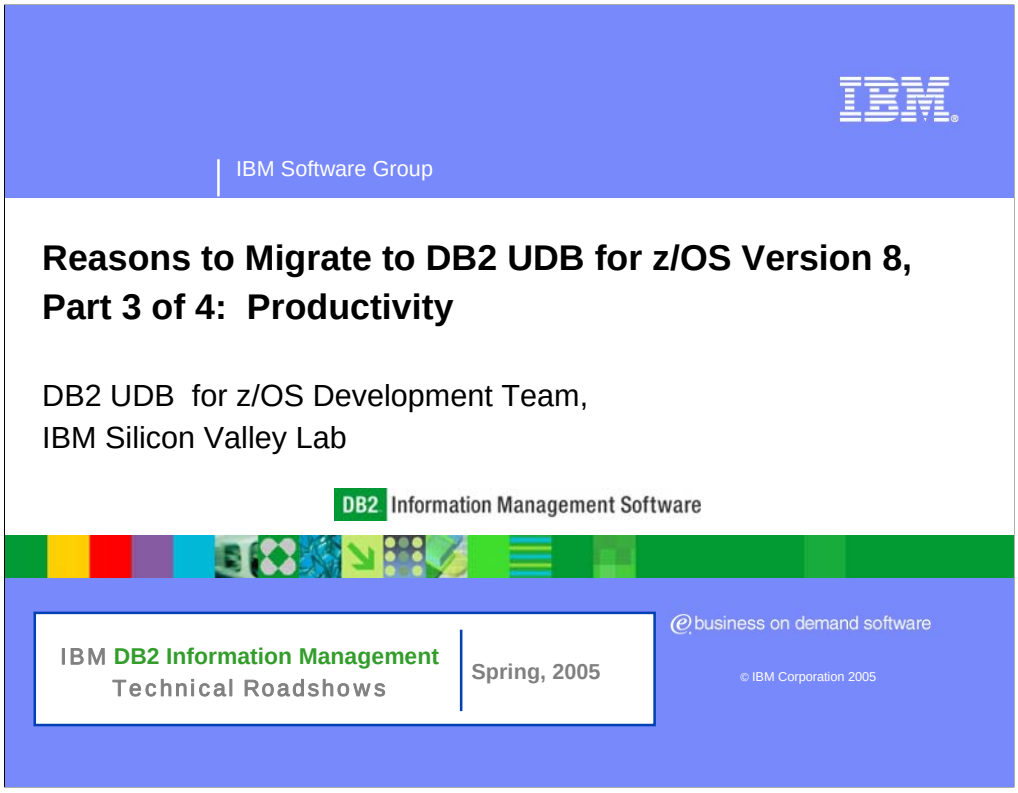
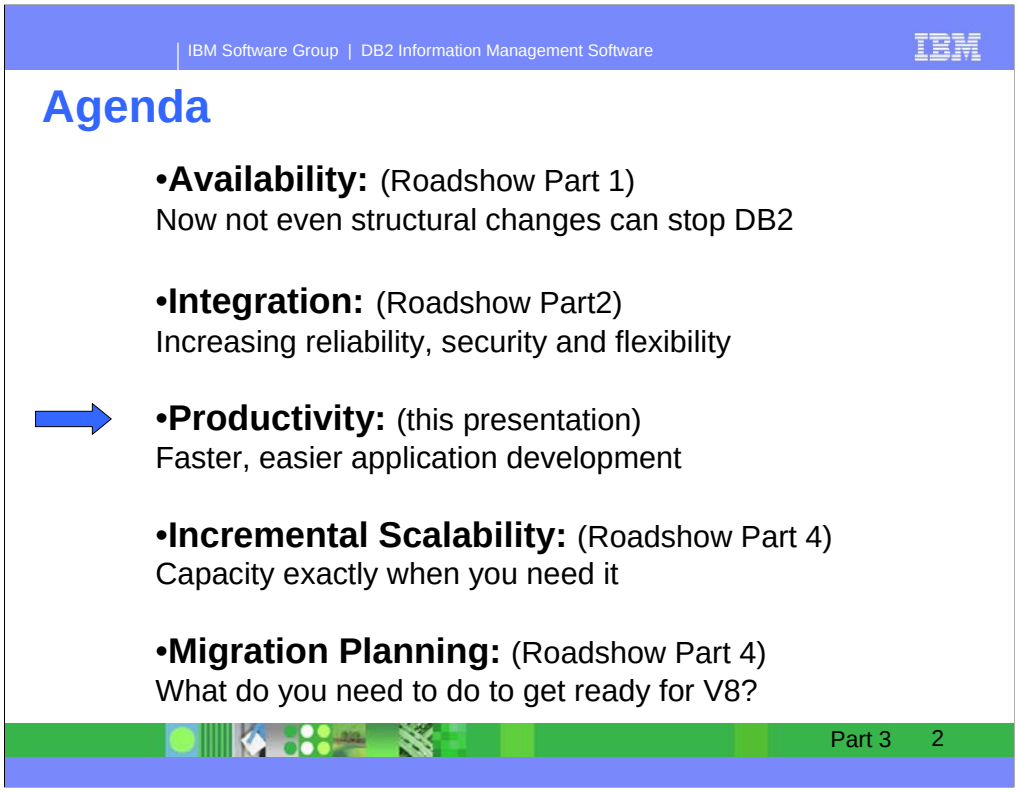
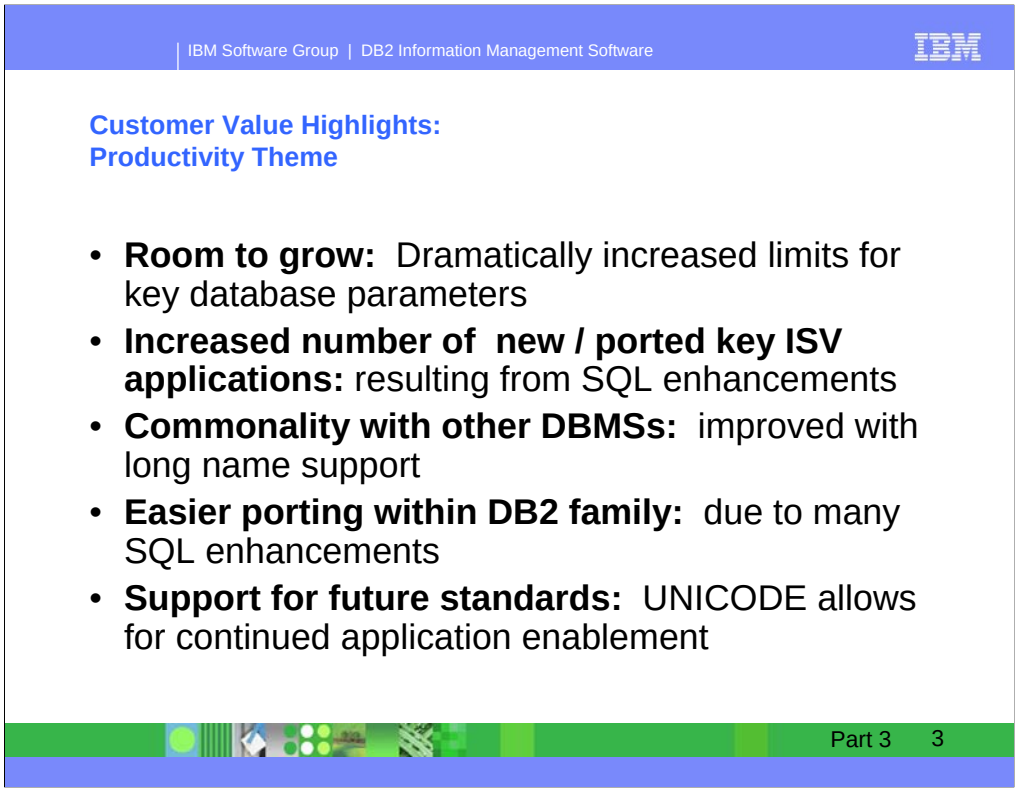


Find out how DB2® Universal Database for z/OS® Version 8 has been re-engineered, with fundamental changes in architecture and structure that will help you manage your very large databases more easily and cost effectively. "Reasons to Migrate to DB2 UDB Version 8" is a day long seminar suited to database technical managers, database administrators, applications developers and systems programmers. It focuses on four critical areas: availability, integration, application development productivity, and flexible growth and incremental scalability. Its information-packed sessions will familiarize you with the enhancements that are enabling organizations to streamline database management, respond more flexibly and quickly to business needs and ensure short and long-term growth capacity.



Many of the items I'm talking about today improve performance, availability, productivity and scalability. We'll try to keep the items that are closely related together, but categories don't always work when you need to choose and the real answer is both.

The agenda for today is roughly
8:15 am Registration and Continental Breakfast
8:45 am Welcome and Introduction
9:00 am Availability: Now not even structural changes can stop DB2
10:30am Break
11:00 am Integration: Increasing reliability, security and flexibility
12:30 pm Lunch
1:30 pm Productivity: Faster, easier application development
3:00pm Break
3:30 pm Incremental Scalability: Capacity exactly when you need it
Migration Planning: How to get there
5:00 pm Close of Program



We just talked about tools and their autonomic capabilities, but this session will concentrate on improved productivity in DB2.

The improvements include increased limits, more consistent and more functional SQL. SQL enhancements make it easier to port an application from other platforms – DB2 for Linux, Unix and Windows or from Oracle.

In many cases, DB2 UDB for z/OS Version 8 has changed the game from one of catch up to one of leapfrog.

SQL and DB2 Family Enhancements

- Multi-row INSERT, FETCH & UPDATE
- GET DIAGNOSTICS
- INSERT within SELECT
- IDENTITY Column enhancements
- SEQUENCES
- CURRENT PACKAGE PATH
- SQL Procedure Language
- Dynamic Scrollable Cursors
- Common Table Expressions, Recursion
- Scalar Fullselect
- Materialized Query Tables
- UNICODE SQL, Multiple CCSIDs
- XML Publishing
- Long names, long statements...

Version 8 is a breakthrough in SQL, with too many new functions to list them all: multi-row INSERT, FETCH & UPDATE, GET DIAGNOSTICS, INSERT within SELECT, IDENTITY Column enhancements, SEQUENCES, CURRENT PACKAGE PATH, Dynamic Scrollable Cursors, Common Table Expressions, Scalar Fullselect, Materialized Query Tables, Unicode SQL, XML Publishing and much more. Support is added for volatile tables, group by expression, multiple DISTINCT clauses, and qualified names for INSERT and UPDATE, Table function improvements, trigger performance, star join sparse index improvements, cost-based parallel sort, better ability to use indexes, longer statements, longer names, longer literals and predicates, session variables, new special registers and SQL procedures improvements. Longer names for tables and columns mean that customers can use more meaningful names, matching standards. Longer SQL statements help with SQL that is generated or used in an SQL procedure. The sum of the changes is a giant leap forward for SQL.

