INternational Committee for
Information Technology Standards (INCITS)
INCITS H2 (Database)

INITIAL PROJECT PROPOSAL
AND SKELETON DRAFT DOCUMENT

SQL/RPR — Row Pattern Recognition
With Application to Streaming Data Queries

H2-2008-027

New Project Proposal
for
Row Pattern Recognition – Amendment to SQL
with
Application to Streaming Data Queries

Title: Information technology – Database Languages – SQL - Row Pattern Recognition (SQL/RPR)

Status: For action at 9 January 2008 H2 teleconference meeting.

Abstract: This proposed project will produce a standard that provides functionality for finding patterns in sequences of rows, possibly including language enhancements (syntax and semantics) that support SQL processing of streaming data. Patterns are defined using familiar syntax of Regular Expressions (RE). RE variables span sub-sequences of rows and are defined using conditions on individual rows and their aggregates. Such pattern recognition is especially valuable when querying streaming data. If consensus is achieved in a timely manner, the standard may include additional functionality associated with streaming data.

References:

[FrameWD] INCITS H2-2007-181/WG3:STR-017, *(Working Draft) Framework for SQL (SQL/Framework)*, September, 2007

[FoundWD] INCITS H2-2007-182/WG3:STR-018, *(Working Draft) Foundation for SQL (SQL/Foundation)*, September, 2007

[Friedl] Jeffrey E.F. Friedl, "Mastering regular expressions", 1997, O'Reilley and Associates

[Proposal] INCITS H2-2007-230, Subproject Proposal for Row Pattern Recognition Amendment with Application to Streaming Data Queries, Jim Melton et. al., October 2007.

[Sadri Ph.D.] M. Reza Sadri, "Optimization of sequence queries in database systems", Ph.D. thesis, UCLA, 2001, http://wis.cs.ucla.edu/theses/thesis_sadri.pdf

[Sadri pods] Reza Sadri et.al., "Optimization of sequence queries in database systems", http://www.cs.ucla.edu/%7Ezaniolo/papers/pods2001.pdf

[Sadri tods] Reza Sadri et.al., "Expressing and optimizing sequence queries in database systems", ACM Transactions on database systems, Vol 29, No. 2, June 2004, pages 282-318, http://cs.ucla.edu/~zaniolo/papers/todsjune04.pdf

[Streambase] Streambase Systems, "Pattern matching for StreamSQL", October 2006

[SQL 2008] ISO/IEC 9075-Parts 1,2,3,4,9,10,11,13 14:2008 (Currently out for FDIS ballot, expected to be approved by JTC 1 and adopted as INCITS/ANS standards within the year.)

[WLG-027] INCITS H2-2005-292r2 = ISO/IEC JTC1/SC32 WG3:WLG-027, *XQuery regular expression support in SQL/Foundation*, Fred Zemke, Weiran Zhang

[Note] The Proposal Abstract above comments on the possibility of including in the Standard additional specifications related to querying streaming data. The present paper does not include specific text (*e.g.*, definitions or existing practice) related to those additional specifications pending achievement of consensus. All necessary justifications and definitions will be provided if/when that consensus is reached before the decision is made to progress this Standard for formal processing.

1. Source of the Proposed Project

1.1. Title

Information technology – Database Languages – SQL - Row Pattern Recognition (SQL/RPR)

1.2. Date Submitted

10 January 2008

1.3. Proposer(s)

INCITS H2 Database

2. Process Description for the Proposed Project

2.1. Project Type (Development or Revision)

D – DEVELOPMENT

2.2. Type of Document

National Standard

2.3. Definitions of Concepts and Special Terms

ROW PATTERN MATCHING: The act of identifying a submultiset of rows within a multiset of rows (*e.g.*, a table) that together match a specified pattern.

ROW PATTERN: A form of Regular Expression.

PARTITION: A localized SQL group of rows within which row pattern matching is performed.

MEASURE: An expression that is either a scalar expression applied to a single row, or an aggregate function applied to a group of rows. Defines the values that can be returned from a row pattern matching query.

MAXIMAL MATCH: A style of matching that determines at most one match of the specified pattern, starting with a selected row of the multiset of rows.

INCREMENTAL MATCH: A style of matching that determines any number of matches of the specified pattern, starting with a selected row of the multiset of rows; the result of the query might contain multiple results that involve the same starting row.

2.4. Expected Relationship with Approved Reference Models, Frameworks, Architectures, etc.

The proposed standard is closely tied to the SQL/relational data model; it will extend the SQL Standard to support Row Pattern Recognition.

2.5. Recommended INCITS Development Technical Committee (Existing or New)

INCITS H2, Database, is the recommended INCITS Development Technical Committee because of its existing responsibility for all parts of the SQL standard.

2.6. Anticipated Frequency and Duration of Meetings

INCITS H2 meets approximately five to eight times per year, two or three of which are face-to-face meetings, the rest being electronic (teleconference) meetings.

2.7. Target Date for Initial Public Review (Milestone 4)

An initial public review is planned for Q4 2008.

2.8. Estimated Useful Life of Standard or Technical Report

The proposed standard, possibly with one or more revisions, should have a useful life exceeding ten years.

3. Business Case for Developing the Proposed Standard or Technical Report

3.1. Description

This standard will specify the syntax and semantics of a new SQL capability to perform complex queries involving the relationships between many rows in a single (virtual or base) table. Detection and use of such relationships are critical aspects of many high-value applications. Sometimes called *complex event processing*, many business processes are driven from sequences of events. For example, security applications require the ability to detect unusual behavior definable with regular expressions. Financial applications that detect stock patterns are widely demanded. Fraud detection applications must recognize patterns in financial and other transactions. RFID processing requires the ability to recognize valid paths for RFID tags.

Extremely high interest in these capabilities has been shown by financial institutions, by the USA Department of Homeland Security and other government agencies, by large retailers and their suppliers, and transportation companies, among others.

Upon approval of the national SQL/RPR standard, which will be written as an amendment to Part 2 Foundation [FoundFDIS] of [SQL 2008], H2 expects to submit it to ISO/IEC JTC 1 possibly using the Fast-Track process with proposed maintenance in JTC 1/SC32 if approved.

3.2. Existing Practice and the Need for a Standard

Existing practice in row pattern recognition is currently limited to a very small number of database management systems with proprietary solutions, a larger number of third-party software vendors who layer the capability on top of a relational database management system, and by a much larger number of customer organizations who are forced to “roll their own” solutions.

A standard for row pattern recognition in the context of relational database management systems would find enthusiastic and immediate market interest. Mere standardization of the syntax and semantics would permit both relational database system implementers *and* third-party software

suppliers to provide the capability in a uniform manner that gives customers more freedom of choice and frees them from choosing among incompatible proprietary solutions to this problem.

3.3. *Implementation Impacts of the Proposed Standard*

3.3.1. Development Costs

Development of the proposed standard is anticipated to occupy perhaps 5 to 8 standardizers, operating at various levels of intensity (5% to 10% of their work time) over a period of 12 to 18 months. H2 may establish Task Groups and/or ad hoc groups to focus on SQL/RPR. Meetings of these subgroups (if any) will be almost exclusively in the cyberspace domain (that is, they will meet by telephone and perhaps web conferences). Consideration for formal proposals will occur at regularly scheduled meetings of INCITS H2, some of which will be face-to-face meetings.

