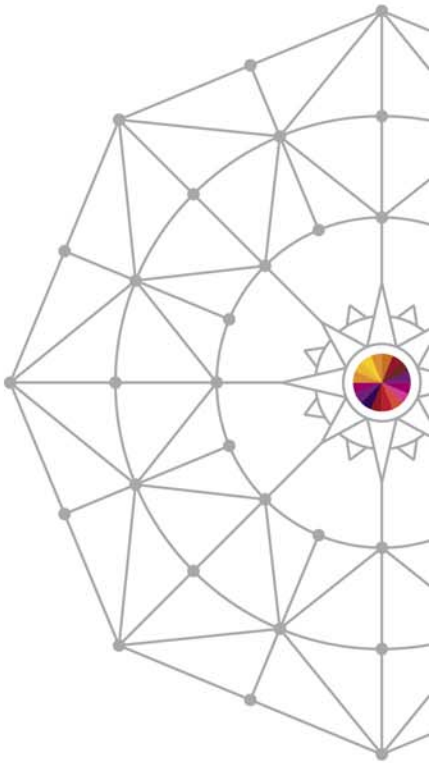
Ten Breakthroughs That Changed DB2 Forever

Craig S. Mullins
Mullins Consulting, Inc.
www.mullinsconsulting.com

August 4, 2014
Session Number 15468



www.SHARE.org

#SHAREorg

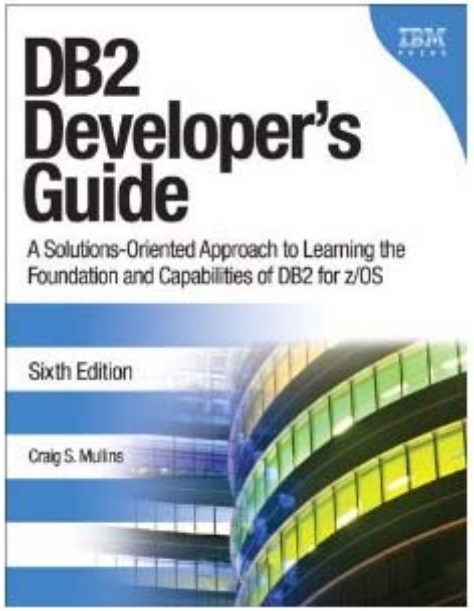
SHARE in Pittsburgh 2014

Author

This presentation was prepared by:

Craig S. Mullins
President & Principal Consultant

Mullins Consulting, Inc.
15 Coventry Ct
Sugar Land, TX 77479
Tel: 281-494-6153
Web: www.mullinsconsulting.com
E-mail: craig@craigsmullins.com



Objectives

- Gain an historical perspective of the features and functionality of DB2.
- We will count down and explain the ten most important technological breakthroughs made to DB2 for z/OS over the years since the introduction of V1 in the early 1980s.
  - Each breakthrough will be covered in chronological order and introduced by the version of DB2 where it was delivered.
  - Each feature, and its significance will be explained along with examples and what was done prior to its introduction.
  - Coverage of several breakthroughs that did not make the top ten will be offered, along with brief explanations of why not, and how things could change based on adoption and usage.
- A discussion of what may come next in terms of future DB2 breakthroughs will be offered at the end.

The History of DB2 for z/OS

Version	GA	EoM	EoS
1.1	1985-04-02		
1.2	1986-03-07		
1.3	1987-06-26		
2.1	1988-09-23		
2.2	1989-09-22		
2.3	1991-10-25		
3	1993-11-16	1999-11-30	
4	1995-10-10	2000-12-01	
5	1997-06-27	2001-12-31	2002-12-31
6	1998-06-30	2002-06-30	2005-06-30
7	2001-03-30	2007-03-05	2008-03-30
8	2004-03-26	2009-09-08	2012-04-30
9	2007-03-06	2012-12-10	2014-06-27
10	2010-10-22	2015-07-06	
11	2013-10-25		

Version vs. Release

- IBM has not delivered a “release” since 1991 with DB2 Version 2 Release 3.

- What is the difference between a version and a release?
  - A new ***version*** of software is a major concern, with many changes and new features.
  - A ***release*** is typically minor, with fewer changes and not as many new features.

In the early days of DB2…

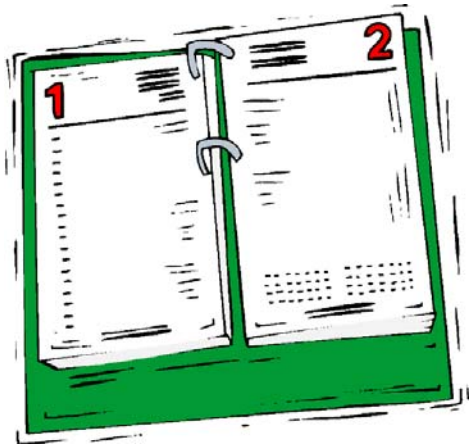
- Throughout the Version 1 releases of DB2 it was considered an “information center” product only.

So, What Features Contributed to DB2's Status Today as…

- A high-performance, feature-laden transaction processing DBMS that also...
- Offers robust data warehousing and analytical capabilities while…
- Providing high availability with...
- On demand administrative and management capabilities?