SQL Limits: DB2 for z/OS

Image of Earth from Moon, Source: NASA (Public Domain)

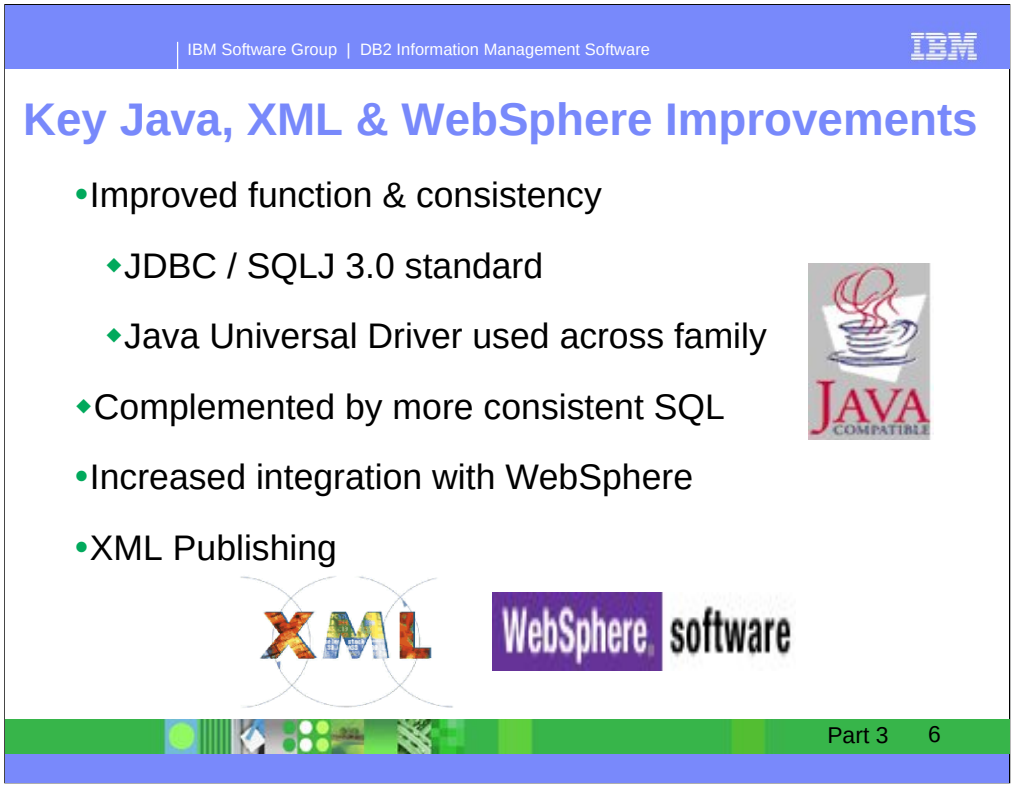
Breaking through limitations - SQL

- **Table name sizes 18 to 128**
- **VIEW & ALIAS names 18 to 128**
- **Column name sizes 18 to 30**
- **SQL statement length 32K to 2MB**
- **Tables in a join 15 to 225**
- **Character Literals 255 to 32704**
- **Index key 255 to 2000**
- **…**

One of the keys to reengineering is breaking through the limits of the current architecture. Increasing some limits improves scalability. Increasing other limits improves productivity, portability & family consistency.

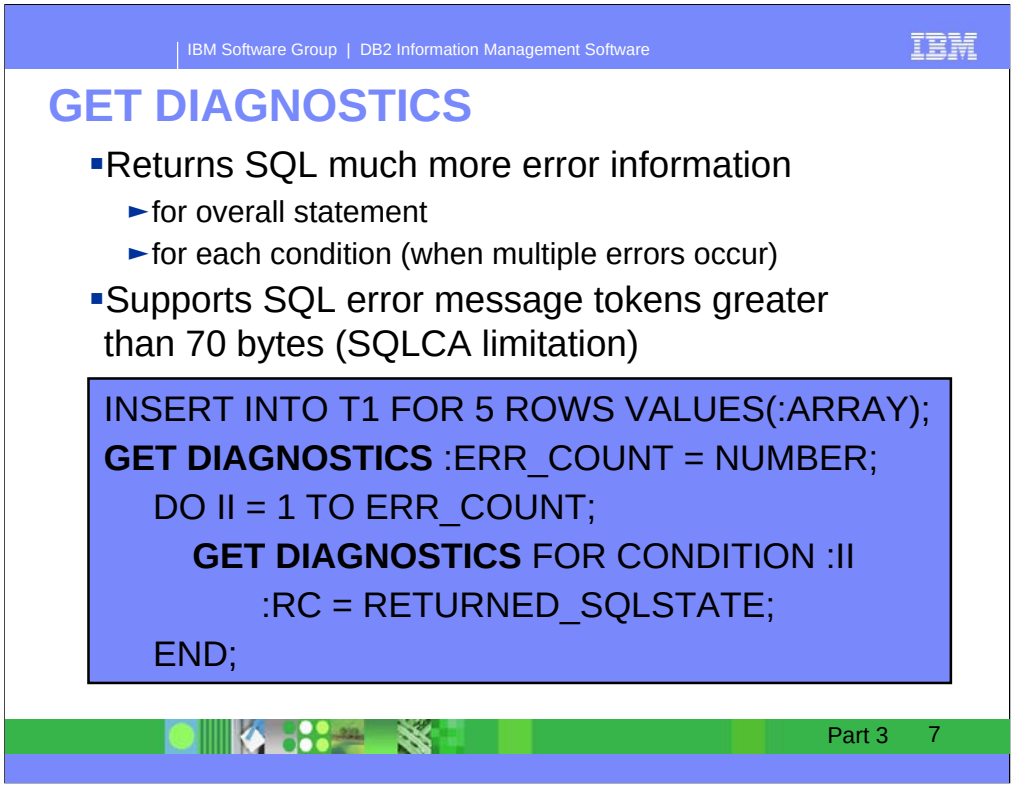
Increasing the amount of virtual storage we can address directly can help with the ability to scale and simplify management for virtual storage. It will require more real memory, but permit increased scalability and availability.

Increasing name sizes & SQL statement lengths makes porting from other DBMS much easier and improves DB2 family compatibility. Increasing the maximum number of partitions helps DB2 scale farther and makes management much easier when you can have one partition per day for 11 years.



Java support will be more consistent across platforms as we use a single code base across the DB2 family. The improved consistency also adds new function to DB2 and improves integration with WebSphere and Java.

The Java Universal Driver is updated to support the JDBC/SQLJ 3.0 standard, including improvements like savepoints, connection pooling improvements, the ability to reuse PreparedStatements, multiple open ResultSets for a single stored procedure, WITH HOLD cursors, and improved BLOB and CLOB support.
ftp://ftp.software.ibm.com/software/db2storedprocedure/db2zos390/techdocs/F01.pdf



The new GET DIAGNOSTICS statement is important to provide the information from all of the extended names and new function. Most programmers will need to switch from using the less standard SQLCA and use this more standard, more capable facility for diagnostic information when their applications include long names or multi-row operations.

The GET DIAGNOSTICS statement can return a lot more information about the statement and / or about conditions and connections. It can return the longer names. It can return multiple conditions for the multi-row statements. It can return the error message associated with an error.

INSERT within SELECT

- Elegant technique for retrieving values created / modified by DB2 during INSERT
  - identity columns, sequence values
  - user-defined defaults, expressions
  - columns modified by triggers
  - ROWIDs, CURRENT TIMESTAMP, …
  - INSERT with return or SELECT from INSERT

EXAMPLE:

```
SELECT C1, C2, C3, C4, C5 FROM
  INSERT (C1, C5) INTO T1
    VALUES('ABC', CURRENT DATE);
```


Have you wanted to return values from the row you just inserted? Applications often need the value of the identity column, the defaults, expression results, current timestamp or the effect of a trigger. Now you can SELECT FROM the inserted row. While there have been some specialized techniques, this is more general and more elegant.

Some programmers think of this as being able to return the generated values from an insert or insert with returns.

Dynamic Scrollable Cursors

- Scrollable cursor provides access to base table rather than work file -- allows viewing of updates by others.
- Defaults to single row fetch, so DDF applications should use:
  - multi-row FETCH
  - positioned update/delete for multi-row FETCH

```
DECLARE C1 SENSITIVE DYNAMIC SCROLL
CURSOR FOR SELECT C1, C2 FROM T1;
```

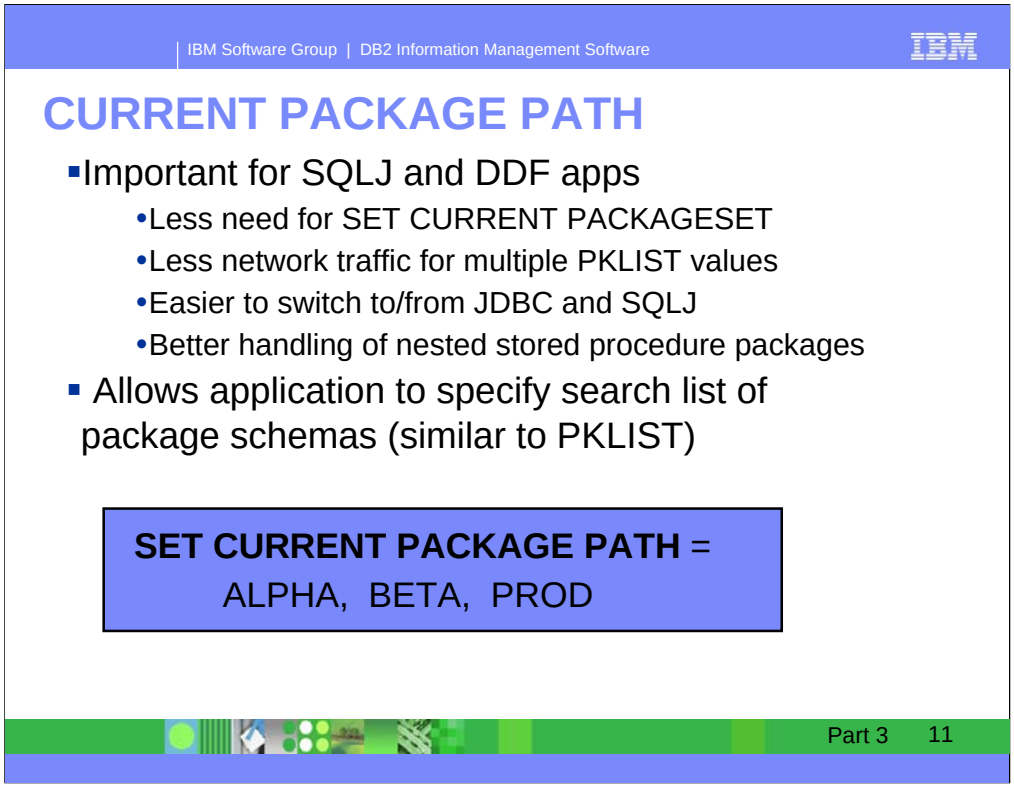

Static scrollable cursors came in V7, with the ability to use a work file for scrolling. Dynamic cursor scrolling is performed directly on the base tables. You can use embedded SQL for scrolling, and this work complements the multi-row SQL statements.