No significant new costs are expected for H2 *per se*, but there will be time costs for those working on this new project. An Editor has been identified who is willing to accept complete responsibility for the editing of the proposed amendment.

3.3.2. Impact on Existing or Potential Markets

Contrasting with the relatively minor costs to develop the proposed standard, the economic benefit to various communities —government, financial, transportation, security, and many others — are expected to be significant. Those communities will be able to build applications that take advantage of patterns in their relational data that they may have suspected but could not readily identify, or patterns that they currently recognize only at great cost (e.g., through the development of in-house software to do the job) or by locking themselves into proprietary solutions.

Hard data regarding the size of the marketplace that is targeted by this proposed standard is remarkably difficult to determine. Financial institutions are reluctant to release information about the number and size of fraudulent transactions. Stock trading institutions prefer to retain supremacy over successful stock trading algorithms. Governments keep secret their methods of detecting nefarious activities as a matter of national security (or, more cynically, to limit the ability of voters to recognize malfeasance within government).

Nonetheless, it is widely agreed that the costs of bank, insurance, and stock fraud is many billions of US dollars per year, that the ability to improve traffic flows and freight carrier routing improves returns measured in millions of hours and hundreds of millions of dollars, and that failure to recognize threats to national security has the potential to cost billions of dollars and thousands of lives. As a result, many industries and other communities are anxious to have pattern recognition capabilities. The value in increased sales of relational database and third-party software products is likely to be in the millions (if not tens of millions) of dollars per year.

3.3.3. Costs and Methods for Conformity Assessment

There are no conformity assessment efforts expected to arise in conjunction with the proposed standard. Therefore, there are no costs or methods associated with this subject.

3.3.4. Return on Investment

It is difficult to estimate the RoI for development of the proposed standard. Very rough estimates of the development costs of the standard (excluding implementations) would range from an average of 0.5 persons for 12 months (approximately US$125,000) to 1.5 persons for 18 months (about US$560,000). It is not unreasonable to expect that these costs can be offset completely by the savings associated with development in a single major release of one or two major SQL systems. Vast additional significant savings would, of course, derive from the reduced application development costs incurred by customers.

3.4. Legal Considerations

3.4.1. Patent Assertions

The proposers of the present proposal are unaware of any patents or other notable intellectual property associated with the proposed standard. However, it must be expected that significant numbers of patents will arise with respect to the implementation techniques used by individual SQL implementations that incorporate the technology. No such patents are expected to affect the development or use of the standard. An analogy can be drawn between the present proposal and the development of the SQL/OLAP amendment adopted in 2000 during which no patent issues arose. The INCITS Secretariat will issue a call for patents as required in processing the national standard.

3.4.2. Dissemination of the Standard or Technical Report

The proposers of the present proposal are unaware of any IPR assertions that exist or are likely to arise with respect to the text of the proposed standard or with respect to the specifications contained therein.

4. Related Standards Activities

4.1. Existing Standards

The existing standard that will be affected is [SQL 2008]; The RPR National Standard will be written as an amendment to Part 2, Foundation [FoundFDIS] of SQL 2008. References for the current versions of this still in-process multi-part standard follow. The authors of this project proposal are unaware of any extant or planned standards that relate to the proposed RPR capabilities.

[FoundFDIS] INCITS H2-2008-016/ WG3:BWU-018p , *(FDIS)* ISO 9075-2 SQL/Foundation *(SQL/Foundation)*, December 2007

[SQL 2008] ISO/IEC 9075-Parts 1,2,3,4,9,10,11,13 14:2008 (December 2007 documents will be submitted for Q1 2008 FDIS ballot, expected to be approved by JTC 1 and adopted as INCITS/ANS standards within the year)

H2-2008-015	WG3:BWU-017p	ISO 9075-1 SQL/Framework FDIS
H2-2008-016	WG3:BWU-018p	ISO 9075-2 SQL/Foundation FDIS
H2-2008-017	WG3:BWU-019p	ISO 9075-3 SQL/CLI FDIS
H2-2008-018	WG3:BWU-020p	ISO 9075-4 SQL/PSM FDIS
H2-2008-019	WG3:BWU-021p	ISO 9075-9 SQL/MED FDIS
H2-2008-020	WG3:BWU-022p	ISO 9075-10 SQL/OLB FDIS
H2-2008-021	WG3:BWU-023p	ISO 9075-11 SQL/Schemata FDIS
H2-2008-022	WG3:BWU-024p	ISO 9075-13 SQL/JRT FDIS
H2-2008-023	WG3:BWU-025p	ISO 9075-14 SQL/XML FDIS

4.2. Related Standards Activity

The authors of the present proposal are unaware of related standards activity.

4.3. Recommendations for Close Liaison

Because INCITS H2 has SQL language development authority under project 1234D and TAG responsibility for ISO/IEC JTC 1 SC 32 and its Database Langauges Working Group (WG3), no requirements for close liaison are anticipated at this time. Of course, liaisons (including those with other INCITS TCs) will be recommended if and when a substantive relationship is identified.

5. Units of Measurement used in the Standard

There are no anticipated specifications in the proposed standard that use any physical measurement concepts. Therefore, the proposed standard is measurement-insensitive.

End of paper

Contents

Page

Tables

Table

Page

Figures

Figure

Page

Foreword

A forward suitable for an INCITS standard will be provided in due course. Please be patient!

This first edition of INCITS/ISO/IEC 9075-15 adds to the SQL standard the ability to find patterns in sequences of rows.

INCITS/ISO/IEC 9075 consists of the following parts, under the general title *Information technology — Database languages — SQL*:

— Part 1: Framework (SQL/Framework)

— Part 2: Foundation (SQL/Foundation)

— Part 3: Call-Level Interface (SQL/CLI)

— Part 4: Persistent Stored Modules (SQL/PSM)

— Part 9: Management of External Data (SQL/MED)

— Part 10: Object Language Bindings (SQL/OLB)

— Part 11: Information and Definition Schema (SQL/Schemata)

— Part 13: SQL Routines and Types Using the Java™ Programming Language (SQL/JRT)

— Part 14: XML-Related Specifications (SQL/XML)

— Part 15: Row Pattern Recognition (SQL/RPR)

Introduction

The organization of this part of INCITS/ISO/IEC 9075 is as follows:

1) Clause 1, “Scope”, specifies the scope of this part of INCITS/ISO/IEC 9075.

2) Clause 2, “Normative references”, identifies additional standards that, through reference in this part of INCITS/ISO/IEC 9075, constitute provisions of this part of INCITS/ISO/IEC 9075.

3) Clause 3, “Definitions, notations, and conventions”, defines the notations and conventions used in this part of INCITS/ISO/IEC 9075.

4) Clause 4, “Concepts”, presents concepts used in the definition of row pattern recognition.

5) Clause 5, “Lexical elements”, defines a number of lexical elements used in the definition of row pattern recognition.

6) Clause 6, “Scalar expressions”, defines a number of scalar expressions used in the definition of row pattern recognition.

