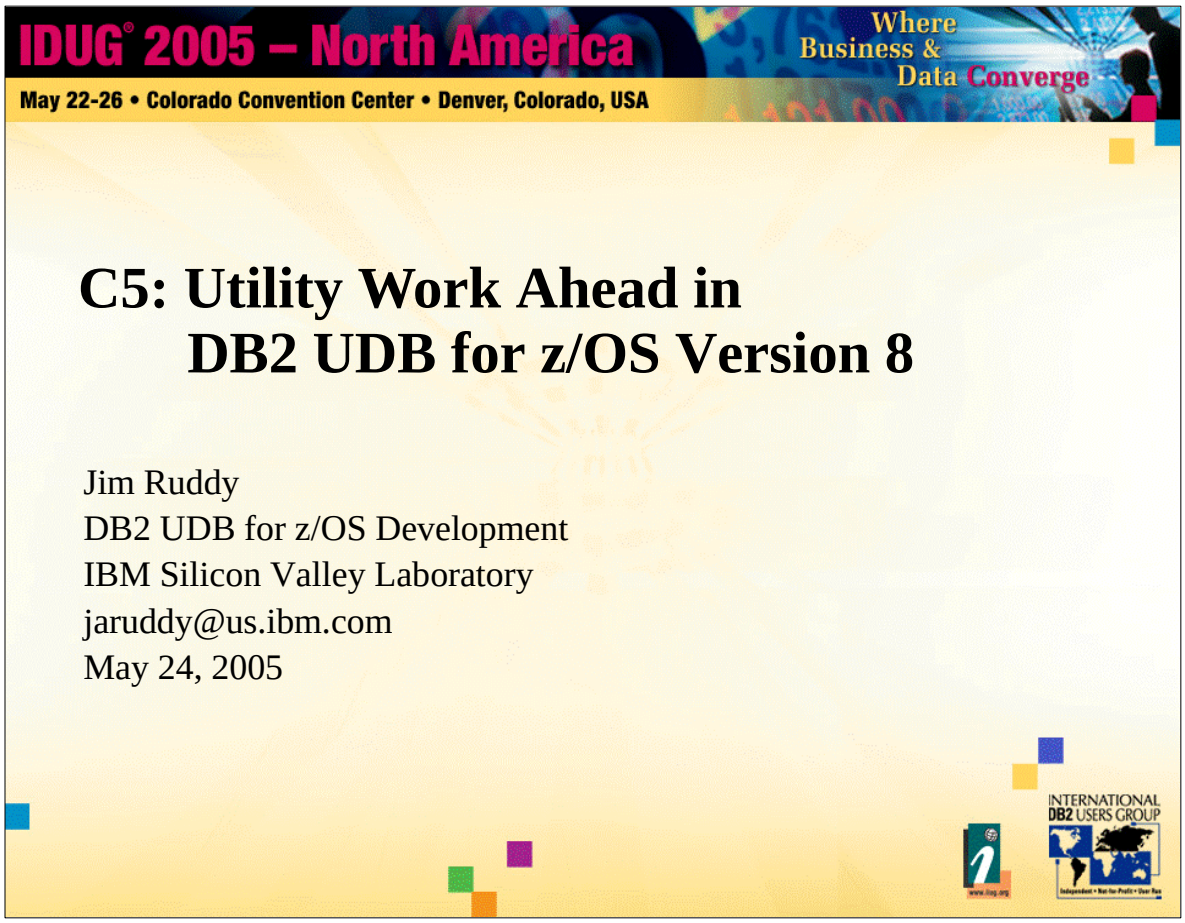
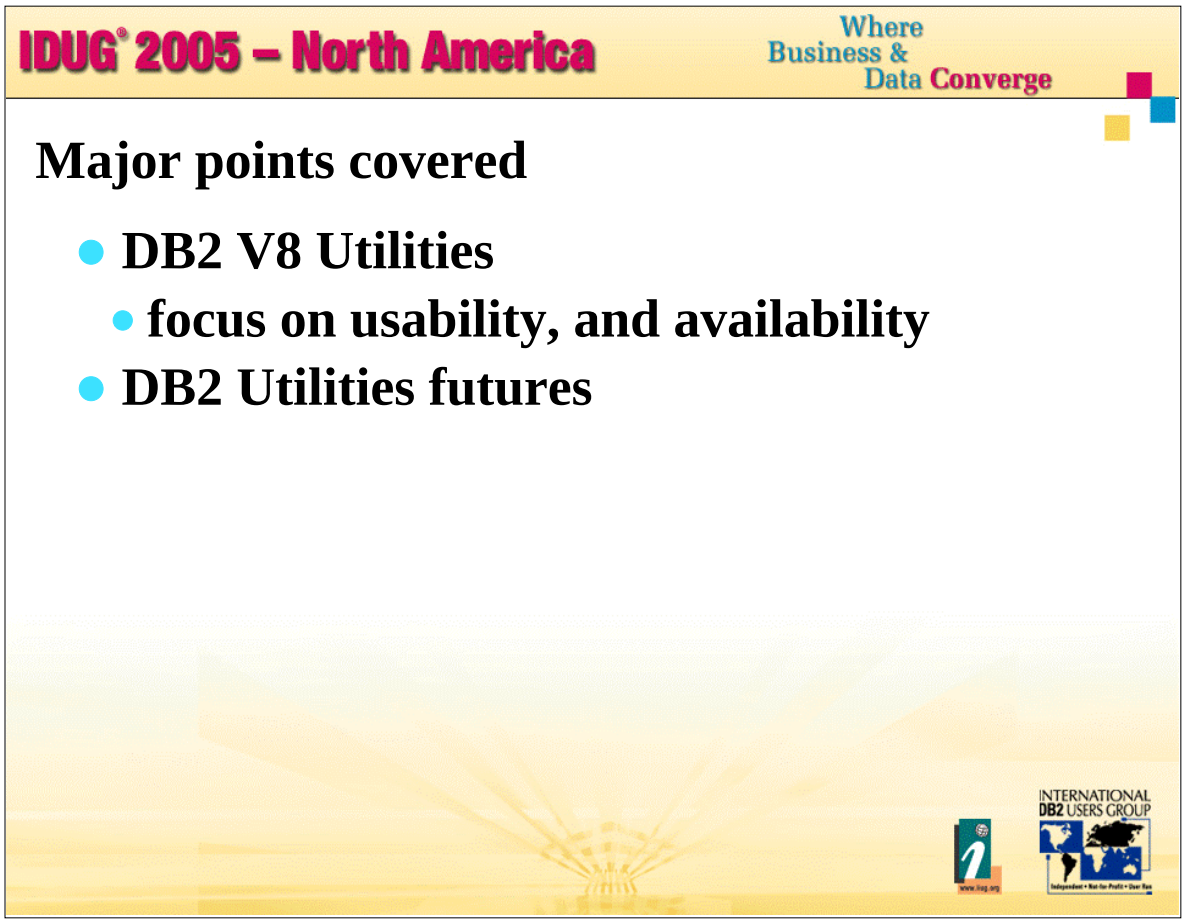


