SQL:1999: Understanding Relational Language Components, ISSN 1046-1698, Jim Melton, Alan R. Simon, Morgan Kaufmann, 2002, 1558604561, 9781558604568, 895 pages. This book is the best way to make the leap from SQL-92 to SQL:1999, but it is much more than just a simple bridge between the two. The latest from celebrated SQL experts Jim Melton and Alan Simon, SQL:1999 is a comprehensive, eminently practical account of SQL's latest incarnation and a potent distillation of the details required to put it to work. Written to accommodate both novice and experienced SQL users, SQL:1999 focuses on the language's capabilities, from the basic to the advanced, and the ways that real applications take advantage of them. Throughout, the authors illustrate features and techniques with clear and often entertaining references to their own custom database, which can be downloaded from the companion Web site. * Gives authoritative coverage from an expert team that includes the editor of the SQL-92 and SQL:1999 standards.* Provides a general introduction to SQL that helps you understand its constituent parts, history, and place in the realm of computer languages.* Explains SQL:1999's more sophisticated features, including advanced value expressions, predicates, advanced SQL query expressions, and support for active databases.* Explores key issues for programmers linking applications to SQL databases.* Provides guidance on troubleshooting, internationalization, and changes anticipated in the next version of SQL.* Contains appendices devoted to database design, a complete SQL:1999 example, the standardization process, and more..


Sql: Abg 3E , Oppel, Jan 1, 2008, , . Written to the SQL:2006 ANSI/ISO standard, this easy-to-follow resource teaches database professionals and programmers how to use SQL. Learn how to retrieve, insert, update ....

Understanding the New SQL A Complete Guide, Jim Melton, 1993, Computers, 536 pages. The only book you'll ever need on SQL. The authors detail the changes in the new standard and provide a thorough guide to programming with SQL 2 for both newcomers and ....


Designing database applications with objects and rules the IDEA methodology, Stefano Ceri, Piero Fraternali, 1997, Computers, 579 pages. Master the latest advances in modern database technology with IDEA, a state-of-the-art methodology for developing, maintaining and applying database systems. The IDEA ....

Advanced ANSI SQL Data Modeling and Structure Processing , Michael M. David, Jan 1, 1999, Computers, 244 pages. This new book is an essential tool for utilizing the ANSI SQL outer join operation, and an indispensable guide to using this operation to perform simple or complex data ....


DB2 : Design & development guide , , , ,
Mastering SQL, Martin Gruber, Feb 1, 2000, Computers, 841 pages. A recognized SQL expert and author of “SQL Instant Reference” provides this guide for database programmers and administrators who need to learn SQL or want to know about ....

The dynamics of data base, William H. Inmon, Thomas J. Bird, 1986, 395 pages. Very Good, No Highlights or Markup, all pages are intact...

How to Build a Business Rules Engine Extending Application Functionality Through Metadata Engineering, Malcolm Chisholm, 2004, Computers, 484 pages. This is the only book that demonstrates how to develop a business rules engine. Covers user requirements, data modeling, metadata, and more. A sample application is used ....

This book is the best way to make the leap from SQL-92 to SQL:1999, but it is much more than just a simple bridge between the two. The latest from celebrated SQL experts Jim Melton and Alan Simon, SQL:1999 is a comprehensive, eminently practical account of SQL's latest incarnation and a potent distillation of the details required to put it to work. Written to accommodate both novice and experienced SQL users, SQL:1999 focuses on the language's capabilities, from the basic to the advanced, and the ways that real applications take advantage of them. Throughout, the authors illustrate features and techniques with clear and often entertaining references to their own custom database, which can be downloaded from the companion Web site.

