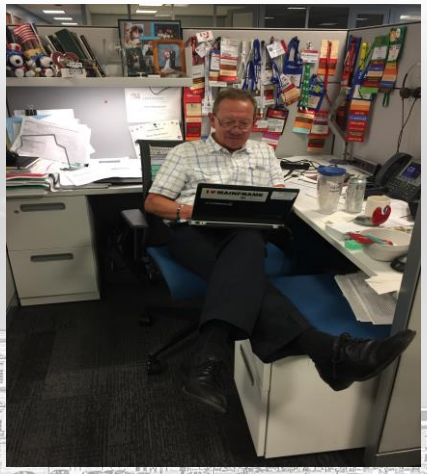
IDUG
Leading the DB2 User Community since 1988

IDUG DB2 EMEA Tech Conference
Lisbon Portugal | October 2017

#IDUGDB2

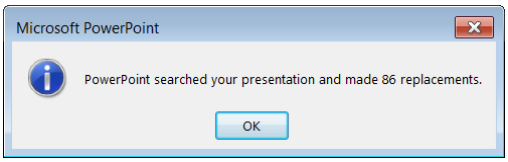
The Db2 12 catalog – what happened since Db2 11

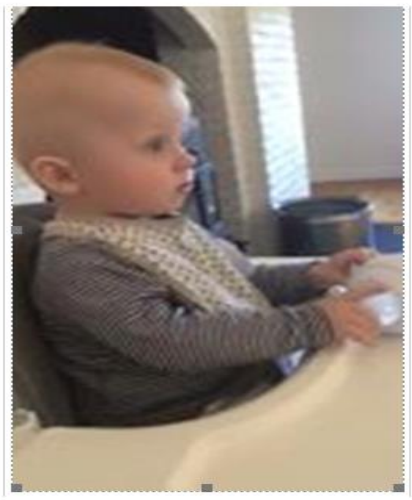


Steen Rasmussen
Principal Engineering Services Architect,

CA technologies

It is Db2 and not DB2





Agenda

- Dropped catalog objects.
- New catalog objects.
- Old Db2 features added to the catalog.
- Modified catalog tablespaces, catalog tables, columns, indexes, relationships.
- New Db2 12 features reflected in the catalog.

This presentation has 5 main topics:

Objects removed from the Db2 catalog.

New tablespaces and tables as well as indexes.

Part of the catalog can adopt Db2 features which were made available in Db2 10.

A few catalog objects have been modified too.

Finally, a glance into existing objects with modified information reflecting the new Db2 12 features.

DISCLAIMER

- Expressions are purely my own – not CA technologies.
- This presentation is based on using a real life Db2 12 system FUNCTION LEVEL(V12R1M500) with maintenance as of November 2016
- Discrepancies do exist comparing with Db2 SQL Reference Guide APPENDIX A Catalog Tables – and this presentation (as of January 2017).

ABSTRACT

- There are many ways to learn about a new Db2 version, but using the catalog is probably the fastest and most enlightening. Once you look at the revision track in the Db2 Catalog and Directory section of the SQL Reference Guide, study the column changes, additions as well as new catalog tables, you have a very good overview of everything worth to know about that specific Db2 version.

The catalog always reflect the features available in a Db2 release, so browsing through the IBM Db2 SQL Reference Guide to look at the new tables and also existing columns with new content as well as existing tables with new columns can provide a quick view of all the changes in that specific Db2 release.

Why do you have to understand the Db2 Catalog changes ?

- You can read the What's New Guide
- The IDUG website has an excellent White Paper
- You have some kind of tooling to query the catalog !
- If you don't know what's IN there – how do you know how to benefit ?
- If using your own SQL or homegrown REXX's or the like:
  - You might have to consider adding additional join predicates
  - You probably need to change FILTERING (WHERE clauses)
  - You might need to code new queries
  - And probably many more reasons . . . . . .

7

You might question why it can be an advantage to understand the catalog changes between releases.

1) After all you can study the release guide

2) IDUG usually provides some kind of white paper highlighting all the new goodies

3) Most Db2 sites have a catalog query tool from IBM or one of the vendors – and some might be using SPUFI or have some homegrown solution in place.

Bottom line, if you don't understand all the details, you might be missing out some great opportunities.

If you are using SPUFI or a homegrown solution, you probably need to change some of the queries or add new ones, in which case its even more important that you have a good understanding of the changes.

Dropped Catalog Objects

Nothing removed from the Catalog

- No additional PBG conversions like Db2 10 and 11
- Still some old multi-table tablespaces and non-PBG

```
SYSALTER with SYSOBDS
SYSCONTX with 3 tables
SYSDDF with 8 tables
SYSEBCDC with 2 tables
SYSGPAUT with SYSRESAUTH
SYSGRTNS with 2 tables
SYSHIST with 11 tables
SYSJAUXA with SYSJARDATA
SYSJAUXB with
SYSJARCLASS_SOURCE
SYSJAVA with 4 tables
SYSPLUXA with SYSROUTINESTEXT
SYSPLUXB with SYSROUTINES_TREE
```

```
SYSROLES with 2 tables
SYSSEQ with SYSSEQUENCES
SYSSEQ2 with 2 tables
SYSSTATS with 9 tables
SYSTARG with SYSKEYTARGETS
SYSXML with 2 tables
```

9

Nothing has been removed from DSNDB06 unlike what was the case in Db2 10 and Db2 11 where some multi-table tablespaces were moved into PBG tablespaces.

Db2 12 has NOT converted any of the remaining multi-table tablespaces, so this is the list of some “good old objects”.

You might question why these haven’t been converted to UTS since many Db2 features require UTS – maybe IBM has no plans to exploit UTS-only features for these objects ??

New catalog objects.

