SQL

SQL (/ˌɛsˌkjuːˈɛl/ (listen) *S-Q-L*,[4] /ˈsiːkwəl/ "sequel"; **Structured Query Language**)[5][6][7] is a domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS). It is particularly useful in handling structured data, i.e. data incorporating relations among entities and variables.

SQL offers two main advantages over older read–write APIs such as ISAM or VSAM. Firstly, it introduced the concept of accessing many records with one single command. Secondly, it eliminates the need to specify *how* to reach a record, e.g. with or without an index.

Originally based upon relational algebra and tuple relational calculus, SQL consists of many types of statements,[8] which may be informally classed as sublanguages, commonly: a data query language (DQL),[a] a data definition language (DDL),[b] a data control language (DCL), and a data manipulation language (DML).[c][9] The scope of SQL includes data query, data manipulation (insert, update and delete), data definition (schema creation and modification), and data access control. Although SQL is essentially a declarative language (4GL), it also includes procedural elements.

SQL was one of the first commercial languages to utilize Edgar F. Codd's relational model. The model was described in his influential 1970 paper, "A Relational Model of Data for Large Shared Data Banks".[10] Despite not entirely adhering to the relational model as described by Codd, it became the most widely used database language.[11][12]

SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987.[13] Since then the standard has been revised to include a larger set of features. Despite the existence of standards, most SQL code requires at least some changes before being ported to different database systems.

Contents

SQL (Structured Query Language)

Paradigm	Declarative
Family	Query language
Designed by	Donald D. Chamberlin Raymond F. Boyce
Developer	ISO/IEC
First appeared	1974
Stable release	SQL:2016 / December 2016
Typing discipline	Static, strong
OS	Cross-platform
Website	www.iso.org/standard/63555.html (https://www.iso.org/standard/63555.html)
Major implementations	
Many	
Dialects	
SQL-86 · SQL-89 · SQL-92 · SQL:1999 · SQL:2003 · SQL:2006 · SQL:2008 · SQL:2011 · SQL:2016	
Influenced by	
Datalog	
Influenced	
CQL, LINQ, SPARQL, SOQL, PowerShell,[1] JPQL, jOOQ, N1QL	
Structured Query Language at Wikibooks	

SQL (file format)

Filename extension	`.sql`
Internet media type	`application/sql`[2][3]

Developed by	ISO/IEC
Initial release	1986
Type of format	Database
Standard	ISO/IEC 9075
Open format?	Yes
Website	www.iso.org/standard/63555.html (https://www.iso.org/standard/63555.html)

History

SQL was initially developed at IBM by Donald D. Chamberlin and Raymond F. Boyce after learning about the relational model from Edgar F. Codd[14] in the early 1970s.[15] 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.[15]

Chamberlin and Boyce's first attempt at a relational database language was Square, but it was difficult to use due to subscript notation. After moving to the San Jose Research Laboratory in 1973, they began work on SEQUEL.[14] The acronym SEQUEL was later changed to SQL because "SEQUEL" was a trademark of the UK-based Hawker Siddeley Dynamics Engineering Limited company.[16]

After testing SQL at customer test sites to determine the usefulness and practicality of the system, IBM began developing commercial products based on their System R prototype including System/38, SQL/DS, and DB2, which were commercially available in 1979, 1981, and 1983, respectively.[17]

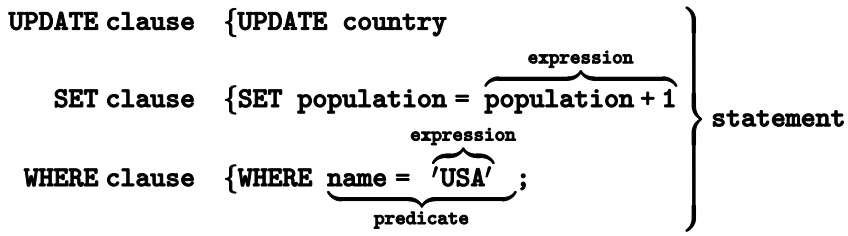
In the late 1970s, Relational Software, Inc. (now Oracle Corporation) saw the potential of the concepts described by Codd, Chamberlin, and Boyce, and developed their own SQL-based RDBMS with aspirations of selling it to the U.S. Navy, Central Intelligence Agency, and other U.S. government agencies. In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

By 1986, ANSI and ISO standard groups officially adopted the standard "Database Language SQL" language definition. New versions of the standard were published in 1989, 1992, 1996, 1999, 2003, 2006, 2008, 2011[14] and, most recently, 2016.

