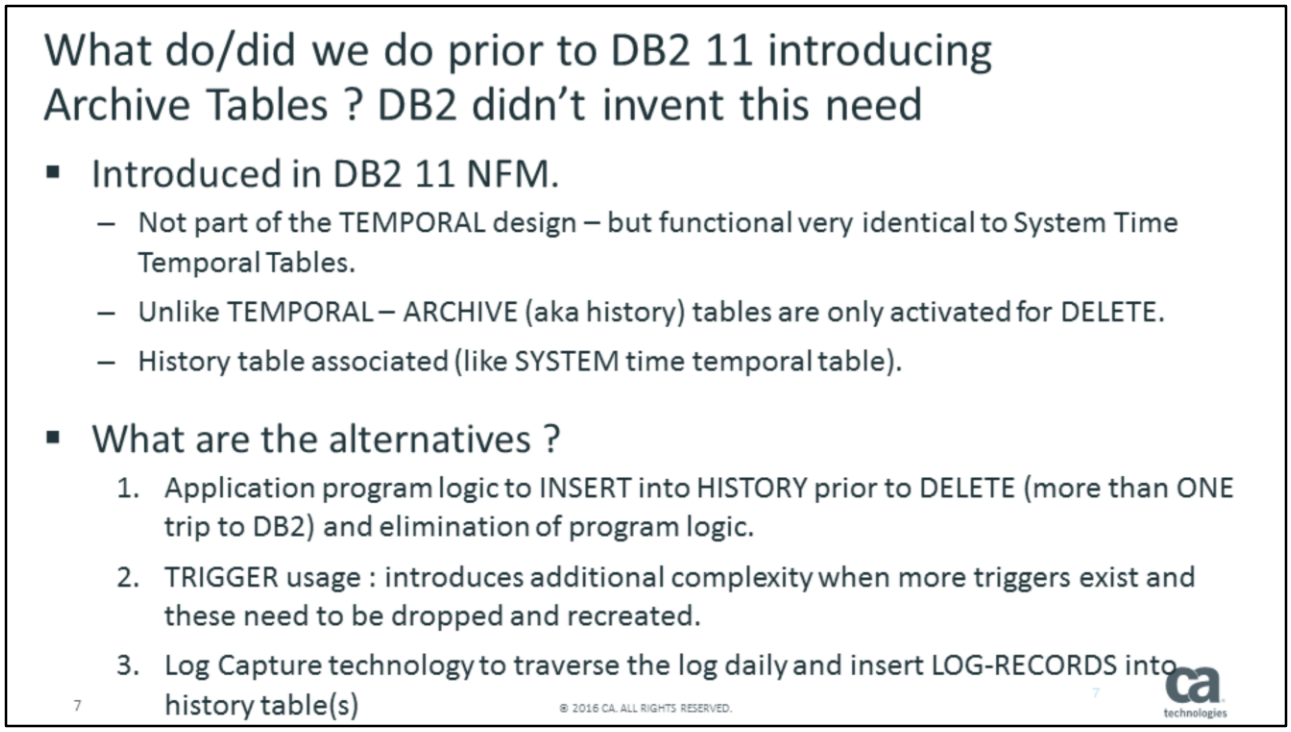
5 5

The TV channel in the US named HISTORY CHANNEL ahs a program called “American Pickers”. Two guys named Mike and Frank tour around the US “picking” from what people have been collecting for decades in order to resell the items for a profit.



The TV channel in the US named HISTORY CHANNEL ahs a program called “American Pickers”. Two guys named Mike and Frank tour around the US “picking” from what people have been collecting for decades in order to resell the items for a profit.