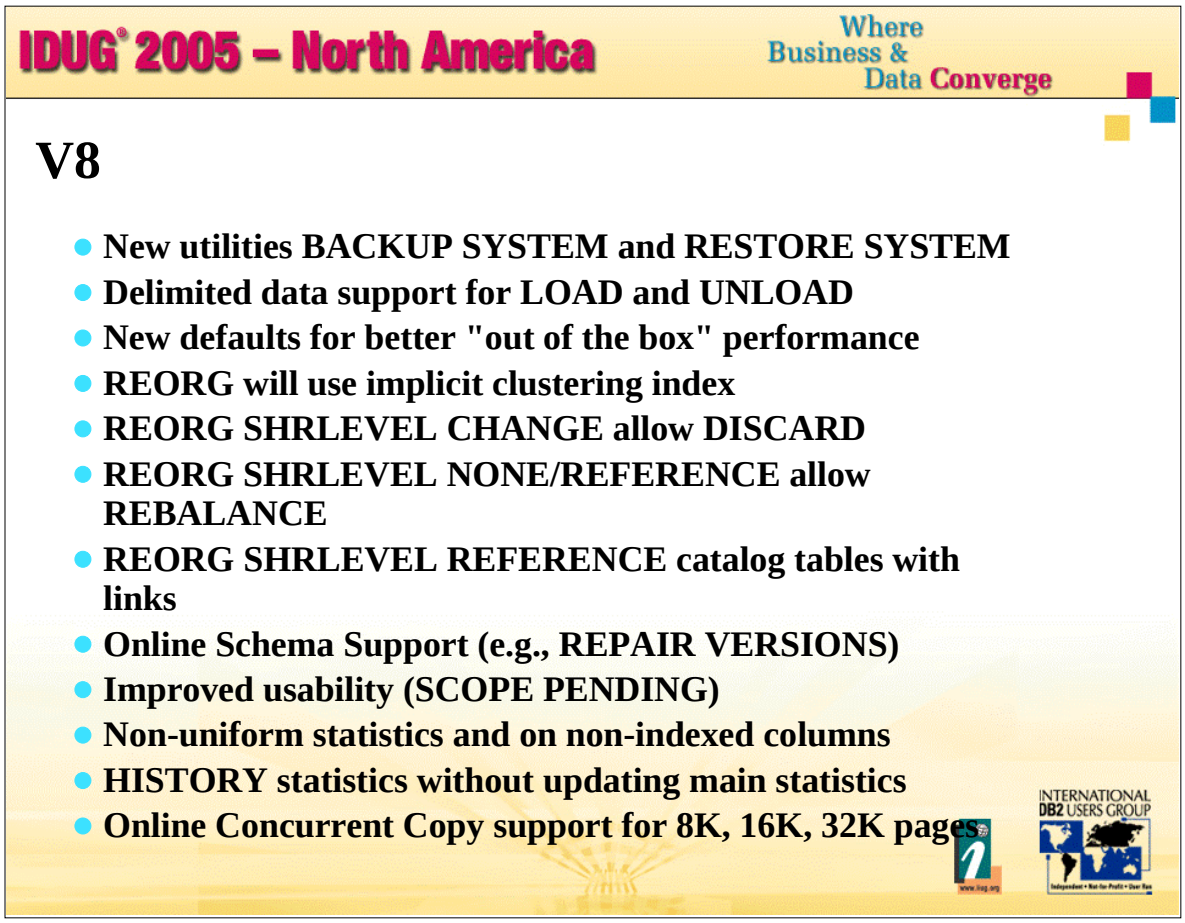
- The function described here is brought to you by the IBM DB2 UDB for z/OS Utilties Development, Test, and Service teams.



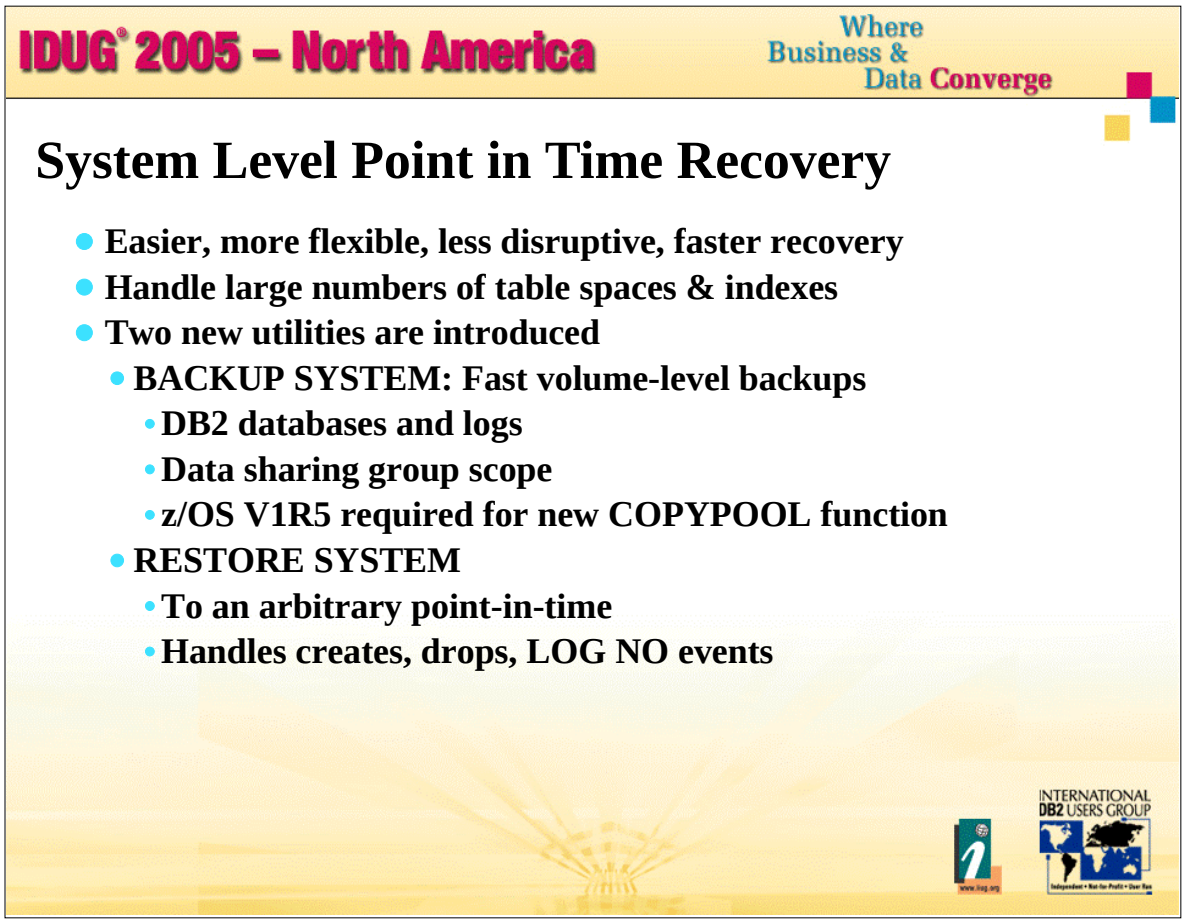
- We are going to cover:

  Details of Version 8 where the focus was on usability, and data availability.

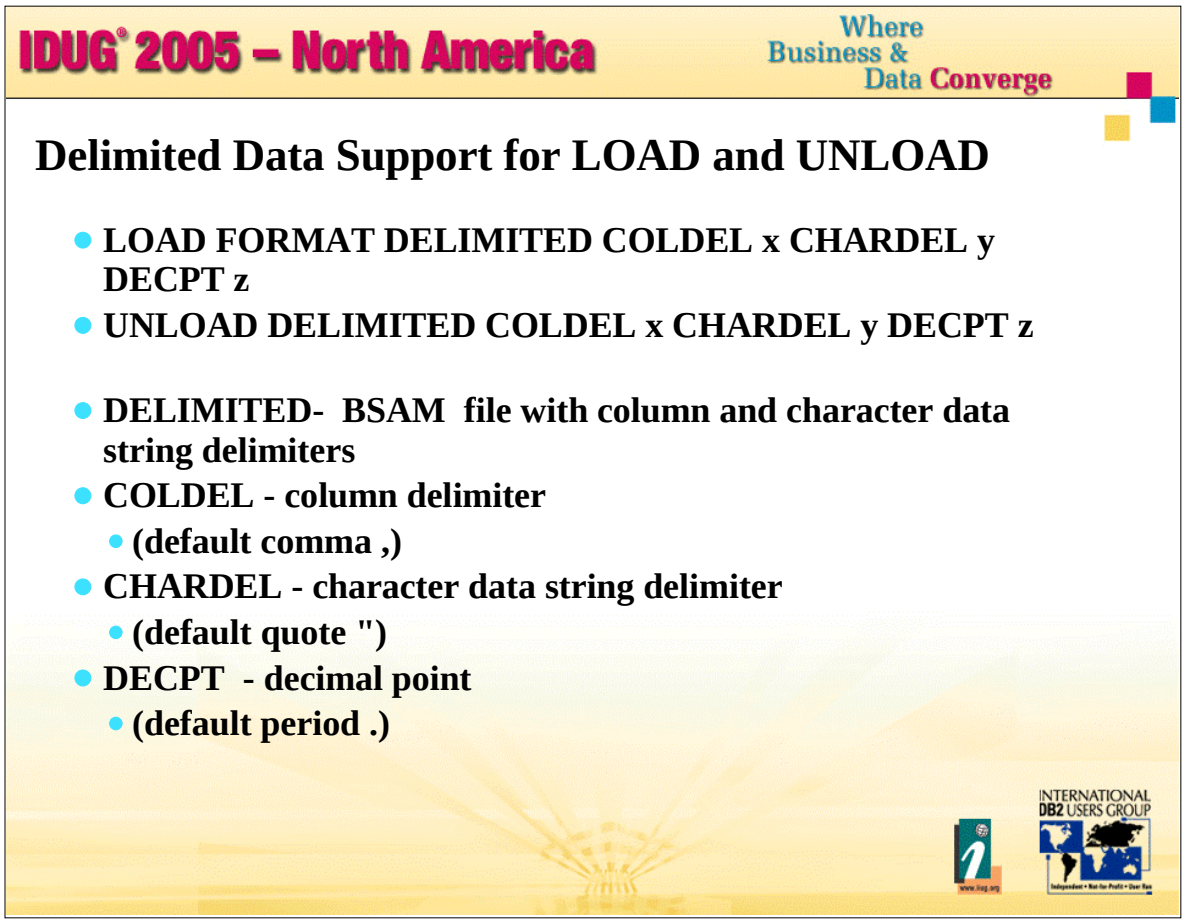
  What the future may bring in DB2 utility enhancements