DATE and TIME Data Types
DB2 V1.3

- Prior to DB2 V1.3, there was no DATE and TIME data type support
- Most designs used the CHAR data type for date/time data
- Drawbacks:
  - No validation of the data
  - No ability to perform date/time arithmetic
  - In general, bad data quality



DATE/TIME Arithmetic

- **DB2 can add and subtract DATE, TIME, and TIMESTAMP values and columns**
  - **and DATE, TIME, and TIMESTAMP durations**
- **Guidelines**:
  - Let DB2 do the hard work for you.
  - Use the proper DB2 data types.
  - Understand durations *(see next slide).*
  - Know your DB2 functions.

Understanding Durations

- **Labeled Durations** – YEAR(S), MONTH(S), DAY(S), HOUR(S), MINUTE(S), SECOND(S), MICROSECOND(S)
  - Example(s): 10 DAYS 2 YEARS 33 MINUTES 1 SECOND
- **Date Durations –yyyymmdd** DECIMAL(8,0)
  - Example: 00201104 **(20 years, 11 months, and 4 days)**
- **Time Durations – hhmmsss** DECIMAL(6,0)
  - Example: 081144 **(8 hours, 11 minutes, and 44 seconds)**
- **TIMESTAMP Durations – yyyyxxddhhmmsszzzzzz**
  - DECIMAL(20,6)
  - Example: 002011040811 44.004351
    (20 years, 11 months, 4 days, 8 hours, 11 minutes, 44 seconds, & 4351 microseconds)

DATE/TIME vs. TIMESTAMP

DATE / TIME

- Requires 2 columns.
- Saves storage: only 7 total bytes required.
- Less precise: seconds.
- DB2 provides formatting options for DATE and TIME (not TS).

TIMESTAMP

- Everything in 1 column.
- Requires 10 bytes of storage.
- More precise: microseconds.
- DATE arithmetic easier using 1 column

DB2 9: New Built-in Timestamp Functions

- TIMESTAMPADD - adds an interval to a timestamp.
- TIMESTAMPDIFF - subtracts two timestamps & returns an interval.
- TIMESTAMP_FORMAT – changes the display format for a timestamp value. Valid formats that can be specified are:
  - 'YYYY-MM-DD'
  - 'YYYY-MM-DD-HH24-MI-SS'
  - 'YYYY-MM-DD-HH24-MI-SS-NNNNNN'

DB2 10: More Flexible Timestamps

- Can now specify a precision to indicate the amount of fractional seconds (default is 6, compatible with old TIMESTAMPs)
- Can optionally specify TIMESTAMP WITH TIME ZONE to add time zone information to your data.

Referential Integrity
DB2 V2.1

- The addition of RI in DB2 V2.1 was a boon to data integrity

CUST_NO SMALLINT	CUST_NAME CHAR(30)	CUST_ADDR CHAR(25)	CUST_CITY CHAR(20)	CUST_STATE CHAR(2)	CUST_ZIP INT	CUST_AREA_CODE SMALLINT	CUST_PHONE INT	CREDIT_CAT CHAR(3)	SALES_REP INT	COLLECT_AGY_NO SMALLINT
1	ACME INC	222 ELM ST	LISLE	IL	60532	708	5551212	AAA	123456	
2	PXI CORP	5 PXI PLAZA	DALLAS	TX	76543	407	8326745	AAA	273818	5
3	SYNC CORP	2254 HAMILTON DR	FT.WASHINGTON	PA	19003	215	8983736	C	583490	
4	FR STANLEY INC	987 BEAVER DR	ANNAPOLIS	MD	30589	301	2734147	AA	123456	
5	BUYLO INC	235 IRON ST	BLOOMSBURG	PA	17815	717	9895280	BB	903757	
7	WHOS AIRCRAFT	8837 DESERT RD	TUCSON	AZ	80345	602	6743333	A	583490	
8	BUSINESS INFO CO	555 WATERSEDGE	LOMBARD	IL	60406	708	4836285	AAA	273818	2
9	CPU INC	1123 PEACH ST	COLUMBUS	OH	50387	216	4823778	C	999475	
10	XYZ CORP	4590 WAYNE RD	LIVONIA	MI	46827	313	3435555	B	999475	

FOREIGN KEY

ASSOCIATION INTEGRITY
Each city/state pair has a valid zip code

DOMAIN INTEGRITY
Each value of CREDIT_CAT is valid

SALES_ORDR_TAB

SALES_ORDR_NO INT	ORDR_DATE DATE	CUST_NO SMALLINT	SALES_HIST_CUST_NO SMALLINT	ORDR_AMT DEC(9,2)
1	1991-09-15		2	1923.45
3	1991-09-23		1	2407.53
4	1991-09-23		10	57613.89
5	1991-09-29	3	5	67000.00
6	1991-10-01	2		42345.88
7	1991-10-02	2		122345.61
8	1991-10-02	7		23007.34
9	1991-10-05	1		9823.55
10	1991-10-07	5		223019.27
11	1991-10-11			78780.99

DEPENDENT TABLE

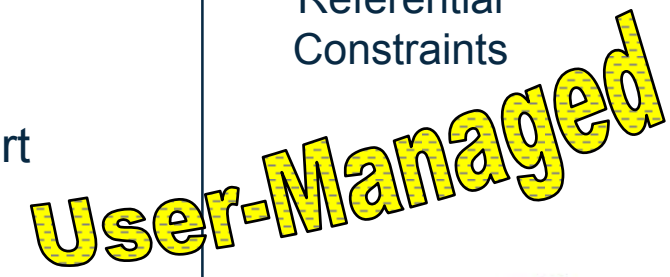
REFERENTIAL INTEGRITY
Each value of CUST_NO exists as a value of CUST_NO in CUST_TAB

RI: System or User-Managed?