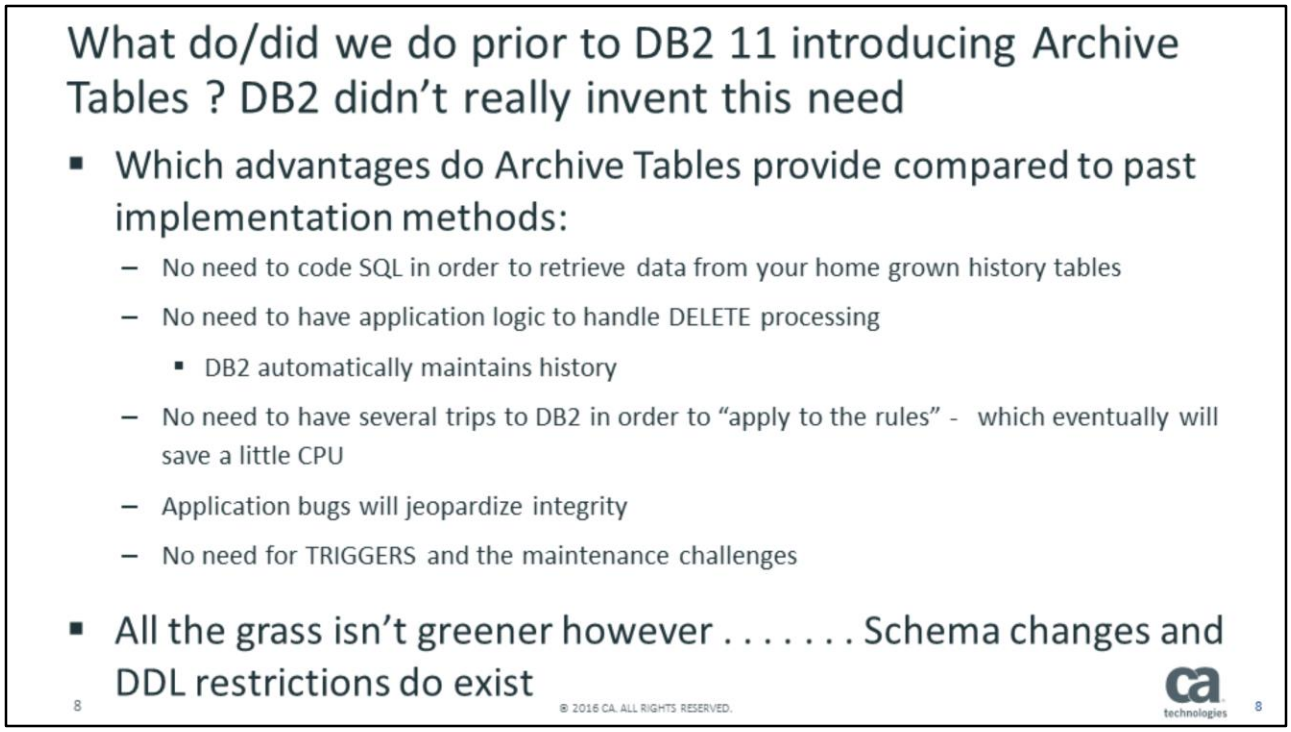
The tv program can be quite educational since the history behind old artifacts are covered.



Just because DB2 11 introduced Archive Tables doesn't really mean we haven't had this need in the past – it has existed since the first day of IT.

What has been introduced in DB2 11 to handle archival of data looks very similar to the DB2 10 feature named TEMPORAL TABLES – especially the System Time Temporal table design.
Both have a HISTORY table associated, but the archival design only saves the image of the base table when a delete is executed unlike System time temporal tables where also UPDATES are saved in the history table.

As already mentioned, the need to archive data has existed since forever. The current implementation used range from pure Application logic to using triggers and using the DB2 log to capture transactions and apply these to history tables via one of the log tools available in the market.



So why is the DB2 11 Archive Table feature more attractive than the current implementations ?

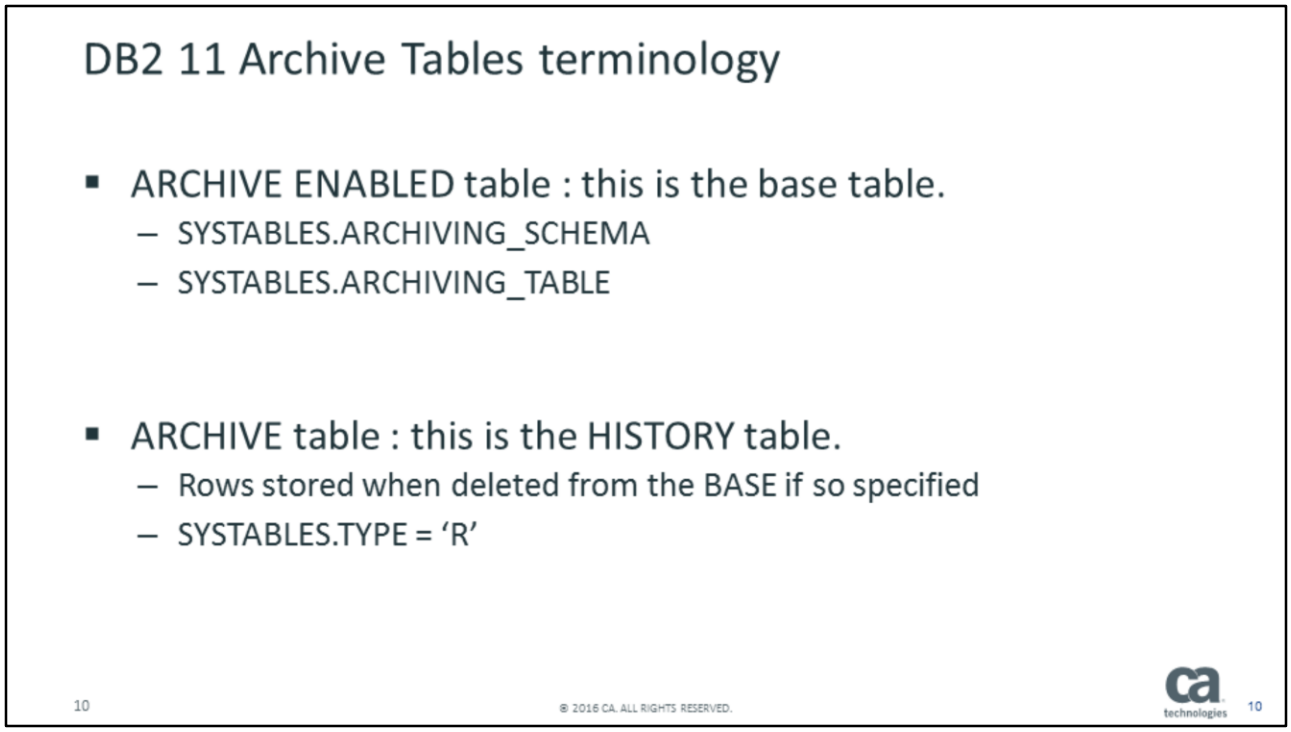
It takes time to code, maintain, test application code to insert rows into a history table prior to deleting. Archive-enabled tables handle this automatically.
One of the most expensive pieces in today's DB2 environment is TRIPS TO DB2. Every trip saved is resources preserved, and the INSERT followed by a DELETE from the application is TWO trips – not to count the retrieval.

