Character Related Features in ISO/ANSI SQL

Dave Birdsall

3/29/01

H2-2001-109

Purpose of This Presentation

- Part of H2 liaison to L2/UTC
- Gives an overview of character set features in standard SQL as a basis for further discussion

Outline

- Description of character set features as they have evolved in the standard
- Some issues raised against SQL-99
- Open discussion

SQL-86

- Document reference: X3.135-1986
- CHAR datatype only -- fixed length strings
- Just one character set; it's implementor-defined
- Can perform assignments, comparisons and pattern matching
- No string manipulation functions

SQL-86 Assignment Semantics

- When assigning a shorter string to a longer one, the shorter string is padded with <space>s
- Assigning a longer string to a shorter column is not allowed
- Assigning a longer string to a shorter host variable results in truncation, with the original length stored in the indicator if one was supplied

SQL-86 Comparison Semantics

- Character strings are compared by:
  - effectively padding the shorter string with <space>s to get two strings of equal length

- comparing each ordinal position from first to last, stopping at the first mismatch (if any)
- result of comparison is result of first mismatch (if any) or equality (if no mismatch)
- collating sequence of individual characters is implementor-defined

SQL-86 Comparison

- Comparison occurs “under the covers” for such SQL operators as DISTINCT, GROUP BY, ORDER BY, MIN(), MAX()

SQL-86 Pattern Matching

- LIKE predicate: X LIKE ‘%xyz_’
- % matches arbitrary substrings
- _ matches arbitrary single characters
- everything other than % and _ matches itself (without <space> padding)
- so X = Y can be true while X LIKE Y can be false (e.g. if X is ‘a’ and Y is ‘a ’)

SQL-92

- Added support for multiple character sets
- Added support for alternative collating sequences
- Added string manipulation functions
- Added CHARACTER VARYING (varying length character string) datatype

SQL-92 Some Motivations

- Many vendors already supported dual character sets
- There was no “universal character set” at the time
- Ironically, because of the self-describing nature of SQL metadata, SQL-92 had to invent a “local universal character set” (my term), i.e. SQL_TEXT

SQL-92 Character Set Notions

- Character repertoire -- a “set” (in the mathematical sense) of characters apart from any encoding

- Form-of-use -- an encoding of a character repetoire
- Character set -- a character repertoire under a specific form-of-use

SQL-92 Character Set Notions II

- SQL_TEXT -- an implementation-defined character set that can represent every SQL language character and all characters in any character set supported by the implementation
- Translation -- allows character strings from one character set to be converted to character strings of another character set

SQL-92 Character Set Notions III

- CHAR and CHAR VARYING data types now have a character set attribute
- NATIONAL CHAR is a syntactic sugar for CHAR CHARACTER SET <a second implementation-defined character set>
- Two character strings may be compared only if they have the same character set

SQL-92 Collations

- “Collation” = “collating sequence”
- A collation defines an ordering of the set of character strings on a particular character repertoire (not character set)
- For every character set, there is a default collation

SQL-92 Collations II

- Collations affect any operation on character strings that involves order or character matching
- Examples: comparison, LIKE, DISTINCT, GROUP BY, ORDER BY, UNION (duplicate elimination), MIN, MAX

SQL-92 Comparison Semantics

- Comparison semantics are encapsulated in the collation
- A collation may use blank padding or no padding
- Two non-identical strings may compare as equal
- Duplicate elimination picks one in an implementation-dependent

fashion

SQL-92 String Manipulation

- concatenation (||)
- SUBSTRING
- POSITION
- TRIM
- UPPER/LOWER

SQL-92 Other Functions on Strings

- CHARACTER_LENGTH
- OCTET_LENGTH

SQL-92 Functions Returning Strings

- CURRENT_USER
- SESSION_USER
- SYSTEM_USER
- These are defined as returning character strings in the SQL_TEXT character set