- Standard declarative implementation.
- Less coding required.
- Easier to modify later. (DDL and CHECK)
- More efficient.
- Ad hoc and planned updates.

- Requires program code to be written.
- Hard to modify later.
- Sometimes there is the possibility for better insert performance.
- Works only for planned updates.

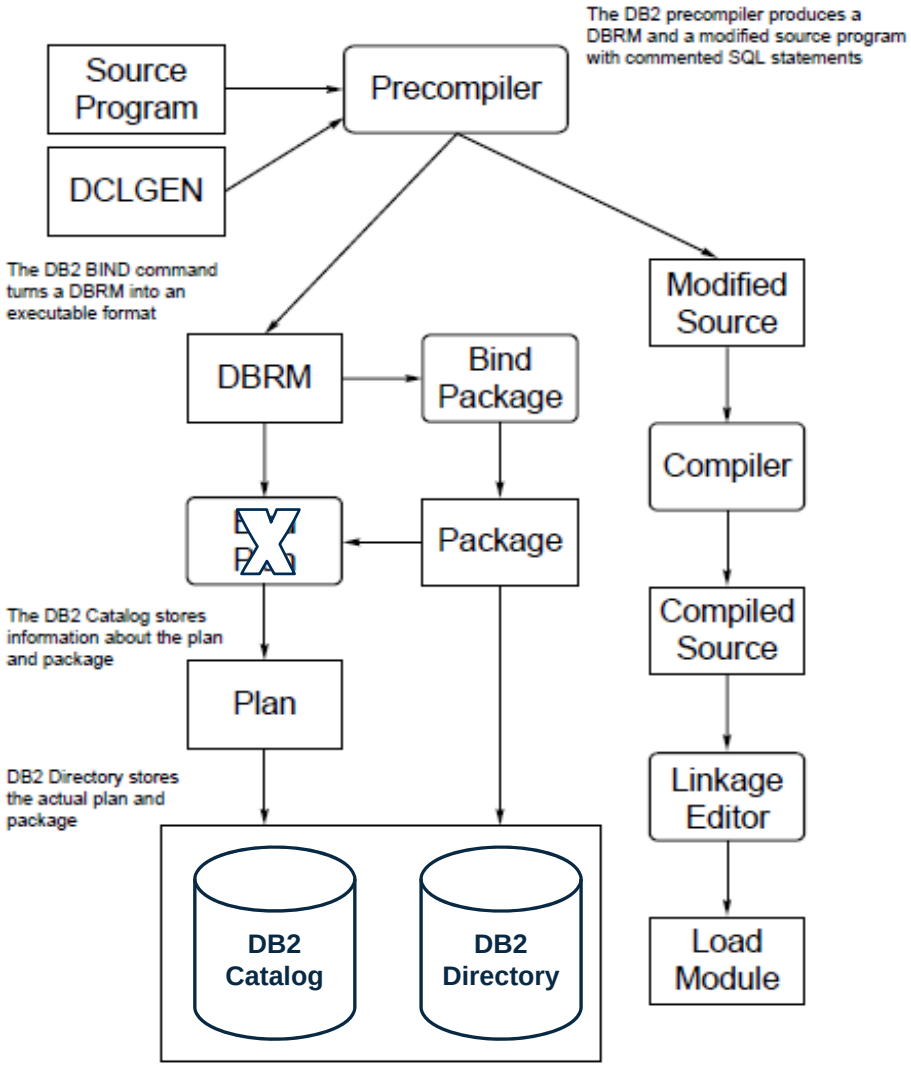
DB2 V8:
Informational Referential Constraints



Packages
DB2 V2.3

- Some newer DB2 programmers may not realize that packages were not *always* part of DB2
  - Packages simplify application development.
- What is package?
  - A single, bound program with optimized access paths.
  - Prior to packages, programs were bound at the plan level.
  - A plan could consist of multiple DBRMs.
  - But what if you only needed to rebind one program?
  - The only way to do it without packages was to rebind the entire plan (including every program).
    - If a plan consists of hundreds (or even thousands) of programs, the rebind can take a long time. And that translates into downtime for the applications using the plan.

Program Preparation

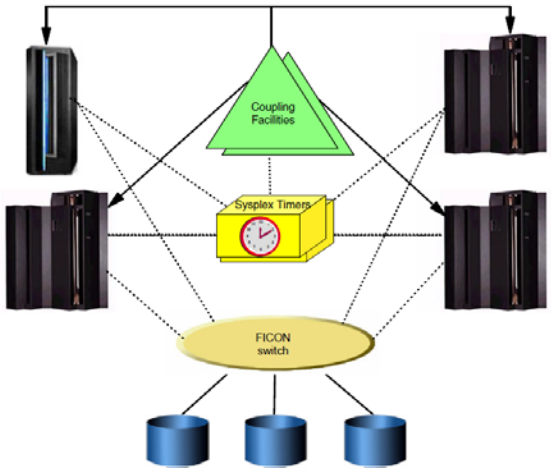


Package Benefits

- Simplified program management
  - BIND and REBIND can be done at the program level instead of the plan level
- Granularity of BIND parameters
  - For example, mulitple packages, each with different ISOLATION or RELEASE specifications, can be bound to single plan
- Versioning
  - Multiple versions of a package can exist