7) Clause 7, “Query expressions”, defines the elements of the language that produce rows and tables of data as used in row pattern recognition.

8) Clause 8, “Additional common elements”, defines additional common elements used in the definition of row pattern recognition.

9) Clause 9, “Schema definition and manipulation”, defines the schema definition and manipulation statements associated with the definition of row pattern recognition.

10) Clause 10, “Data manipulation”, defines data manipulation operations associated with row pattern recognition.

11) Clause 11, “Diagnostics management”, defines enhancements to the facilities used with row pattern recognition.

12) Clause 12, “Information Schema”, defines the Information and Definition Schema objects associated with row pattern recognition.

13) Clause 13, “Definition Schema”, defines base tables on which the viewed tables containing schema information depend.

14) Clause 14, “Status codes”, defines SQLSTATE values related to row pattern recognition.

15) Clause 15, “Conformance”, defines the criteria for conformance to this part of INCITS/ISO/IEC 9075.

16) Annex A, “SQL Conformance Summary”, is an informative Annex. It summarizes the conformance requirements of the SQL language.

17) Annex B, “Implementation-defined elements”, is an informative Annex. It lists those features for which the body of this part of INCITS/ISO/IEC 9075 states that the syntax, the meaning, the returned results, the effect on SQL-data and/or schemas, or any other behavior is partly or wholly implementation-defined.

18) Annex C, “Implementation-dependent elements”, is an informative Annex. It lists those features for which the body of this part of INCITS/ISO/IEC 9075 states that the syntax, the meaning, the returned results, the effect on SQL-data and/or schemas, or any other behavior is partly or wholly implementation-dependent.

19) Annex D, “Deprecated features”, is an informative Annex. It lists features that the responsible Technical Committee intend will not appear in a future revised version of this part of INCITS/ISO/IEC 9075.

20) Annex E, “Incompatibilities with INCITS/ISO/IEC 9075:2008”, is an informative Annex. It lists incompatibilities with the previous version of this part of INCITS/ISO/IEC 9075.

21) Annex F, “SQL feature taxonomy”, is an informative Annex. It identifies features of the SQL language specified in this part of INCITS/ISO/IEC 9075 by an identifier and a short descriptive name. This taxonomy is used to specify conformance and may be used to develop other profiles involving the SQL language.

22) Annex G, “Defect reports not addressed in this edition of this part of INCITS/ISO/IEC 9075”, is an informative Annex. It describes the Defect Reports that were known at the time of publication of this part of this International Standard. Each of these problems is a problem carried forward from the previous edition of INCITS/ISO/IEC 9075. No new problems have been created in the drafting of this edition of this International Standard.

In the text of this part of INCITS/ISO/IEC 9075, Clauses begin a new odd-numbered page, and in Clause 5, “Lexical elements”, through Clause 15, “Conformance”, Subclauses begin a new page. Any resulting blank space is not significant.

(Blank page)

Information technology — Database languages — SQL —

Part 15:
Row Pattern Recognition (SQL/RPR)

1 Scope

This part of International Standard INCITS/ISO/IEC 9075 specifies the syntax and semantics of database language facilities that support row pattern matching using regular expressions.

The database language facilities that support row pattern recognition include:

— A subset of regular expression syntax.

— Row expression variables that span subsequences of rows, defined using conditions on individual rows and on aggregates of rows.

— A major new syntax element, the MATCH_RECOGNIZE clause, that can be applied to table expressions and can be used in definitions of windows.

It also includes the definition of the Information Schema tables that contain schema information pertaining to row pattern recognition.

**** Editor's Note (number 1) ****

This specification *looks like* a new Part 15 of the SQL standard. That is purely an artifact of the relatively new XML environment in which SQL standard material is coded and built. Specifically, the Principal Editor has not yet added to that environment features to support Amendments to the SQL standard. Needless to say, if this material is to be processed as an Amendment, that work must be done.

It is hoped that participants can be tolerant of this issue until the Editor has the cycles to do that work.

(Blank page)

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

2.1 ISO and IEC standards

[ISO9075-1] INCITS/ISO/IEC 9075-1:2008, *Information technology — Database languages — SQL — Part 1: Framework (SQL/Framework)*.

[ISO9075-2] INCITS/ISO/IEC 9075-2:2008, *Information technology — Database languages — SQL — Part 2: Foundation (SQL/Foundation)*.

[ISO9075-11] INCITS/ISO/IEC 9075-11:2008, *Information technology — Database languages — SQL — Part 11: Information and Definition Schemas (SQL/Schemata)*.

(Blank page)

3 Definitions, notations, and conventions

This Clause modifies Clause 3, "Definitions, notations, and conventions", in ISO/IEC 9075-2.

3.1 Definitions

This Subclause modifies Subclause 3.1, "Definitions", in ISO/IEC 9075-2.

3.1.1 Definitions provided in Part 15

For the purposes of this document, the following definitions apply.

3.1.1.1 **incremental match**: A mode of row pattern matching that returns the multiset union of all maximal matches evaluated after each tuple in the sequence of tuples. In an incremental match, row pattern matching is performed in full after each tuple in the sequence has been retrieved.

3.1.1.2 **maximal match**: A mode of row pattern matching that returns the result of row pattern matching on the entire collection of tuples.

3.1.1.3 **measure**: An exported column the value for which is specified by an expression that is defined with respect to the pattern matching variables. Those expressions can reference partition columns, order by columns, singleton variables, aggregates of group variables, and aggregates on the columns of the source table.

3.1.1.4 **partition (in the context of row pattern matching)**: A collection of rows in which all rows have equal values for a set of one or more columns, called the *partition columns*.

3.1.1.5 **row pattern**: A form of regular expressions that specifies criteria (patterns) used to identify collections of rows in a table based on their relationships to one another.

3.1.1.6 **row pattern matching**: The process of identifying collections of rows in a table based on the patterns specified in row patterns.

3.1.1.7 **TO BE PROVIDED**: TO BE PROVIDED.

(Blank page)

4 Concepts

This Clause modifies Clause 4, "Concepts", in ISO/IEC 9075-2.

4.1 Row pattern matching

Business process applications are composed of complex sequences of possibly many events. For example:

— Security applications must be able to identify unusual behaviors that may comprise efforts to defeat a system's security.

— Financial applications require the ability to detect fraud in, for example, stock trades.

— Material handling and shipping applications are expected to track material and packages through all possible valid paths.

In modern information systems technology, a *regular expression* is typically used to represent patterns for searching character strings and a variety of similar purposes. A form of regular expression can be used to detect patterns in rows of a table that is ordered in some manner (often sequentially by time of row creation).

Regular expressions are used to specify patterns of interest and the ordered rows of the table are scanned to discover whether the patterns exist and, if so, to retrieve detailed or aggregated information contained in the rows that match certain components of the pattern. Variables defined as part of the regular expressions are used to reference those rows that match the components of the pattern represented by the variables.

Columns of the rows referenced by each variable in a row pattern can be used in ordinary value expressions whose values can be returned upon successful matching of that pattern.