SQL-92 Assignment Semantics

- Assigning a longer string to a shorter host variable now results in a warning diagnostic as well as truncation
- Assigning a longer string to a shorter column is OK as long as the characters truncated are <space>s; an exception occurs otherwise

SQL-99

- Redefined many concepts in terms of Unicode 2.0
- Added a few more string manipulation functions
- Added character large objects (CLOBs)
- Defined several additional character sets

SQL-99 String Manipulation

- UPPER/LOWER redefined in terms of Unicode
- New OVERLAY string manipulation function added
- New SIMILAR regular expression pattern matching predicate

added

SQL-99 CLOBs

- A CLOB is essentially just a long character string
- Not all of the normal character string semantics are supported for CLOBs (presumably because of size difficulties)
  - examples: no GROUP BY, ORDER BY, unique constraints, join predicates

SQL-99 Additional Character Sets

- SQL_CHARACTER -- the 88 characters that appear in SQL syntax (apart from identifiers and string literals); a subset of ISO 646:1991
- GRAPHIC_IRV -- SQL_CHARACTER + seven more characters from ISO 646
- LATIN1 -- the 191 graphic characters from ISO 8859-1

SQL-99 Additional Character Sets II

- ISO8BIT -- all of ISO 8859-1 except the character encoded as 0 (255 characters)
- UTF16 -- Unicode 2.0/ISO IEC 10646:1993, using the UTF-16 encoding
- UTF8 -- Unicode 2.0/ISO 10646, using UTF-8 encoding
- UCS2 -- Unicode 2.0/ISO 10646, using the UCS2 encoding

SQL-99 Additional Character Sets III

- SQL_TEXT -- as in SQL-92; a superset of SQL_CHARACTER; may include implementation-defined additional characters
- SQL_IDENTIFIER -- all the characters that an implementation supports in identifiers; a superset of SQL_CHARACTER and either a subset or all of SQL_TEXT

SQL-99 Additional Character Sets IV

- To reiterate: SQL_CHARACTER, GRAPHIC_IRV, LATIN1, ISO8BIT, UTF16, UTF8 and UCS2 are precise sets of characters

- SQL_TEXT and SQL_IDENTIFIER may include additional characters (and are implementation-defined)

SQL-99 Conformance

- A conforming implementation need only support character set names if it claims to conform to feature F451, “Character set definition” and Feature F461, “Named character sets”

SQL-99 Identifiers

- Identifiers redefined in terms of characters having Unicode properties (e.g. “alphabetic”, “ideographic”)
- Case folding rules are redefined in terms of Unicode notions of lower, title and upper case

SQL-99 Meta-Data

- Identifiers appearing in meta-data are defined as having the character set SQL_IDENTIFIER
- General character data (e.g. constraint text) in meta-data are defined as having the character set SQL_TEXT
- The other named character sets don’t appear much in the standard beyond their definition

SQL-99 Tweaks

- CURRENT_USER etc. now return strings of character set SQL_IDENTIFIER (instead of SQL_TEXT)

Some Issues Raised

- Issues raised by L2/UTC (for which hopefully more specifics will be forthcoming as a result of this presentation)
- Issues raised by J. M. Sykes in ISO DBL
- There’s some overlap, so I’ll just list the issues without attribution on next slides

Some Issues Raised II

- Bring SQL up-to-date (e.g. to reference Unicode 3.1 instead of 2.0, use more recent terminology)

- Character model: Should SQL character model be changed? (e.g. perhaps a SQL character should correspond to a Unicode code point)

Some Issues Raised III

- Identifier case folding rules OK?
- Today standard SQL knows nothing of forms of normalization in Unicode. Should this change? What forms of normalization would be appropriate for SQL to process?
- Other issues?

Open Discussion

- Looking for specifics from L2/UTC

Conclusion

- Have summarized the evolution of character-related features in standard SQL
- Have noted some issues raised against SQL-99
- Have invited discussion, looking for specifics