application array authorization identifier base tables Chapter character set character string CHARACTER VARYING CHECK constraint clause COBOL collation column name constant SQLSTATE_TYPE constraint contains CREATE TABLE current_dvd_sale_price cursor Data Definition Language data type database system datetime DBMS declaration default defined definition DELETE descriptor area diagnostics area discussed dynamic parameters dynamic SQL embedded SQL execute external routine FOREIGN KEY function grouping host language host variable implementation implementation-defined implementation-dependent INTEGER invoked keyword length module movie titles movie_titles movies_stars null value operations OUTER JOIN predicate PRIMARY KEY privileges procedure query expression recursive reference referential referential integrity regular expression result set retrieve role name row type savepoint schema search condition SELECT statement specify SQL standard SQL statement SQL-invoked routines SQL-session SQL's SQLCHAR SQLINTEGER SQLRETURN SQLSMALLINT SQLSTATE statement handle stored subquery Syntax temporary table tion transaction trigger user identifier value expression virtual table Yes SYSTEM SCREEN.TUNES

Jim Melton is editor of all parts of ISO/IEC 9075 (SQL) and is a representative for database standards at Oracle Corporation. Since 1986, he has been his company's representative to ANSI INCITS Technical Committee H2 for Database and a US representative to ISO/IEC JTC1/SC32/WG3 (Database Languages). In addition, Jim has participated in the W3C's XML Query Working Group since 1998 and is currently co-Chair of that Working Group. He is also Chair of the WG's Full-Text Task Force, co-Chair of the Update Language Task Force, and co-editor of two XQuery-related specifications. He is the author of several SQL books.

Alan Simon is a leading authority on data warehousing and database technology. He is the author of 26 books, including the previous edition of this book and the forthcoming Data Warehousing and Business Intelligence for e-Commerce, available from Morgan Kaufmann Publishers in early 2001. He currently provides data warehousing-related consulting services to clients.

Originally based upon relational algebra and tuple relational calculus, SQL consists of a data definition language and a data manipulation language. The scope of SQL includes data insert, query, update and delete, schema creation and modification, and data access control. Although SQL is often described as, and to a great extent is, a declarative language (4GL), it also includes procedural elements.

SQL was one of the first commercial languages for Edgar F. Codd's relational model, as described
in his influential 1970 paper "A Relational Model of Data for Large Shared Data Banks". Despite not entirely adhering to the relational model as described by Codd, it became the most widely used database language.

SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standards (ISO) in 1987. Since then, the standard has been enhanced several times with added features. Despite these standards, code is not completely portable among different database systems, which can lead to vendor lock-in. The different makers do not perfectly adhere to the standard, for instance by adding extensions, and the standard itself is sometimes ambiguous.

SQL was initially developed at IBM by Donald D. Chamberlin, Donald C. Messerly, and Raymond F. Boyce in the early 1970s. This version, initially called SEQUEL (Structured English Query Language), was designed to manipulate and retrieve data stored in IBM's original quasi-relational database management system, System R, which a group at IBM San Jose Research Laboratory had developed during the 1970s. The acronym SEQUEL was later changed to SQL because "SEQUEL" was a trademark of the UK-based Hawker Siddeley aircraft company.

In the late 1970s, Relational Software, Inc. (now Oracle Corporation) saw the potential of the concepts described by Codd, Chamberlin, and Boyce and developed their own SQL-based RDBMS with aspirations of selling it to the U.S. Navy, Central Intelligence Agency, and other U.S. government agencies. In June 1979, Relational Software, Inc. introduced the first commercially available implementation of SQL, Oracle V2 (Version2) for VAX computers.

The most common operation in SQL is the query, which is performed with the declarative SELECT statement. SELECT retrieves data from one or more tables, or expressions. Standard SELECT statements have no persistent effects on the database. Some non-standard implementations of SELECT can have persistent effects, such as the SELECT INTO syntax that exists in some databases.

A query includes a list of columns to be included in the final result immediately following the SELECT keyword. An asterisk ("*") can also be used to specify that the query should return all columns of the queried tables. SELECT is the most complex statement in SQL, with optional keywords and clauses that include:

The following is an example of a SELECT query that returns a list of expensive books. The query retrieves all rows from the Book table in which the price column contains a value greater than 100.00. The result is sorted in ascending order by title. The asterisk (*) in the select list indicates that all columns of the Book table should be included in the result set.