Syntax

The SQL language is subdivided into several language elements, including:

- *Clauses*, which are constituent components of statements and queries. (In some cases, these are optional.)[18]



A chart showing several of the SQL language elements that compose a single statement

- *Expressions*, which can produce either scalar values, or tables consisting of columns and rows of data
- *Predicates*, which specify conditions that can be evaluated to SQL three-valued logic (3VL) (true/false/unknown) or Boolean truth values and are used to limit the effects of statements and queries, or to change program flow.
- *Queries*, which retrieve the data based on specific criteria. This is an important element of *SQL*.
- *Statements*, which may have a persistent effect on schemata and data, or may control transactions, program flow, connections, sessions, or diagnostics.
  - SQL statements also include the semicolon (";") statement terminator. Though not required on every platform, it is defined as a standard part of the SQL grammar.
- *Insignificant whitespace* is generally ignored in SQL statements and queries, making it easier to format SQL code for readability.

Procedural extensions

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

Source	Abbreviation	Full name
ANSI/ISO Standard	SQL/PSM	SQL/Persistent Stored Modules
Interbase / Firebird	PSQL	Procedural SQL
IBM DB2	SQL PL	SQL Procedural Language (implements SQL/PSM)
IBM Informix	SPL	Stored Procedural Language
IBM Netezza	NZPLSQL[19]	(based on Postgres PL/pgSQL)
Invantive	PSQL[20]	Invantive Procedural SQL (implements SQL/PSM and PL/SQL)
MariaDB	SQL/PSM, PL/SQL	SQL/Persistent Stored Module (implements SQL/PSM), Procedural Language/SQL (based on Ada)[21]
Microsoft / Sybase	T-SQL	Transact-SQL
Mimer SQL	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)
MySQL	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)
MonetDB	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)
NuoDB	SSP	Starkey Stored Procedures
Oracle	PL/SQL	Procedural Language/SQL (based on Ada)
PostgreSQL	PL/pgSQL	Procedural Language/PostgreSQL Structured Query Language (based on reduced PL/SQL)
SAP R/3	ABAP	Advanced Business Application Programming
SAP HANA	SQLScript	SQLScript
Sybase	Watcom-SQL	SQL Anywhere Watcom-SQL Dialect
Teradata	SPL	Stored Procedural Language

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. Microsoft 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 unmanaged extended stored procedures primarily written in C. PostgreSQL lets users write functions in a wide variety of languages—including Perl, Python, Tcl, JavaScript (PL/V8) and C.[22]

Interoperability and standardization

Overview

SQL implementations are incompatible between vendors and do not necessarily completely follow standards. In particular date and time syntax, string concatenation, `NULL`s, and comparison case sensitivity vary from vendor to vendor. Particular exceptions are PostgreSQL[23] and Mimer SQL[24] which strive for standards compliance, though PostgreSQL does not adhere to the standard in how folding of unquoted names is done.

The folding of unquoted names to lower case in PostgreSQL is incompatible with the SQL standard,[25] which says that unquoted names should be folded to upper case.[26] Thus, `Foo` should be equivalent to `FOO` not `foo` according to the standard.

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`,[27][28] and lacks a `TIME` type)[29] and MS SQL Server (before the 2008 version). As a result, SQL code can rarely be ported between database systems without modifications.

Reasons for incompatibility

There are several reasons for this lack of portability between database systems:

- The complexity and size of the SQL standard means that most implementors do not support the entire standard.
- The standard does not specify database behavior in several important areas (e.g. indexes, file storage...), leaving implementations to decide how to behave.
- The SQL standard precisely specifies the syntax that a conforming database system must implement. However, the standard's specification of the semantics of language constructs is less well-defined, leading to ambiguity.
- Many database vendors have large existing customer bases; where the newer version of the SQL standard conflicts with the prior behavior of the vendor's database, the vendor may be unwilling to break backward compatibility.
- There is little commercial incentive for vendors to make it easier for users to change database suppliers (see vendor lock-in).
- Users evaluating database software tend to place other factors such as performance higher in their priorities than standards conformance.

Standardization history

SQL was adopted as a standard by the American National Standards Institute (ANSI) in 1986 as SQL-86[30] and the International Organization for Standardization (ISO) in 1987.[13] It is maintained by *ISO/IEC JTC 1, Information technology, Subcommittee SC 32, Data management and interchange*.