Common Table Expressions (CTE) & Recursion

- Refer to table-name later in SQL
- Allow in SELECT, CREATE VIEW, INSERT
- WITH table-name [column-list] AS fullselect
- For recursion fullselect of CTE refers to itself
  - fullselect that does not refer to CTE
  - UNION ALL
  - fullselect that refers to CTE

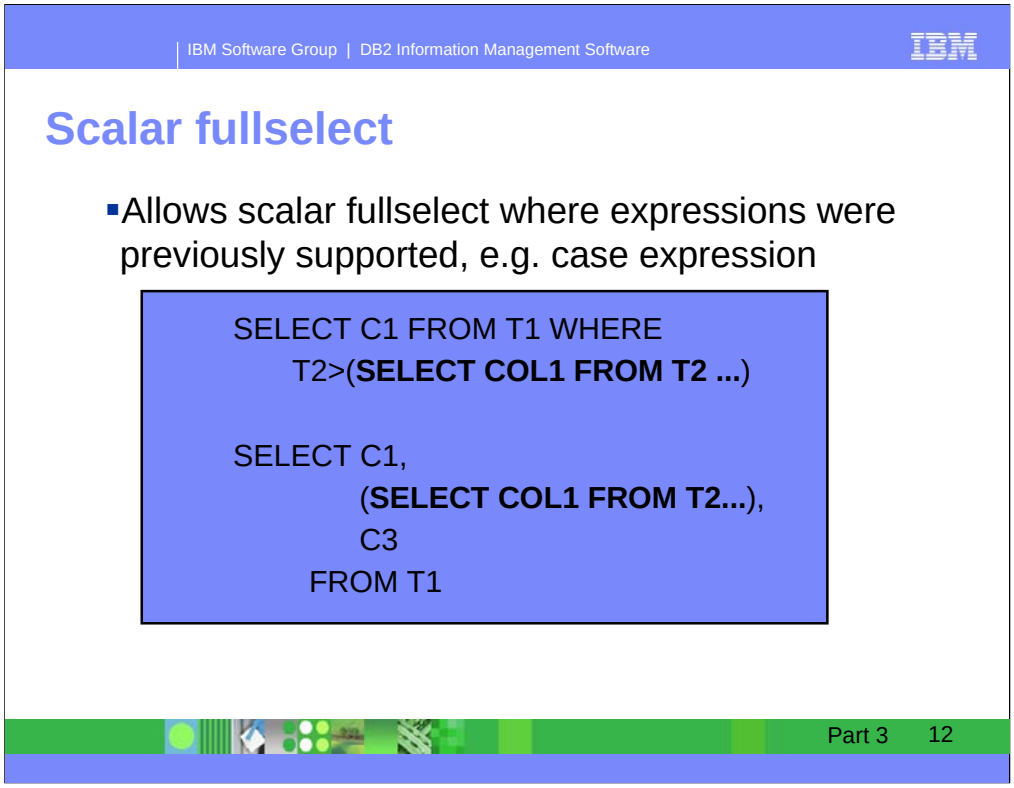
Common Table Expressions provide improved usability and DB2 family consistency. In some cases they can be used to improve performance as well, computing a value once, rather than several times.

The common table expression is defined and used within an SQL statement. Each common table expression can be referenced many times in the statement, and all references to a common table expression in an SQL statement share the same result table. This is unlike regular views or nested table expressions which are derived each time they are referenced.



Package switching and versioning for static SQL applications is critical to DB2 for z/OS customers. SQLJ access will increase the need for these types of control. A new special register, CURRENT PACKAGE PATH, provides a means to specify a list of collections to search for the appropriate package.

The semantics are similar to the PKLIST Bind Option, except that the PACKAGE PATH list is processed at the server. This new special register will provide control for applications that do not run under a DB2 plan.



The scalar fullselect or a fullselect that results in a single scalar value to be used where an expression is allowed today. The scalar fullselect can be used in the SELECT clause, in the WHERE clause, or in a CASE expression, for instance. This improves DB2 function, the ability to pert applications and DB2 family compatibility.

A scalar fullselect, as supported in an expression, is a fullselect, enclosed in parentheses, that returns a single row consisting of a single column value. If the fullselect does not return a row, the result of the expression is the null value.

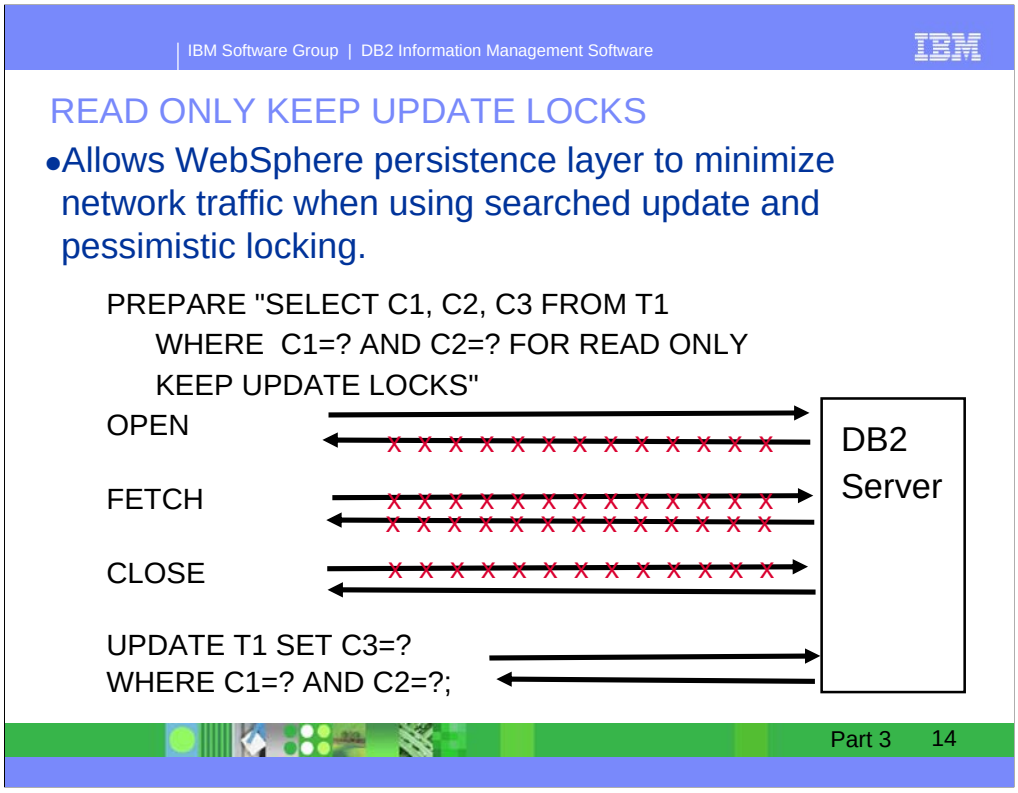
VOLATILE Table Support

- Encourages index access for tables that have unpredictable cardinality
- Significant performance improvement for some SAP applications

```
CREATE TABLE XYZ .....   VOLATILE;
ALTER TABLE XYZ .....    VOLATILE;
```

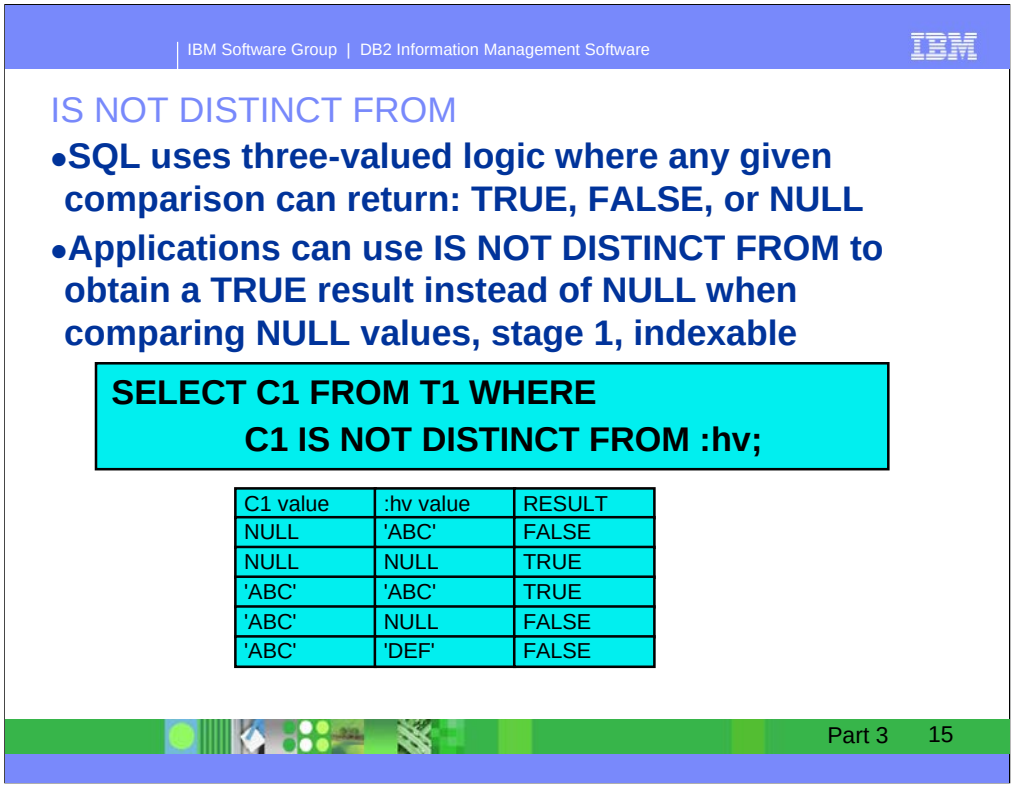

Where applications have tables that are volatile, it is difficult or perhaps impossible to find a good time to gather statistics. Volatile tables have a wide range of cardinality. For example, a table might have no rows part of the time and a million rows at other times.

