

PHP

PHP (a recursive acronym for PHP: Hypertext Preprocessor) is a computer scripting language, originally designed for producing dynamic web pages. It is for server-side scripting, but can be used from a command line interface or in standalone graphical applications.

While PHP was originally created by Rasmus Lerdorf in 1995, the main implementation of PHP is now produced by The PHP Group and serves as the de facto standard for PHP as there is no formal specification. Released under the PHP License, the Free Software Foundation considers it to be free software.

PHP is a widely-used general-purpose scripting language that is especially suited for web development and can be embedded into HTML. It generally runs on a web server, taking PHP code as its input and creating web pages as output. It can be deployed on most web servers and on almost every operating system and platform free of charge. PHP is installed on more than 20 million websites and 1 million web servers. It is also the most popular Apache module among computers using Apache as a web server. The most recent major release of PHP was version 5.2.6 on May 1, 2008.

PHP originally stood for Personal Home Page. It began in 1994 as a set of Common Gateway Interface binaries written in the C programming language by the Danish/Greenlandic programmer Rasmus Lerdorf. Lerdorf initially created these Personal Home Page Tools to replace a small set of Perl scripts he had been using to maintain his personal homepage. The tools were used to perform tasks such as displaying his résumé and recording how much traffic his page was receiving. He combined these binaries with his Form Interpreter to create PHP/FI, which had more functionality. PHP/FI included a larger C implementation and could communicate with databases enabling the building of simple, dynamic web applications. He released PHP publicly on June 8, 1995 to speed up the finding of bugs and improving the code. This release was named PHP version 2 and already had the basic functionality that PHP has today. This included Perl-like variables, form handling, and the ability to embed HTML. The syntax was similar to Perl but was more limited, simpler, and less consistent.

Zeev Suraski and Andi Gutmans, two Israeli developers at the Technion IIT, rewrote the parser in 1997 and formed the base of PHP 3, changing the language's name to the recursive initialism PHP: Hypertext Preprocessor. The development team officially released PHP/FI 2 in November 1997 after months of beta testing. Afterwards, public testing of PHP 3 began, and the official launch came in June 1998. Suraski and Gutmans then started a new rewrite of PHP's core, producing the Zend Engine in 1999. They also founded Zend Technologies in Ramat Gan, Israel, which manages the development of PHP.

On May 22, 2000, PHP 4, powered by the Zend Engine 1.0, was released. On July 13, 2004, PHP 5 was released, powered by the new Zend Engine II. PHP 5 included new features such as improved support for object-oriented programming, the PHP Data Objects extension (which defines a lightweight and consistent interface for accessing databases), and numerous performance enhancements. The most recent update released by The PHP Group is for the older PHP version 4 code branch. As of January 2008, this branch is up to version 4.4.8. PHP 4 is no longer under active development but will be supported by security updates until August 8, 2008.

In 2008, PHP 5 became the only stable version under development. Late static binding has been missing from PHP and will be added in version 5.3.

Alongside PHP 5, PHP 6 is also under active development. Major changes include the removal of register_globals, magic quotes, and safe mode.

PHP does not have complete native support for Unicode or multibyte strings; unicode support will be included in PHP 6. Many high profile open source projects ceased to support PHP 4 in new code as of February 5, 2008, due to the GoPHP5 initiative, provided by a consortium of PHP developers promoting the transition from PHP 4 to PHP 5. It runs in both 32-bit and 64-bit environments, but on Windows the only official distribution is 32-bit, requiring Windows

32-bit compatibility mode to be enabled while using IIS in a 64-bit Windows environment. There is a third-party distribution available for 64-bit Windows.

Release history

Meaning

Red

Old release; not supported

Yellow

Old release; still supported

Green

Current release

Blue

Future release

Major Version

Minor Version

Release date

Notes

1.0

1.0.0

1995-06-08

Officially called "Personal Home Page Tools (PHP Tools)". This is the first use of the name "PHP".

2.0

2.0.0

1996-04-16

Considered with its creator as the "fastest and simplest tool" for creating dynamic web pages.

3.0

3.0.0

1998-06-06

Development moves from one person to multiple developers. Zeev Suraski and Andi Gutmans rewrite the base for this version.

4.0

4.0.0

2000-05-22

Added more advanced two-stage parse/execute tag-parsing system called the Zend engine.

4.1.0

2001-12-10

Introduced 'superglobals' (\$_GET, \$_POST, \$_SESSION, etc.)

4.2.0

2002-04-22

Disabled register_globals by default. Data received over the network is not inserted directly into the global namespace anymore, closing possible security holes in applications.

4.3.0

2002-12-27

Introduced the CLI, in addition to the CGI.

4.4.0

2005-07-11

Added man pages for phpize and php-config scripts.

4.4.8

2008-01-03

Several security enhancements and bug fixes. End of life release for PHP 4. Security updates only until 2008-08-08, if

necessary.