Until 1996, the National Institute of Standards and Technology (NIST) data management standards program certified SQL DBMS compliance with the SQL standard. Vendors now self-certify the compliance of their products.[31]

The original standard declared that the official pronunciation for "SQL" was an initialism: /ˌɛsˌkjuːˈɛl/ ("ess cue el").[11] Regardless, many English-speaking database professionals (including Donald Chamberlin himself[32]) use the acronym-like pronunciation of /ˈsiːkwəl/ ("sequel"),[33] mirroring the language's pre-release development name, "SEQUEL".[15][16][32][15]
The SQL standard has gone through a number of revisions:

Year	Name	Alias	Comments
1986	SQL-86	SQL-87	First formalized by ANSI.
1989	SQL-89	FIPS 127-1	Minor revision that added integrity constraints, adopted as FIPS 127-1.
1992	SQL-92	SQL2, FIPS 127-2	Major revision (ISO 9075), *Entry Level* SQL-92 adopted as FIPS 127-2.
1999	SQL:1999	SQL3	Added regular expression matching, recursive queries (e.g. transitive closure), triggers, support for procedural and control-of-flow statements, non-scalar types (arrays), and some object-oriented features (e.g. structured types). Support for embedding SQL in Java (SQL/OLB) and vice versa (SQL/JRT).
2003	SQL:2003		Introduced XML-related features (SQL/XML), window functions, standardized sequences, and columns with auto-generated values (including identity-columns).
2006	SQL:2006		ISO/IEC 9075-14:2006 defines ways that SQL can be used 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 lets applications integrate queries into their SQL code with XQuery, the XML Query Language published by the World Wide Web Consortium (W3C), to concurrently access ordinary SQL-data and XML documents.[34]
2008	SQL:2008		Legalizes ORDER BY outside cursor definitions. Adds INSTEAD OF triggers, TRUNCATE statement,[35] FETCH clause.
2011	SQL:2011		Adds temporal data (PERIOD FOR)[36] (more information at: Temporal database#History). Enhancements for window functions and FETCH clause.[37]
2016	SQL:2016		Adds row pattern matching, polymorphic table functions, JSON.
2019	SQL:2019		Adds Part 15, multidimensional arrays (MDarray type and operators).

Current Standard

The standard is commonly denoted by the pattern: *ISO/IEC 9075-n:yyyy Part n: title*, or, as a shortcut, *ISO/IEC 9075*.

ISO/IEC 9075 is complemented by *ISO/IEC 13249: SQL Multimedia and Application Packages* (SQL/MM), which defines SQL based interfaces and packages to widely spread applications like video, audio and spatial data. Interested parties may purchase SQL standards documents from ISO,[38] IEC or ANSI. A draft of SQL:2008 is freely available as a zip archive.[39]

Anatomy of SQL Standard

The SQL standard is divided into ten parts. There are gaps in the numbering due to the withdrawal of outdated parts.

- ISO/IEC 9075-1:2016 Part 1: *Framework* (SQL/Framework). It provides logical concepts.[40]
- ISO/IEC 9075-2:2016 Part 2: *Foundation* (SQL/Foundation). It contains the most central elements of the language and consists of both *mandatory and optional* features.
- ISO/IEC 9075-3:2016 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.

- ISO/IEC 9075-4:2016 Part 4: *Persistent stored modules* (SQL/PSM). It standardizes procedural extensions for SQL, including flow of control, condition handling, statement condition signals and resignals, cursors and local variables, and assignment of expressions to variables and parameters. In addition, SQL/PSM formalizes declaration and maintenance of persistent database language routines (e.g., "stored procedures"). This part of the standard consists solely of *optional* features.
- ISO/IEC 9075-9:2016 Part 9: *Management of External Data* (SQL/MED). It provides extensions to SQL that define foreign-data wrappers and datalink types to allow SQL to manage external data. External data is data that is accessible to, but not managed by, an SQL-based DBMS. This part of the standard consists solely of *optional* features.
- ISO/IEC 9075-10:2016 Part 10: *Object language bindings* (SQL/OLB). It defines the syntax and semantics of SQLJ, which is SQL embedded in Java (see also part 3). The standard also describes mechanisms to ensure binary portability of SQLJ applications, and specifies various Java packages and their contained classes. This part of the standard consists solely of *optional* features. Unlike SQL/OLB JDBC defines an API and is not part of the SQL standard.
- ISO/IEC 9075-11:2016 Part 11: *Information and definition schemas* (SQL/Schemata). It defines the Information Schema and Definition Schema, providing a common set of tools to make SQL databases and objects self-describing. These tools include the SQL object identifier, structure and integrity constraints, security and authorization specifications, features and packages of ISO/IEC 9075, support of features provided by SQL-based DBMS implementations, SQL-based DBMS implementation information and sizing items, and the values supported by the DBMS implementations.[41] This part of the standard contains both *mandatory and optional* features.
- ISO/IEC 9075-13:2016 Part 13: *SQL Routines and types using the Java TM programming language* (SQL/JRT). It specifies the ability to invoke static Java methods as routines from within SQL applications ('Java-in-the-database'). It also calls for the ability to use Java classes as SQL structured user-defined types. This part of the standard consists solely of *optional* features.
- ISO/IEC 9075-14:2016 Part 14: *XML-Related Specifications* (SQL/XML). It specifies SQL-based extensions for using XML in conjunction with SQL. The *XML* data type is introduced, as well as several routines, functions, and XML-to-SQL data type mappings to support manipulation and storage of XML in an SQL database.[34] This part of the standard consists solely of *optional* features.
- ISO/IEC 9075-15:2019 Part 15: *Multi-dimensional arrays* (SQL/MDA). It specifies a multidimensional array type (MDarray) for SQL, along with operations on MDarrays, MDarray slices, MDarray cells, and related features. This part of the standard consists solely of *optional* features.