- SYSTEM BACKUP and SYSTEM RESTORE require DFSMS 1.5 and provide full system backup and recovery.
- Delimited data support for LOAD and UNLOAD provides family compatibility with DB2 for Linux, Unix, Windows
- New defaults for better "out of the box" performance - no longer have to add these parameters to get the best utility performance
- REORG SORTDATA is now allowed on 32K records if DFSORT is used
- REORG will now order the data by the same implicit clustering index as SQL inserts if no index on the table has been created with or altered to CLUSTER
- DISCARD during Online REORG is now supported
- REORG can automatically evenly distribute data across partitions
- REORG SHRLEVEL REFERENCE is now supported on catalog and directory tables with links
- Online Schema Support
- REPAIR VERSIONS - allows reconciliation between DB2 catalog/directory and tablespace or indexes copied from another system via DSN1COPY or other means.
- SCOPE PENDING - REORG and REBUILD are enhanced to limit operation to partitions or list items in REORG pending or REBUILD pending without the user have to list only those objects.
- More non-uniform statistics and on non-indexed columns - RUNSTATS is enhanced to also collect least frequent column values and to provide most and least frequent column values on non-indexed columns.



- Enhancements to system-level point-in-time recovery for DB2 provide improved usability, more flexibility, and faster recovery. You can now recover your data to any point in time, regardless of whether you have uncommitted units of work. As a result, data recovery time improves significantly for large DB2 systems that contain many thousands of objects. Two new utilities provide system-level point-in-time recovery:
- The BACKUP SYSTEM utility provides fast volume-level copies of DB2 databases and logs. It relies on new DFSMShsm services in z/OS Version 1 Release 5 that automatically keep track of the volumes that need to be copied. BACKUP SYSTEM is less disruptive than using the SET LOG SUSPEND command for copy procedures. An advantage for data sharing is that BACKUP SYSTEM is group scope.
- The RESTORE SYSTEM utility recovers a DB2 system to an arbitrary point in time. RESTORE SYSTEM automatically handles any creates, drops, and LOG NO events that might have occurred between the backup and the recovery point in time.



- A delimited file is a stream of characters consisting of cell values ordered by row , then by column. Rows are separated by row delimiters. For z/OS, a row is a single BSAM record. Columns within each row are separated by column delimiters.
- Requirement is to make it easier to replicate or migrate data from other platforms to DB2 for z/OS. Delimited file format ("DEL" to DB2 UDB) is common among non-z/Series RDBMSs. Currently difficult to coerce data from other platforms into LOAD's positional format. Direct support of DEL format will make moving data much easier.
- Delimiters are mutually exclusive, a delimiter can not be binary zero, and a default decimal point (.) can not be a string delimiter. Double character delimiters recognition is supported. Applies to CHAR, CLOB, VARCHAR

- "what a ""nice "" day" LOADs as what a "nice " day
- I am 6" tall. UNLOADs as "I am 6"" tall."

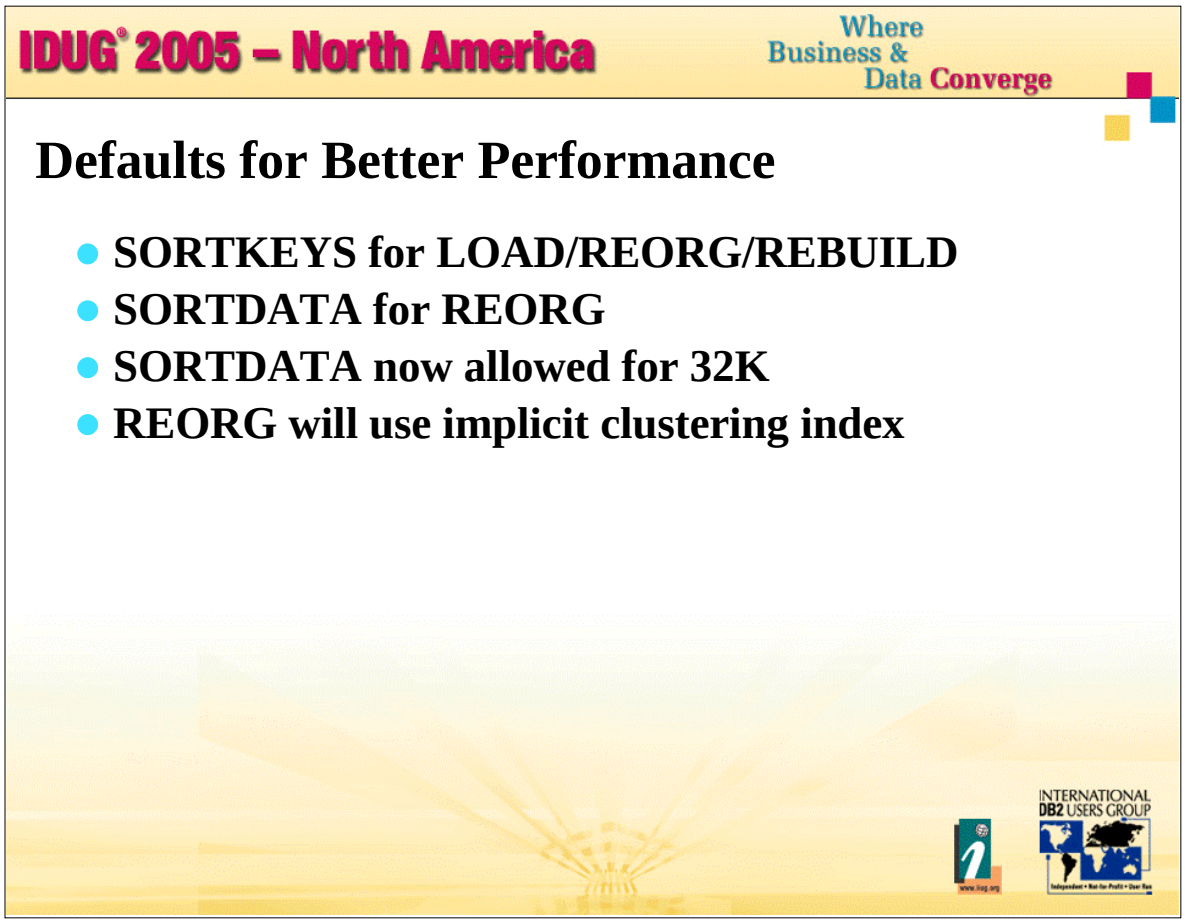
Delimited Data Support for LOAD and UNLOAD

- **LOAD FORMAT DELIMITED**
  - **Field specification POSITION is ignored**
  - **Single LOAD INTO TABLE statement only**
  - **All string data treated as VARCHAR or VARGRAPHIC**
  - **All numeric (and EBCDIC GRAPHICs) will be considered EXTERNAL**
- **UNLOAD DELIMITED**
  - **Field specification POSITION is ignored**
  - **Single UNLOAD FROM TABLE only**
  - **Fields are in character string or numeric external format**
  - **String data enclosed by character delimiter (NOPAD)**
  - **No header record is created. Header none is forced.**
  - **No length byte preceding any data**
  - **Null value is indicated by the absence of a cell value.**