4.1.1 Matching rows with a pattern

The process of row pattern matching operates on an ordinary table (a base table, a view, a virtual table produced as an intermediate result during query processing) and produces another (virtual) table as its "output". In this, it may be considered analogous to various components of <table expression>s. Because multiple tables that are joined in a <from clause> can be the subject of row pattern matching, the syntax that specifies row pattern matching is an optional clause available for use on most of the alternatives of <table primary>. Those alternatives are called *source tables* in this document, and the result of row pattern matching is called an *output table*.

Row pattern recognition is specified through the use of the <row pattern recognition clause>. Although the process of using row patterns to match rows of a table has several optional steps, some of which complicate the process significantly, the basic process is relatively straightforward:

1) If specified (<row pattern order by>), the source table is partitioned based on the values of the column or columns specified as *partition columns*; each partition has equal values of the partition column or columns in all of its rows. If not specified, the source table is considered to have only one partition.

2) If specified (<row pattern partition by>), each partition is ordered based on the values of the column or columns specified as *ordering columns*; such ordering may be specified to be ascending or descending on a column-by-column basis (as cursor ordering is done). If not specified, then each the ordering of rows in each partition is implementation-dependent.

   NOTE 1 — Failure to order partitions decreases the likelihood that queries involving row pattern matching will behave the same in different SQL-implementations, and possibly from query execution to query execution on a given SQL-implementation.

3) Rows of the ordered or unordered, partitioned or not partitioned table are inspected for potential matches to the specified row pattern (<row pattern>).

   a) The inspection scans the table to find a row that meets the criteria of the first element (<row pattern term>, <row pattern factor>, <row pattern primary>, as modified by <row pattern quantifier>)) of the row pattern; that element requires or allows additional consecutive rows that match those criteria, those rows are identified by scanning further in the table.

   b) When a sequence of rows have been matched to the first element of the row pattern, those rows are associated with a correlation name that is specified as part of the row pattern.

   c) When there are no more matches to the first element of the row pattern, or that first element has been fully satisfied, the scan continues using the next element of the row pattern.

   d) That process continues until the row pattern has been fully satisfied by row matches in the source table, in which case the row pattern match is successful and the result table is returned, or until there are no more rows in the source table, in which care the row pattern match is unsuccessful. (In fact, row pattern matches can be partly successful by returning rows that match elements of the row pattern as they are found.)

4) When the pattern has been fully matched, or all rows of the source table have been inspected, the output table is generated. The correlation names specified in the row pattern are (if any matching rows were found for the corresponding component of the row pattern) associated with sequences of matched rows. Rows of the output table are generated by specifying one or more columns (<row pattern measure column>s) whose values are derived from the values of columns in the matched rows corresponding to one or more of the correlation names.

Additional optional syntax (<row pattern rows per match>) specifies whether a single row is generated in the output table for all rows of the table corresponding to a complete match of the row pattern, or a row is generated for each row that is matched by some component of the row pattern. In the former case, the sequence of matching rows is treated similarly to the collection of rows created by the <group by clause>; that is, only partition columns, ordering columns, and columns created explicitly for the output table are allowed to be columns of the output table. In the latter case, all columns of the input table, as well as columns created explicitly for the output table, are allowed to be columns of the output table.

Further optional syntax (<row pattern skip to>) specifies whether further matches of the row pattern are sought by scanning from the first row of the table following the last row matched by that pattern, or from the row immediately following the first row matched by that pattern. (Other options are provided as well.)

Still more optional syntax (<row pattern match specification>) determines whether a given row that participates matches one component of a row pattern might also match a different component of that row pattern.

4.1.2 Row pattern matching illustrated

Suppose it is necessary to examine a table containing the identifier of a particular stock, the time at which trades of shares of that stock were performed, and the price at which such trades took place. The business goal is to identify sequences of trades that involve a trade as some (unspecified) price, followed by one or more trades at declining prices, followed by one or more trades at increasing prices, the last one or more of which are greater than the price of the initial trade. The business requirement is to identify the time and price of the first trade in the sequence that matches the pattern, the time at which the decline in prices finished and prices started to rise along with the lowest price, and the time and price of the last trade matched by the pattern.

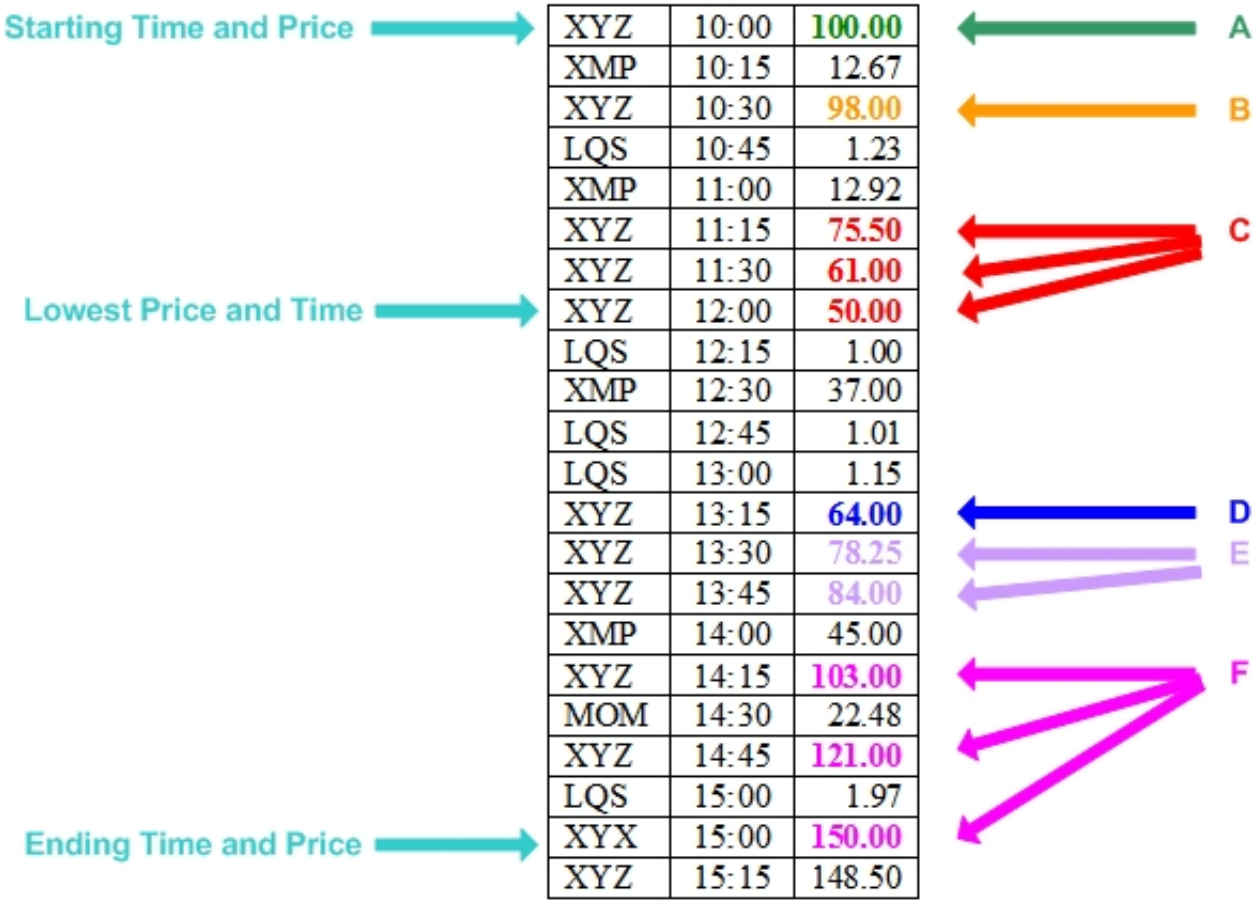
For example, that pattern describes a trading day during which the stock of XYZ Corp. began the day at a price of $100/share, then steadily fell throughout the morning until it hit a price of $50/share at noon, after which the price rose throughout the afternoon until it hit a peak of $150/share at 15:00 and then began to decline again. The information to be returned would be 10:00 (starting time), $100 (starting price), 12:00 (time of turnaround), $50 (lowest price), 15:00 (time at which the share price was no longer rising), and $150 (final price).