Extensions to the ISO/IEC Standard

ISO/IEC 9075 is complemented by ISO/IEC 13249 *SQL Multimedia and Application Packages*. This closely related but separate standard is developed by the same committee. It defines interfaces and packages based on SQL. The aim is a unified access to typical database applications like text, pictures, data mining or spatial data.

- ISO/IEC 13249-1:2016 Part 1: *Framework*
- ISO/IEC 13249-2:2003 Part 2: *Full-Text*
- ISO/IEC 13249-3:2016 Part 3: *Spatial*
- ISO/IEC 13249-5:2003 Part 5: *Still image*
- ISO/IEC 13249-6:2006 Part 6: *Data mining*
- ISO/IEC 13249-7:2013 Part 7: *History*

- ISO/IEC 13249-8:xxxx Part 8: *Metadata Registry Access* MRA (https://www.iso.org/standard/73181.html) (work in progress)

ISO/IEC 9075 is also accompanied by a series of Technical Reports, published as ISO/IEC TR 19075 in 8 parts. These Technical Reports explain the justification for and usage of some features of SQL, giving examples where appropriate. The Technical Reports are non-normative; if there is any discrepancy from 9075, the text in 9075 holds. Currently available 19075 Technical Reports are:

- ISO/IEC TR 19075-1:2011 Part 1: XQuery Regular Expression Support in SQL
- ISO/IEC TR 19075-2:2015 Part 2: SQL Support for Time-Related Information
- ISO/IEC TR 19075-3:2015 Part 3: SQL Embedded in Programs using the Java™ programming language
- ISO/IEC TR 19075-4:2015 Part 4: SQL with Routines and types using the Java™ programming language
- ISO/IEC TR 19075-5:2016 Part 5: Row Pattern Recognition in SQL
- ISO/IEC TR 19075-6:2017 Part 6: SQL support for JavaScript Object Notation (JSON)
- ISO/IEC TR 19075-7:2017 Part 7: Polymorphic table functions in SQL
- ISO/IEC TR 19075-8:2019 Part 8: Multi-Dimensional Arrays (SQL/MDA)

Alternatives

A distinction should be made between alternatives to SQL as a language, and alternatives to the relational model itself. Below are proposed relational alternatives to the SQL language. See navigational database and NoSQL for alternatives to the relational model.

- .QL: object-oriented Datalog
- 4D Query Language (4D QL)
- Datalog: critics suggest that Datalog has two advantages over SQL: it has cleaner semantics, which facilitates program understanding and maintenance, and it is more expressive, in particular for recursive queries.[42]
- HTSQL: URL based query method
- IBM Business System 12 (IBM BS12): one of the first fully relational database management systems, introduced in 1982
- ISBL
- jOOQ: SQL implemented in Java as an internal domain-specific language
- Java Persistence Query Language (JPQL): The query language used by the Java Persistence API and Hibernate persistence library
- JavaScript: MongoDB implements its query language in a JavaScript API.
- LINQ: Runs SQL statements written like language constructs to query collections directly from inside .Net code.
- Object Query Language
- QBE (Query By Example) created by Moshè Zloof, IBM 1977
- Quel introduced in 1974 by the U.C. Berkeley Ingres project.
- Tutorial D
- XQuery

Distributed SQL processing

Distributed Relational Database Architecture (DRDA) was designed by a work group within IBM in the period 1988 to 1994. DRDA enables network connected relational databases to cooperate to fulfill SQL requests.[43][44]

An interactive user or program can issue SQL statements to a local RDB and receive tables of data and status indicators in reply from remote RDBs. SQL statements can also be compiled and stored in remote RDBs as packages and then invoked by package name. This is important for the efficient operation of application programs that issue complex, high-frequency queries. It is especially important when the tables to be accessed are located in remote systems.

The messages, protocols, and structural components of DRDA are defined by the Distributed Data Management Architecture.