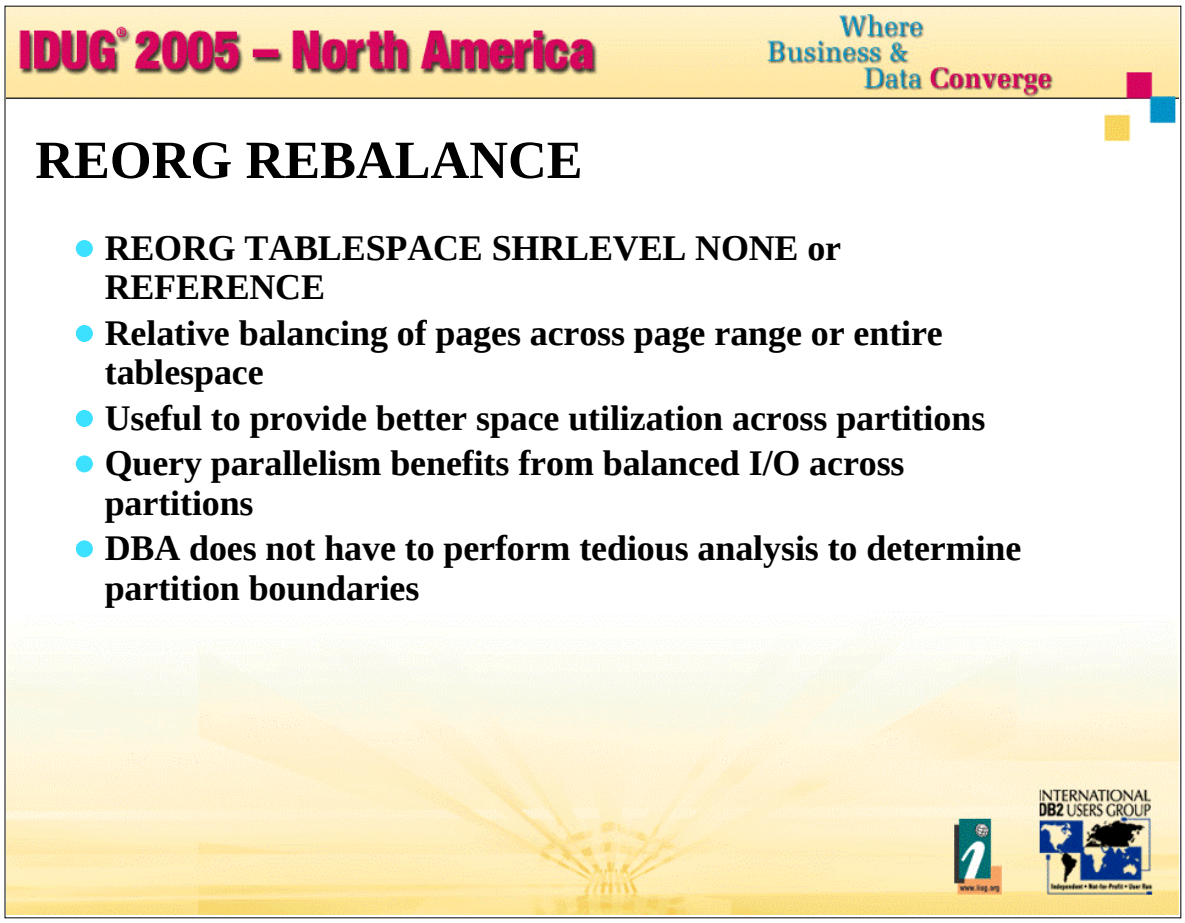
- Default Delimiters Values and maximum value allowed

	EBCDIC		ASCII/UNICODE	
	SBCS	DBCS/MBCS	SBCS	MBCS
	----	----------	-----	-----
Character String	'7F'x	'7F'x	'22'X	'22'X
Decimal point	'4B'x	'4B'x	'2E'X	'2E'X
Column	'6B'x	'6B'x	'2C'x	'2C'x
Maximum value allowed	None	'3F'x	None	'7F'x

- note: The hex values specified in the above EBCDIC delimiters are double quote ("), period (.), and comma (,)in most EBCDIC codepages.



- Over several releases keywords have been added to several of the utilities to activate methods providing higher performance. Unfortunately, changing hundreds of utilities to add these keywords is impractical for most customers. Also, it was not obvious to new utility users that these keywords should be specified. So in V8 we force on SORTKEYS for LOAD, REORG, and REBUILD and SORTDATA in REORG. With extensions in DFSORT, we are also able to use SORTDATA for 32K records.
- REORG will no longer require an index defined CLUSTER YES to order the data - if none exists the first index created will be used just as DB2 uses for inserts. Along with this, Online Schema provides the ability to alter the cluster attributes of indexes.

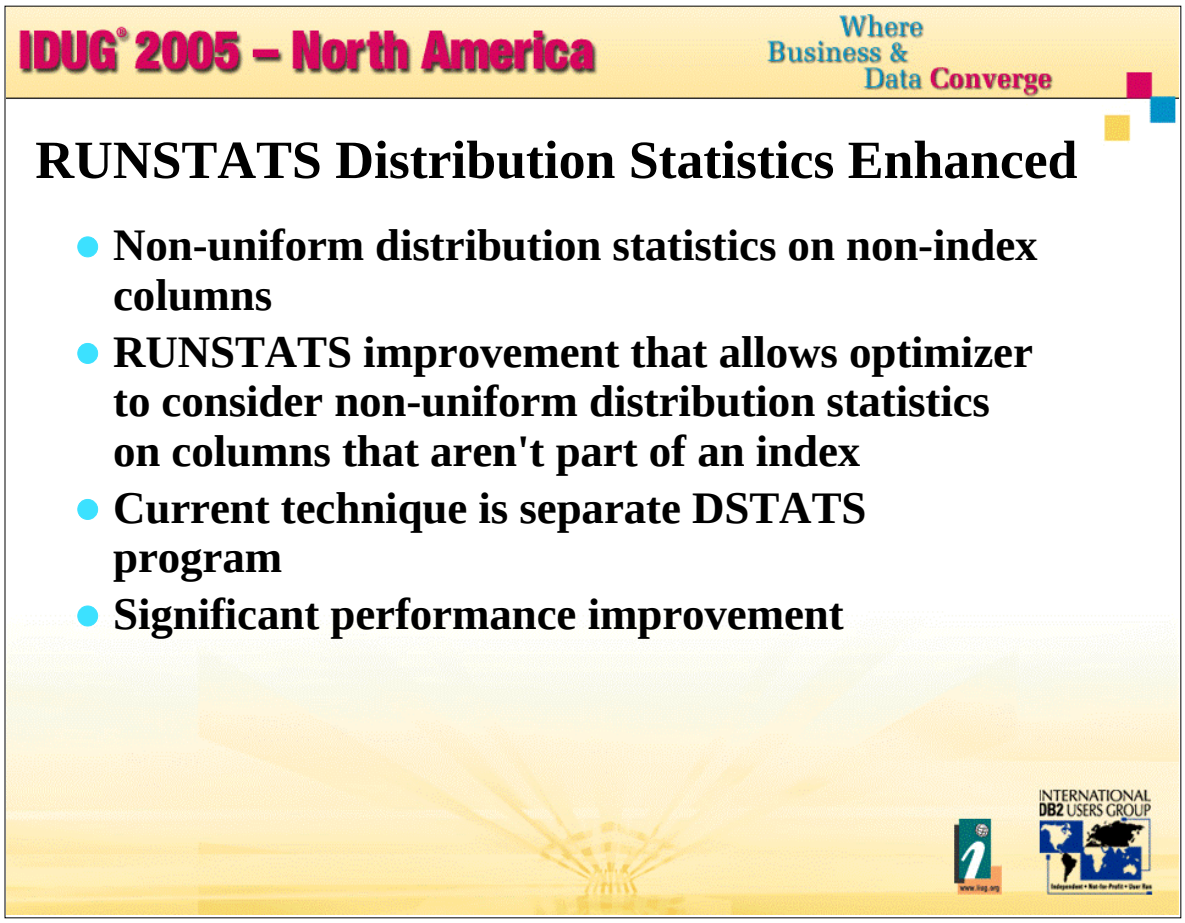


- As described in US Patent 6,269,375, before reloading the data REORG with REBALANCE will calculate an approximate number of pages expected to be populated across all the partitions taking into account the percent free space allowed on each page as well as the free pages specified. During reloading of the data, REORG will note the value of the partitioning columns as each partition reaches the page count threshold. At the end of a successful REORG, the catalog and directory are updated with the new partition boundary values.