Data Sharing
DB2 V4

- Data Sharing is perhaps the single biggest improvement in DB2's three decade life span
  - Runs in a z/OS Parallel Sysplex
  - Allows concurrent read/write data access from multiple DB2s
  - Data must reside on shared devices
    - Single DB2 Catalog/Directory
  - Application can run on any member DB2



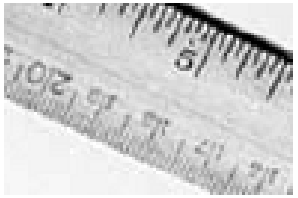
The Importance of Data Sharing

- The Critical Mass
  - Prior to Data Sharing, everything that required access to DB2 had to reside on the same CPU
    - DB2, TSO, IMS/TM, CICS…
  - When errors occurred you could not move subsystems independently to another LPAR without losing access to DB2
  - A large shop could quickly exhaust machine resources this way

Benefits of Data Sharing

- No special programming required
- Improved data availability
- Extended processing capacity
- Flexible system configuration
- Improved transaction rates

Type 2 Indexes
DB2 V4



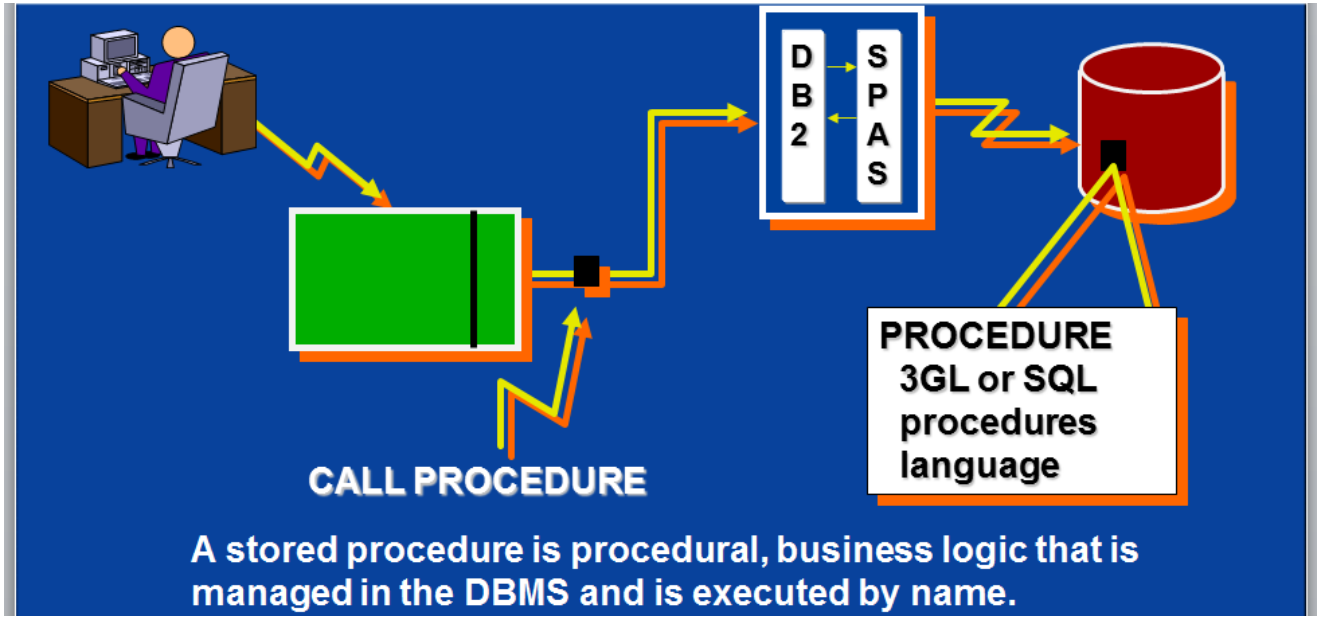
- Version 4 was a very significant release because it also brought Type 2 indexes
- The old indexes (Type 1) required locking, whereas Type 2 indexes do not
  - With Type 2 indexes, a lock on the data page or row acts as a lock on the index key
- Index locking was one of the most significant barriers to DB2 performance

Type 1 vs. Type 2 Index

- Type 1 indexes supported subpages
  - A way to break up a 4K page into smaller units
  - Because index rows are smaller than table rows locking a full 4K index page could cause significant bottlenecks
    - 1, 2, 4, 8, or 16 subpages
- Type 2 indexes do not require subpages
  - No locking

Stored Procedures
DB2 V4

- DB2 was late to the game with stored procedure support
  - But starting with V4, and with significant improvements over subsequent releases DB2 offers stellar stored procedure support today



Stored Procedure Benefits

- Reduce network traffic for distributed applications
- Consistent behavior
  - When any application calls the stored procedure, it processes data in a consistent way
- Ease of maintenance
  - If you need to change the rules, you only need to make the change once in the stored procedure, not in every application that calls the stored procedure.
- Flexible development choices
  - Can be coded in C, C++, COBOL, Assembler, PL/I, REXX, SQL Procedures language and Java.
- Can access non-DB2 resources
  - VSAM files, flat files, IMS or CICS transactions, DL/I databases, MVS/APPC conversations