Triggers can reduce the previous mentioned pieces, but then again – triggers add complexity in a DB2 environment.

Despite all the GREAT stuff introduced, as we will see later, there are some schema changes we need to be aware of and consider how we will cope with these.

There are many reasons why IT organizations want to archive data:

1) DASD savings despite the fact that DASD has dropped dramatically over the years.
2) One of my favorites is less INDEX LEVELS – every level decreased is one less I/O for EVERY statement using the index.
3) GETPAGE activity – another favorite similar to the previous one – if you have 25% less rows there's a good change you will save 25% GETP from DASD into the bufferpool unless doing single-row random access.
4) UTILITY COSTS can be a major factor since every LOAD, REORG, RUNSTATS, RECOVER and potentially other utilities will need to access fewer pages.
5) If we really look back – when new applications are implemented, NOBODY think about cleaning up or archiving – and the fact that regulations might change over time so you MIGHT have to retain deleted rows.
6) Once the application is implemented – what does it cost to implement a DATA ARCHIVE DATA solution ????? In fact it never happens – well almost.



Let's start with the official IBM DB2 terminology:

The table where your BASE/Production data is (the current table having your transactional data is named the ARCHIVAL-ENABLED table – I like to name it the BASE TABLE – and the ARCHIVE TABLE which is where the DELETED rows end up is named the ARCHIVE table – I like to name it the HISTORY table.

DB2 Archive Tables DDL – how does it all work

- Tablespaces created for BASE and HISTORY
  - Tablespace attributes can be different

```
CREATE  TABLESPACE ARCHBASE
    IN IDUG2015
    USING STOGROUP SYSDEFLT
                ERASE NO
      FREEPAGE 0
      PCTFREE 0
      MAXPARTITIONS   1
      BUFFERPOOL BP1
      LOCKSIZE ANY
      CLOSE YES
      SEGSIZE 04
      LOCKMAX 0
  ;
```

```
CREATE  TABLESPACE ARCHHIST
    IN IDUG2015
    USING STOGROUP SYSDEFLT
                ERASE NO
      FREEPAGE 0
      PCTFREE 0
      NUMPARTS   9
      BUFFERPOOL BP1
      LOCKSIZE ANY
      CLOSE YES
      SEGSIZE 04
      LOCKMAX 0
      CCSID EBCDIC
      DSSIZE 4G
  ;
```