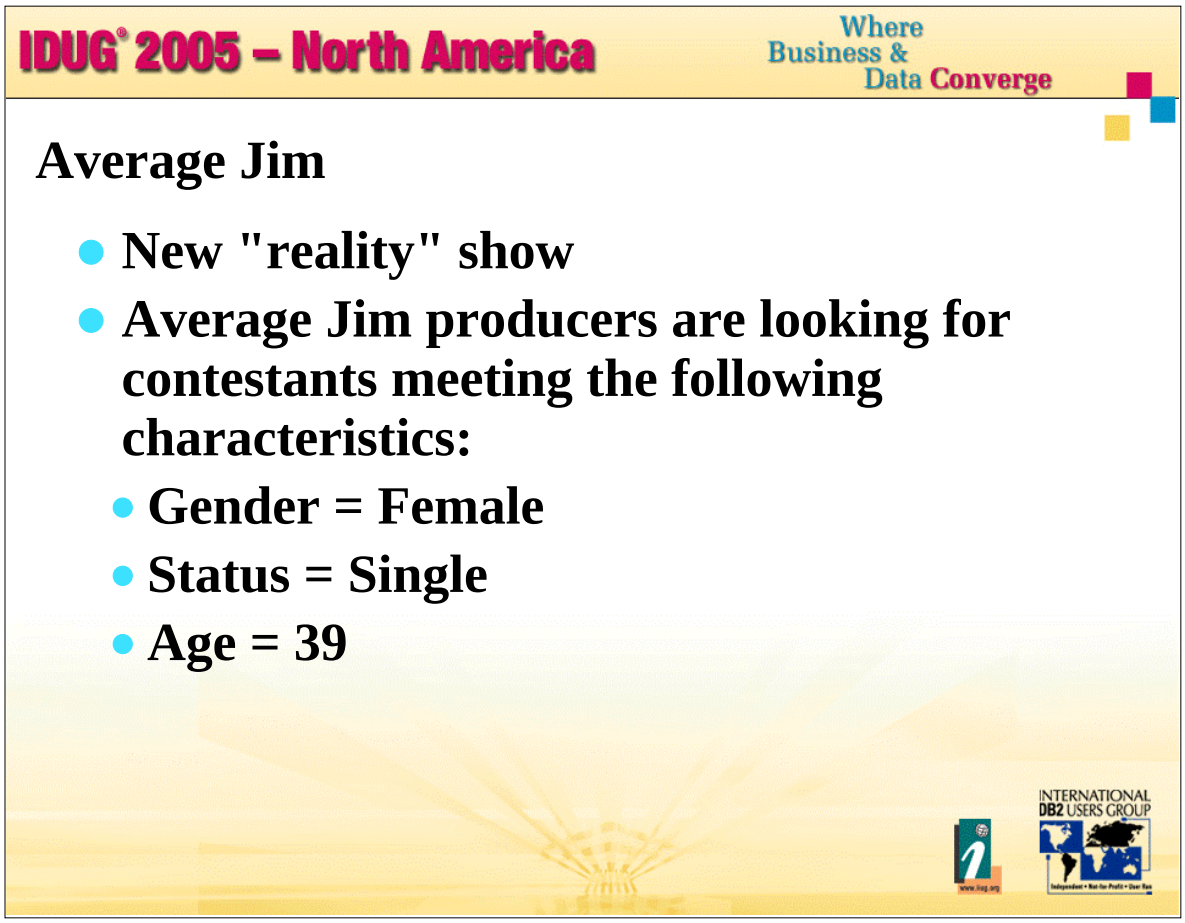
Online Schema Support

- REPAIR VERSIONS - Updates the versions in the catalog and directory from the information in the table space or index. Use VERSIONS when you perform the following tasks:
  - When you use the OBIDXLAT option of DSN1COPY to move objects from one system to another.
  - As part of version number management for objects that do not use the IBM REORG utility.
- Improved usability for objects placed in pending states
  - SCOPE PENDING for REORG and REBUILD
  - REORG SCOPE PENDING will operate only on objects in
    - REORP, AREO*
  - REBUILD SCOPE PENDING will operate only on objects in
    - RBDP, AREO*, RECP

- REPAIR VERSIONS helps resolve version number inconsistencies when data is moved from one system to another or where manipulated outside of DB2. For a description of the considerations please see the DB2 Utility Guide and Reference.
- Tablespace, table and index alterations may place tablespaces and indexes in advisory or restrictive states. So the user does not have to issue display database commands to find which objects are in states advising or requiring REORG or REBUILD, the SCOPE PENDING directive has been added to these two utilities. When specified, only objects in the listed states will be reorganized or rebuilt. The default SCOPE ALL directive will result in the utility operating on the objects as in prior releases.



- Skewed data distributions are responsible for a high proportion of performance problems with DB2 queries, especially in ad hoc queries. Symptoms can be less than optimal join sequences, too much synchronous I/O, and long response times. When there is asymmetrical distribution of data, not having distribution statistics on non-leading indexed columns and/or non-indexed columns can cause DB2 to make sub-optimal table join order and table join method decisions.
- Collecting distribution statistics for non-leading indexed columns and/or non-indexed columns allows DB2 to use these statistics for better access path selection. Better index selections can be made, when there are screening predicates or there are matching in-list / in-subq predicates which break up matching equals predicates.
- No longer have to run separate DSTATS program to rescan the data to collect the statistics.



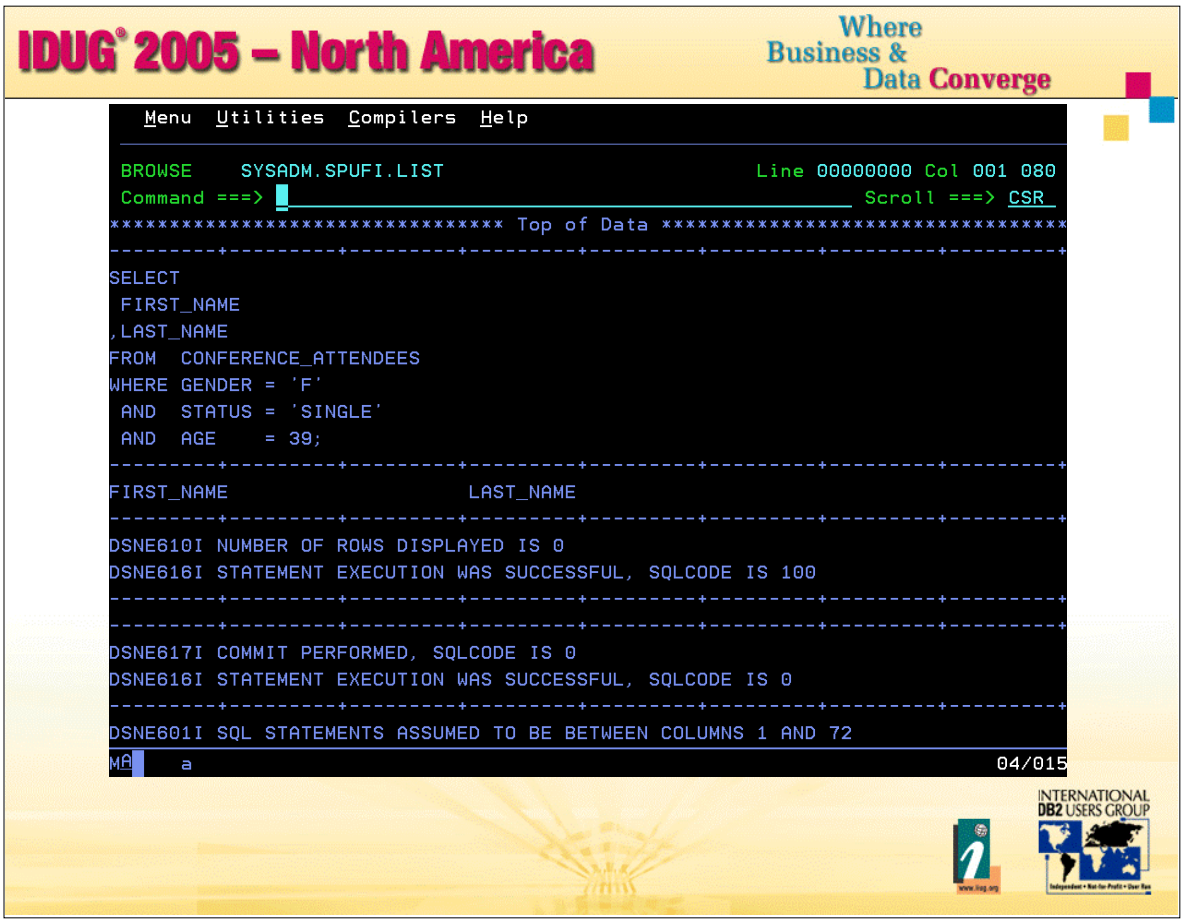
- To better illustrate non-uniform distribution statistics and review the way the DB2 optimizer computes access paths, lets look at this in the paradigm of the now ubiquitous reality show

Average Jim Data Statistics

	Pred	Dflt FF	Col card	Calc FF	NUD FF
Gender	=	1/25	2	1/2	2/5
Status	=	1/25	2	1/2	2/15
Age	=	1/25	39	1/40	1/4
Est * Card		<1		9	20