Customers or vendors can use this option on CREATE TABLE to indicate that the table is volatile, and index access should be encouraged.



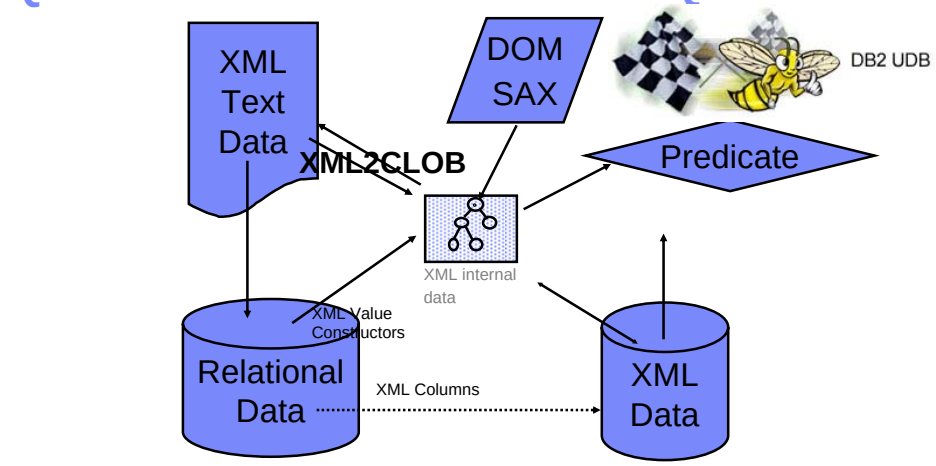
The WebSphere persistence layer currently uses ISOLATION(RS) to retrieve one or more rows with the FOR UPDATE clause when loading the WebSphere entity beans. The Java application is then allowed to perform updates on those beans, and the updates are subsequently sent to DB2 as searched UPDATE statements. WebSphere uses this approach to minimize the number of open cursors at runtime.

With DB2 V8, we will be able to obtain exclusive locks with a FOR READ ONLY query. This will allow the JDBC driver and DDF to use block fetch for the SELECT (eliminating the extra network messages required with a FOR UPDATE query), while still obtaining and holding the locks WebSphere needs for the searched UPDATE statement. This will provide significant CPU and elapsed time improvements.



This is often called the distinct predicate. By definition a null value is unknown and this makes it unequal to all other values, including other null values. The only way to test for null values is to use the IS NULL predicate, as in "WHERE col IS NULL". A predicate of the form "WHERE col = :hv :nullind" will never match a null value in "col", even if the host variable "nullind" contains a null indicator. Unfortunately, this is not intuitively obvious. The IS NOT DISTINCT FROM predicate provides an explicit technique to compare and treat two NULL values as equal.

SQL/XML - XML extensions to SQL



XML value constructors:

XMLELEMENT (xmlattributes), XMLFOREST, XMLCONCAT, XMLAGG, XMLGEN, XMLNAMESPACES

XML Publishing Functions provide a set of SQL built-in functions that allow applications to generate XML data from relational data with high performance. These functions reduce application development efforts in generating XML data for data integration, information exchange, and web services, thus enhance the leadership position of DB2 for z/OS as an enterprise database server.

Transparent ROWID column

- Eliminates the need to explicitly declare a ROWID column in tables that include LOBs
- DB2 will generate a "hidden" ROWID column, which is not visible on SELECT *
- Simplifies porting of LOB applications from other platforms

DB2 UDB for z/OS currently requires a ROWID column to be included in tables which have LOBs. In V8, DB2 will generate a ROWID column if one is not present, and will not include this column in a SELECT *. These changes make application porting simpler with LOBs.

ON COMMIT DROP enhancement

- Declared Temporary Table improvement
- Temp table dropped automatically at COMMIT if no HELD cursors open on the temp table
- Significant improvement for DDF threads that use CMTSTAT=INACTIVE
  - Allows thread to go INACTIVE

This change is a significant change for distributed work that can be switched from active to inactive. Temporary tables that do not have any HELD cursors open are dropped automatically at COMMIT. This will allow the threads to be switched to INACTIVE. The ON COMMIT DROP TABLE clause is added to the DECLARE GLOBAL TEMPORARY TABLE SQL statement. The clause indicates that the declared global temporary table should be implicitly dropped at COMMIT if there are no open cursors on the table that are defined as WITH HOLD.

SQL statements up to 2M bytes

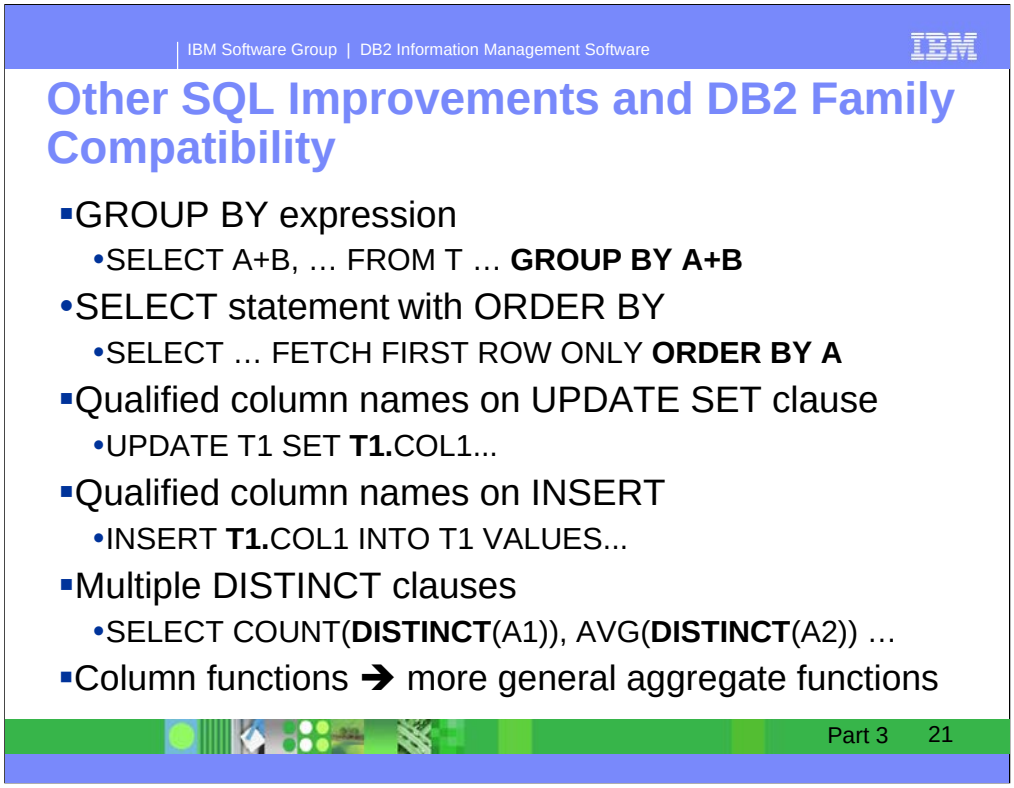
- SQL statements can be up to 2M bytes in length
- Parse tree has been completely rearchitected to reduce SQL too complex SQLCODE -101 scenarios due to SQL statement length
- Important for SQL Procedure Language applications
- Needed for generated SQL, long names, increased numbers of partitions, …

In addition to other limit-breaking support in Version 8, you can now have SQL statements that are up to 2 MB in length. A number of the Version 8 capabilities stretch the limit on the size of an SQL statement. Long names and 4096 partitions take much more space. An SQL Procedure must be completely stated in a single SQL statement, and was limited to 32K. Other changes in DB2 allow much larger structures and thus much larger statements. SQL statements that are too large or too complex should be very unusual, but still possible, with the larger statements.

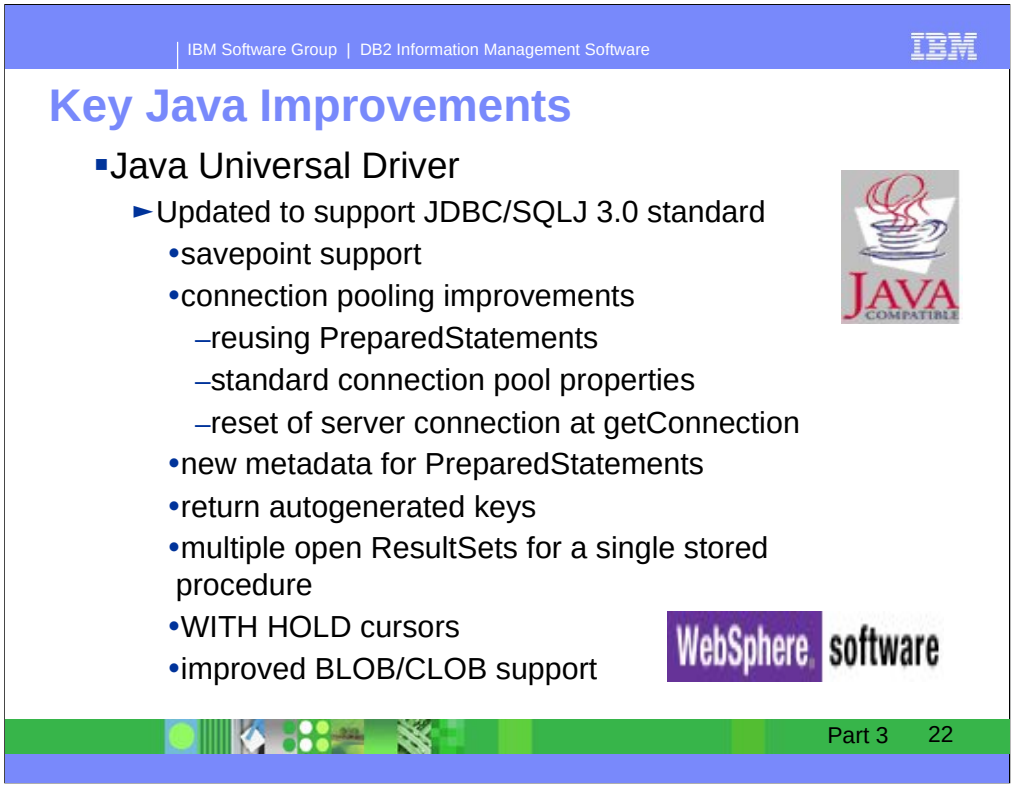
SQL Procedure Language extensions

- RETURN statement
- SIGNAL/RESIGNAL support
- CREATE PROCEDURE up to 2M bytes
- Integrated debugger
- GET DIAGNOSTICS for all fields in SQLCA
- Get values from RETURN
- Enhanced support for labels
- Iterate statement

Several of the changes to SQL are very important for SQL procedures. The limit of 32 kilobytes for an SQL statement was for the entire SQL procedure, and that limit is now 2 megabytes. The ability to return information to the caller of an SQL procedure is improved with the RETURN statement issued from within an SQL procedure. The GET DIAGNOSTICS statement (already valid within SQL procedures) is extended to support returning the status information from a RETURN statement of an SQL procedure (the new RETURN_STATUS keyword). Other DB2 platforms support the RETURN statement and the GET DIAGNOSTICS enhancement within SQL procedures, so these changes improved DB2 family consistency.