Catalog objects

- A historical view of the evolution in terms of #objects

	#TS	#LOB Cols	#TB	#TB Cols	#IX	#Constraints
Db2 V1R1M0	11	0	25	291	27	
. . . . .						
Db2 V8	23	2	85	1333	133	
Db2 V9	26	3	104	1714	166	
Db2 10	77	18	116	2036	234	192
Db2 11	115	21	151	2231	250	197
Db2 12	149	33	185	2934	275	233

- Not all objects included (SYSTSTAB, DSNPROGAUTH, Accelerator tables, …)
- DSNDB01 included since Db2 10
- Constraints include RI and table check constraints
- New tablespaces follow the “rule” from Db2 10/11 : PBG and understandable naming convention

This is a short overview illustrating the Db2 catalog objects over the past +30 years. For comparison reasons, SYSTSTAB and a few other tablespace are excluded since they are not “mandatory”.

DSNDB01 (the Directory) is included beginning with Db2 10 when these “tables” became select’able.

The good news in Db2 12 is, the naming convention and tablespace attributes follow the standard introduced in Db2 10 and continued in Db2 11.

SYSLEVELUPDATES catalog table

- A few ways to identify where you are on the Db2 Continuous Delivery Model : one being DISPLAY GROUP

```
---------------------- DB2 Command Processor ---------------------- 2016/12/27 06:37
COMMAND ===>                                                        SCROLL ===> PAGE

------------------------------------------------------------------  User ID: RASST02
      | -DIS GROUP                                                                 |
      |                                                                            |
      |                                                                            |
********************************** TOP OF DATA ***********************************
DSN7100I  !D12A DSN7GCMD
*** BEGIN DISPLAY OF GROUP(........) CATALOG LEVEL(V12R1M500)
                  CURRENT FUNCTION LEVEL(V12R1M500)
                  HIGHEST ACTIVATED FUNCTION LEVEL(V12R1M500)
                  HIGHEST POSSIBLE FUNCTION LEVEL(V12R1M500)
                  PROTOCOL LEVEL(2)
                  GROUP ATTACH NAME(....)
------------------------------------------------------------------------------
DB2          SUB                          DB2    SYSTEM    IRLM
MEMBER   ID  SYS  CMDPREF   STATUS   LVL    NAME      SUBSYS IRLMPROC
-------- --- ---- --------  -------- ------ --------  ----   --------
........   0 D12A !D12A     ACTIVE   121500 CA31      I12A   D12AIRLM
------------------------------------------------------------------------------
SPT01 INLINE LENGTH:       32138
*** END DISPLAY OF GROUP(........)
DSN9022I  !D12A DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
********************************** BOTTOM OF DATA ********************************
```

12

One method to find out where you are and where you have been as well as where you can go is to execute the DIS GROUP command.

This is one method where you can verify:

1) Where you are in terms of maintenance – what is the highest possible FUNCTION LEVEL which can be activated due to the current maintenance.
2) Where you were previously
3) What function level you can use

SYSLEVELUPDATES catalog table

- Another method is looking at this new catalog table.
  - Where are you : FUNCTION_LVL
  - Where did you come from : PREV_FUNCTION_LVL
  - Have you been in the future in the past : HIGH_FUNCTION_LVL
  - Where is the Db2 catalog : CATALOG_LVL
  - No indexes and might not be needed for years

```
RUBROWF  19.0   ---------- RC/Browse: Form Mode ---------- 2016/12/27 06:43:30
COMMAND ===>                                                SCROLL ===> CSR

For Table    => SYSIBM.SYSLEVELUPDATES
Browse Mode => F
SSID: D12A ----------FETCH STATUS: COMPLETE-----
##.COLUMN NAME          NULL  DATA FOR ROW # 1
A1.FUNCTION_LVL               V12R1M500
A2.PREV_FUNCTION_LVL          V12R1M100
A3.HIGH_FUNCTION_LVL          V12R1M500
A4.CATALOG_LVL                V12R1M500
A5.OPERATION_TYPE             M
A6.EFFECTIVE_TIME             2016-09-01-18.27.36.821189395019
A7.EFFECTIVE_LRSN             ......W.C4
A8.OPERATION_TEXT             CATMAINT PROCESSING - DB2DEV
A9.GROUP_MEMBER
******************************** BOTTOM OF DATA ********************************
```

OPERATION TYPE:
C: Catalog level change.
F: Function level change.
M: Code level change.

One new table is SYSLEVELUPDATES which is very useful due to the agile approach Db2 has taken – becoming a continuous delivery model.

This table basically provides the same information as the DIS GROUP command, but I expect this table to also illustrate the historical path – still to be determined once new function levels become available.

The historical view of OPERATION TYPE and FUNCTION LEVEL could provide a lot of valuable information once Db2 12 gets “older”.

System Time Temporal available for RTS

- Optional to enable history of RTS tables.
  - Two new tablespaces and tables

TABLE_NAME	CREATOR	DATABASE	TBLSPACE
SYSIXSPACESTATS_H	SYSIBM	DSNDB06	SYSTSISH
SYSTABSPACESTATS_H	SYSIBM	DSNDB06	SYSTSTSH

- History table name length kept less than 18 byte (unfortunately ?)

```
ALTER TABLE SYSIBM.SYSINDEXSPACESTATS
ADD VERSIONING USE HISTORY TABLE SYSIBM.SYSIXSPACESTATS_H;

ALTER TABLE SYSIBM.SYSTABLESPACESTATS
ADD VERSIONING USE HISTORY TABLE SYSIBM.SYSTABSPACESTATS_H;
```

14

Db2 10 introduced three types of Temporal Tables – and now Db2 12 offers the same functionality for two catalog tables :The RTS (Real Time Statistics) tables.