A row pattern matching query that solves that particular problem is illustrated here:

```
SELECT a_symbol,      /* stock symbol */
       a_tstamp,      /* start time */
       a_price,       /* start price */
       max_c_tstamp,  /* inflection time */
       last_c_price,  /* lowest price */
       max_f_tstamp,  /* end time */
       last_c_price,  /* end price */
       matchno
FROM stock_ticker
  MATCH_RECOGNIZE (
  PARTITION BY symbol
  ORDER by time_stamp
  MEASURES A.symbol      AS a_symbol,
           A.tstamp      AS a_tstamp,
           A.price       AS a_price,
           MAX(C.tstamp) AS max_c_tstamp,
           LAST(C.price) AS last_c_price,
           MAX(F.tstamp) AS max_f_tstamp,
           MATCH_NUMBER  AS matchno
  ONE ROW PER MATCH
  AFTER MATCH SKIP PAST LAST ROW
  MAXIMAL MATCH
  PATTERN ( A B C* D E* F+ )
  DEFINE /* A is unspecified, defaults to TRUE, matches any row */
         B AS (B.price < PREV (B.price)),
         C AS (C.price <= PREV (C.price)),
         D AS (D.price > PREV (D.price)),
         E AS (E.price >= PREV (E.price)),
```

```
F AS (F.price >= PREV (F.price)
  AND F.price > A.price)
```

Figure 1 — Illustration of important concepts in example query



XYZ	10:00	100.00
XMP	10:15	12.67
XYZ	10:30	98.00
LQS	10:45	1.23
XMP	11:00	12.92
XYZ	11:15	75.50
XYZ	11:30	61.00
XYZ	12:00	50.00
LQS	12:15	1.00
XMP	12:30	37.00
LQS	12:45	1.01
LQS	13:00	1.15
XYZ	13:15	64.00
XYZ	13:30	78.25
XYZ	13:45	84.00
XMP	14:00	45.00
XYZ	14:15	103.00
MOM	14:30	22.48
XYZ	14:45	121.00
LQS	15:00	1.97
XYX	15:00	150.00
XYZ	15:15	148.50

4.1.3 Row partitioning

When a <row pattern recognition clause> contains a <row pattern partition by>, the rows of the source table are partitioned into one or more groups, or *partitions*. In each of those partitions, the values of the set of *ordering columns* are equal in all rows of the partition.

If no <row pattern partition by> is specified, then all rows of the source table are in a single partition.

NOTE 2 — The syntax and semantics of <row pattern partition by> are the same as the syntax and semantics of <window partition clause>.

4.1.4 Row ordering

When a <row pattern recognition clause> contains a <row pattern order by>, the rows of each partition of the source table are ordered by the values of the *ordering columns*.

If no <row pattern order by> is specified, or if the ordering specified by <row pattern order by> does not determine the relative order of two or more rows in the partition, then the ordering of those rows is implementation-dependent.

NOTE 3 — The syntax and semantics of <row pattern order by> are the same as the syntax and semantics of <order by clause>.

NOTE 4 — The syntax of <row pattern recognition clause> makes the <row pattern order by> optional. In practice, very few applications will omit <row pattern order by>.

4.1.5 Row measures

When a <row pattern recognition clause> contains a <row pattern measures>, that <row pattern measures> defines "exported" columns that contain expressions over the pattern variables. The expressions can reference partition columns, order by columns, singleton variables, aggregates on the group variables, and aggregates on the columns of the table that is the source of the <row pattern recognition clause> clause.

4.1.6 Number of rows per match

The <row pattern recognition clause> supports row pattern matching that returns one row per match. When <row pattern rows per match> specifies ONE ROW PER MATCH (which is the default when <row pattern rows per match> is not specified), a single row is returned after completion of a match. Columns of that row are referenced by column names that are specified by <row pattern partition by>, <row pattern order by>, and <row pattern measures>.

When <row pattern rows per match> specifies ALL ROWS PER MATCH, for each row identified by each match, a row is returned. Columns of that row are referenced by column names that are specified by <row pattern partition by>, <row pattern order by>, and <row pattern measures>, and by column names that identify columns of the source table.

When <row pattern rows per match> specifies ALL ROWS PER MATCH, the columns of the row returned by each row identified by each match may also include a column whose value is the name of the row pattern matching variable that the row matched. In addition, the columns of that row may include a column whose value is the ordinal number of the row within the partition to which it belongs.

4.1.7 Skipping rows after matching

Once a row pattern match has been made, row pattern matching continues searching for additional matches (unless the entire source table has been processed. Row pattern matching can continue at any of five places, as specified in the optional <row pattern skip to> component of <row pattern recognition clause>:

1) At the row immediately following the first row of the current match.

2) At the row immediately following the last row of the current match.

3) At the row that is classified by a singleton pattern variable.

4) At the first row of the group of rows matched by a group pattern variable.

5) At the last row of the group of rows matched by a group pattern variable.

4.2 Pattern matching mode

The optional <row pattern match specification> component of <row pattern recognition clause> specifies whether row pattern matching is done across an entire collection of rows, or if it is done by repeated matching incrementally (*e.g.*, as new rows are added to the collection).

When the pattern matching mode is *maximal*, the row pattern recognition process takes place once and operates on the entire collection of rows.

When the pattern matching mode is *incremental*, the process starts off with the first row of a sequence and tries to match that row. Then the second row of the sequence is included and an attempt is made to match a row pattern against those two rows. Then a third row is included and an attempt to match against those three rows is made. This incremental process continues until all rows of the sequence have been included in the matching effort.

NOTE 5 — Under most conditions when a source table is either a persistent base table or a view, applications will use maximal row pattern recognition mode. If the source table is a virtual table whose rows are made available incrementally and continuously (*e.g.*, streaming data), applications should choose to use incremental row pattern recognition mode. Application authors should be aware that maximal row pattern recognition and incremental row pattern recognition do not necessarily produce the same results, even when all other conditions are identical.