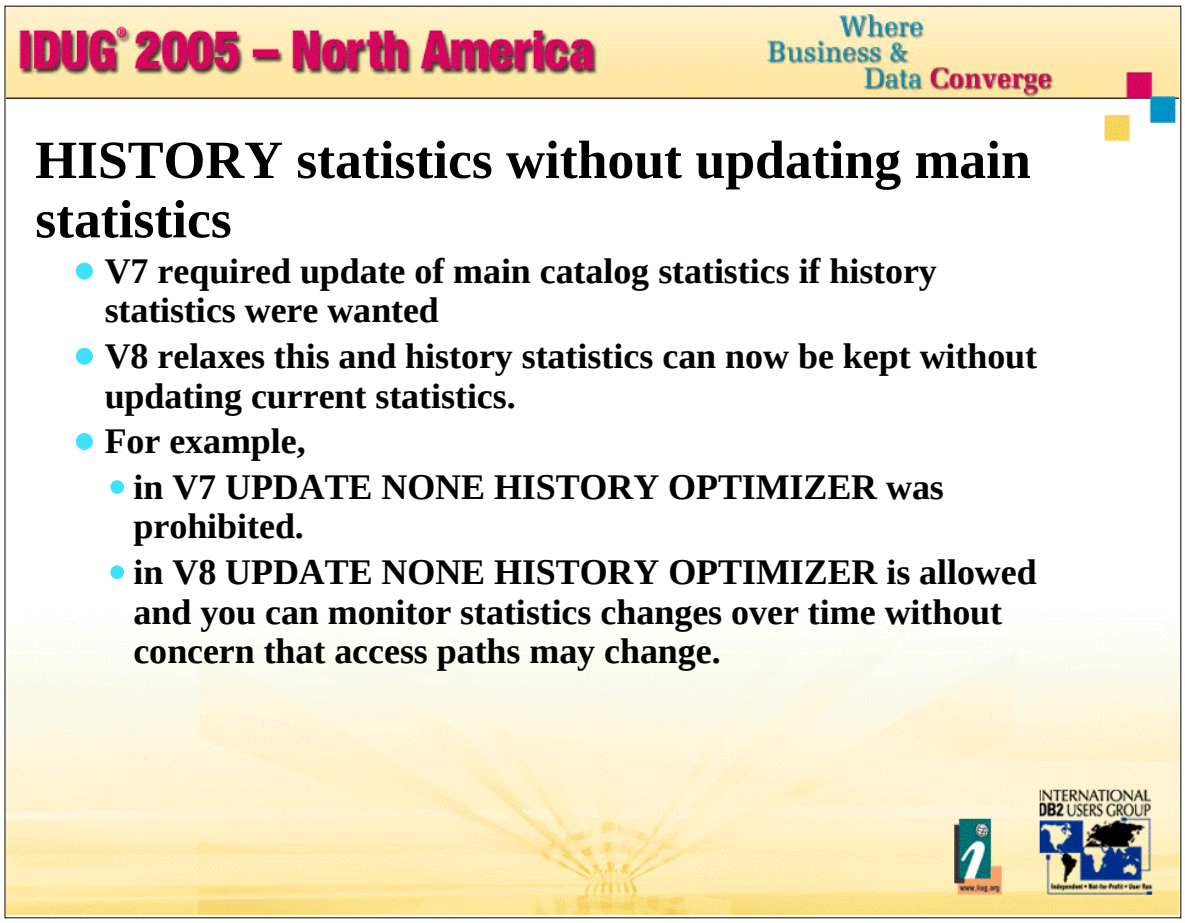
*Database cardinality = 1500

- In a paper published by IBM Research in 1979 title "Access Path Selection in a Relational Database Management System" the basic algorithms for access path filter factor and join orders were described. In this paper there is a very interesting statement that for a column equal value predicate "This assumes an even distribution of tuples among the index key values". Experience has taught us that this only occurs with test data.

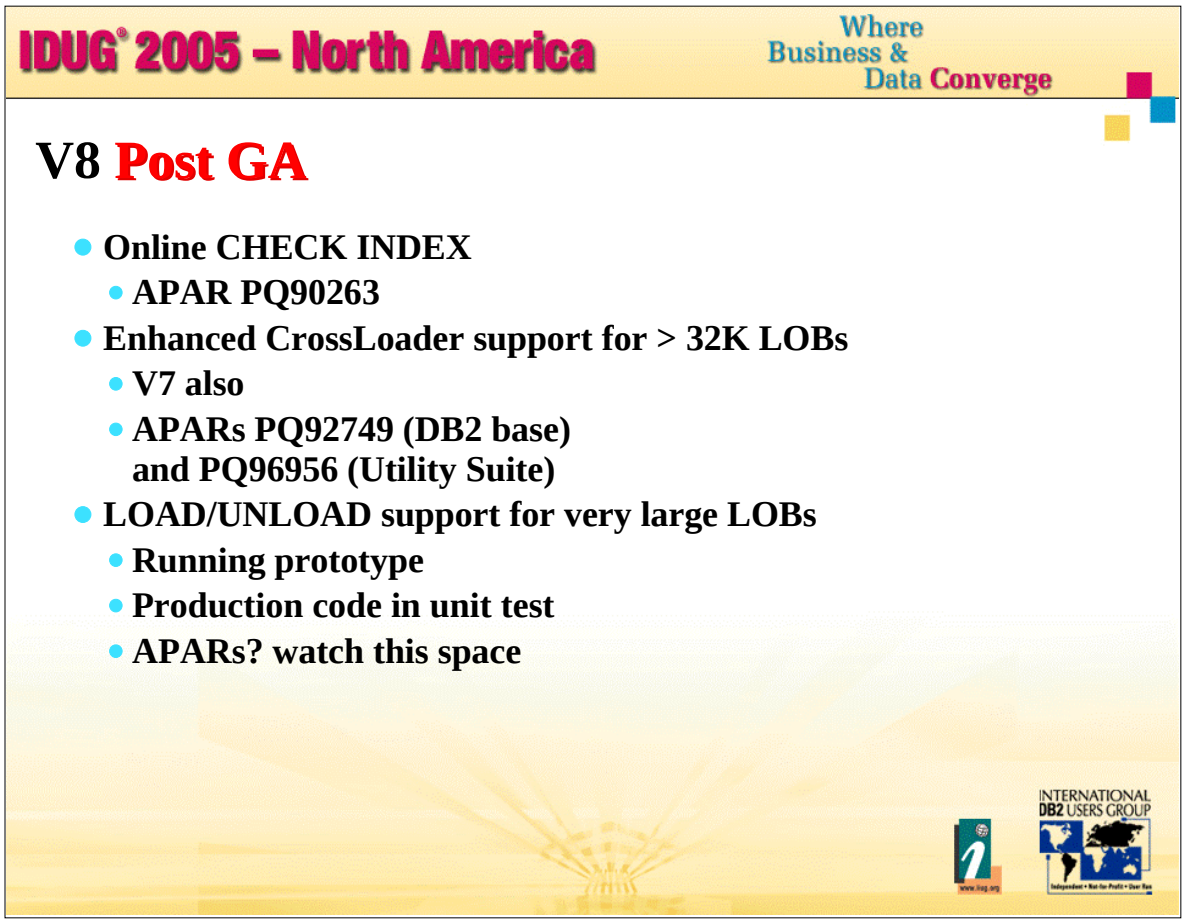


- We run the query and..... what happened?

  Well, as it turns out the data is even more non-uniform than we thought. If we had collected statistics on all three columns in our predicate we would have seen that all of the single females are less than or equal to 38 or they are greater than or equal to 40. Go figure.



- Version 8 relaxes the Version 7 requirement that statistics history would only be collected if the main catalog statistics were also updated. This greater flexibility allows the user to keep track of statistics changes overtime without the concern that main statistics changes could result in access path changes, especially for dynamic SQL.