DB2 Archive Tables DDL – how does it all work

Tables for BASE and HISTORY (have to be identical –

NO differences – well, sort of !!!)

```
CREATE  TABLE IDUG15.ARCHBASE
      ( EMPNO CHAR ( 6 ) NOT NULL
      , FIRSTNME VARCHAR ( 12 ) NOT NULL
      , MIDINIT CHAR ( 1 ) NOT NULL
      , LASTNAME VARCHAR ( 15 ) NOT NULL
      , WORKDEPT CHAR ( 3 )
      , PHONENO CHAR ( 4 )
      , HIREDATE DATE
      , JOB CHAR ( 8 )
        FOR SBCS DATA
      , EDLEVEL SMALLINT
      , SEX CHAR ( 1 )
      , BIRTHDATE DATE
      , SALARY DECIMAL ( 9 , 2 )
      , BONUS DECIMAL ( 9 , 2 )
      , COMM  DECIMAL ( 9 , 2 )
      )
      IN IDUG2015.ARCHBASE
  DATA CAPTURE CHANGES   CCSID   EBCDIC ;
```

```
CREATE  TABLE IDUG15.ARCHHIST
      ( EMPNO CHAR ( 6 ) NOT NULL
      , FIRSTNME VARCHAR ( 12 ) NOT NULL
      , MIDINIT CHAR ( 1 ) NOT NULL
      , LASTNAME VARCHAR ( 15 ) NOT NULL
      , WORKDEPT CHAR ( 3 )
      , PHONENO CHAR ( 4 )
      , HIREDATE DATE
      , JOB CHAR ( 8 )
        FOR SBCS DATA
      , EDLEVEL SMALLINT
      , SEX CHAR ( 1 )
      , BIRTHDATE DATE
      , SALARY DECIMAL ( 9 , 2 )
      , BONUS DECIMAL ( 9 , 2 )
      , COMM  DECIMAL ( 9 , 2 )                )
      IN IDUG2015.ARCHHIST
PARTITION BY ( HIREDATE )
      ( PARTITION 1     ENDING AT ( '2010-12-31' )
      , PARTITION 2     ENDING AT ( '2011-12-31' )
      , PARTITION 3     ENDING AT ( '2012-12-31' )
      , PARTITION 4     ENDING AT ( '2013-12-31' )
      , PARTITION 5     ENDING AT ( '2014-12-31' )
      , PARTITION 6     ENDING AT ( '2015-12-31' )
      , PARTITION 7     ENDING AT ( '2016-12-31' )
      , PARTITION 8     ENDING AT ( '2017-12-31' )
      , PARTITION 9     ENDING AT ( '2018-12-31' ) )
  DATA CAPTURE CHANGES      CCSID       EBCDIC    ;
```

DB2 Archive Tables DDL – how does it all work

- Indexes created for both base and history

```
RQTXC  18.0   -------- RC/Q Table Index Column list --------- 2015/08/04 12:36
COMMAND ===>                                                  SCROLL ===> PAGE

 DB2 Object ===> T                     Option  ===> XC   Where => N
 Table Name ===> ARC%                 > Creator ===> IDUG15                 >
  Qualifier ===> *                    > N/A     ===> *                      >
Loc: LOCAL ---------- SSID: D11A ----------RASST02 -          LINE 1 OF 9    >
CMD      TABLE NAME      INDEX NAME      INDEXED COLUMN  COLSEQ ORD  CLS  UNQ
________ ARCHBASE
________                 ARCHBASEIX
________                                 EMPNO              1    A    N    D
________ ARCHHIST
________                 ARCHHISTIX
________                                 EMPNO              1    A    N    D
________                 ARCHHISTIX2
________                                 EMPNO              1    A    N    U
________                                 HIREDATE           2    A    N    U
****************************** BOTTOM OF DATA ********************************
```


13

DB2 Archive Tables DDL – how does it all work

- TIE the BASE and HISTORY to ENABLE *(or disable)* archiving is very simple.

```
ALTER TABLE  IDUG15.ARCHBASE
  ENABLE  (DISABLE)  ARCHIVE
    USE    IDUG15.ARCHHIST ;
```

- Now it's time to start INSERT/DELETE/UPDATE processing and see how data flows.

TABLE NAME	CREATOR	DATABASE	TBLSPACE	NUMBER OF ROWS
ARCHBASE	IDUG15	IDUG2015	ARCHBASE	45
ARCHHIST	IDUG15	IDUG2015	ARCHHIST	0

DB2 Archive Tables DML – how does it all work

- Insert / Update BASE doesn't impact the ARCHIVE table

COLUMN NAME	NULL	DATA FOR ROW #2
EMPNO		900000
FIRSTNME		STEEN
MIDINIT		
LASTNAME		RASMUSSEN
WORKDEPT	N	E21
PHONENO	N	4667
HIREDATE	N	1995-04-01
JOB	N	FIELDREP
EDLEVEL	N	16
SEX	N	M
BIRTHDATE	N	1961-04-13
SALARY	N	85000.00
BONUS	N	25000.00
COMM	N	320000.00

```
UPDATE IDUG15.ARCHBASE
SET SEX='M' ,
HIREDATE = '2000-01-15'
Where EMPNO='000015' ;
```

```
 SELECT * FROM IDUG15.ARCHHIST ;

DSNT404I SQLCODE = 100, NOT FOUND:  ROW
NOT FOUND FOR FETCH, UPDATE, OR DELETE,
OR THE RESULT OF A QUERY IS AN EMPTY
TABLE
```


15

Special registers / Global variables :

- Two new GLOBAL VARIABLEs will help manage DML:

- SYSIBMADM.GET_ARCHIVE = Y / N
  - Should SQL against BASE include rows from ARCHIVE table ?
- SYSIBMADM.MOVE_TO_ARCHIVE
  - E : Delete from BASE results in INSERT into ARCHIVE table
  - Y : SQL INSERT/UPDATE on ARCHIVE ENABLED / HISTORY table will result in SQL-ERROR
  - N : Delete from BASE will not insert into ARCHIVE