4.3 Row patterns

The heart of row pattern recognition is the row pattern itself. A *row pattern* is a form of regular expression in which the items for which matches are sought are not characters, sequences of characters, ranges of characters, and the like. Instead, the items for which matches are sought are expressed in terms of *match variables*, each of which represents a Boolean condition (see Subclause 4.5, "Defining boolean conditions") that in turn expresses the relationship between pairs of rows in the source table.

The regular expressions used in row pattern matching are very similar to those used in character string pattern matching. For example:

— Concatenation: indicated by the absence of any operator sign between two successive items in a pattern.

— Quantifiers: quantifiers are postfix operators with the following choices:

- `*` — zero (0) or more matches
- `+` — one (1) or more matches
- `?` — no match or one (1) match, optional
- `{ n }` — exactly *n* matches
- `{ n, }` — *n* or more matches

- `{ n, m }` — between *n* and *m* (inclusive) matches
- `{ , m }` — between zero (0) and *m* (inclusive) matches

— Reluctant quantifiers: indicated by an additional question mark (*e.g.*, `*?`, `+?`, `??`, `{ n, m }?`). *Reluctant quantifiers* try to match as few rows as possible, whereas non-reluctant quantifiers (*greedy qualifiers*) try to match as many rows as possible.

— `{- ... -}`: indicates that matching rows are to be excluded from the output.

— Alternation: indicated by a vertical bar " `|` ".

— Grouping: indicated by parentheses

— `^`: indicates the start of a partition

— `$`: indicates the end of a partition

The row pattern implicitly defines the match variables, which are assigned meaning (Boolean conditions) in a subsequent clause of the <row pattern match specification>.

4.4 Unions of variables

The optional <row pattern subset clause> component of <row pattern match specification> is used to define additional variables that represent unions of the rows matched to the match variables defined in the row pattern proper.

4.5 Defining boolean conditions

The Boolean conditions used to define what rows are matched to specific parts of a row pattern are associated with match variables. One or more match variables are specified using syntax that requires the name to be given each variable and the Boolean condition associated with that variable. Each Boolean condition is expressed in terms of match variables. The Boolean condition associated with a particular match variable can reference the match variable with which it is associated as well as other match variables.

More precisely, each match variable name is used as a kind of qualifier (analogous to <correlation name>s) for the names of columns of the source table. The Boolean conditions are then defined as comparisons of the values of columns in rows of the source table, the specific rows being used for the comparisons determined by the rows referenced by each match variable name.

NOTE 6 — Suppose a table whose name is `T` has three columns whose names are `C1`, `C2`, and `C3`, respectively. A Boolean condition associated with a match variable whose name is `V1` can reference the values of those columns using a notation like this one: "`V1.C1`". A subsequent Boolean condition associated with a match variable whose name is `V2` can reference the values of those columns with either "`V1.C1`" or "`V2.C1`". The values represented as "`V1.C1`" are those matched by the Boolean condition associated with `V1` and those represented as "`V2.C1`" are those matched by the Boolean condition associated with `V2`.

Match patterns may use the names of match variables that are not declared. Such match variables always evaluate to *True* and thus will match any row.

Consider the following Boolean condition definitions:

```
A AS A.price > 100,
```

```
B AS B.price > A.price,
C AS C.price < AVG (B.price),
D AS D.price > PREV(D.price)
```

The match variable named A identifies a single row – the first row whose column named `price` has a value that is greater than 100. The match variable named B identifies zero or more rows (which must be consecutive rows following the single row identified by match variable named A), all of whose `price` column have values greater than the value of the price column associated with match variable A. Because A matched exactly one row, the definition of the Boolean condition associated with B is allowed to reference individual columns of the row matched by A (all columns, not only the one or ones used in the definition of A).

However, because B represents possibly many rows, the definition of C is not permitted to reference individual columns of the rows represented by B. The definition of C can, of course, reference aggregates of the columns of the rows identified by B. In this example, C will match all rows immediately following those identified by B that all have values of the `price` column that are greater than the average of the values of the `price` column in the rows identified by B

Finally, the definition of D illustrates how a match variable can be defined in terms of itself. In this case, D represents all rows following the last row represented by C whose `price` column's value is greater than the previous row's `price` column's value. (PREV is a function that is used to access the values of the previous row of the partition.)

4.6 Row pattern classifiers

(To Be Supplied.)

4.7 Row pattern match number

(To Be Supplied.)

4.8 Row pattern aggregates

(To Be Supplied.)

4.9 To be supplied

(To Be Supplied.)

5 Lexical elements

This Clause modifies Clause 5, "Lexical elements", in ISO/IEC 9075-2.

5.1 <token> and <separator>

This Subclause modifies Subclause 5.2, "<token> and <separator>", in ISO/IEC 9075-2.

Function

Specify lexical units (tokens and separators) that participate in SQL language.

Format

```
<non-reserved word> ::=
    !! All alternatives from INCITS/ISO/IEC 9075-2

  | PAST

<reserved word> ::=
    !! All alternatives from INCITS/ISO/IEC 9075-2

  | AGGREGATES

  | CLASSIFIER

  | DEFINE

  | INCREMENTAL | INITIAL

  | MATCH_NUMBER | MATCH_RECOGNIZE | MAXIMAL | MEASURES

  | ONE

  | PATTERN | PER

  | SEEK | SKIP | SUBSET
```

**** Editor's Note (number 2) ****

Readers should note that not all keywords specified in this section as <reserved word>s are necessarily required to be reserved. Reviewers are invited to inform the editor of such words that can be moved to <non-reserved word>.

Syntax Rules

No additional Syntax Rules.

Access Rules

No additional Access Rules.

General Rules

No additional General Rules.

Conformance Rules

No additional Conformance Rules.

6 Scalar expressions

6.1 <set function specification>

Function

Specify a value derived by the application of a function to an argument.

Format

```
<set function specification> ::=
    <aggregate function>
  | <grouping operation>
  | <row pattern matching function>

<row pattern matching function> ::=
  PREV <left paren> <row pattern variable name> <period> <column reference> <right paren>
```

Syntax Rules

1) If <row pattern matching function> *RPMF* is specified, then:

   a) Let *RPVN* be the <row pattern variable name> immediately contained in *RPMF*.

   b) Let *RPD* be the <row pattern definition> that contains *RPMF* and let *RPDL* be the <row pattern definition list> that contains *RPD*.

   c) *RPVN* shall be equivalent to exactly one of the following:

      i) The <row pattern variable name> immediately contained in *RPD*.

      ii) The <row pattern variable name> immediately contained in some <row pattern definition> *RPDE* that is immediately contained in *RPDL* prior to *RPD*.

Access Rules

No additional Access Rules.