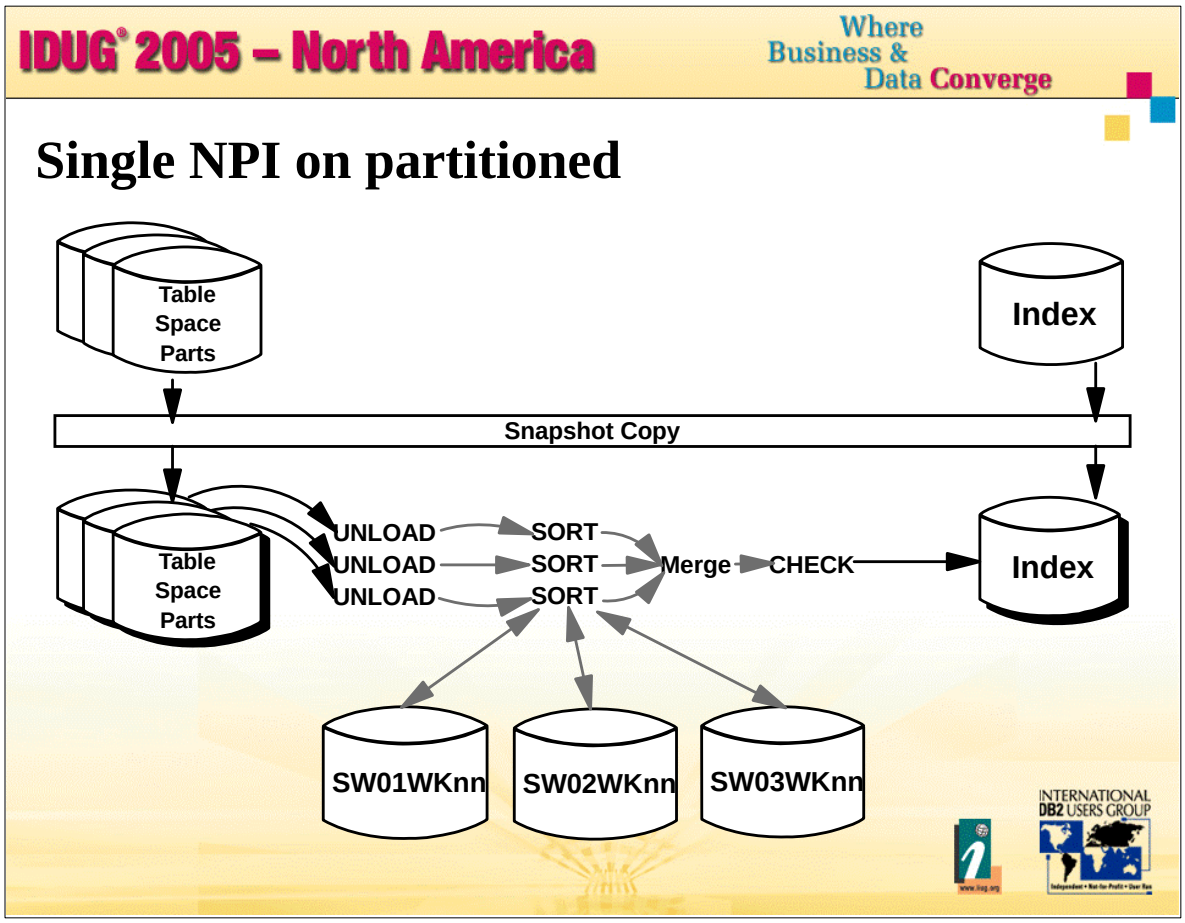


- You thought you knew what function was in DB2 Version 8 - but WAIT! We have more!
- Online CHECK INDEX was planned to V8 but due to resource constraints but testing could not be completed in time.
- You ALSO thought you knew what function was in DB2 Version 7 - but WAIT! There is still more to come!
- Architectural limits within LOAD/UNLOAD does not allow for a record greater than 32K to be loaded or unloaded.
- The demand for improved support for large LOBs is so great we are delivering improvements in two stages
  - The CrossLoad function of the LOAD utility is first
  - LOAD and UNLOAD of very large LOBs from and to data sets comes a little later

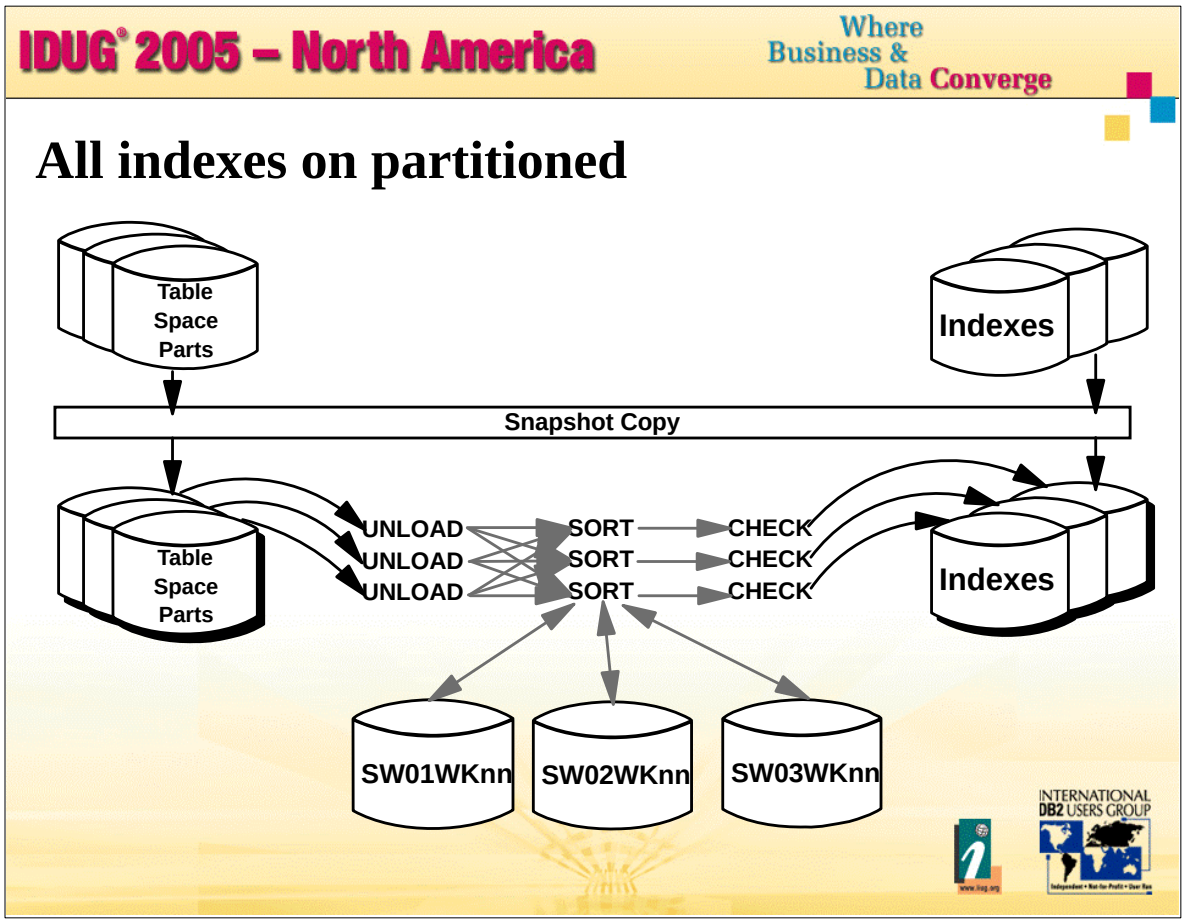
Online CHECK INDEX

- Current CHECK INDEX is slow and data and indexes are unavailable for update for the duration.
- Online CHECK INDEX different design than Online REORG
- Claim as reader for target data and indexes
- Create shadow datasets when DB2 managed
  - same dataset naming convention as Online REORG
  - cannot run Online CHECK INDEX on two logical parts of NPI
- Drain writers for target data and indexes
- Flash data and indexes from target to shadows
- After copy logically complete for ALL,
  - dedrain target data and indexes
  - run parallel check index on shadow data and indexes same parallel design as REBUILD INDEX
- At utilterm delete shadow datasets when DB2 managed