SQL includes operators and functions for calculating values on stored values. SQL allows the use of expressions in the select list to project data, as in the following example which returns a list of books that cost more than 100.00 with an additional sales_tax column containing a sales tax figure calculated at 6% of the price.

Queries can be nested so that the results of one query can be used in another query via a relational operator or aggregation function. A nested query is also known as a subquery. While joins and other table operations provide computationally superior (i.e. faster) alternatives in many cases, the use of subqueries introduces a hierarchy in execution which can be useful or necessary. In the following example, the aggregation function AVG receives as input the result of a subquery:

Since 1999 the SQL standard allows named subqueries called common table expression (named and designed after the IBM DB2 version 2 implementation; Oracle calls these subquery factoring). CTEs can be also be recursive by referring to themselves; the resulting mechanism allows tree or graph traversals (when represented as relations), and more generally fixpoint computations.

The concept of Null was introduced into SQL to handle missing information in the relational model.
The word NULL is a reserved keyword in SQL, used to identify the Null special marker. Comparisons with Null, for instance equality (=) in WHERE clauses, results in an Unknown truth value. In SELECT statements SQL returns only results for which the WHERE clause returns a value of True; i.e. it excludes results with values of False and also excludes those whose value is Unknown.

Along with True and False, the Unknown resulting from direct comparisons with Null thus brings a fragment of three-valued logic to SQL. The truth tables SQL uses for AND, OR, and NOT correspond to a common fragment of the Kleene and Lukasiewicz three-valued logic (which differ in their definition of implication, however SQL defines no such operation).[20]

There are however disputes about the semantic interpretation of Nulls in SQL because of its treatment outside direct comparisons. As seen in the table above direct equality comparisons between two NULLs in SQL (e.g. NULL = NULL) returns a truth value of Unknown. This is in line with the interpretation that Null does not have a value (and is not a member of any data domain) but is rather a placeholder or "mark" for missing information. However, the principle that two Nulls aren't equal to each other is effectively violated in the SQL specification for the UNION and INTERSECT operators, which do identify nulls with each other.[21] Consequently, these set operations in SQL may produce results not representing sure information, unlike operations involving explicit comparisons with NULL (e.g. those in a WHERE clause discussed above). In Codd's 1979 proposal (which was basically adopted by SQL92) this semantic inconsistency is rationalized by arguing that removal of duplicates in set operations happens "at a lower level of detail than equality testing in the evaluation of retrieval operations."[20] However, computer science professor Ron van der Meyden concluded that "The inconsistencies in the SQL standard mean that it is not possible to ascribe any intuitive logical semantics to the treatment of nulls in SQL."[21]

Additionally, since SQL operators return Unknown when comparing anything with Null directly, SQL provides two Null-specific comparison predicates: IS NULL and IS NOT NULL test whether data is or is not Null.[22] Universal quantification is not explicitly supported by SQL, and must be worked out as a negated existential quantification.[23][24][25] There is also the "<row value expression> IS DISTINCT FROM <row value expression>" infixed comparison operator which returns TRUE unless both operands are equal or both are NULL. Likewise, IS NOT DISTINCT FROM is defined as "NOT (<row value expression> IS DISTINCT FROM <row value expression>)". SQL:1999 also introduced BOOLEAN type variables, which according to the standard can also hold Unknown values. In practice, a number of systems (e.g. PostgreSQL) implement the BOOLEAN Unknown as a BOOLEAN NULL.

COMMIT and ROLLBACK terminate the current transaction and release data locks. In the absence of a START TRANSACTION or similar statement, the semantics of SQL are implementation-dependent. The following example shows a classic transfer of funds transaction, where money is removed from one account and added to another. If either the removal or the addition fails, the entire transaction is rolled back.