Stored Procedures – Through The Years

- Originally coded using traditional programming language only and executed via SPAS
  - WLM replaced SPAS in V5
- Could not return result sets until V5
- Originally had to be manually registered to DB2 Catalog
  - CREATE PROCEDURE added in V6
- SQL stored procedures - supported as of V6
  - But SQL procedures were converted to C programs
  - Native SQL stored procedures
    - Supported as of DB2 9 for z/OS
    - No external load module is created
    - Run within the DB2 engine (DBM1 address space)
      - *No WLM/SPAS required*
    - zIIP eligible when called via distributed connection

Stored Procedure Usage Guidelines

- Keep it simple - make each procedure do one thing only
- Use stored procedures to create reusable “components” of business logic
- Putting multiple types of business logic into a single stored procedure makes it more difficult to tune, modify, and understand
- Document the purpose of each procedure as well as any required input, outputs, and each change made to the stored procedure

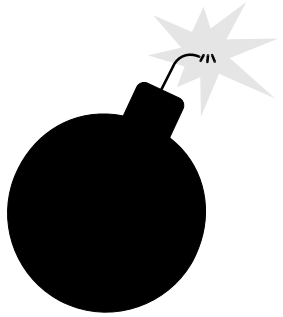
SP Languages: Quick Analysis

	Prgming Strength	Thruput / Perf	Billable Cost	Runs in DB2 Engine	Specialty Processor Support
COBOL	High	High	Low	No	No zAAP Some zIIP
C/C++	High	Better than COBOL	Lower than COBOL	No	No zAAP Some zIIP
SQLJ	High	Below native SQL	Higher than COBOL	No	zAAP Some zIIP
JDBC	High	Below External SQL	Higher than External SQL	No	zAAP Some zIIP
Enternal SQL	Low	Below SQLJ	Higher than SQLJ	No	No zAAP Some zIIP
Native SQL	Low	Below COBOL	Lowest	Yes	Significant zIIP

Triggers
DB2 V6

- Execution is automatic and implicit
- Triggers implement “active” database systems
- Trigger is executed, or “fired,” based upon a pre-defined “firing” activity:
  - Database Modification (INSERT, UPDATE, DELETE)
- Trigger uses include:
  - Perform subsequent modification
  - Implement business rules
  - Maintain redundant data
  - Maintain derived data
  - Validate data
- Like with stored procedures, DB2 was late to the game with trigger support, too