DB2 Archive Tables DML – how does it all work

- DELETE processing without special registers enabled.
  - Lets “FIRE” Steen Rasmussen
  - Without setting the special REGISTER – ARCHIVE table not maintained !!!

```
COLUMN NAME          NULL  DATA FOR ROW #
44
EMPNO                      900000
FIRSTNME                   STEEN
MIDINIT
LASTNAME                   RASMUSSEN
WORKDEPT             N     E21
PHONENO              N     4667
HIREDATE             N     1995-04-01
JOB                  N     FIELDREP
EDLEVEL              N     16
SEX                  N     M
BIRTHDATE            N     1961-04-13
SALARY               N     85000.00
BONUS                N     25000.00
COMM                 N     320000.00
```

```
DELETE FROM IDUG15.ARCHBASE WHERE
EMPNO='900000' ;

DSNT400I SQLCODE = 000,  SUCCESSFUL
EXECUTION

COMMIT;

DSNT400I SQLCODE = 000,  SUCCESSFUL
EXECUTION

SELECT * FROM IDUG15.ARCHHIST    ;

Nothing displays !
```

DB2 Archive Tables DML – how does it all work

- Let's ACTIVATE the special REGISTERS – and DELETE !!
  - What happened ?
  - The PARTITION LIMITKEY for the HISTORY table did NOT accept.

```
SET SYSIBMADM.MOVE_TO_ARCHIVE = 'E' ;
 DSNT400I SQLCODE = 000,  SUCCESSFUL EXECUTION
DELETE FROM IDUG15.ARCHBASE WHERE EMPNO < '000030' ;
 DSNT408I SQLCODE = -327, ERROR:  THE ROW CANNOT BE INSERTED BECAUSE IT
          IS OUTSIDE THE BOUND OF THE PARTITION RANGE FOR THE LAST
          PARTITION


COLUMN NAME         NULL  DATA FOR ROW # 2
EMPNO                     000015
FIRSTNME                  Bob
MIDINIT                   I
LASTNAME                  Smith
WORKDEPT            N     A00
PHONENO             N     3978
HIREDATE            Y     ----------   <---------- NULL VALUE
JOB                 N     PRES
EDLEVEL             N     18
SEX                 N     F
BIRTHDATE           Y     ----------
SALARY              N     90200.00
BONUS               N     20000.00
COMM                N     4220.00
****************************** BOTTOM OF DATA **********
```


18

DB2 Archive Tables DML – how does it all work

- Let's ACTIVATE the special REGISTERS – and DELETE !!
  - In the mean time – HIREDATE updated to be NOT NULL
  - TWO rows qualified for the DELETE
  - And these have been INSERT'ed into the ARCHIVE table.

```
SET SYSIBMADM.MOVE_TO_ARCHIVE = 'E' ;
 DSNT400I SQLCODE = 000,  SUCCESSFUL EXECUTION
DELETE FROM IDUG15.ARCHBASE WHERE EMPNO < '000030' ;
 DSNT400I SQLCODE = 000,  SUCCESSFUL EXECUTION
 COMMIT WORK ;
 DSNT400I SQLCODE = 000,  SUCCESSFUL EXECUTION


SELECT * FROM IDUG15.ARCHHIST ;

EMPNO  FIRSTNME       MIDINIT LASTNAME        N:WORKDEPT N:PHONENO N
------------------------------------------------------------------
000015 Bob            I       Smith           N A00      N 3978    N
000020 MICHAEL        L       THOMPSON        N B01      N 3476    N
```


19

DB2 Archive Tables – RETRIEVE data

- You can select from BASE or HISTORY as you like

```
SELECT * FROM IDUG15.ARCHHIST    ;
 DSNT400I SQLCODE = 000,  SUCCESSFUL EXECUTION
SELECT * FROM IDUG15.ARCHBASE    ;
 DSNT400I SQLCODE = 000,  SUCCESSFUL EXECUTION

EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE
000015 Bob          I       Smith           A00      3978    2000-01-15
000020 MICHAEL      L       THOMPSON        B01      3476    1973-10-10

 43 ROWS RETRIEVED
EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE
000340 JASON        R       GOUNOT          E21      5698    1947-05-05
000030 SALLY        B       KWAN            C01      4738    1975-04-05
000050 JOHN         B       GEYER           E01      6789    1949-08-17
000060 IRVING       F       STERN           D11      6423    1973-09-14
000070 EVA          D       PULASKI         D21      7831    1980-09-30
000090 EILEEN       W       HENDERSON       E11      5498    1970-08-15
000100 THEODORE     Q       SPENSER         E21      0972    2005-06-19
. . . . . .
```