General Rules

1) If <row pattern matching function> *RPMF* is specified, then:

   a) Let *RPVN* be the <row pattern variable name> immediately contained in *RPMF* and let *CR* be the <column reference> immediately contained in *RPMF*.

b) Let *RPD* be the <row pattern definition> that contains *RPMF* and let *RPDL* be the <row pattern definition list> that contains *RPD*.

c) Let *TP* be the <table primary> that simply contains the <row pattern recognition clause> *RPRC* that contains *RPDL*. If *RPRC* simply contains <row pattern order by>, then let *RPOB* be that <row pattern order by>; otherwise, let *RPOB* be the zero-length string.

d) Let *TU* be the table identified by the <table or query name>, <derived table>, <lateral derived table>, <collection derived table>, <table function derived table>, or <only spec> immediately contained in *TP*. Let *TO* be the table that results from ordering *TU* according to *RPOB*.

e) Let *N* be the number of rows in *TO*. Let R_i, 1 (one) $\le i \le N$, be the rows of *T* in order.

f) Let R_i be the row of *TO* being evaluated when *RPMF* is invoked; the value of *RPVN* is the value of the column identified by *CR* in row R_{i-1}.

Conformance Rules

1) Without Feature XXXX, "row pattern recognition", conforming SQL language shall not contain a <row pattern matching function>.

7 Query expressions

This Clause modifies Clause 7, "Query expressions", in ISO/IEC 9075-2.

7.1 <table reference>

This Subclause modifies Subclause 7.6, "<table reference>", in ISO/IEC 9075-2.

Function

Reference a table.

Format

```
<table primary> ::=
    <table or query name> [ <correlation or recognition>
        [ <left paren> <derived column list> <right paren> ] ]
  | <derived table> <correlation or recognition>
      [ <left paren> <derived column list> <right paren> ]
  | <lateral derived table> <correlation or recognition>
      [ <left paren> <derived column list> <right paren> ]
  | <collection derived table> <correlation or recognition>
      [ <left paren> <derived column list> <right paren> ]
  | <table function derived table> <correlation or recognition>
      [ <left paren> <derived column list> <right paren> ]
  | <only spec> [ <correlation or recognition>
      [ <left paren> <derived column list> <right paren> ] ]
  | <parenthesized joined table>

<correlation or recognition> ::=
    [ AS ] <correlation name>
  | <row pattern recognition clause>
```

Syntax Rules

No additional Syntax Rules.

Access Rules

No additional Access Rules.

General Rules

No additional General Rules.

Conformance Rules

1) Without Feature XXXX, “row pattern recognition”, conforming SQL language shall not contain a <row pattern recognition clause>.

7.2 <row pattern recognition clause>

Function

Match row patterns.

Format

```
<row pattern recognition clause> ::=
  MATCH_RECOGNIZE <left paren>
      [ <row pattern partition by> ]
      [ <row pattern order by> ]
      [ <row pattern measures> ]
      [ <row pattern rows per match> ]
      [ <row pattern skip to> ]
      [ <row pattern match specification> ]
      PATTERN <left paren> <row pattern> <right paren>
      [ <row pattern subset clause> ]
      DEFINE <row pattern definition list>
      [ <row pattern classifier> ]
      [ <row pattern MATCH_NUMBER> ]
      [ <row pattern aggregates> ]
      <right paren>

<row pattern partition by> ::=
  PARTITION BY <row pattern partition list>

<row pattern partition list> ::=
  <row pattern partition column> [ { <comma> <row pattern partition column> }... ]

<row pattern partition column> ::=
  <column reference> [ <collate clause> ]

<row pattern order by> ::=
  ORDER BY <sort specification list>

<row pattern measures> ::=
  MEASURES <row pattern measure list>

<row pattern measure list> ::=
  <row pattern measure column> [ { <comma> <row pattern measure column> }... ]

<row pattern measure column> ::=
  <column name> AS <expression>

<row pattern rows per match> ::=
    ONE ROW PER MATCH
  | ALL ROWS PER MATCH

<row pattern skip to> ::=
    SKIP TO NEXT ROW
  | SKIP PAST LAST ROW
  | SKIP TO FIRST <variable name>
  | SKIP TO LAST <variable name>
  | SKIP TO <variable name>
```

```
<row pattern match specification> ::=
    INCREMENTAL MATCH
  | MAXIMAL MATCH

<row pattern> ::=
    <row pattern term>
  | <row pattern> <vertical bar> <row pattern term>

<row pattern term> ::=
    <row pattern factor>
  | <row pattern term> <row pattern factor>

<row pattern factor> ::=
  <row pattern primary> [ <row pattern quantifier> ]

<row pattern quantifier> ::=
    <asterisk> [ <question mark> ]
  | <plus sign> [ <question mark> ]
  | <question mark> [ <question mark> ]
  | <left brace> [ <unsigned integer> ] <comma> [ <unsigned integer> ] <right brace> [
  <question mark> ]

<row pattern primary> ::=
    <variable name>
  | <left paren> <row pattern> <right paren>

<row pattern subset> ::=
  SUBSET <row pattern subset list>

<row pattern subset list> ::=
  <row pattern subset item> [ { <comma> <row pattern subset item> }... ]

<row pattern subset item> ::=
  <variable name> <equals operator> <left paren> <row pattern subset rhs> <right paren>

<row pattern subset rhs> ::=
  <variable name> [ { <comma> <variable name> }... ]

<row pattern definition list> ::=
  <row pattern definition> [ { <comma> <row pattern definition> }... ]

<row pattern definition> ::=
  <row pattern variable name> AS <search condition>

<row pattern variable name> ::=
  <variable name>

<row pattern classifier> ::=
  CLASSIFIER <column name>

<row pattern MATCH_NUMBER> ::=
  MATCH_NUMBER <column name>

<row pattern aggregates> ::=
  AGGREGATES <row pattern aggregate list>

<row pattern aggregate list> ::=
  <row pattern aggregate definition> [ { <comma> <row pattern aggregate definition> }...
  ]
```

```
<row pattern aggregate definition> ::=
  <value expression> AS <column name>
```

Syntax Rules

1) **To Be Supplied**

Access Rules

No additional Access Rules.

General Rules

1) **To Be Supplied**

Conformance Rules

1) **To Be Supplied**

7.3 <window clause>

Function

Specify one or more window definitions.

Format

```
<window specification details> ::=
  [ <existing window name> ]
      [ <window partition clause> ]
      [ <window order clause> ]
      [ <window frame clause> ]
      [ <window pattern recognition clause> ]

<window pattern recognition clause> ::=
  MATCH_RECOGNIZE <left paren>
      [ <row pattern partition by> ]
      [ <row pattern order by> ]
      [ <row pattern measures> ]
      <row pattern initial or seek> PATTERN <left paren> <row pattern> <right paren>
      [ <row pattern subset clause> ]
      DEFINE <row pattern definition list>
      [ <row pattern classifier> ]
      [ <row pattern aggregates> ]
      <right paren>

<row pattern initial or seek> ::=
    INITIAL
  | SEEK
```

Syntax Rules

1) **To Be Supplied**

Access Rules

No additional Access Rules.

General Rules

1) **To Be Supplied**

Conformance Rules

1) **To Be Supplied**

8 Additional common elements

This Clause modifies Clause 10, "Additional common elements", in ISO/IEC 9075-2.

(Blank page)

9 Schema definition and manipulation

This Clause modifies Clause 11, "Schema definition and manipulation", in ISO/IEC 9075-2.

(Blank page)

10 Data manipulation

This Clause modifies Clause 14, "Data manipulation", in ISO/IEC 9075-2.

(Blank page)

11 Diagnostics management

This Clause modifies Clause 23, "Diagnostics management", in ISO/IEC 9075-2.