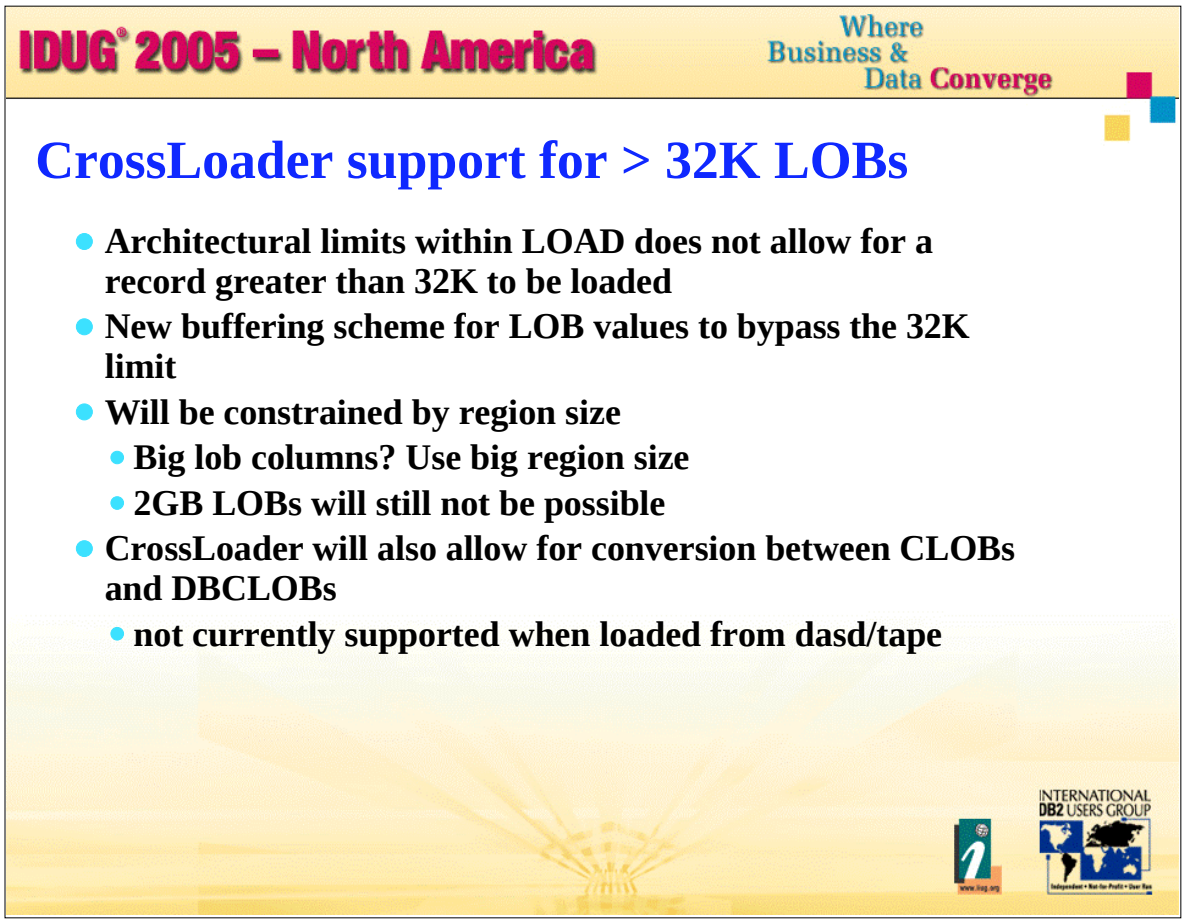
- Version 8 Utility Guide and Reference already has syntax changes documented but you knew that already, right?



- In this example of checking a single non-partitioning index on a partitioned tablespace, three unload subtasks will pipe index keys to three sort subtasks. The three sort subtasks will each pipe sorted keys to a merge subtask. The merge subtask will pipe the sorted and merged keys to an index check subtask.



- In this example of checking all the indexes on a partitioned tablespace, three unload subtasks will pipe index keys to the appropriate of the three sort subtasks. The three sort subtasks will pipe sorted keys to three companion index check subtasks.

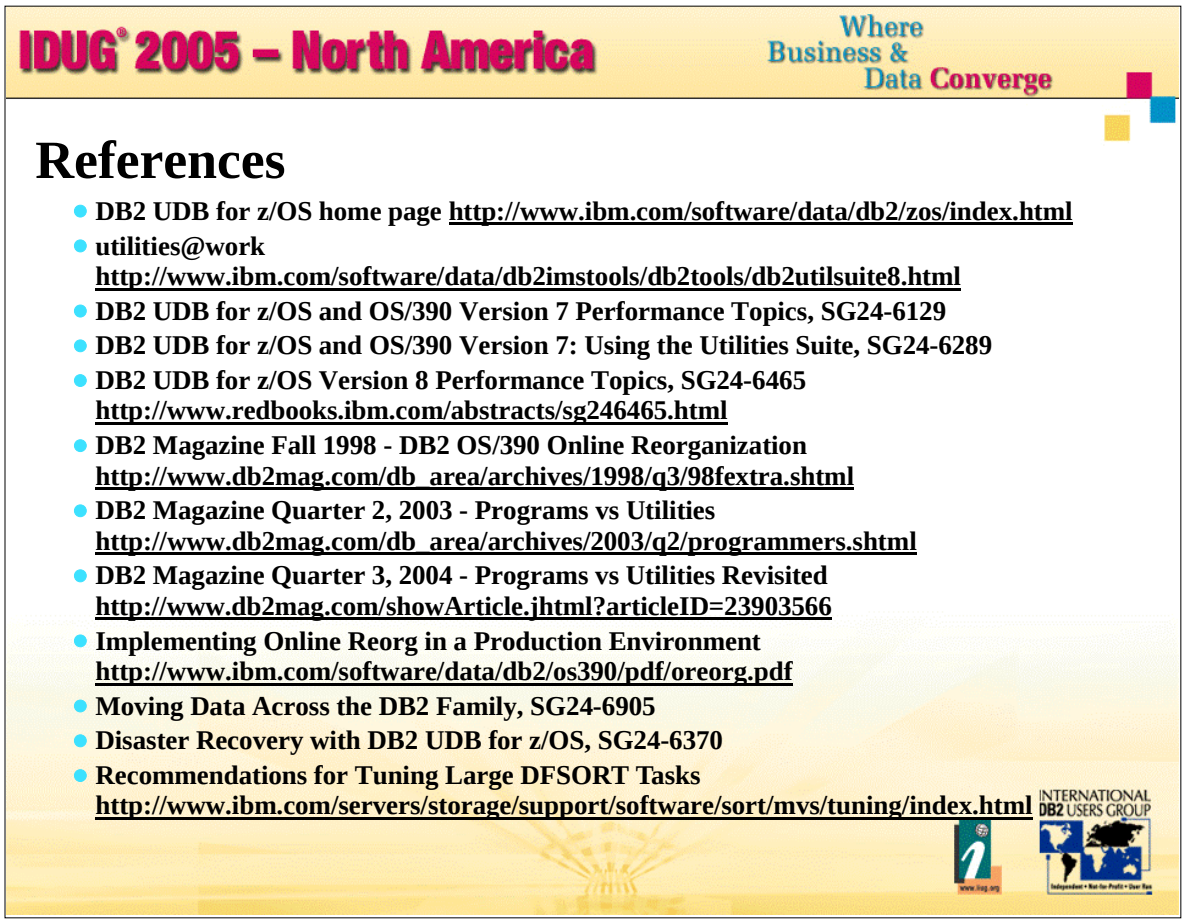


- To get around architectural limits within the LOAD utility, we will use a design where the lob values are stored in a separate buffer.
- We will now be limited by the region size "above the line" where we will reserve half the available storage in the batch address space for the lob values.
- Since DB2 does not have support for 64 bit addressing in applications, we cannot take advantage of "above the bar" storage so there is no room to store 1GB (a little less actually) of LOB column values per row
- We have also extended the support for conversion from CLOBs to DBCLOBs but only within the CrossLoad function at this time.

LOAD/UNLOAD support for very large LOBs

- **Requirement is to move LOBs from one z/OS system to another z/OS system**
- **Need to support millions of rows**
- **Typical LOB sizes are 25K, 200K, 1MB**
- **Need to allow user to limit LOAD at target with WHEN clause**
- **LOB column values will be stored as separate PDS member, PDS/E member, of HFS directory member.**
- **Values from each LOB column will be stored in separate PDS, PDS/E, or HFS directory**
- **LOB column values from each row will have identical member names in each PDS, PDS/E, or HFS**
- **Data set name stored in output record**
- **Design fits well with File Reference Variables where LOB values are in individual datasets**

- LOAD and UNLOAD of very large LOBs is driven by the requirement to move millions of rows containing very large LOBs from one z/OS system to another z/OS system.
- Also have to be able to "efficiently" support the WHEN clause capability of LOAD to restrict loading of only a portion of the records in the dataset.
- LOB column values will be stored as PDS members, PDS/E members (allowing for greater capacity than a PDS), and files within an HFS directory
- If there are multiple LOB columns then there will be multiple PDS, PDS/E, or HFS source destinations
- UNLOAD will store the LOB columns values from each row with identical "member" names in each PDS. PDS/E or HFS destination
- The fully qualified data set name including member will be contained in the input record for LOAD and output record for UNLOAD
- This design is consistent with the File Reference Variable support for SQL applications which may be delivered in a future release.



- To learn more about DB2 for z/OS and usage and tuning information about the the DB2 for z/OS Utilities, please visit these sources on the Web.

Utility Work Ahead in DB2 UDB for z/OS Version 8
C5

Jim Ruddy
IBM DB2 for z/OS Development
jaruddy@us.ibm.com