We are making big strides in SQL improvements, including: GET DIAGNOSTICS, SEQUENCE, Dynamic scrollable cursors, Scalar fullselect, Multiple DISTINCT clauses, GROUP BY expression, Qualified column names on INSERT and UPDATE SET clause, Unicode for SQL, intermixing EBCDIC, ASCII, or Unicode columns in a single SQL statement and SELECT from an INSERT statement
In addition, we are making major changes in our SQL system limits by extending support for long names in SQL objects to 30 characters for column names and to 128 characters for most other SQL objects; expanding support for SQL statements up to 2 megabytes in length; increasing the length of literals and predicates to 32 kilobytes; and extending JOIN to allow up to 225 tables in a single statement.
www7b.boulder.ibm.com/dmdd/library/techarticle/0209cotner/0209cotner.html



Java support will be more consistent across platforms as we use a single code base across the DB2 family. The improved consistency also adds new function to DB2 for z/OS and improves integration with WebSphere and Java.

We will be supporting both Type 2 and Type 4 Java drivers which will be updated to support the JDBC/SQLJ 3.0 standard, including improvements like savepoints, connection pooling improvements, the ability to reuse PreparedStatements, multiple open ResultSets for a single stored procedure, WITH HOLD cursors, and improved BLOB and CLOB support.

DB2 Family SQL Reference

Version 2 new as of Sept. 22, 2004

IBM DB2 Universal Database

SQL Reference

for Cross Platform Development

z/OS OS/390 OS/400 AIX HP-UX Solaris Linux Windows

SQL Reference book for DB2 UDB family, not just one platform

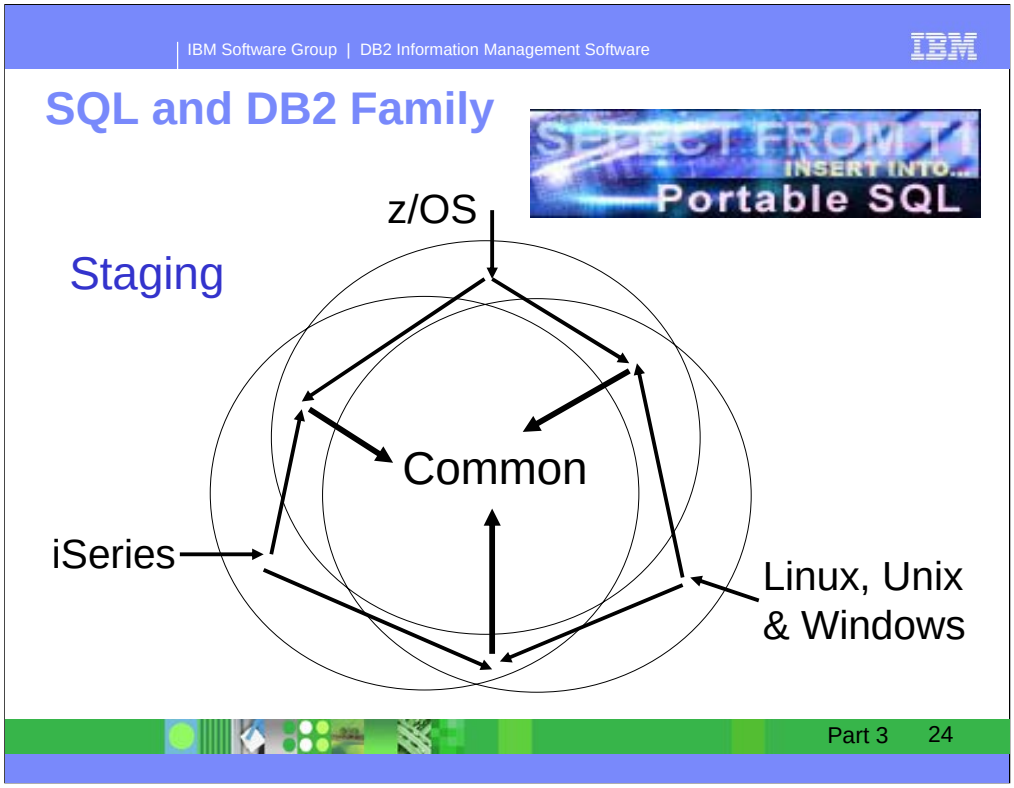
http://www.ibm.com/developerworks/db2/library/techarticle/0206sqlref/0206sqlref.html

Do you want to design or write applications for the entire DB2 family, rather than for just one of the platforms? If so, you need the IBM DB2 Universal Database SQL Reference for Cross-Platform Development. Version 1 & (1.1) covers these product versions:
DB2 UDB for Linux, UNIX & Windows V7 R2 (V8.1)

DB2 UDB for z/OS V7

DB2 UDB for iSeries V5 R1 (V5 R2)
SQL consistency across the DB2 family has improved substantially in the past few versions, while significant new common function has been added. SQL that is common to the DB2 UDB relational database products and the SQL 1999 Core standard is much more comprehensive. The 778 page book describes the rules and limits for

preparing portable programs for these versions. More is coming.
www7b.software.ibm.com/dmdd/library/techarticle/0206sqlref/0206sqlref.html



In the DB2 family, we develop our releases independently, so that when a new version delivers, there is generally staging of new function and common function. Each new release of a product tends to have items that are already implemented on other platforms. This moves function from being implemented by one other platforms to two or from two to the common. Each new release also has some items that are the first delivery of the new function. DB2 for z/OS Version 8 has many items for each type, as do the new releases DB2 for iSeries 5.3 and DB2 for Linux, Unix & Windows 8.2.

DB2 Family SQL

z z/OS V7
common
UW Linux, Unix & Windows V8.2

z

common

Inner and Outer Joins, Table Expressions, Subqueries, GROUP BY, Complex Correlation, Global Temporary Tables, CASE, 100+ Built-in Functions, Limited Fetch, Insensitive Scroll Cursors, UNION Everywhere, MIN/MAX Single Index Support, Self Referencing Updates with Subqueries, Sort Avoidance for ORDER BY, and Row Expressions, Call from trigger, statement isolation

UW

Updateable UNION in Views, ORDER BY/FETCH FIRST in subselects & table expressions, GROUPING SETS, ROLLUP, CUBE, INSTEAD OF TRIGGER, EXCEPT, INTERSECT, 16 Built-in Functions, MERGE, Native SQL Procedure Language, SET CURRENT ISOLATION, BIGINT data type, file reference variables, SELECT FROM UPDATE, DELETE & MERGE, multi-site join, 2M Statement Length, GROUP BY Expression, Sequences, Scalar Fullselect, Materialized Query Tables, Common Table Expressions, Recursive SQL, CURRENT PACKAGE PATH, VOLATILE Tables, Star Join Sparse Index, Qualified Column names, Multiple DISTINCT clauses, ON COMMIT DROP, Transparent ROWID Column, FOR READ ONLY KEEP UPDATE LOCKS, SET CURRENT SCHEMA, Client special registers, long SQL object names, SELECT from INSERT

This text just shows the relationship of DB2 for Linux, Unix & Windows with DB2 for z/OS and OS/390 Version 7, comparing a March 2001 z/OS version with an October 2004 LUW version. The next step in the process is DB2 UDB for z/OS Version 8 and then DB2 UDB for Linux, Unix & Windows V8.2. There are three sets of SQL noted above, with none that is unique to DB2 UDB for z/OS in the first group, SQL that is common across DB2 UDB for Linux, Unix, Windows and z/OS in the small group in the middle, then SQL that is unique to DB2 UDB for Linux, Unix and Windows in the bottom group. Sheryl Larsen provided the base for this information.

DB2 Family SQL

z z/OS V8
common
UW Linux, Unix & Windows 8.2

z Multi-row INSERT, FETCH & multi-row cursor UPDATE, Dynamic Scrollable Cursors, GET DIAGNOSTICS, Enhanced UNICODE for SQL, join across encoding schemes, IS NOT DISTINCT FROM, Session variables

common Inner and Outer Joins, Table Expressions, Subqueries, GROUP BY, Complex Correlation, Global Temporary Tables, CASE, 100+ Built-in Functions including SQL/XML, Limited Fetch, Insensitive Scroll Cursors, UNION Everywhere, MIN/MAX Single Index Support, Self Referencing Updates with Subqueries, Sort Avoidance for ORDER BY, and Row Expressions, 2M Statement Length, GROUP BY Expression, Sequences, Scalar Fullselect, Materialized Query Tables, Common Table Expressions, Recursive SQL, CURRENT PACKAGE PATH, VOLATILE Tables, Star Join Sparse Index, Qualified Column names, Multiple DISTINCT clauses, ON COMMIT DROP, Transparent ROWID Column, Call from trigger, statement isolation, FOR READ ONLY KEEP UPDATE LOCKS, SET CURRENT SCHEMA, Client special registers, long SQL object names, SELECT from INSERT

UW Updateable UNION in Views, ORDER BY/FETCH FIRST in subselects & table expressions, GROUPING SETS, ROLLUP, CUBE, INSTEAD OF TRIGGER, EXCEPT, INTERSECT, 16 Built-in Functions, MERGE, Native SQL Procedure Language, SET CURRENT ISOLATION, BIGINT data type, file reference variables, SELECT FROM UPDATE, DELETE & MERGE, multi-site join

This text just shows the changed relationship of DB2 for Linux, Unix & Windows with DB2 for z/OS Version 8, comparing the z/OS version from March 2004 with the LUW version from October 2004. There are three sets of SQL noted above, as in the prior foil, with some that is unique to DB2 UDB for z/OS in the first group, SQL that is common across DB2 UDB for Linux, Unix, Windows and z/OS in the large group in the middle, then SQL that is unique to DB2 UDB for Linux, Unix and Windows in the bottom group. Sheryl Larsen provided the base for this information.