This is an optional feature not being mandated.

In order to activate SYSTEM TIME temporal tables for the RTS tables, two simple ALTER commands are needed (described in the SQL REFERENCE GUIDE).

From my opinion, I would have preferred to simply add " _H " to the RTS tables for the history tables instead of shortening them.

System Time Temporal – future planning ?

- What is IBM planning ? Maybe this provides a clue ? !

- Catalog Table changes in the past has not very often indicated what we can expect in the future – Db2 12 may be different
- We have had history tables for several releases covering ***statistics history*** :

  SYSIBM.SYSxxxxxx**_HIST**

We just mentioned RTS can be enabled with system time temporal tables, but what's quite interesting is to have a closer look at which tablespaces and tables exist in the Db2 12 catalog but are not referenced in the SQL Reference Guide's Appendix A section describing the catalog tables.

This is probably the first time that the Db2 catalog tables are prepared for the future and kind of eludes what to expect.

The _HIST tables have existed for many releases providing the ability to store historical data for RUNSTATS. There are a complete new set of DIFFERENT history tables – let's have a closer look.

System Time Temporal – future planning ?

- One common component for these OLD and NEW objects – all have the same columns appended (see next slide).

Tablespace	Table	AUX	Suggested Base Table	Suggested Base Tablespace
SYSTSADH	SYSAUDITPOLICIES_H		SYSAUDITPOLICIES	SYSTSADT
SYSTSFAH	SYSCOLAUTH_H		SYSCOLAUTH	SYSTSFAU
SYSTSCXH	SYSCONTEXTAUTHID_H		SYSCONTEXTAUTHIDS	SYSCONTX
SYSTSCNH	SYSCONTEXT_H		SYSCONTEXT	SYSTSCON
SYSTSCTH	SYSCONTROLS_H		SYSCONTROLS	SYSTSCTL
SYSTSCHD	SYSCONTROLS_DESC_H	X	SYSCONTROLS_DESC	SYSTSCTD
SYSTSCHR	SYSCONTROLS_RTXT_H	X	SYSCONTROLS_RTXT	SYSTSCTR
SYSTSTAH	SYSCTXTTRUSTATTR_H		SYSCTXTTRUSTATTRS	SYSCONTX
SYSTSDBH	SYSDBAUTH_H		SYSDBAUTH	SYSTSDBU
SYSTSPKH	SYSPACKAUTH_H		SYSPACKAUTH	SYSTSPKA
SYSTSPLH	SYSPLANAUTH_H		SYSPLANAUTH	SYSTSPLA
SYSTSAUH	SYSRESAUTH_H		SYSRESAUTH	SYSGPAUT
SYSTSRAH	SYSROUTINEAUTH_H		SYSROUTINEAUTH	SYSTSRAU
SYSTSSCH	SYSSCHEMAAUTH_H		SYSSCHEMAAUTH	SYSTSSCM
SYSTSSAH	SYSSEQUENCEAUTH_H		SYSSEQUENCEAUTH	SYSSEQ2
SYSTSTBH	SYSTABAUTH_H		SYSTABAUTH	SYSTSTAU
SYSTSUAH	SYSUSERAUTH_H		SYSUSERAUTH	SYSUSER
SYSTSVDH	SYSVIEWDEP_H		SYSVIEWDEP	SYSTSVWD

16

To me these tables seem to indicate that we can expect all the tables holding authorizations (GRANT's) to be temporal as well – they all have a couple of attributes in common (please see next page).

System Time Temporal – future planning ?

- These appended columns could indicate a potential enablement of System Time Temporal Table (just like the RTS tables can be enabled in Db2 12)

```
CREATE TABLE "SYSIBM  ".SYSSEQUENCEAUTH
    (GRANTOR VARCHAR(128)      FOR MIXED DATA NOT NULL
    ,GRANTEE VARCHAR(128)      FOR MIXED DATA NOT NULL
    ,"SCHEMA" VARCHAR(128)     FOR MIXED DATA NOT NULL
    ,"NAME" VARCHAR(128)       FOR MIXED DATA NOT NULL
    ,GRANTEETYPE CHARACTER(1)  FOR MIXED DATA NOT NULL
    ,AUTHHOWGOT CHARACTER(1)   FOR MIXED DATA NOT NULL
    ,ALTERAUTH CHARACTER(1)    FOR MIXED DATA NOT NULL
    ,USEAUTH CHARACTER(1)      FOR MIXED DATA NOT NULL
    ,"COLLID" VARCHAR(128)     FOR MIXED DATA NOT NULL
    ,CONTOKEN CHARACTER(8)     FOR BIT DATA NOT NULL
    ,GRANTEDTS TIMESTAMP (6)   WITHOUT TIME ZONE NOT NULL
    ,IBMREQD CHARACTER(1)      FOR MIXED DATA NOT NULL
    ,GRANTORTYPE CHARACTER(1)  FOR MIXED DATA NOT NULL WITH DEFAULT
    ,SYS_START TIMESTAMP (12) WITHOUT TIME ZONE NOT NULL
                               GENERATED ALWAYS AS ROW BEGIN
    ,SYS_END TIMESTAMP (12)   WITHOUT TIME ZONE NOT NULL
                               GENERATED ALWAYS AS ROW END
    ,TRANS_START TIMESTAMP (12) WITHOUT TIME ZONE   NOT NULL
                               GENERATED ALWAYS AS TRANSACTION START ID
    ,PERIOD SYSTEM_TIME
     ( SYS_START, SYS_END )
     )
     IN DSNDB06.SYSSEQ2
```

17

All these tables have had the special columns/attributes appended which are needed to turn them into temporal tables.

However – if you try to enable history, Db2 will block you from doing so.