Criticisms

Design

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).

Critics argue that SQL should be replaced with a language that returns strictly to the original foundation: for example, see *The Third Manifesto*. However, no known proof exists that such uniqueness cannot be added to SQL itself,[45] or at least a variation of SQL. In other words, it's quite possible that SQL can be "fixed" or at least improved in this regard such that the industry may not have to switch to a completely different query language to obtain uniqueness. Debate on this remains open.

Other criticisms

Chamberlin discusses four historical criticisms of SQL in a 2012 paper:[14]

Orthogonality and completeness

Early specifications did not support major features, such as primary keys. Result sets could not be named, and sub-queries had not been defined. These were added in 1992.[14]

Null

The concept of Null is the subject of some debates. The Null marker indicates that there is no value, even no 0 for an integer column or a string of length 0 for a text column. The concept of Nulls enforces the 3-valued-logic in SQL, which is a concrete implementation of the general 3-valued logic.

Duplicates

Another popular criticism is that it allows duplicate rows, making integration with languages such as Python, whose data types might make it difficult to accurately represent the data,[14] difficult in terms of parsing and by the absence of modularity.[46]

This can be avoided declaring a unique constraint with one or more fields that identifies uniquely a row in the table. That constraint could also become the primary key of the table.

Impedance mismatch

In a similar sense to Object-relational impedance mismatch, there is a mismatch between the declarative SQL language and the procedural languages that SQL is typically embedded in.

Data Integrity Categories

Main data integrity categories of each RDBMS.

Entity integrity

Establishes that within the table the primary key has a unique value for each row, checking the uniqueness of the value of the primary key avoiding that there are duplicated rows in a table.

Domain integrity

Restricts the type, format, and value range that applies to valid entries for a column within a table

Referential integrity

Makes rows in a table that are being used by other records impossible to delete

User-defined integrity

Other specific rules not included above apply

SQL data types

The SQL standard defines three kinds of data types:

- predefined data types
- constructed types
- user-defined types.

Predefined data types

- Character Types
  - Character (CHAR)
  - Character Varying (VARCHAR)
  - Character Large Object (CLOB)
- Binary Types

    - Binary (BINARY)
    - Binary Varying (VARBINARY)
    - Binary Large Object (BLOB)

- Numeric Types
    - Exact Numeric Types (NUMERIC, DECIMAL, SMALLINT, INTEGER, BIGINT)
    - Approximate Numeric Types (FLOAT, REAL, DOUBLE PRECISION)

- Datetime Types (DATE, TIME, TIMESTAMP)
- Interval Type (INTERVAL)
- Boolean
- XML
- JSON

Constructed types

Constructed types are one of ARRAY, MULTISET, REF(erence), or ROW.
User-defined types are comparable to classes in object-oriented language with their own constructors, observers, mutators, methods, inheritance, overloading, overwriting, interfaces, and so on.

See also

- Relational database
- Object database
- Object-relational database
- List of relational database management systems
- Comparison of relational database management systems
- Comparison of object-relational database management systems
- D (data language specification)
- D4 (programming language)
- Query by Example
- SQL syntax
- Oracle PL/SQL
- Microsoft Transact-SQL (T-SQL)
- Online transaction processing (OLTP)
- Online analytical processing (OLAP)
- Data warehouse
- Relational data stream management system
- NoSQL
- MUMPS
- Hierarchical model
- Star schema
- Snowflake schema

Notes

a. Formally, "SQL-data" statements *excluding* "SQL-data change" statements; this is primarily the Select statement.
b. Formally, "SQL-schema" statements.
c. Formally, "SQL-data change" statements