DB2 Archive Tables – RETRIEVE data

- Implement the ARCHIVE feature into your applications without rewriting SQL in the “good old programs”
- The special REGISTERS come to help
- Modifications to application code NOT needed – just one small addition (below the result without special register).

```
SELECT * FROM IDUG15.ARCHBASE WHERE EMPNO < '000040';
 DSNT400I SQLCODE = 000,  SUCCESSFUL EXECUTION

EMPNO  FIRSTNME     MIDINIT LASTNAME
000030 SALLY        B       KWAN


SELECT * FROM IDUG15.ARCHHIST WHERE EMPNO < '000040';
 DSNT400I SQLCODE = 000,  SUCCESSFUL EXECUTION

EMPNO  FIRSTNME     MIDINIT LASTNAME
000015 Bob          I       Smith
000020 MICHAEL      L       THOMPSON
```


21

DB2 Archive Tables – RETRIEVE data

- Use GLOBAL VARIABLE to describe if rows retrieved from BASE and HISTORY

```
SET SYSIBMADM.GET_ARCHIVE  =  'Y' ;

SELECT * FROM IDUG15.ARCHBASE WHERE EMPNO < '000040';

EMPNO  FIRSTNME     MIDINIT LASTNAME
000030 SALLY        B       KWAN
```

- Didn't you expect to see three rows in total ?
- One additional parameter is the BIG BOSS.
  - Bind parameter ARCHIVESENSITIVE(YES) was missing.

DB2 Archive Tables – RETRIEVE data

- Let's try again.

```
SET SYSIBMADM.GET_ARCHIVE  =  'Y' ;

DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0

SELECT * FROM IDUG15.ARCHBASE WHERE EMPNO < '000040';

EMPNO   FIRSTNME      MIDINIT  LASTNAME         WORKDEPT
---------+---------+---------+---------+---------+-------
000030  SALLY         B        KWAN             C01
000015  Bob           I        Smith            A00
000020  MICHAEL       L        THOMPSON         B01
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 1
```

DB2 Archive Tables – RETRIEVE data

- What does DB2 do when GET_ARCHIVE and ARCHIVESENSITIVE in place ?
- EXPLAIN uncovers the truth.

```
SET SYSIBMADM.GET_ARCHIVE  =  'Y' ;
EXPLAIN PLAN SET QUERYNO = 5002 FOR
SELECT * FROM IDUG15.ARCHBASE WHERE EMPNO < '000040';
COMMIT;
SELECT * FROM PLAN_TABLE WHERE QUERYNO = 5002 ;
```

- DB2 rewrites the single select to become a UNION ALL

QUERYNO	QBLOCKNO	PLANNO	CREATOR	TNAME	TABNO	ACCESSTYPE	MATCHCOLS
5002	1	1	IDUG15	ARCHHIST	2	R	0
5002	5	1	IDUG15	ARCHBASE	1	I	1
5002	2	1			0		0

DB2 Archive Tables – manipulate data

- It does matter how MOVE_TO_ARCHIVE is set

```
SET SYSIBMADM.MOVE_TO_ARCHIVE='Y' ;
UPDATE IDUG15.ARCHBASE SET SEX ='X' WHERE SEX='M' ;

---------+---------+---------+---------+---------+---------+---------+---------
DSNT408I SQLCODE = -20555, ERROR:  AN ARCHIVE-ENABLED TABLE IS NOT ALLOWED IN
         THE SPECIFIED CONTEXT. REASON CODE  2
DSNE618I ROLLBACK PERFORMED, SQLCODE IS 0


SET SYSIBMADM.MOVE_TO_ARCHIVE='E' ;
UPDATE IDUG15.ARCHBASE SET SEX ='X' WHERE SEX='M' ;
---------+---------+---------+---------+---------+---------+------
DSNE615I NUMBER OF ROWS AFFECTED IS 24
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

  - Both “E” and “Y” will insert DELETEs into the associated history table.
  - Only UPDATE / INSERT are impacted

Limitations, Restrictions and Challenges and . . . . .

- If the SYSIBMADM.GET_ARCHIVE global variable is set to 'Y' and the ARCHIVESENSITIVE bind option is set to 'Y', an archive-enabled table cannot be referenced in an inline SQL table function or in the definition of a row permission or column mask that is activated by a data change statement or query

- A SELECT statement can't reference both BASE and HISTORY (I am confused !!)