System Time Temporal – future planning ?

- Another view of “preparation for the future”
- System Time changes – but not enabled to history tables.

```
-- SYSXXX.SYSAUDITPOLICIES WILL BE ALTERED VIA THESE NATIVE Db2 COMMANDS
-- .

ALTER   TABLE SYSXXX.SYSAUDITPOLICIES
        ADD SYS_START TIMESTAMP ( 12 )
          NOT NULL GENERATED ALWAYS AS  ROW BEGIN
            ;

ALTER   TABLE SYSXXX.SYSAUDITPOLICIES
        ADD SYS_END TIMESTAMP ( 12 )
          NOT NULL GENERATED ALWAYS AS  ROW END
            ;

ALTER   TABLE SYSXXX.SYSAUDITPOLICIES
        ADD TRANS_START TIMESTAMP ( 12 )
          NOT NULL GENERATED ALWAYS AS  TRANSACTION START ID
            ;

ALTER   TABLE SYSXXX.SYSAUDITPOLICIES
        ADD PERIOD SYSTEM_TIME ( SYS_START , SYS_END ) ;
```


Doing a schema compare between these tables in Db2 11 and Db2 12 illustrated that the Db2 upgrade process has appended the needed columns to the base catalog tables.

System Time Temporal – future planning ?

- Maybe we are getting ready to get an "autonomic audit trail" of GRANT/REVOKE
- If so – changes are required for some catalog objects.
  - SYSSEQUENCEAUTH sits in a tablespace with TWO tables
  - VCAT defined tablespace (like prior to other tablespaces converted to PBG)

```
RQTST  19.0   ------- RC/Q Table Space Table Inquiry -------- 2016/12/31 09:17
COMMAND ===>                                                 SCROLL ===> CSR

 Db2 Object ===> TS                    Option  ===> T    Where => N
Table Space ===> SYSSEQ2              > Creator ===> *                      >
  Data Base ===> DSNDB06              > N/A     ===> *                      >
Loc: LOCAL ---------- SSID: D12A ----------RASST02 -          LINE 1 OF 3    >
CMD          NAME               CREATOR   DATABASE          N. ROWS
________ SYSSEQ2                SYSIBM    DSNDB06
________     SYSSEQUENCEAUTH    SYSIBM                        252
________     SYSSEQUENCESDEP    SYSIBM                        284
****************************** BOTTOM OF DATA *******************************
```


My personal opinion is we're getting prepared to provide an automatic method to audit who is granting/revoking which auth's from who.

If this is going to happen, there are a couple of schema changes needed since temporal tables have to reside in UTS tablespaces and violations do exist :

1) VCAT defined tablespaces not yet converted to PBG
2) Some tablespaces have more than one table.

System Time Temporal – future planning ?

- Another interesting topic is SYSVIEWDEP_H
  - This is not a table holding authorizations
  - Maybe due to TRANSFER OWNERSHIP ?
- Your guess as good as mine

When looking at the previous pages, there clearly seems to be a theme.

SYSVIEWDEP doesn't really fall into the category of holding AUTH's, so why does this table follow the same pattern ?

New Catalog Objects

- Dynamic Query Stabilization
  - An extension to what Db2 11 delivered for static queries.
  - Two tables and five LOBs.

```
RQTL   19.0   --------------- RC/Q Table List --------------- 2016/12/31 09:45
COMMAND ===>                                                  SCROLL ===> CSR

 Db2 Object ===> T                      Option  ===> L    Where => N
 Table Name ===> SYSDYN%               > Creator ===> SYSIBM                 >
  Qualifier ===> *                     > N/A     ===> *                      >
Loc: LOCAL ---------- SSID: D12A ----------RASST02 -          LINE 1 OF 7    >
CMD      TABLE NAME          CREATOR   DATABASE  TBLSPACE      NUMBER OF ROWS
________ SYSDYNQRY           SYSIBM    DSNDB06   SYSTSDQY                 176
________ SYSDYNQRYDEP        SYSIBM    DSNDB06   SYSTSDQD                 653
________ SYSDYNQRY_EXPL      SYSIBM    DSNDB06   SYSTSDQE                 232
________ SYSDYNQRY_OPL       SYSIBM    DSNDB06   SYSTSDQO                 232
________ SYSDYNQRY_SHTEL     SYSIBM    DSNDB06   SYSTSDQH                 232
________ SYSDYNQRY_SPAL      SYSIBM    DSNDB06   SYSTSDQS                  27
________ SYSDYNQRY_TXTL      SYSIBM    DSNDB06   SYSTSDQT                 232
****************************** BOTTOM OF DATA ********************************
```

21

Let's have a look at the NEW catalog tables which really can be used.

I have divided them into the Db2 12 themes to hopefully get a better overview of the new Db2 12 features.

Db2 11 offered the opportunity to lock down the access path for static packages/queries. Db2 12 has extended this to DYNAMIC SQL.

In order to support this – 5 new tables and 2 AUX objects are available.

New Catalog Objects

- SYSDYNQRY and SYSDYNQRYDEP indexes
  - Indexes seem to follow “normal standard”
  - Might consider reversing first two columns of DSNDQX04

TABLE_NAME	INDEX_NAME	INDEXED_COLUMN	UNQ
SYSDYNQRY			
	DSNDQX01		
		SDQ_STMT_ID	P
		COPYID	P
	DSNDQX02		
		CURSCHEMA	D
		QUERY_HASH	D
		COPYID	D
		RELBOUND	D
	DSNDQX11		
		STBLGRP	U
		SDQ_STMT_ID	U
		COPYID	U

TABLE_NAME	INDEX_NAME	INDEXED_COLUMN	UNQ
SYSDYNQRYDEP			
	DSNDQX03	SDQ_STMT_ID	D
		COPYID	D
	DSNDQX04		
		BQUALIFIER	D
		BNAME	D
		BTYPE	D
		SDQ_STMT_ID	D
		COPYID	D
	DSNDQX05		
		CLASS	D
		AUTHID	D
		AUTHID_TYPE	D
	DSNDQX12		
		CLASS	D
		BTYPE	D
		BAUTH	D
		AUTHID_TYPE	D
		AUTHID	D
		BQUALIFIER	D
		BNAME	D

22

The indexes created for these tables seem to match what’s needed if this feature will be heavily exploited.