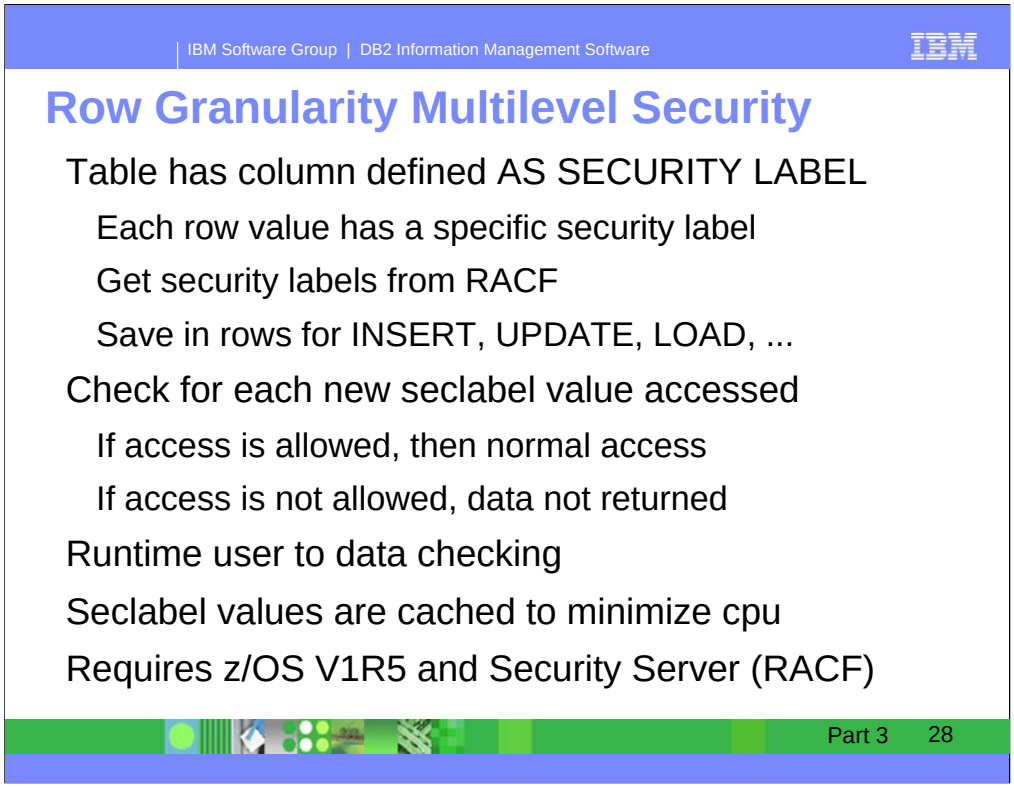
If you want to improve DB2 family consistency, then DB2 UDB for z/OS Version 8 is a big step, changing the game from one of catch up to one of leapfrog. Now we are helping our family members to leap ahead even as we work on our next leap.

Improved Security

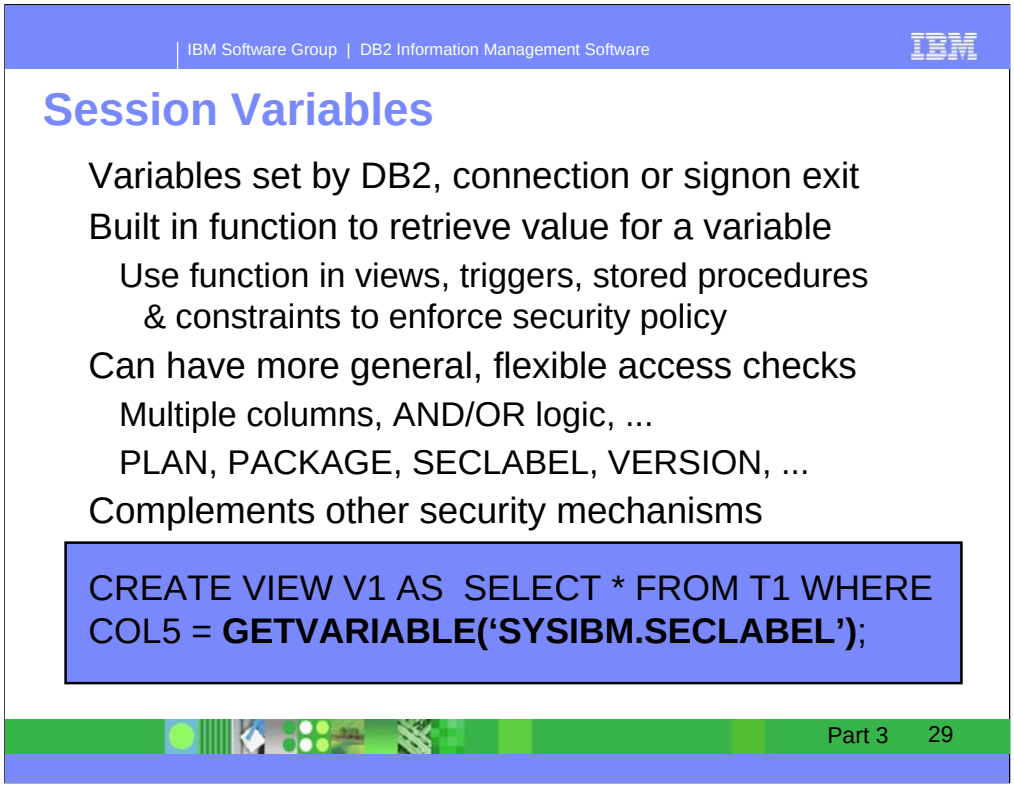
- Multilevel security with row level granularity
- Session variables
- Special registers for identification
- RACF access control & DB2 operator commands
- Encryption and decryption built-in functions
- Encryption for DRDA on the wire

Everyone seems to be more aware of security today. Improving integration and making security more robust and easier to manage are very important.

Customers asked for a wide range of enhancements for security. DB2 V8 provides new options for higher security, more granularity, and more information for additional flexibility in applications and SQL. Improvements include improvement for RACF access control, encryption and decryption built-in functions and encryption for data on the DRDA wire.



We have had many requests for row-level security for applications that need more granular security schemes. For example, in organizational hierarchies, it is desirable to set up a hierarchy in which employees can see their own payroll data, a first line manager can see his or her payroll information and all of the employees reporting to that manager, and so on. In addition, government security schemes often include a security hierarchy such as TOP SECRET, SECRET, or UNCLASSIFIED. We are currently developing support for row-level security that includes support for these types of hierarchical security schemes. You can use this support by adding a column that acts as the security label.



Session Variables provide another way to provide information to applications. Some variables will be set by DB2. Others can be set in the connection and signon exits to set these session variables

A new built-in function GETVARIABLE is added to retrieve the values of a session variable. This function can be used in views, triggers, stored procedures and constraints to help enforce a security policy. If your primary security need is more general, flexible controls, this information complements other security mechanisms.

For example, you can have a view which provides data that is at the current security label.

New Special Registers

Client information for this connection

Provided by sqleseti, Java methods, RRS SIGNON & SET_CLIENT_ID

- CLIENT_ACCTNG accounting string
- CLIENT_APPLNAME value of application name
- CLIENT_USERID client user ID
- CLIENT_WRKSTNNAME workstation name

Four new SPECIAL REGISTERS are added to the product. These special registers are CLIENT_ACCTNG, CLIENT_APPLNAME, CLIENT_USERID, and CLIENT_WRKSTNNAME. The information is provided through a number of application programming interfaces.

These special registers were added to DB2 for Linux, UNIX & Windows V8.

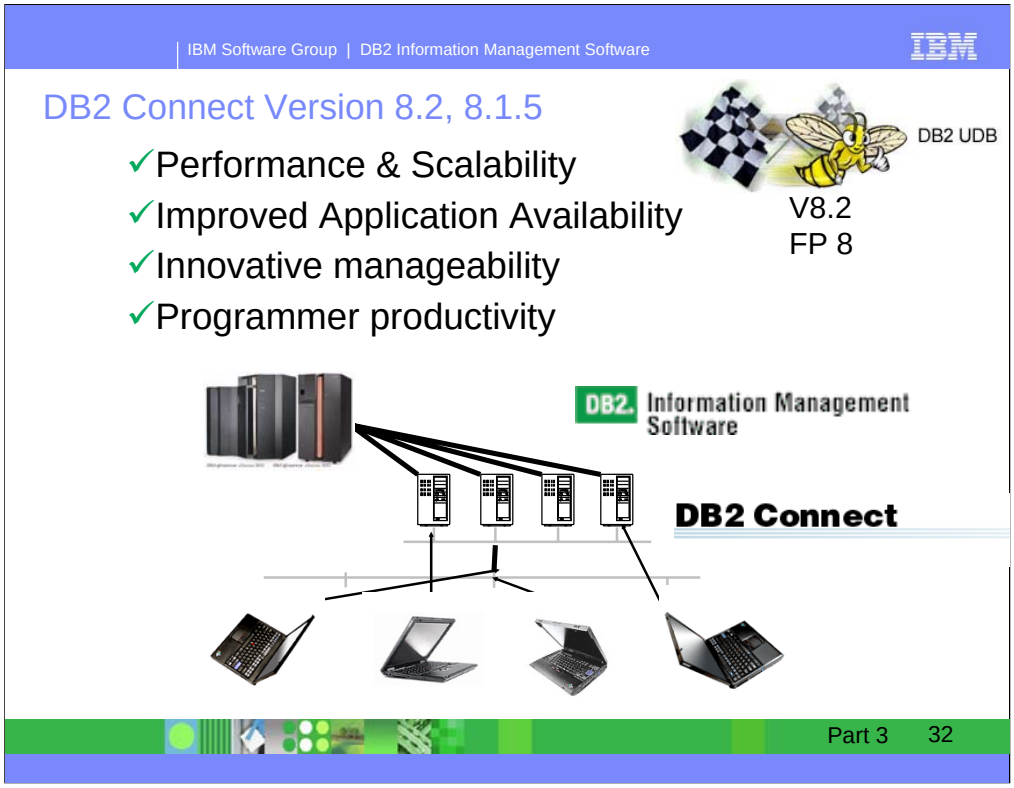
New Built-in Functions

- ENCRYPT_TDES: encrypt column in a table with a user-provided encryption password
- ENCRYPTION PASSWORD special register
- DECRYPT_BIT, DECRYPT_CHAR, DECRYPT_DB
- GET_HINT: obtain hint to help remember ENCRYPTION PASSWORD
- GENERATE_UNIQUE creates CHAR(13) FOR BIT DATA value that is unique across Sysplex
- DRDA encryption on the wire

Functions ENCRYPT_TDES (triple DES), DECRYPT_BIN, DECRYPT_CHAR, and GETHINT are added. The SET ENCRYPTION PASSWORD statement allows the application to specify a password

The ability to generate a unique value is also included. These changes came in DB2 for Linux, UNIX and Windows V8, so this change improves DB2 family consistency.