```
SELECT * FROM IDUG15.ARCHBASE  UNION ALL SELECT * FROM IDUG15.ARCHHIST
  ORDER BY EMPNO
---------+---------+---------+---------+---------+---------+---------+-------
EMPNO    FIRSTNME      MIDINIT  LASTNAME          WORKDEPT  PHONENO  HIREDATE
---------+---------+---------+---------+---------+---------+---------+-------
000015  Bob           I        Smith             A00       3978     2000-01-15
000020  MICHAEL       L        THOMPSON          B01       3476     1973-10-10
000030  SALLY         B        KWAN              C01       4738     1975-04-05
000050  JOHN          B        GEYER             E01       6789     1949-08-17
000060  IRVING        F        STERN             D11       6423     1973-09-14
000070  EVA           D        PULASKI           D21       7831     1980-09-30

SELECT A.* FROM IDUG15.ARCHBASE A, IDUG15.ARCHHIST B   WHERE A.EMPNO=B.EMPNO
This one also works
```

Limitations, Restrictions and Challenges and . . . . .

- Let's do the UNION with GET_ARCHIVE='Y'

```
SET SYSIBMADM.GET_ARCHIVE  =  'Y' ;
SELECT * FROM IDUG15.ARCHBASE  UNION ALL SELECT * FROM IDUG15.ARCHHIST
        ORDER BY EMPNO
--------+--------+--------+--------+--------+--------+--------+-------
EMPNO   FIRSTNME      MIDINIT  LASTNAME        WORKDEPT  PHONENO  HIREDATE
--------+--------+--------+--------+--------+--------+--------+-------
000015  Bob           I        Smith           A00       3978     2000-01-15
000015  Bob           I        Smith           A00       3978     2000-01-15
000020  MICHAEL       L        THOMPSON        B01       3476     1973-10-10
000020  MICHAEL       L        THOMPSON        B01       3476     1973-10-10
000030  SALLY         B        KWAN            C01       4738     1975-04-05
000050  JOHN          B        GEYER           E01       6789     1949-08-17
000060  IRVING        F        STERN           D11       6423     1973-09-14
```

  - Be prepared to get duplicates
  - Due to GET_ARCHIVE = 'Y' the HISTORY table is accessed twice

Limitations, Restrictions and Challenges and . . . . .

- Lets do the JOIN with GET_ARCHIVE='Y'

```
SET SYSIBMADM.GET_ARCHIVE  =  'Y' ;
SELECT A.* FROM IDUG15.ARCHBASE A
              ,IDUG15.ARCHHIST B
  WHERE A.EMPNO=B.EMPNO
  ORDER BY A.EMPNO
---------+---------+---------+---------+---------+---------+---------+-------
EMPNO    FIRSTNME      MIDINIT  LASTNAME         WORKDEPT  PHONENO  HIREDATE
---------+---------+---------+---------+---------+---------+---------+-------
000015  Bob           I        Smith            A00       3978     2000-01-15
000020  MICHAEL       L        THOMPSON         B01       3476     1973-10-10
DSNE610I NUMBER OF ROWS DISPLAYED IS 2
```

  - Now you can see which rows have been DELETED from the BASE and RE-INSERTED

28

Limitations, Restrictions and Challenges and . . . . .

- Let's INSERT into BASE from HISTORY – and RE-DELETE

```
SET SYSIBMADM.MOVE_TO_ARCHIVE = 'E' ;
---------+---------+---------+---------+---------+---------+---------+------
INSERT INTO IDUG15.ARCHBASE SELECT * FROM IDUG15.ARCHHIST WHERE
           EMPNO = '000015' ;
---------+---------+---------+---------+---------+---------+---------+------
DSNE615I NUMBER OF ROWS AFFECTED IS 1

COMMIT;


DELETE FROM IDUG15.ARCHBASE WHERE EMPNO='000015' ;
---------+---------+---------+---------+---------+---------+---------+---------+
DSNT408I SQLCODE = -803, ERROR:  AN INSERTED OR UPDATED VALUE IS INVALID
         BECAUSE INDEX IN INDEX SPACE ARCH1U64 CONSTRAINS COLUMNS OF THE TABLE
         SO NO TWO ROWS CAN CONTAIN DUPLICATE VALUES IN THOSE COLUMNS.
         RID OF EXISTING ROW IS X'0000000201'.
```

  - Think about UNIQUE indexes for the HISTORY table !!

Limitations, Restrictions and Challenges

- You can't:
  - ADD column to history ➔ ADD column to BASE, DB2 will handle history
  - SECURITY LABEL column or ROW ACCESS CONTROL
  - Column attribute alterations for any of them
  - RENAME column or RENAME table
  - DROP column
  - ADD PERIOD (temporal) for any of them
  - ADD versioning (temporal)
  - PK / FK for history
  - ROTATE partition

Limitations, Restrictions and Challenges

- You can't:
  - ADD column LONG VARCHAR / VARGRAPHIC
  - ALTER CCSID
  - MQT involved
  - ADD CLONE
  - AUX objects
  - IDENTITY column
  - PENDING CHANGES ?!
  - VIEW and DGT
  - Only one table in tablespace ➔ has to be UTS

```
ALTER TABLESPACE IDUG2015.ARCHBASE
      SEGSIZE 08;

DSNT404I SQLCODE = 610, WARNING:
A CREATE/ALTER ON OBJECT
IDUG2015.ARCHBASE HAS PLACED OBJECT
IN ADVISORY REORG PENDING
```


31

How to cope with Schema Changes

- Schema changes can be a challenge as noted on the previous slides