For example, the number 123.45 has a precision of 5 and a scale of 2. The precision is a positive integer that determines the number of significant digits in a particular radix (binary or decimal). The scale is a non-negative integer. A scale of 0 indicates that the number is an integer. For a decimal number with scale S, the exact numeric value is the integer value of the significant digits divided by 10^S.

SQL provides several functions for generating a date / time variable out of a date / time string (TO_DATE, TO_TIME, TO_TIMESTAMP), as well as for extracting the respective members (seconds, for instance) of such variables. The current system date / time of the database server can be called by using functions like NOW.

SQL is designed for a specific purpose: to query data contained in a relational database. SQL is a set-based, declarative query language, not an imperative language like C or BASIC. However, there are extensions to Standard SQL which add procedural programming language functionality, such as
control-of-flow constructs. These include:

In addition to the standard SQL/PSM extensions and proprietary SQL extensions, procedural and object-oriented programmability is available on many SQL platforms via DBMS integration with other languages. The SQL standard defines SQL/JRT extensions (SQL Routines and Types for the Java Programming Language) to support Java code in SQL databases. SQL Server 2005 uses the SQLCLR (SQL Server Common Language Runtime) to host managed .NET assemblies in the database, while prior versions of SQL Server were restricted to using unmanaged extended stored procedures that were primarily written in C. PostgreSQL allows functions to be written in a wide variety of languages including Perl, Python, Tcl, and C.[28]

SQL deviates in several ways from its theoretical foundation, the relational model and its tuple calculus. In that model, a table is a set of tuples, while in SQL, tables and query results are lists of rows: the same row may occur multiple times, and the order of rows can be employed in queries (e.g. in the LIMIT clause). Furthermore, additional features (such as NULL and views) were introduced without founding them directly on the relational model, which makes them more difficult to interpret.

Critics argue that SQL should be replaced with a language that strictly returns to the original foundation: for example, see The Third Manifesto. Other critics suggest that Datalog has two advantages over SQL: it has a cleaner semantics which facilitates program understanding and maintenance, and it is more expressive, in particular for recursive queries.[29]

Popular implementations of SQL commonly omit support for basic features of Standard SQL, such as the DATE or TIME data types. The most obvious such examples, and incidentally the most popular commercial and proprietary SQL DBMSs, are Oracle (whose DATE behaves as DATETIME,[31][32] and lacks a TIME type)[33] and MS SQL Server (before the 2008 version). As a result, SQL code can rarely be ported between database systems without modifications.

SQL was adopted as a standard by the American National Standards Institute (ANSI) in 1986 as SQL-86[34] and the International Organization for Standardization (ISO) in 1987. Nowadays the standard is subject to continuous improvement by the Joint Technical Committee ISO/IEC JTC 1, Information technology, Subcommittee SC 32, Data management and interchange which affiliate to ISO as well as IEC. It is commonly denoted by the pattern: ISO/IEC 9075-n:yyyy Part n: title, or, as a shortcut, ISO/IEC 9075.

ISO/IEC 9075-14:2006 defines ways in which SQL can be used in conjunction with XML. It defines ways of importing and storing XML data in an SQL database, manipulating it within the database and publishing both XML and conventional SQL-data in XML form. In addition, it enables applications to integrate into their SQL code the use of XQuery, the XML Query Language published by the World Wide Web Consortium (W3C), to concurrently access ordinary SQL-data and XML documents.[38]

ISO/IEC 9075-3:2008 Part 3: Call-Level Interface (SQL/CLI). It defines interfacing components (structures, procedures, variable bindings) that can be used to execute SQL statements from applications written in Ada, C respectively C++, COBOL, Fortran, MUMPS, Pascal or PL/I. (For Java see part 10.) SQL/CLI is defined in such a way that SQL statements and SQL/CLI procedure calls are treated as separate from the calling application's source code. Open Database Connectivity is a well-known superset of SQL/CLI. This part of the standard consists solely of mandatory features.