DRDA is extended to allow encryption of the data being sent.



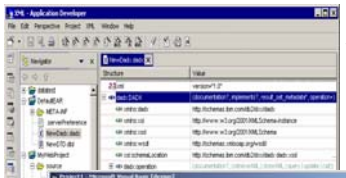
DB2 Connect improvements are noted in announcements and in the What's New in DB2 UDB Version 8.1? book.
ibm.com/software/data/db2/udb/pdfs/db2q0.pdf

Plan to use V8.1.5 (fixpak 5) or later for DB2 Connect. If you want the DRDA encryption support, then you need V8.2 (fixpak 8 preferred, 7A) or later. Version 7.2 end of service was September 2004 and then extended for customers moving to V8. While DB2 V8 works with DB2 Connect V7.2 (fixpak 10, 11 or 12), most of the improvements only deliver with V8.1.4 or fixpak 4 or later. I recommend adding the fixes from fixpak 5 at a minimum. Also see the web:
ibm.com/software/data/db2/db2connect/
http://www.ibm.com/software/data/db2/udb/support/downloadv8.html

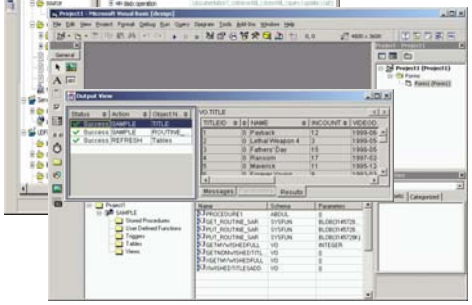
Productivity for Application Development

Developer Productivity for Both Worlds: Java & .NET

- **Flexible Access Style**
  - SQL
  - JDBC, Java Beans, SQLJ
  - ADO, ODBC, OLE
  - Web Services
- **Seamless Integration with Integrated Development Environments**
  - Java…WebSphere Studio
  - .NET…Visual Studio .NET
  - DB2…Development Center



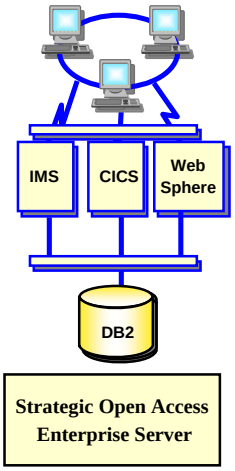
WebSphere Studio



Microsoft Visual Studio .NET

DB2 works with application development to meet a wide variety of application needs, using varied languages, varied vendors, and varied application development styles. Whether you choose Java or Microsoft .NET, regardless of the access style, DB2 database access must be seamless. Programmers will use their integrated development environment, language, access style, and still interact easily with DB2. DB2 provides plug-ins to the IDEs.

Network Computing Improvements (1 of 2)

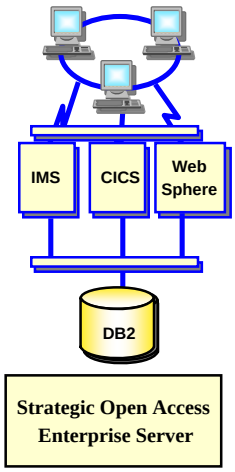


- Application Connectivity for JAVA
- Improve data sharing member routing
- LOCATION aliases at requester & server
- Extended function, standards
- Reduced processing
- Stored Procedure Enhancements
- Java Universal Driver

DB2 V8 also provides many improvements for network computing: security, availability, usability and performance. The improved data sharing member routine is more robust. Having the ability to use multiple names for a server adds flexibility. Increased levels of standards are implemented. Performance improvements will reduce cpu costs substantially. Improved granularity for stored procedures help with managing your work load, using the WLM.

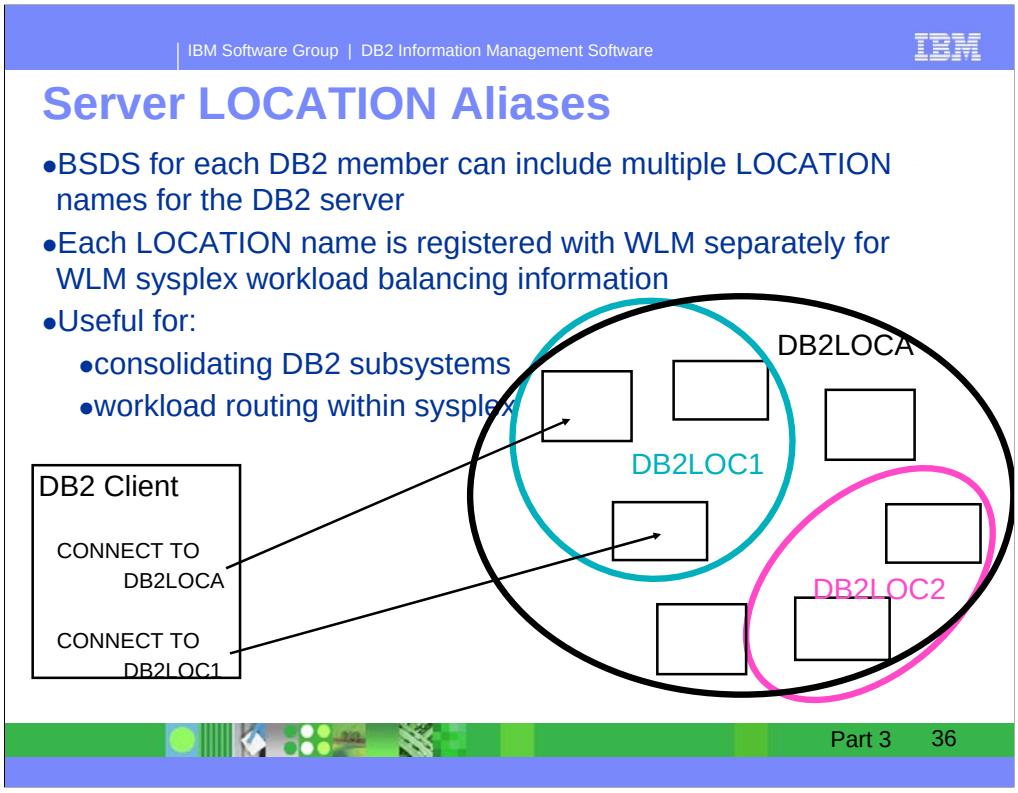
The Java Universal Driver is used across the DB2 family.

Network Computing Improvements (2 of 2)



- Encrypt data flowing on the wire
- CDB improvements for data sharing member routing
- LOCATION aliases at requester and server (e.g. CONNECT)
- Extended DESCRIBE for CLI & Java
- LOB streaming
- SQL Cancel
- Auto-close for WITH HOLD cursors
- Reduced block FETCH CPU cost
- Reduced TCP/IP CPU cost
- Stored Procedure Enhancements
- Java Universal Driver

Version 8 also includes many improvements for network computing: security, availability, usability and performance. Performance improvements will reduce cpu costs substantially.



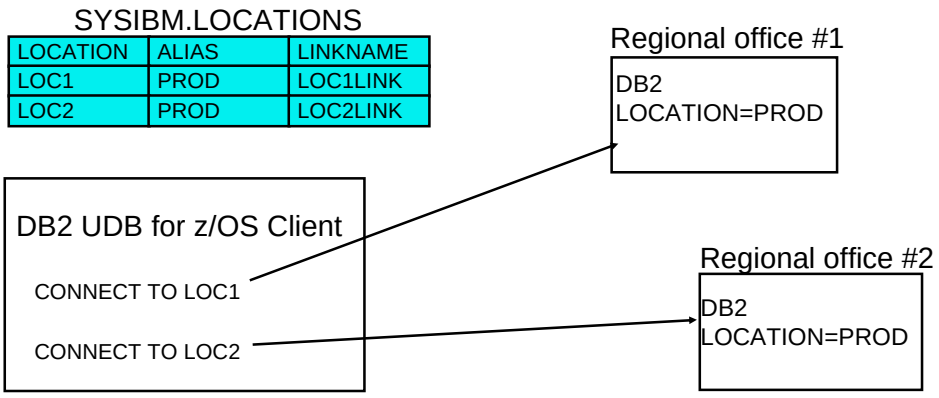
The new server location alias capability allows you to assign multiple LOCATION names to a give DB2 data sharing group, and have those LOCATION names managed separately by WLM for workload routing purposes.

As an example, you might want to have one LOCATION name for handling SQL requests from CICS, and another LOCATION for ad-hoc query users. This gives you the ability to dedicate a subset of the DB2 data sharing members to a specific set of clients (e.g. DB2LOC1 consists of 2 members processing CICS requests, DB2LOC2 consists of 2 members for ad-hoc query, etc.).