Trigger Impact on DBAs

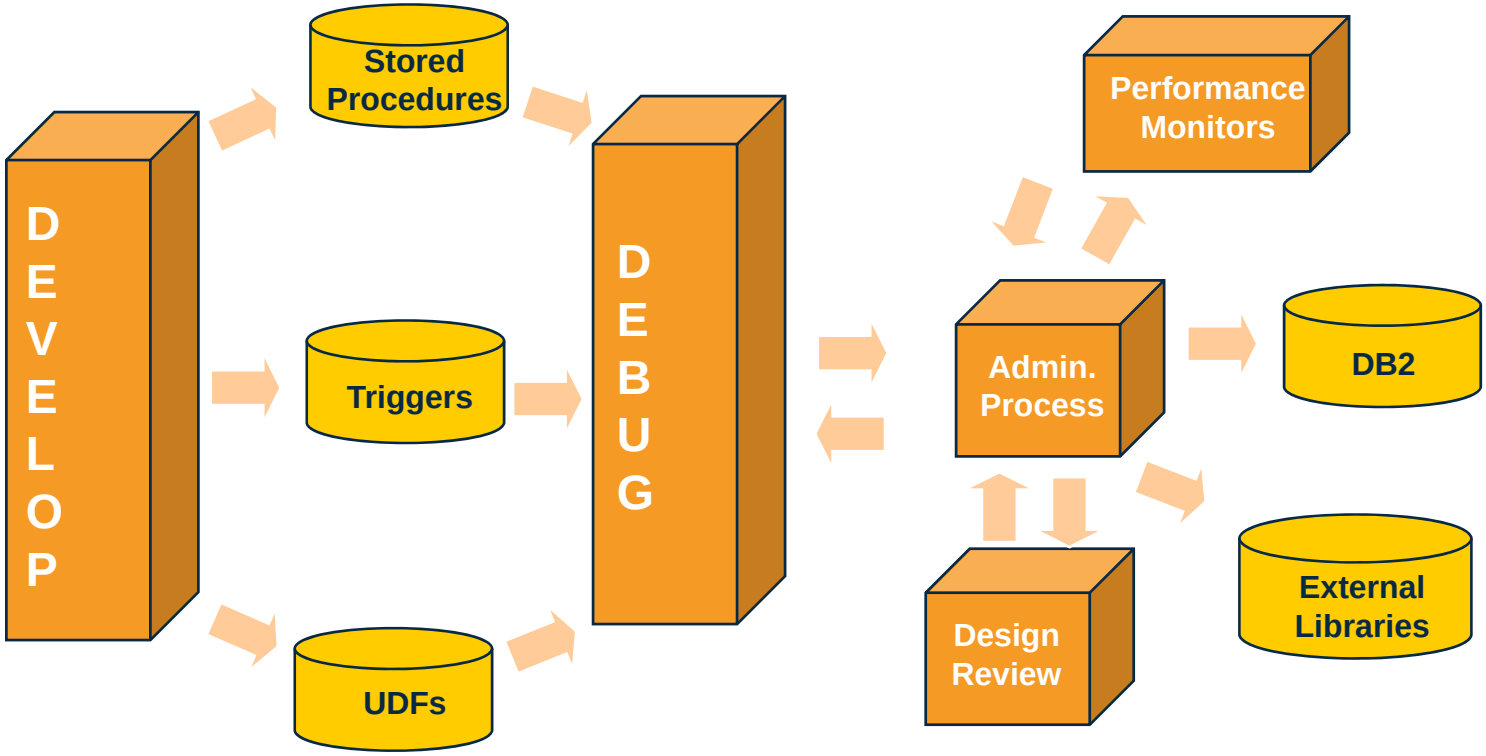


- Performance Implications
  - Monitoring and Tuning
  - Statement vs. “Row-at-a-time” Behavior
    - Statement triggers fire even if no rows are impacted.
    - Not so with ROW triggers; only if rows are impacted.
- Cascading triggers
  - Can change a lot of data with one SQL modification
- Trigger execution order (when multiple triggers exist)
  - Of the same type on the same table. . .
  - Oldest to newest - be careful
- Trigger Access Paths
  - Must REBIND TRIGGER PACKAGE … EXPLAIN YES

Additional Trigger Impacts and Issues

- Utilities do not activate triggers
  - `LOAD` can be troublesome
- Trigger Dependencies:
  - If you `DROP` any object used by the trigger the trigger package will become invalid, and the trigger will not run
- When data already exists?
  - If you add a trigger to a table that already contains data, the trigger will not be fired for the existing data (data integrity problems may exist unless they are addressed by other means)
- SQL termination character
  - DSNTEP2: `SET TERMINATOR`
  - SPUFI Defaults: Option #1 SQL Termination

Procedural DBA Requirements



Role of the Procedural DBA

DBCO = Database Code Object
shorthand for trigger, UDF, stored procedure

DBCO Implementation
(COMMIT in proc, write or guide)

Ensuring Reuse

Design Reviews

DBCO Administration
(trigger firing order, proc set)

EXPLAIN Analysis

Coding Complex Queries

On Call for DBCO Abends

Tuning SQL

Debugging SQL

Schema Resolution

INSTEAD OF Triggers

- DB2 V9 added a new type of trigger: INSTEAD OF triggers
  - INSTEAD OF triggers can only be defined on VIEWs.
  - INSTEAD OF triggers enable views that would not otherwise be modifiable to support INSERTs, UPDATEs, and DELETEs.
    - Typically, a view that consists of multiple base tables cannot be updated.
  - With an INSTEAD OF trigger you can code logic to direct inserts, updates and deletes to the appropriate underlying tables that comprise the view.
    - Each requested modification made against the view is replaced by the trigger logic. The trigger performs the insert, update, or delete on behalf of the view.

http://www.ibm.com/developerworks/db2/library/techarticle/0210rielau/0210rielau.html

Multiple Buffer Pools (& other BP improvements) DB2 V3 thru V6

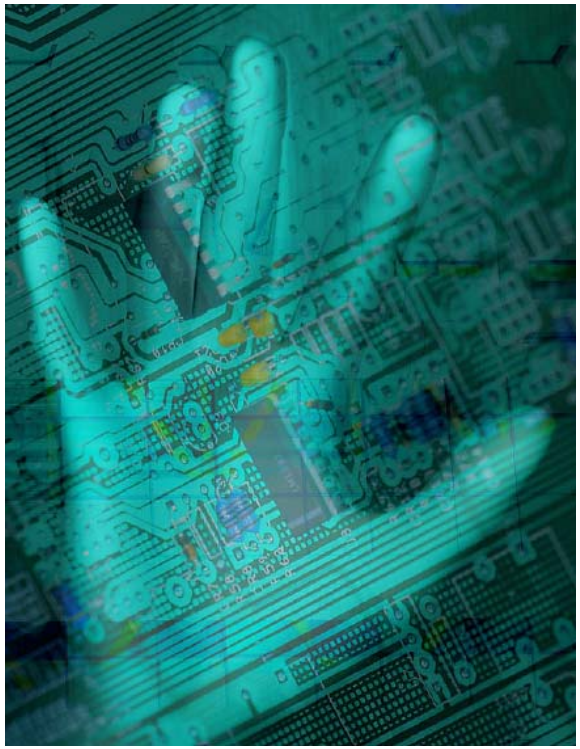
- In the earliest days of DB2, the “best practice” advice was to lump ***everything*** into BP0 and “let DB2 manage it”
- This might have been reasonable in the days when only 5 buffer pools were available
  - BP0, BP1, BP2, BP3, and BP32K

Use Multiple Buffer Pools

- IBM provides 80 buffer pools for a reason
  - 4K: BP0 thru BP49
  - 8K: BP8K0 thru BP8K9
  - 16K: BP16K0 thru BP16K9
  - 32K: BP32K thru BP32K9
- Using them effectively optimizes performance
- Ideas:
  - isolate the catalog in BP0
  - separate indexes from table spaces
  - isolate heavily hit data
  - isolate sort work area
  - optimize BP strategy for your data & app processing mix: sequential vs. random
  - there is no “silver bullet” approach