11.1 <get diagnostics statement>

This Subclause modifies Subclause 23.1, "<get diagnostics statement>", in ISO/IEC 9075-2.

Function

Get exception or completion condition information from the diagnostics area.

Format

```
No additional Format items.
```

Syntax Rules

1) *Table 1, "<condition information item name>s for use with <get diagnostics statement>", modifies Table 30, "<statement information item name>s for use with <get diagnostics statement>", in [ISO9075-2].*

Table 1 — <condition information item name>s for use with <get diagnostics statement>

<identifier>	Data Type
All alternatives from [ISO9075-2]	
TO BE SPECIFIED	*TO BE SPECIFIED*

Access Rules

No additional Access Rules.

General Rules

1) *Table 2, "SQL-statement codes", modifies Table 32, "SQL-statement codes", in [ISO9075-2].*

Table 2 — SQL-statement codes

SQL-statement	Identifier	Code
All alternatives from [ISO9075-2]		
TO BE SPECIFIED	*TO BE SPECIFIED*	*TO BE SPECI-FIED*

Conformance Rules

No additional Conformance Rules.

12 Information Schema

This Clause modifies Clause 5, "Information Schema", in ISO/IEC 9075-11.

12.1 TO BE SPECIFIED

Function

TO BE SPECIFIED

Definition

```
TO BE SPECIFIED
```

Conformance Rules

No additional Conformance Rules.

12.2 Short name views

This Subclause modifies Subclause 5.77, "Short name views", in ISO/IEC 9075-11.

Function

Provide alternative views that use only identifiers that do not require Feature F391, "Long identifiers".

Definition

```
TO BE SPECIFIED
```

Conformance Rules

No additional Conformance Rules.

13 Definition Schema

This Clause modifies Clause 6, "Definition Schema", in ISO/IEC 9075-11.

(Blank page)

14 Status codes

This Clause modifies Clause 24, "Status codes", in ISO/IEC 9075-2.

14.1 SQLSTATE

This Subclause modifies Subclause 24.1, "SQLSTATE", in ISO/IEC 9075-2.

Table 3, "SQLSTATE class and subclass values", modifies Table 33, "SQLSTATE class and subclass values", in [ISO9075-2].

Table 3 — SQLSTATE class and subclass values

Category	Condition	Class	Subcondition	Subclass
	All alternatives from INCITS/ISO/IEC 9075-2			
X	*TO BE SPECIFIED*	*TO BE SPECI-FIED*	(*TO BE SPECIFIED*)	*TO BE SPECI-FIED*

(Blank page)

15 Conformance

15.1 Claims of conformance to SQL/RPR

In addition to the requirements of [ISO9075-1], Clause 8, "Conformance", a claim of conformance to this part of INCITS/ISO/IEC 9075 shall:

1) Claim conformance to Feature XXXX, "*TO BE SPECIFIED*".

15.2 Implied feature relationships of SQL/PSM

Table 4 — Implied feature relationships of SQL/PSM

Feature ID	Feature Name	Implied Feature ID	Implied Feature Name
TO BE SPECI-FIED	Stored *TO BE SPECIFIED*	*TO BE SPECI-FIED*	*TO BE SPECIFIED*

(Blank page)

Annex A
(informative)

SQL Conformance Summary

This Annex modifies Annex A, "SQL Conformance Summary", in ISO/IEC 9075-2.

The contents of this Annex summarizes all Conformance Rules, ordered by Feature ID and by Subclause.

1) Specifications for Feature XXXX, "row pattern recognition":
   a) Subclause 6.1, "<set function specification>":
      i) Without Feature XXXX, "row pattern recognition", conforming SQL language shall not contain a <row pattern matching function>.
   b) Subclause 7.1, "<table reference>":
      i) Without Feature XXXX, "row pattern recognition", conforming SQL language shall not contain a <row pattern recognition clause>.

(Blank page)

Annex B
(informative)

Implementation-defined elements

This Annex modifies Annex B, "Implementation-defined elements", in ISO/IEC 9075-2.

This Annex references those features that are identified in the body of this part of INCITS/ISO/IEC 9075 as implementation-defined.

None.

(Blank page)

Annex C
(informative)

Implementation-dependent elements

This Annex modifies Annex C, "Implementation-dependent elements", in ISO/IEC 9075-2.

This Annex references those places where this part of INCITS/ISO/IEC 9075 states explicitly that the actions of a conforming implementation are implementation-dependent.

None.

(Blank page)

Annex D
(infomative)

Deprecated features

This Annex modifies Annex D, "Deprecated features", in ISO/IEC 9075-2.

It is intended that the following features will be removed at a later date from a revised version of this part of INCITS/ISO/IEC 9075:

None.

(Blank page)

Annex E
(informative)

Incompatibilities with INCITS/ISO/IEC 9075:2008

This Annex modifies Annex E, "Incompatibilities with ISO/IEC 9075:2008", in ISO/IEC 9075-2.

This edition of this part of INCITS/ISO/IEC 9075 introduces some incompatibilities with the earlier version of Database Language SQL as specified in INCITS/ISO/IEC 9075-15:2008.

Except as specified in this Annex, features and capabilities of Database Language SQL are compatible with INCITS/ISO/IEC 9075-15:2008.

None.

(Blank page)

Annex F
(informative)

SQL feature taxonomy

This Annex modifies Annex F, "SQL feature taxonomy", in ISO/IEC 9075-2.

This Annex describes a taxonomy of features defined in this part of INCITS/ISO/IEC 9075.

Table 5, "Feature taxonomy for optional features", contains a taxonomy of the optional features of the SQL language that are specified in this part of INCITS/ISO/IEC 9075.

In this table, the first column contains a counter that may be used to quickly locate rows of the table; these values otherwise have no use and are not stable — that is, they are subject to change in future editions of or even Technical Corrigenda to INCITS/ISO/IEC 9075 without notice.

The "Feature ID" column of this table specifies the formal identification of each feature and each subfeature contained in the table.

The "Feature Name" column of this table contains a brief description of the feature or subfeature associated with the Feature ID value.

Table 5, "Feature taxonomy for optional features", does not provide definitions of the features; the definition of those features is found in the Conformance Rules that are further summarized in Annex A, "SQL Conformance Summary".

Table 5 — Feature taxonomy for optional features

	Feature ID	Feature Name
1	***TO BE SPECI-FIED***	***TO BE SPECIFIED***

(Blank page)

Annex G
(informative)

Defect reports not addressed in this edition of this part of INCITS/ISO/IEC 9075

Each entry in this Annex describes a reported defect in the previous edition of this part of INCITS/ISO/IEC 9075 that remains in this edition.

None.

(Blank page)

Index

Index entries appearing in **boldface** indicate the page where the word, phrase, or BNF nonterminal was defined; index entries appearing in *italics* indicate a page where the BNF nonterminal was used in a Format; and index entries appearing in roman type indicate a page where the word, phrase, or BNF nonterminal was used in a heading, Function, Syntax Rule, Access Rule, General Rule, Leveling Rule, Table, or other descriptive text.

— S —

— T —

— W —

— X —