The only modification which should be considered is to create one additional index reversing BQUALIFIER and BNAME depending on how much this will be used and how many distinct object qualifiers do exist (pretty much like CREATOR,NAME for SYSINDEXES, SYSTABLES etc.

New Catalog Objects

- IN-MEMORY indexes (aka FTB) – another autonomic feature with control.
  - Eligible for UNIQUE indexes4 where key size < 64K
  - DSNZPARM : INDEX_MEMORY_CONTROL
  - Control which indexes not to be considered / when
  - TS=SYSTSICO
  - TYPE could indicate future options (no check what's inserted – NO RI)

```
CREATE TABLE "SYSIBM".SYSINDEXCONTROL
       (SSID CHARACTER(4) FOR MIXED DATA WITH DEFAULT NULL
       ,"PARTITION" SMALLINT WITH DEFAULT NULL
       ,IXNAME VARCHAR(128) FOR MIXED DATA NOT NULL
       ,IXCREATOR VARCHAR(128) FOR MIXED DATA NOT NULL
       ,"TYPE" CHARACTER(1) FOR MIXED DATA NOT NULL WITH DEFAULT 'F'
       ,ACTION CHARACTER(1) FOR MIXED DATA NOT NULL WITH DEFAULT 'A'
       ,MONTH_WEEK CHARACTER(1) FOR MIXED DATA WITH DEFAULT NULL
       ,"MONTH" SMALLINT WITH DEFAULT NULL
       ,"DAY" SMALLINT  WITH DEFAULT NULL
       ,FROM_TIME TIME  WITH DEFAULT NULL
       ,TO_TIME TIME    WITH DEFAULT NULL
       ) IN DSNDB06.SYSTSICO
```

Force FTB creation
Disable FTB creation
Automatic FTB creation

23

Another new table is SYSINDEXCONTROL which is used to administer a new feature for in-memory indexes.

The look and feel is pretty similar to SYSINDEXCLEANUP in the nature that you can specify which indexes should take advantage or not of this FTB feature as well as when.

Looking at the TYPE column, this could indicate preparation for the future since only 'F' is current a valid option, so why need a type ?

New Catalog Objects

- SYSINDEXCONTROL has no indexes defined
- You might consider creating one depending on usage:
  - If you don't let Db2 decide when to use
  - If you want to highly customize which indexes to use FTB
  - If you want to be very specific about which weeks/days and timeframe

```
CREATE TABLE "SYSIBM".SYSINDEXCONTROL
       (SSID CHARACTER(4) FOR MIXED DATA WITH DEFAULT NULL
       ,"PARTITION" SMALLINT WITH DEFAULT NULL
       ,IXNAME VARCHAR(128) FOR MIXED DATA NOT NULL
       ,IXCREATOR VARCHAR(128) FOR MIXED DATA NOT NULL
       ,"TYPE" CHARACTER(1) FOR MIXED DATA NOT NULL WITH DEFAULT 'F'
       ,ACTION CHARACTER(1) FOR MIXED DATA NOT NULL WITH DEFAULT 'A'
       ,MONTH_WEEK CHARACTER(1) FOR MIXED DATA WITH DEFAULT NULL
       ,"MONTH" SMALLINT WITH DEFAULT NULL
       ,"DAY" SMALLINT  WITH DEFAULT NULL
       ,FROM_TIME TIME  WITH DEFAULT NULL
       ,TO_TIME TIME    WITH DEFAULT NULL
       ) IN DSNDB06.SYSTSICO
```


There are no indexes defined for SYSINDEXCONTROL, so if you plan on really controlling this feature instead of letting Db2 master the decision, you should consider a user-defined index based on usage.

New Catalog Objects

- Maintain session data on target server to reduce network traffic.
  - Three base tables and two AUX tables.
  - Supports transaction rerouting and pooling.
  - Distributed clients need session data (special registers and global variables) to be returned.

```
Db2 Object ===> T                      Option  ===> L    Where => N
 Table Name ===> SYSSESS%             > Creator ===> SYSIBM                >
  Qualifier ===> *                    > N/A     ===> *                     >
Loc: LOCAL ---------- SSID: D12A ----------RASST02 -          LINE 1 OF 5    >
CMD      TABLE NAME          CREATOR   DATABASE  TBLSPACE
________ SYSSESSION          SYSIBM    DSNDB06   SYSTSSES
________ SYSSESSION_DATA     SYSIBM    DSNDB06   SYSTSSXL
________ SYSSESSION_EX       SYSIBM    DSNDB06   SYSTSSNX
________ SYSSESSION_GV       SYSIBM    DSNDB06   SYSTSSNL
________ SYSSESSION_STATUS   SYSIBM    DSNDB06   SYSTSSTA
****************************** BOTTOM OF DATA ********************************
```

25

If you have applications where you need to reduce network traffic by maintaining session data, three base tables and two AUX objects are delivered to support this feature.