DB2 10 Buffer Pool Enhancements

- Taking advantage of System z and z/OS improvements
- Improved memory allocation
  - Memory is allocated as data is brought into the buffer pool instead of at DB2 startup
- New System z10 1 Megabyte page size
  - 1MB page size enables DB2 to manage larger buffer pools better



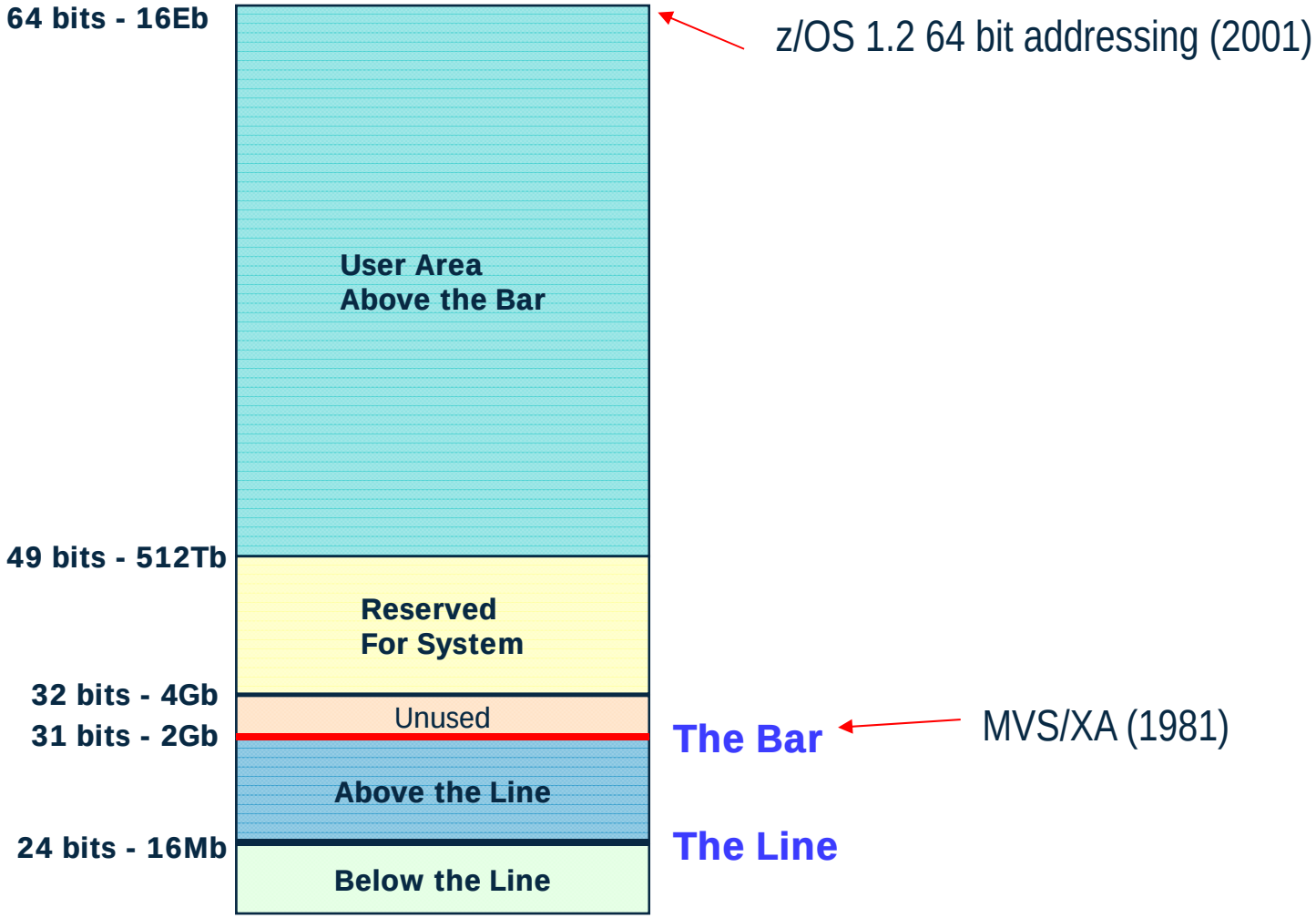
Expanding Architectural Limits DB2 V8

- The number 9 DB2 breakthrough is a bit more nebulous than the previous 8 we've walked through, but it is just as important, if not moreso
- DB2 V8 was re-architected to support 64 bit
  - Required new hardware, new OS, and Unicode (in DB2 Catalog)
  - With 64-bit addressing DB2 can theoretically support up to 16 exabytes
    - 1 EB = 1,000,000,000,000,000,000B = 10^{18} bytes = 1,000,000,000 gigabytes = 1,000,000 terabytes
- Other architectural improvements in DB2 V8 included:
  - Long names (up to 128 bytes)
    - Columns 30 bytes, table spaces 8 bytes, packages 8 bytes (except trigger packages 128 bytes).
  - DB2 Catalog table spaces with larger page sizes (>4K)

The EDM Pool and V8, V9

- V8: EDM Pool split into three specific pools:
  - **Below the 2GB Bar**
    - **EDMPOOL: EDM Pool stores only CTs, PTs, SKCTs, SKPTs**
      - ***Should be able to reduce the size of this EDM pool***
      - ***Provide some VSCR for below the 2GB Bar storage***
  - **Above the 2GB Bar**
    - **EDMDBDC: DBDs**
    - **EDMSTMTC: Cached Dynamic Statements**
- V9: Introduced additional changes
  - **Above the 2GB Bar: EDM_SKELETON_POOL**
    - All SKCTs and SKPTs
  - **A portion of the CT and PT is moved above the bar, too**