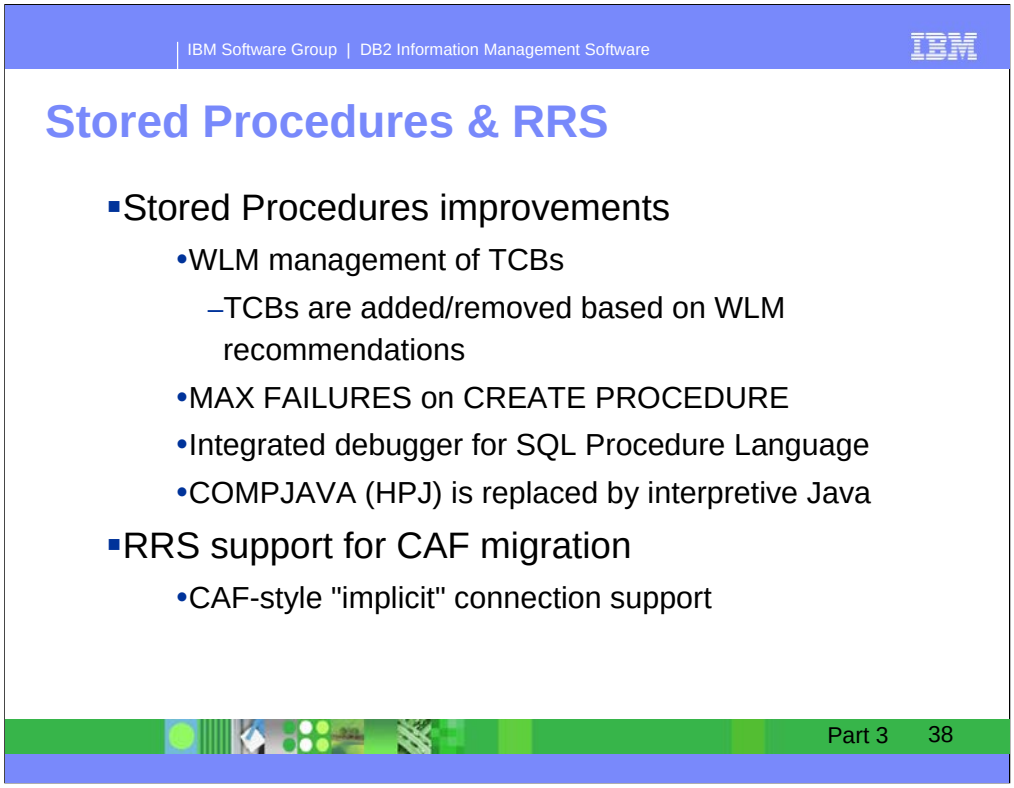
This capability is also useful when combining multiple DB2 subsystems into a single data sharing group, since it allows applications to continue to connect to DB2 using whatever DB2 LOCATION name was used previously.

Client LOCATION Aliases

- CDB is enhanced so client LOCATION name does not have to match remote server's LOCATION name



Client LOCATION alias support allows you to refer a remote DB2 server by more than one LOCATION name. This can be particularly useful when accessing regional DB2 systems that are all configured with the same LOCATION name.



z/OS Workload Manager function can determine appropriate resource utilization and provide a method of changing the number of tasks within a stored procedures address space.

Customers can set a limit on how many times each stored procedure or user defined function can fail before it is stopped. We already allow the customer to specify a max abend value for all stored procedures and user defined functions on a single DB2 image.

DB2 for z/OS Version 8 will no longer support LANGUAGE COMPJAVA stored procedures.

Applications that have been written to use CAF implicit connect can be easily converted to RRSAF.

Automatic Storage Management

- No need to specify space parameters
- Start with small disk space
  - Start with small secondary extents
  - Larger secondary extents as table grows
- Improves DBA productivity
- Avoids application outage
- Reduces need to reorganize

Currently, database administrators need to specify primary and secondary space quantity. Then they need to monitor the data set sizes and extents to avoid an outage. Since the secondary extents are a fixed size, they must make the tradeoff between wasted space for many small tables and not having adequate space for rapidly growing tables. We can manage the extent size based upon the growth pattern of the table or index.

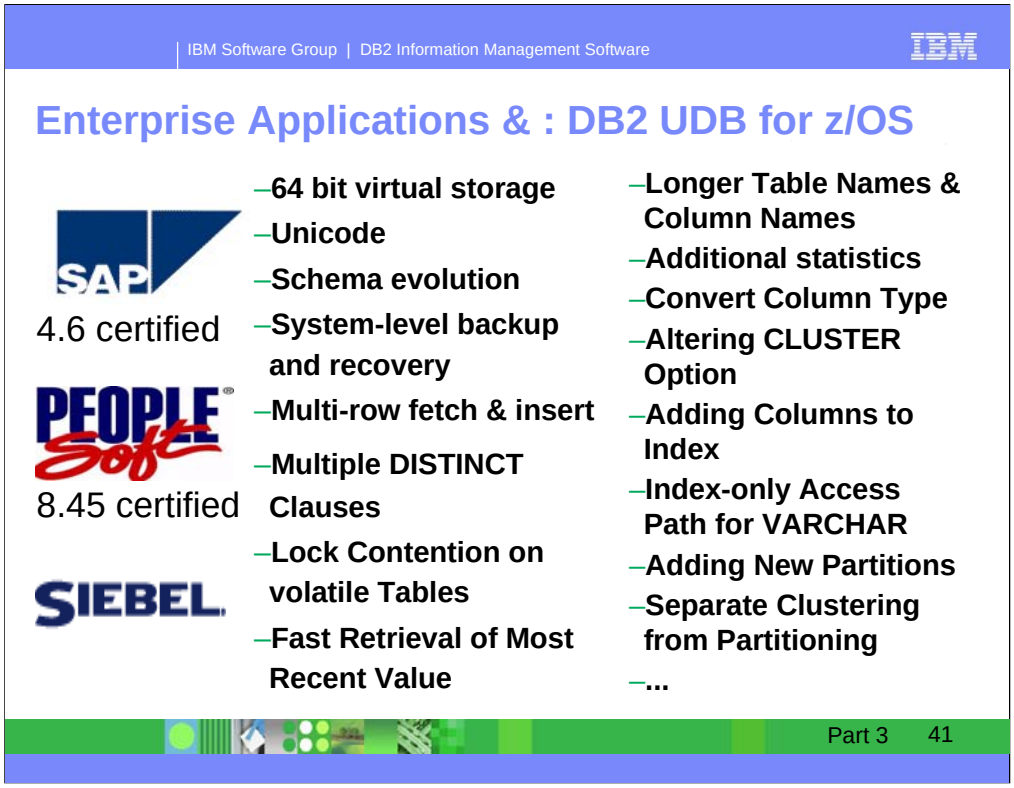
Unicode Enhancements in Version 8

- Able to join Unicode with EBCDIC or ASCII
- SQL in Unicode, statements, literals, object names, mixed CCSIDs
- Most DB2 character catalog columns in Unicode
- Collating sequence change
- Lengths and maximum lengths can change

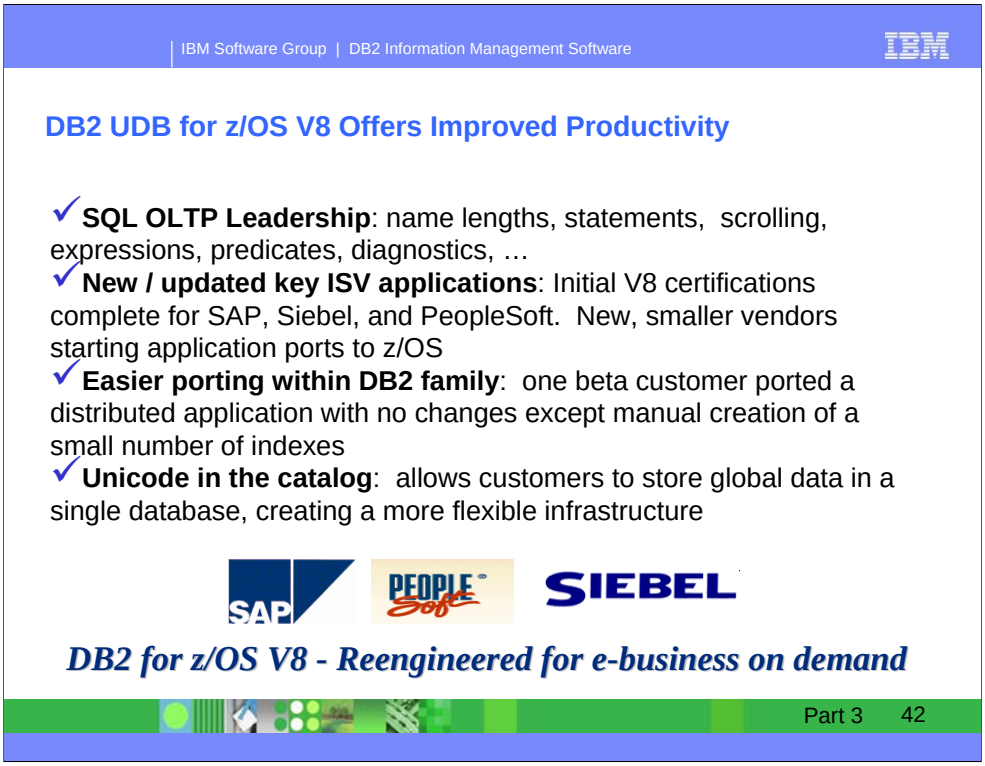
www.unicode.org

The key improvements for Unicode allow much more flexibility, with the ability to join a Unicode table to one that is ASCII or EBCDIC. SQL statements and literals can be Unicode or EBCDIC. Many of the DB2 catalog character columns will be converted to Unicode, so Unicode is for everyone. SQL is converted to Unicode before parsing to allow code-page dependent parsing.

The collating sequence for Unicode is similar to ASCII, with numbers sorting lower than letters. The lengths and maximum lengths of strings can change as they are converted to Unicode, so treat lengths as variable.



Enterprise Resource Management (ERM) includes Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM). Most of the key items in this version help our key enterprise application partners: SAP, PeopleSoft and Siebel, but also many other applications and customers. The 2 GB address space limit is a major problem for customers who have large dynamic statement cache, many concurrently open data sets and long running units of work. Unicode helps vendors support multinational companies and improve character conversions. These applications often insert and fetch many rows. Using a technique called array fetch & insert or multirow fetch & insert can be as much as 50% less cpu time locally, but improves remote access more. SQL flexibility improvements allow DB2 to be efficient in performance and in productivity for our partners. There are roughly 50 items, with some differences, for each partner. See the book DB2 for z/OS V8: Through the Looking Glass & What SAP Found, SG24-7088 for more detail. SAP R/3 4.6 and PeopleSoft PeopleTools 8.45 were certified for V8, less than four months after general availability, and more certifications are expected.



DB2 UDB for z/OS V8 has been re-engineered for e-business on demand, with many fundamental changes in architecture and structure. Key improvements enhance scalability, application porting, security, architecture, and continuous availability. Management for very large databases is made much easier, while 64-bit virtual storage support makes management simpler and improves scalability and availability. This new version breaks through many old limitations in the definition of DB2 objects, including SQL improvements, online schema evolution, longer names for tables and columns, longer SQL statements, enhanced Java and Unicode support, enhanced utilities, more log data sets, more partitions, and many more advantages.

Version 8 includes dozens of changes in SQL, improving family consistency in many cases, playing leapfrog in others, pushing DB2 SQL beyond current boundaries for enhanced application portability & open standards. These changes improve our customer productivity, consistency across the DB2 family and ability to port applications.

Version 8 brings new synergy with the zSeries hardware, for example using z/OS 64-bit virtual addressing capabilities. Many enhancements improve our integration with key vendor applications like PeopleSoft, SAP & Siebel