New Catalog Objects

- SYSSESSION indexes seem to be appropriate and no immediate need to create your own for performance:

TABLE_NAME	INDEX_NAME	INDEXED_COLUMN	COLSEQ	ORD	CLS
SYSSESSION					
	DSNSNX02				
		TOKEN	1	A	N
SYSSESSION_DATA					
	DSNSNX03				
		AUXID	1	A	N
		AUXVER	2	A	N
SYSSESSION_EX					
	DSNSNX04				
		TOKEN	1	A	N
	DSNSNX05				
		TOKEN	1	A	N
		GVID	2	A	N
		LOCATOR	3	A	N
SYSSESSION_GV					
	DSNSNX01				
		AUXID	1	A	N
		AUXVER	2	A	N
SYSSESSION_STATUS					
	DSNSNX06				
		TOKEN	1	A	N

The indexes delivered with SESSION DATA tables seem to be sufficient and I see no need to create your own indexes for performance reasons.

New Catalog Objects

- New indexes on existing tables:
  - DSNDCX06 on SYSCOLUMNS
    (TBCREATOR ASC, TBNAME ASC , COLNO ASC)
  - DSNOVX04 on SYSVARIABLES
    (TYPESCHEMA ASC, TYPENAME ASC)
  - DSNQYX04 on SYSQUERY
    (QUERY_SEC_HASH ASC, SCHEMA ASC, SOURCE ASC)
- Changed indexes:
  - ALTER INDEX DSNATX02 ADD COLUMN (UNLOADAUTH ASC)
    - SYSTABAUTH non-unique index
  - ALTER INDEX DSNOTX01 ADD COLUMN (VERSION ASC)
    - Unique Clustering for SYSTRIGGERS

27

Three new indexes are created on existing catalog tables and two existing indexes have an additional column appended.
SYSTABAUTH has a new column appended – UNLOADAUTH – which is another new Db2 12 feature.
SYSTRIGGERS has VERSION appended which also is a new column for a new feature.

Modified catalog objects

(Db2 12 <> Db2 11 compare)

Modified Objects

- Encoding scheme a new attribute on the column level

```
ALTER   TABLE SYSIBM.SYSCOLUMNS
        ADD ENCODING_SCHEME CHAR ( 1 )
        NOT NULL WITH DEFAULT FOR MIXED DATA ;

ALTER   TABLE SYSIBM.SYSCONTROLS
        ADD REGENERATETS TIMESTAMP ( 12 )
        NOT NULL WITH DEFAULT ;
```

- Row permissions and column masks can be REGENERATED via ALTER command and timestamp recorded *(alter command did exist in Db2 11 as well)*

29

Another new feature is that it is now possible to have the CCSID on the column level and not being dictated on the table level.

To support this, SYSCOLUMNS has a new column=ENCODING_SCHEME appended.

Even though Db2 11 had an ALTER command to REGENERATE row permissions and column masks, SYSCONTROLS has a new column added illustrating when this command was executed.

Modified Objects

- Two new columns for SYSENVIRONMENT – not described in the SQL Reference Guide “Catalog Tables”.

```
ALTER   TABLE SYSIBM.SYSENVIRONMENT
        ADD CREATEDTS TIMESTAMP ( 12 )
        NOT NULL WITH DEFAULT ;

ALTER   TABLE SYSIBM.SYSENVIRONMENT
        ADD APPLCOMPAT VARCHAR ( 10 )
        NOT NULL WITH DEFAULT 'V11R1' FOR MIXED DATA;
```


SYSENVIRONMENT has two new columns appended, but these are not described in appendix A in the SQL Reference Guide, so I am not sure whether they are used or not.

One new column is when the row was inserted and another column is APPLCOMPAT.

To me it seems like these two columns are useful however.

Modified Objects

- SYSINDEXES ready for partition wise attributes and new tablespace type *(Italic marked reserved for the future)*.

```
DSSIZE          INTEGER ;
PAGENUM         CHAR ( 1 ) NOT NULL WITH DEFAULT 'A';
PARTKEYCOLNUM   SMALLINT NOT NULL WITH DEFAULT;
STATUS          CHAR ( 1 ) NOT NULL WITH DEFAULT;
INDEXSTATUS     VARCHAR ( 30 ) NOT NULL WITH DEFAULT;
PARTITIONS      SMALLINT;
PQTY            INTEGER;
STORTYPE        CHAR ( 1 );
STORNAME        VARCHAR ( 128 );
VCATNAME        VARCHAR ( 24 );
FREEPAGE        SMALLINT;
PCTFREE         SMALLINT;
GBPCACHE        CHAR ( 1 );
SECQTYI         INTEGER;
ENFORCED_CONS   CHAR ( 1 ) NOT NULL WITH DEFAULT;
IMPLICIT        CHAR ( 1 );
REGENERATETS    TIMESTAMP ( 12 ) NOT NULL WITH DEFAULT;
```


A large number of new columns have been added to SYSINDEXES.

All the columns in the box marked in ITALIC are considered for future use.

These new columns are related to the new tablespace type (PBR2 or tablespaces with relative addressing) where partition level attributes will be available etc.

Modified Objects

- SYSINDEXPART also ready for relative page numbering.

```
DSSIZE            INTEGER;
PAGENUM           CHAR ( 1 ) NOT NULL WITH DEFAULT 'A';
LIMITKEY_EXTERNAL VARCHAR ( 765 ) NOT NULL WITH DEFAULT ;
```


The same goes for SYSINDEXPART which is ready for the new tablespace type PBR2.

Modified Objects

- Real Time Statistics tables (besides System Time Temporal columns)

```
ALTER   TABLE SYSIBM.SYSINDEXSPACESTATS
        ADD GETPAGES BIGINT WITH DEFAULT  0 ;

ALTER   TABLE SYSIBM.SYSTABLESPACESTATS
        ADD GETPAGES BIGINT WITH DEFAULT  0 ;
```


We already covered the Real Time Statistics changes where it's now possible to enable System Time Temporal.
Besides this change both RTS tables now also counts the GETPAGE activity.