Of Mainframe Storage: Lines, Bars, and Beams



Database Definition on Demand

Online Schema Change

DB2 V8 thru 10

SHARE
Educate · Network · Influence

- Modifying database structures after they have been created had always been somewhat troubling before online schema evolution in DB2 V8
  - Many types of common changes were not supported by ALTER
  - Required dropping and re-creating the object
    - As well as all related objects, security, etc. that cascaded DROP
  - Resulted in significant outages to make database changes
- Renamed Database Definition on Demand in DB2 V9
  - Each new version of DB2 adds more DDOD capabilities

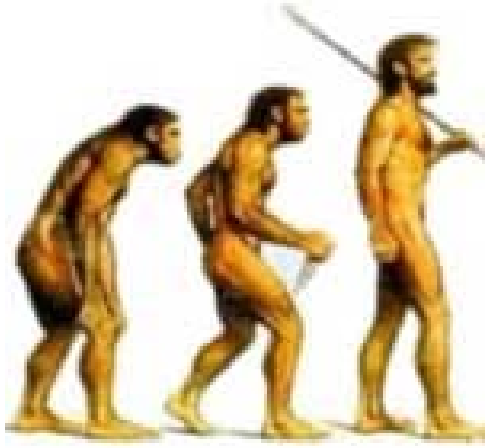
SHARE in Pittsburgh 2014

Why This is a Breakthrough: An Example

- Steps to change CHAR(10) column to CHAR(15) (pre V8)
  1. Unload data and extract DDL and authorizations
  2. Drop the table
  3. Modify the DDL to change the column to CHAR(15)
  4. Re-create the table
  5. Re-create all dependent objects
  6. Rebuild authorization for the table
  7. Rebuild authorizations for all dependent objects
  8. Reload the data (using the new 15 byte column)
  9. Run utilities (RUNSTATS and COPY)
  10. REBIND all impacted programs
  11. Test it all to make sure it worked

Online Schema Evolution
DB2 V8

- ALTER can extend the length of a CHAR column to a greater size
- Can change data type within family
  - CHAR → CHAR, numeric → numeric, date/time → date/time
- Can add a column to an index and change padding
- Can change the clustering index
- Partitioning changes
  - Add partitions
  - Rotate partitions
  - Change partition boundaries
  - Rebalance partitions



Database Definition on Demand Post V8

- IBM continues to advance the ability to make changes to DB2 databases without an outage
  - DB2 9
    - Elimination of REORG BUILD2 phase
    - Cloning tables
    - Rename a column within a table and rename indexes
  - DB2 10
    - ALTER a table space's page size, data set size, segment size, and/or MEMBER CLUSTER structure
    - ALTER a simple or segmented table space with only one table to a partition-by-growth Universal table space
    - ALTER a classic partitioned table space to a partition-by-growth or range-partitioned Universal table space
    - ALTER an index's page size
  - DB2 11
    - PiT recoveries can be done prior to a materializing REORG – *more flexibility*
    - DROP COLUMN support
    - Improved availability when altering LIMITKEYs
    - Allow BIND, REBIND and DDL to break-in persistent threads

Other Important Breakthroughs?

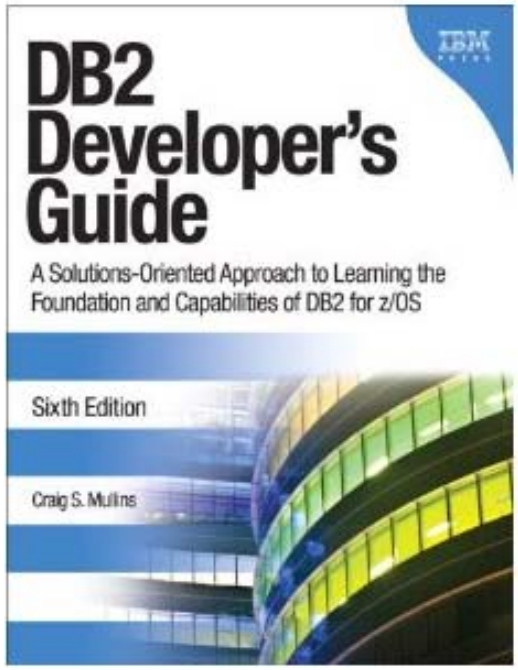
- Temporal – DB2 10
- pureXML – DB2 9
- Real Time Statistics – DB2 V7
- LOB support – DB2 V6
- Parallelism
  - I/O parallelism V3
  - CP parallelism V4
  - Sysplex parallelism V5
- DB2 LUW (UDB)
  - OS/2 EE DBM

What Lies Ahead?

- More changes, more availability, more usability
- But what will DB2 V30 look like in the year 2041?*

* There is no guarantee that Version 30 will be out before the end of 2041 and mention of it in this presentation should not indicate an actual future version or release of DB2

Thank You



Craig S. Mullins
Mullins Consulting, Inc.
15 Coventry Court
Sugar Land, TX 77479

http://www.mullinsconsulting.com

← 6th edition available now…
covering up thru DB2 V10!