References

1. Paul, Ryan. "A guided tour of the Microsoft Command Shell" (https://arstechnica.com/business/news/2005/10/msh.ars/4). *Ars Technica*. Retrieved 10 April 2011.
2. "Media Type registration for application/sql" (http://www.iana.org/assignments/media-types/application/sql). Internet Assigned Numbers Authority. 10 April 2013. Retrieved 10 April 2013.
3. "The application/sql Media Type, RFC 6922" (http://tools.ietf.org/html/rfc6922). Internet Engineering Task Force. April 2013. p. 3. Retrieved 10 April 2013.
4. Beaulieu, Alan (April 2009). Mary E Treseler (ed.). *Learning SQL* (2nd ed.). Sebastopol, CA, USA: O'Reilly. ISBN 978-0-596-52083-0.
5. "SQL" (http://www.britannica.com/EBchecked/topic/569684/SQL). *Britannica.com*. Retrieved 2013-04-02.
6. "SQL" (http://oxforddictionaries.com/definition/american_english/SQL). *Oxforddictionaries.com*. Retrieved 2017-01-16.
7. "Structured Query Language (SQL)" (http://msdn.microsoft.com/en-gb/library/windows/desktop/ms714670(v=vs.85).aspx). *Msdn.microsoft.com*. Retrieved 2017-01-16.
8. SQL-92, 4.22 SQL-statements, 4.22.1 Classes of SQL-statements "There are at least five ways of classifying SQL-statements:", 4.22.2, SQL statements classified by function "The following are the main classes of SQL-statements:"; SQL:2003 4.11 SQL-statements, and later revisions.
9. Chatham, Mark (2012). *Structured Query Language By Example - Volume I: Data Query Language*. p. 8 (https://books.google.com/books?id=64MBBAAAQBAJ&pg=PA8). ISBN 978-1-29119951-2.
10. Codd, Edgar F. (June 1970). "A Relational Model of Data for Large Shared Data Banks". *Communications of the ACM*. **13** (6): 377–87. CiteSeerX 10.1.1.88.646 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.88.646). doi:10.1145/362384.362685 (https://doi.org/10.1145%2F362384.362685).
11. Chapple, Mike. "SQL Fundamentals" (http://databases.about.com/od/sql/a/sqlfundamentals.htm). *Databases*. About.com. Retrieved 2009-01-28.
12. "Structured Query Language (SQL)" (http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp?topic=com.ibm.db2.udb.admin.doc/doc/c0004100.htm). International Business Machines. October 27, 2006. Retrieved 2007-06-10.
13. "ISO 9075:1987: Information technology – Database languages – SQL – Part 1: Framework (SQL/Framework)" (https://www.iso.org/standard/16661.html). 1987-06-01.
14. Chamberlin, Donald (2012). "Early History of SQL". *IEEE Annals of the History of Computing*. **34** (4): 78–82. doi:10.1109/MAHC.2012.61 (https://doi.org/10.1109%2FMAHC.2012.61).
15. Chamberlin, Donald D; Boyce, Raymond F (1974). "SEQUEL: A Structured English Query Language" (https://web.archive.org/web/20070926212100/http://www.almaden.ibm.com/cs/people/chamberlin/sequel-1974.pdf) (PDF). *Proceedings of the 1974 ACM SIGFIDET Workshop on Data Description, Access and Control*. Association for Computing Machinery: 249–64. Archived from the original (http://www.almaden.ibm.com/cs/people/chamberlin/sequel-1974.pdf) (PDF) on 2007-09-26. Retrieved 2007-06-09.
16. Oppel, Andy (February 27, 2004). *Databases Demystified* (http://www.mhprofessional.com/product.php?cat=112&isbn=0071469605). San Francisco, CA: McGraw-Hill Osborne Media. pp. 90–1. ISBN 978-0-07-146960-9.