Modified Objects

- SYSPACKAGE has some very nice pieces of information
  - How the BIND was performed (eg. Auto-Rebind)
  - If CONCENTRATE statement is enabled
  - When APREUSE was disabled
  - Db2 FUNCTION_LEVEL when the row was inserted

```
ORIGIN CHAR ( 1 ) NOT NULL WITH DEFAULT;
APREUSE_NO_FL     VARCHAR ( 10 ) NOT NULL WITH DEFAULT;
APREUSE_NO_TS     TIMESTAMP NOT NULL WITH DEFAULT;
CONC_STMT         CHAR ( 1 ) NOT NULL WITH DEFAULT 'N';
FUNCTION_LVL      VARCHAR ( 10 ) NOT NULL WITH DEFAULT;
```

- SYSPACKCOPY has the same columns appended.
- SYSQUERY has FUNCTION_LVL appended.

34

SYSPACKAGE has some new columns which seem extremely useful.

1) How the package got bound : REBIND, BIND, Automatic Rebind etc.
2) FUNCTION-LEVEL when APREUSE(NO) was executed as well as timestamp.
3) If CONCENTRATE_STATEMENT was specified for literals/constants.
4) Finally the Db2 FUNCTION-LEVEL when the row was inserted.

Modified Objects

- SYSPACKSTMT has new columns matching SYSQUERY

```
QUERYID     BIGINT NOT NULL WITH DEFAULT – 1 ;

QUERY_HASH  CHAR ( 16 ) NOT NULL WITH DEFAULT
        X'00000000000000000000000000000000' FOR BIT DATA;

QUERY_HASH_VERSION INTEGER NOT NULL WITH DEFAULT -1 ;
```

- SYSPENDINGDDL partition-range for logical parts to be reorganized instantiating pending changes

```
REORG_SCOPE_LOWPART SMALLINT ;

REORG_SCOPE_HIGHPART SMALLINT ;
```

35

SYSPACKSTMT has been appended with the same columns as SYSQUERY so it's possible to join these.

Since attributes can be altered for PBR2 partitions independently, SYSPENDINGDDL now keeps track of the highest and lowest partition with PENDING changes since adjacent logical partitions must be reorganized together to materialize pending definition changes.

Modified Objects

- SYSROUTINES describes if obfuscation is active

```
ALTER   TABLE SYSIBM.SYSROUTINES
        ADD WRAPPED CHAR ( 1 ) NOT NULL WITH DEFAULT ;

ALTER   TABLE SYSIBM.SYSROUTINES
        ADD REGENERATETS TIMESTAMP ( 12 )
        NOT NULL WITH DEFAULT ;
```

  - And yet another catalog table has REGENERATE timestamp appended.

- SYSTABLES also has REGENERATE timestamp.

Routines can be obfuscated and REGENERATED using ALTER so these are reflected as new columns.

Modified Objects

- SYSTABLEPART reflects the new tablespace type (absolute or relative page numbering)
- Partition level physical attributes reflected.

```
TYPE           CHAR ( 1 ) ;
PAGENUM        CHAR ( 1 ) NOT NULL WITH DEFAULT 'A' ;
BPOOL          CHAR ( 8 ) ;
PGSIZE         SMALLINT ;
DSSIZE         INTEGER ;
MEMBER_CLUSTER CHAR ( 1 );
COMPRESSRATIO  SMALLINT NOT NULL WITH DEFAULT - 1 ;
```

37

Due to the new tablespace type (PBR2 or relative page addressing), SYSTABLEPART has 7 new columns reflecting this feature along with the attributes which now can be different for the individual partitions.

Modified Objects

- SYSTABLESPACE.
  - Relative/absolute page numbering.
  - Interesting that this table now has inherited columns from SYSTABLEPART.

```
PAGENUM           CHAR ( 1 ) NOT NULL WITH DEFAULT 'A';
INSERTALG         SMALLINT NOT NULL WITH DEFAULT;
PQTY              INTEGER;
STORTYPE          CHAR ( 1 );
STORNAME          VARCHAR ( 128 );
VCATNAME          VARCHAR ( 24 );
FREEPAGE          SMALLINT;
PCTFREE           SMALLINT;
COMPRESS          CHAR ( 1 );
GBPCACHE          CHAR ( 1 );
TRACKMOD          CHAR ( 1 );
SECQTYI           INTEGER;
PCTFREE_UPD       SMALLINT;
PCTFREE_UPD_CALC SMALLINT;
COMPRESSRATIO     SMALLINT NOT NULL WITH DEFAULT - 1;
```


SYSTABLESPACE also reflects the new tablespace type.

What is interesting is that some attributes are inherited from SYSTABLEPART like PCTFREE, SECQTY etc.

Modified Objects

- SYSTRIGGERS
  - Versioning available (like NSP's)
  - Trigger text can be obfuscated

```
VERSION          VARCHAR ( 122 ) NOT NULL WITH DEFAULT ;
ORIGINAL_CONTOK  CHAR ( 8 ) FOR BIT DATA ;
REGENERATETS     TIMESTAMP NOT NULL WITH DEFAULT ;
ACTIVE           CHAR ( 1 ) FOR MIXED DATA ;
WRAPPED CHAR     ( 1 ) NOT NULL WITH DEFAULT ;
```


Triggers can now be versioned like Native Stored Procedures and they can also be obfuscated.