```
ALTER TABLE IDUG15.ARCHHIST ADD COLUMN NEWCOL CHAR(4) NOT NULL
WITH DEFAULT;

DSNT408I SQLCODE = -20525, ERROR:  THE REQUESTED ACTION IS NOT
VALID FOR TABLE IDUG15.ARCHHIST BECAUSE THE TABLE IS THE WRONG
TYPE OF TABLE. REASON CODE = 13

ALTER TABLE IDUG15.ARCHBASE ADD COLUMN NEWCOL CHAR(4) NOT NULL
WITH DEFAULT;

DSNT408I SQLCODE = -20385, ERROR:  THE STATEMENT CANNOT BE
PROCESSED BECAUSE THERE ARE PENDING DEFINITION CHANGES FOR
OBJECT IDUG2015.ARCHBASE OF TYPE  TABLESPACE (REASON 2)
```


32

How to cope with Schema Changes

- Once the PENDING REORG is fixed:

```
ALTER TABLE IDUG15.ARCHBASE
ADD COLUMN NEWCOL CHAR(4)
NOT NULL WITH DEFAULT;

DSNE616I STATEMENT EXECUTION
WAS SUCCESSFUL, SQLCODE IS 0
```

- DB2 does really add the new column to the history automatically

How to cope with Schema Changes

- The method to handle schema changes which are allowed for archive-enabled and history tables is very identical to those related to temporal tables:
  - Remove the tie : ALTER TABLE ARCHBASE DISABLE ARCHIVE
  - Potentially unload data from both base and history
  - Perform schema changes – and assure the restrictions aren't violated.
  - Potentially load the data back (no need to think about PERIODOVERRIDE and TRANSIDOVERRIDE)
  - Alter table ENABLE ARCHIVE.
  - Copy, Runstats, Rebind etc.

34

Utilities and LISTDEF processing

- LISTDEF has a new parameter – it is a great feature – but . . . . .

```
OPTIONS PREVIEW
LISTDEF TEMPOR1
        INCLUDE    TABLESPACE IDUG2015.ARCHBASE
LISTDEF TEMPOR2
        INCLUDE    TABLESPACE IDUG2015.ARCHBASE ALL
LISTDEF TEMPOR3
        INCLUDE    TABLESPACE IDUG2015.ARCHBASE ARCHIVE
LISTDEF TEMPOR4
        INCLUDE    TABLESPACE IDUG2015.ARCHBASE ALL ARCHIVE
LISTDEF TEMPOR0
        INCLUDE    TABLESPACE IDUG2015.ARCHBASE
        INCLUDE    TABLESPACE IDUG2015.ARCHBASE ALL ARCHIVE
LISTDEF TEMPOR5
        INCLUDE    TABLESPACE IDUG2015.ARCHHIST
LISTDEF TEMPOR6
        INCLUDE    TABLESPACE IDUG2015.ARCHHIST ALL
LISTDEF TEMPOR7
        INCLUDE    TABLESPACE IDUG2015.ARCHHIST ARCHIVE
LISTDEF TEMPOR8
        INCLUDE    TABLESPACE IDUG2015.ARCHHIST ALL ARCHIVE
```


35

Utilities and LISTDEF processing

```
INCLUDE base              : only BASE selected
INCLUDE base ALL          : only BASE selected
INCLUDE base ARCHIVE      : NONE selected
INCLUDE base ALL ARCHIVE  : only HISTORY selected


INCLUDE hist              : only HISTORY selected
INCLUDE hist ALL          : only BASE selected
INCLUDE hist ARCHIVE      : only HISTORY selected
INCLUDE hist ALL ARCHIVE  : only HISTORY selected
```

Utilities and LISTDEF processing

- Always include “special object” keywords.
- Best practices – no need to memorize.
- Might be overkill . . . But better safe than sorry

```
OPTIONS PREVIEW
LISTDEF TEMPOR0

        INCLUDE    TABLESPACE DAP02DB.DAP02TS

        INCLUDE    TABLESPACE DAP02DB.DAP02TS
                        ALL ARCHIVE

        INCLUDE    TABLESPACE DAP02DB.DAP02TS
                      ALL HISTORY
```


37

Utilities and LISTDEF processing

- Some of the newer object types have special considerations regarding utilities.
  - RUNSTATS for CLONE related objects
  - LOAD for Temporal tables
  - RECOVER for Temporal tables

- Archive tables seem a lot easier to deal with
  - No restrictions identified when loading data – including REPLACE of HISTORY
  - Archive-enabled and archive table can be recovered independently

WRAP UP

- Implementing ARCHIVE tables is very straight forward
- Minimal application changes to adopt this technology
- No need to be scared implementing archive tables

- Hoping this session was useful and that you can benefit right away.

THANK YOU !!

39

Steen Rasmussen
Sr. Engineering Services Architect
Steen.rasmussen@ca.com