17. "History of IBM, 1978" (http://www-03.ibm.com/ibm/history/history/year_1978.html). *IBM Archives*. IBM. Retrieved 2007-06-09.
18. ANSI/ISO/IEC International Standard (IS). Database Language SQL—Part 2: Foundation (SQL/Foundation). 1999.
19. "IBM PureData System for Analytics, Version 7.0.3" (http://pic.dhe.ibm.com/infocenter/ntz/v7r0m3/index.jsp?topic=%2Fcom.ibm.nz.sproc.doc%2Fc_sproc_stored_procs.html).
20. "Invantive Procedural SQL" (https://www.invantive.com/products/invantive-sql/grammar).
21. "CREATE PROCEDURE" (https://mariadb.com/kb/en/library/create-procedure/). *MariaDB KnowledgeBase*. Retrieved 2019-04-23.
22. PostgreSQL contributors (2011). "PostgreSQL server programming" (http://www.postgresql.org/docs/9.1/static/server-programming.html). *PostgreSQL 9.1 official documentation*. postgresql.org. Retrieved 2012-03-09.
23. PostgreSQL contributors (2012). "About PostgreSQL" (http://www.postgresql.org/about/). *PostgreSQL 9.1 official website*. PostgreSQL Global Development Group. Retrieved March 9, 2012. "PostgreSQL prides itself in standards compliance. Its SQL implementation strongly conforms to the ANSI-SQL:2008 standard"
24. "Mimer SQL, Built on Standards" (http://developer.mimer.com/features/feature_6.htm). *Mimer SQL official website*. Mimer Information Technology. 2009.
25. "4.1. Lexical Structure" (https://www.postgresql.org/docs/current/static/sql-syntax-lexical.html#SQL-SYNTAX-IDENTIFIERS). *PostgreSQL documentation*. 2018.
26. "(Second Informal Review Draft) ISO/IEC 9075:1992, Database Language SQL, Section 5.2, syntax rule 11" (http://www.contrib.andrew.cmu.edu/~shadow/sql/sql1992.txt). 30 July 1992.
27. Lorentz, Diana; Roeser, Mary Beth; Abraham, Sundeep; Amor, Angela; Arora, Geeta; Arora, Vikas; Ashdown, Lance; Baer, Hermann; Bellamkonda, Shrikanth (October 2010) [1996]. "Basic Elements of Oracle SQL: Data Types" (http://download.oracle.com/docs/cd/E11882_01/server.112/e17118/sql_elements001.htm#sthref154). *Oracle Database SQL Language Reference 11g Release 2 (11.2)*. Oracle Database Documentation Library. Redwood City, CA: Oracle USA, Inc. Retrieved December 29, 2010. "For each `DATE` value, Oracle stores the following information: century, year, month, date, hour, minute, and second"
28. Lorentz, Diana; Roeser, Mary Beth; Abraham, Sundeep; Amor, Angela; Arora, Geeta; Arora, Vikas; Ashdown, Lance; Baer, Hermann; Bellamkonda, Shrikanth (October 2010) [1996]. "Basic Elements of Oracle SQL: Data Types" (http://download.oracle.com/docs/cd/E11882_01/server.112/e17118/sql_elements001.htm#sthref154). *Oracle Database SQL Language Reference 11g Release 2 (11.2)*. Oracle Database Documentation Library. Redwood City, CA: Oracle USA, Inc. Retrieved December 29, 2010. "The datetime data types are `DATE`..."
29. Lorentz, Diana; Roeser, Mary Beth; Abraham, Sundeep; Amor, Angela; Arora, Geeta; Arora, Vikas; Ashdown, Lance; Baer, Hermann; Bellamkonda, Shrikanth (October 2010) [1996]. "Basic Elements of Oracle SQL: Data Types" (http://download.oracle.com/docs/cd/E11882_01/server.112/e17118/sql_elements001.htm#i54335). *Oracle Database SQL Language Reference 11g Release 2 (11.2)*. Oracle Database Documentation Library. Redwood City, CA: Oracle USA, Inc. Retrieved December 29, 2010. "Do not define columns with the following SQL/DS and DB2 data types, because they have no corresponding Oracle data type:... `TIME`"
30. "Finding Aid" (http://special.lib.umn.edu/findaid/xml/cbi00168.xml). *X3H2 Records, 1978–95*. American National Standards Institute.
31. Doll, Shelley (June 19, 2002). "Is SQL a Standard Anymore?" (https://web.archive.org/web/20120705163024/http://www.techrepublic.com/article/is-sql-a-standard-anymore/1046268). *TechRepublic's Builder.com*. TechRepublic. Archived from the original (http://articles.techrepublic.com.com/5100-10878_11-1046268.html) on 2012-07-05. Retrieved 2016-04-12.
32. Gillespie, Patrick. "Pronouncing SQL: S-Q-L or Sequel?" (http://patorjk.com/blog/2012/01/26/pronouncing-sql-s-q-l-or-sequel/). Retrieved 12 February 2012.