New Db2 12 features reflected in the catalog

(existing columns with additional value/meaning)

Existing columns with new content

- Compared to the past couple of Db2 releases – Db2 12 has far less changes.
- Only the most important covered here:

SYSCOLUMNS	PERIOD	C : Now EXCLUSIVE end period
		I : New INCLUSIVE end period
	ENCODING_SCHEME	Ascii , Ebcdic , Unicode
SYSCOPY	STYPE	C : Index Compression enabled (besides existing explanations)
	TTYPE	C : Index Compression enabled (besides existing explanations)
		CMP=Y/N : ALTER INDEX COMPRESS YES/NO
		When ICTYPE=A and STYPE=P : tablespace converted to RELATIVE
SYSDEPENDENCES	DTYPE	B is now BASIC Trigger
		1 is ADVANCED Trigger
SYSKEYCOLUSE	PERIOD	I/C : INCLUSIVE/EXCLUSIVE Business end period
SYSKEYS	PERIOD	I/C : INCLUSIVE/EXCLUSIVE Business end period
SYSPACKAGE	TYPE	1 : Advanced Trigger Package
SYSPACKDEP	DTYPE	1 : Advanced Trigger Package
SYSRELS	CHECKEXISTINGDATA	T : Immediately check existing data for a temporal referential constraint

41

Db2 12 seems to have less changes to existing columns compared to the past couple of releases. This section will only cover some of these.

PERIOD in syscolumns, syskeycoluse and syskeys : For temporal tables, ENDING can now be either inclusive or exclusive.

SYSDEPENDENCES has an indicator whether the trigger is the old style or it's an advanced trigger and the same goes for syspackage and syspackdep.

User Defined Functions available

- Some of these existed since several Db2 releases
  - Execute highlvl.SDSNSAMP(DSNTEJ2U)
  - Requires C / C++ compiler license
  - Use as samples to create you own – or modify
  - Here are just a few of the available UDF's

Function Name	Description
ALTDATE	Returns the current date or a user-specified date in a user-specified format
ALTTIME	Returns the current time or a user-specified time in a user-specified format
CURRENCY	Returns a floating-point number as a currency value
DAYNAME	Returns the name of the day of the week on which a date in ISO format falls
MONTHNAME	Returns the name of the month in which a date in ISO format falls
TABLE_LOCATION	Returns the location name of a table or view after resolving any aliases
TABLE_NAME	Returns the unqualified name of a table or view after resolving any aliases
TABLE_SCHEMA	Returns the schema name of a table or view after resolving any aliases
WEATHER	Shows how to use a user-defined table function to make non-relational data available for SQL manipulation

UDF's are not widely used as far as I have experienced, but I stumbled across this section while preparing this presentation.

Most of these routines are not new, but I am positive some of the attendees will find these useful.

Source code do exist for all of these (only a fraction listed here) so you can easily pick up what's needed and then change to do whatever you need.

User Defined Functions available

- In member DSNTEJ2U – find the UDF needed and look for EXTERNAL NAME
- This is the example for ALTDATE using member DSN8DUAD

```
/*********************************************************************
* Module name = DSN8DUAD (DB2 sample program)                        *
*                                                                    *
* DESCRIPTIVE NAME = Current date reformatter (UDF)                  *
*                                                                    *
*                                                                    *
*     LICENSED MATERIALS - PROPERTY OF IBM                           *
*     5625-DB2                                                       *
*     (C) COPYRIGHT 1998, 2003 IBM CORP.  ALL RIGHTS RESERVED.       *
*                                                                    *
*     STATUS = VERSION 8                                             *
*                                                                    *
*                                                                    *
* Function: Returns the current date in one these 34 formats:        *
*                                                                    *
*              D MONTH YY   D MONTH YYYY   DD MONTH YY   DD MONTH YYYY *
*                  D.M.YY   D.M.YYYY       DD.MM.YY      DD.MM.YYYY  *
*                  D-M-YY   D-M-YYYY       DD-MM-YY      DD-MM-YYYY  *
*                  D/M/YY   D/M/YYYY       DD/MM/YY      DD/MM/YYYY  *
*                  M/D/YY   M/D/YYYY       MM/DD/YY      MM/DD/YYYY  *
*                  YY/M/D   YYYY/M/D       YY/MM/DD      YYYY/MM/DD  *
*                  YY.M.D   YYYY.M.D       YY.MM.DD      YYYY.MM.DD  *
*                           YYYY-M-D                     YYYY-MM-DD  *
*                           YYYY-D-XX                    YYYY-DD-XX  *
*                           YYYY-XX-D                    YYYY-XX-DD  *
*                                                                    *
*           where:                                                   *
*                                                                    *
*             D: Suppress leading zero if the day is less than 10    *
*            DD: Retain leading zero if the day is less than 10      *
*             M: Suppress leading zero if the month is less than 10  *
*            MM: Retain leading zero if the month is less than 10    *
*         MONTH: Use English-language name of month                  *
*            XX: Use a capital Roman numeral for month               *
*            XX: Use a capital Roman numeral for month               *
*            YY: Use non-century year format                         *
*          YYYY: Use century year format                             *
```


Have a look at highlvl.SDSNSAMP (DSNTEJ2U) and find the UDF you can take advantage from and look for EXTERNAL NAME.
This is a snippet from the ALTDATE routine which can automate the conversion of a date into 30+ formats.

References used

- A live Db2 12 system
- Db2 12 for z/OS SQL Reference Guide SC27-8859-00
- White Paper : Db2 12 for z/OS - an IDUG Perspective
  http://www.idug.org/p/bl/ar/blogaid=527
  (you have to be logged in to the IDUG.ORG website)

44

This presentation was prepared using primarily a Db2 12 system FUNCTION LEVEL V12R1M500.
I did take a peek into the IBM Db2 SQL Reference Guide to validate new columns and column content.

Also, please take a look at the white paper created late 2016 by a lot of famous Db2'ers around the world.

Wrap Up

Thank You - QUESTIONS ?