33. Melton, Jim; Alan R Simon (1993). "1.2. What is SQL?" (https://archive.org/details/understandingnew00melt). *Understanding the New SQL: A Complete Guide* (https://archive.org/details/understandingnew00melt/page/536). Morgan Kaufmann. p. 536 (https://archive.org/details/understandingnew00melt/page/536). ISBN 978-1-55860-245-8. "SQL (correctly pronounced "ess cue ell," instead of the somewhat common "sequel")..."
34. Wagner, Michael (2010). *SQL/XML:2006 - Evaluierung der Standardkonformität ausgewählter Datenbanksysteme*. Diplomica Verlag. p. 100. ISBN 978-3-8366-9609-8.
35. "SQL:2008 now an approved ISO international standard" (https://web.archive.org/web/20110628130925/http://iablog.sybase.com/paulley/2008/07/sql2008-now-an-approved-iso-international-standard/). Sybase. July 2008. Archived from the original (http://iablog.sybase.com/paulley/2008/07/sql2008-now-an-approved-iso-international-standard/) on 2011-06-28.
36. Krishna Kulkarni, Jan-Eike Michels (September 2012). "Temporal features in SQL:2011" (http://cs.ulb.ac.be/public/_media/teaching/infoh415/tempfeaturessql2011.pdf) (PDF). *SIGMOD Record*. **41** (3).
37. Fred Zemke (2012). "What's new in SQL:2011" (https://sigmodrecord.org/publications/sigmodRecord/1203/pdfs/10.industry.zemke.pdf) (PDF). Oracle Corporation.
38. "ISO/IEC 9075-2:2016: Information technology -- Database languages -- SQL -- Part 2: Foundation (SQL/Foundation)" (https://www.iso.org/standard/63556.html). December 2016.
39. *SQL:2008 draft* (http://www.wiscorp.com/sql200n.zip) (Zip), Whitemarsh Information Systems Corporation
40. "ISO/IEC 9075-1:2016: Information technology – Database languages – SQL – Part 1: Framework (SQL/Framework)" (https://www.iso.org/standard/63555.html).
41. *ISO/IEC 9075-11:2008: Information and Definition Schemas (SQL/Schemata)* (https://www.iso.org/standard/38645.html)
42. Fernando Saenz-Perez. "Outer Joins in a Deductive Database System" (http://lbd.udc.es/jornadas2011/actas/PROLE/PROLE/S5/13_article.pdf) (PDF). *Lbd.udc.es*. Retrieved 2017-01-16.
43. Reinsch, R. (1988). "Distributed database for SAA". *IBM Systems Journal*. **27** (3): 362–389. doi:10.1147/sj.273.0362 (https://doi.org/10.1147%2Fsj.273.0362).
44. *Distributed Relational Database Architecture Reference*. IBM Corp. SC26-4651-0. 1990.
45. "Khan Academy | Free Online Courses, Lessons & Practice" (https://www.khanacademy.org/). *Khan Academy*. Retrieved 2020-05-29.
46. Schauder, Jen. "Why SQL Sucks" (http://blog.schauderhaft.de/2010/02/15/why-sql-sucks/). *Schauderhaft*. Retrieved 3 February 2018.

Sources

- Codd, Edgar F (June 1970). "A Relational Model of Data for Large Shared Data Banks" (https://web.archive.org/web/20070612235326/http://www.acm.org/classics/nov95/toc.html). *Communications of the ACM*. **13** (6): 377–87. doi:10.1145/362384.362685 (https://doi.org/10.1145%2F362384.362685). Archived from the original (http://www.acm.org/classics/nov95/toc.html) on 2007-06-12.
- Discussion on alleged SQL flaws (C2 wiki)
- C. J. Date with Hugh Darwen: *A Guide to the SQL standard : a users guide to the standard database language SQL, 4th ed.*, Addison Wesley, USA 1997, ISBN 978-0-201-96426-4

SQL standards documents

ITTF publicly available standards and technical reports

The ISO/IEC Information Technology Task Force publishes publicly available standards (http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html) including SQL. Technical Corrigenda (corrections) and Technical Reports (discussion documents) are published there.

SQL -- Part 1: Framework (SQL/Framework) (http://standards.iso.org/ittf/PubliclyAvailableStandards/c053681_ISO_IEC_9075-1_2011.zip)

Draft documents

Formal SQL standards are available from ISO and ANSI for a fee. For informative use, as opposed to strict standards compliance, late drafts often suffice.

- SQL:2011 draft (http://www.wiscorp.com/sql20nn.zip)
- SQL-92 draft (http://www.contrib.andrew.cmu.edu/~shadow/sql/sql1992.txt)

External links

- *1995 SQL Reunion: People, Projects, and Politics*, by Paul McJones (ed.) (http://www.mcjones.org/System_R/SQL_Reunion_95/sqlr95.html): transcript of a reunion meeting devoted to the personal history of relational databases and SQL.
- American National Standards Institute. X3H2 Records, 1978–1995 (http://special.lib.umn.edu/findaid/xml/cbi00168.xml) Charles Babbage Institute Collection documents the H2 committee's development of the NDL and SQL standards.
- Oral history interview with Donald D. Chamberlin (http://purl.umn.edu/107215) Charles Babbage Institute In this oral history Chamberlin recounts his early life, his education at Harvey Mudd College and Stanford University, and his work on relational database technology. Chamberlin was a member of the System R research team and, with Raymond F. Boyce, developed the SQL database language. Chamberlin also briefly discusses his more recent research on XML query languages.
- Comparison of Different SQL Implementations (http://troels.arvin.dk/db/rdbms/) This comparison of various SQL implementations is intended to serve as a guide to those interested in porting SQL code between various RDBMS products, and includes comparisons between SQL:2008, PostgreSQL, DB2, MS SQL Server, MySQL, Oracle, and Informix.
- Event stream processing with SQL (https://web.archive.org/web/20140706215458/http://www.sqlstream.com/stream-processing-with-sql/) - An introduction to real-time processing of streaming data with continuous SQL queries
- BNF Grammar for ISO/IEC 9075:2003, part 2 SQL/Framework (https://github.com/ronsavage/SQL)