5.0

5.0.0

2004-07-13

Zend Engine II with a new object model.

5.1.0

2005-11-24

Performance improvements with introduction of compiler variables in re-engineered PHP Engine.

5.2.0

2006-11-02

Enabled the filter extension by default.

5.2.6

2008-05-01

Several security enhancements and bug fixes

5.3.0

No date set

Namespace support; Improved XML support through use of XMLReader and XMLWriter; SOAP support, Late static bindings, Jump label (limited goto)

6.0

6.0.0

No date set

Unicode support; removal of 'ereg ()', 'register_globals', 'magic_quotes' and 'safe_mode'; Alternative PHP Cache; Removal of mime_magic and rewrite of fileinfo() for better MIME support

Usage

PHP is a general-purpose scripting language that is especially suited for web development. PHP generally runs on a web server, taking PHP code as its input and creating web pages as output. It can also be used for command-line scripting and client-side GUI applications. PHP can be deployed on most web servers, many operating systems and platforms, and can be used with many relational database management systems. It is available free of charge, and the PHP Group provides the complete source code for users to build, customize and extend for their own use.

PHP primarily acts as a filter, taking input from a file or stream containing text and/or PHP instructions and outputs another stream of data; most commonly the output will be HTML. It can automatically detect the language of the user. From PHP 4, the PHP parser compiles input to produce byte code for processing by the Zend Engine, giving improved performance over its interpreter predecessor.

Originally designed to create dynamic web pages, PHP's principal focus is server-side scripting, and it is similar to other server-side scripting languages that provide dynamic content from a web server to a client, such as Microsoft's ASP.NET system, Sun Microsystems' JavaServer Pages, and mod_perl. PHP has also attracted the development of many frameworks that provide building blocks and a design structure to promote rapid application development (RAD). Some of these include CakePHP, PRADO, Symfony and Zend Framework, offering features similar to other web application frameworks.

The LAMP architecture has become popular in the web industry as a way of deploying web applications. PHP is commonly used as the P in this bundle alongside Linux, Apache and MySQL, although the P may also refer to Python or Perl.

As of April 2007, over 20 million Internet domains were hosted on servers with PHP installed, and PHP was recorded as the most popular Apache module.

Security

Insecure scripts written in PHP are popular targets of hackers who exploit poorly built applications written in PHP. Software vulnerabilities related to PHP are identified among the CVE (Common Vulnerabilities and Exposures) records, available from the National Vulnerability Database. The proportion of vulnerabilities related to PHP, out of the total of all

common vulnerabilities, amounted to: 12% in 2003, 20% in 2004, 28% in 2005, 43% in 2006, 36% in 2007, and 33.8% for the first quarter of 2008. More than a quarter of all software vulnerabilities listed in this database are related to scripts written in PHP, and more than a third of vulnerabilities listed recently. Most of these vulnerabilities can be exploited remotely, that is without being logged on the computer hosting the vulnerable application. Such exploitation is made possible due to poor programming habits, such as failing to check data before entering it into a database, and features of the language such as register globals, which is now deprecated. These result in code injection, cross-site scripting and other application security issues. Such attacks are not exclusive to PHP and most can be avoided simply by following proper coding techniques and principles.

Syntax

Syntax-highlighted PHP code

PHP only parses code within its delimiters. Anything outside its delimiters is sent directly to the output and is not parsed by PHP. The most common delimiters are `<?php` and `?>`, which are open and close delimiters respectively. `<script language="php">` and `</script>` delimiters are also available. Short tags (`<?` or `<?=` and `?>`) are also commonly used, but like ASP-style tags (`<%` or `<%=` and `%>`), they are less portable as they can be disabled in the PHP configuration. For this reason, the use of short tags and ASP-style tags is discouraged. The purpose of these delimiters is to separate PHP code from non-PHP code, including HTML. Everything outside the delimiters is ignored by the parser and is passed through as output.

Variables are prefixed with a dollar symbol and a type does not need to be specified in advance. Unlike function and class names, variable names are case sensitive. Both double-quoted ("") and heredoc strings allow the ability to embed a variable's value into the string. PHP treats newlines as whitespace in the manner of a free-form language (except when inside string quotes), and statements are terminated by a semicolon. PHP has three types of comment syntax: `/* */` serves as block comments, and `//` as well as `#` are used for inline comments.

To output text to the browser, either the `print` function or the `echo` function is used. Both functions are nearly identical; the major difference is that `print` is slower than `echo` because the former will return the integer value of 1 in order to behave more like a function, rather than a language construct, whereas the latter does not return a status and only returns the text for output.

Because `print` will behave more like a function, it can be used in expressions where `echo` cannot. It is important to understand however that both are considered language constructs and not functions, the only difference is in their behavior, as outlined above.

Data types

PHP stores whole numbers in a platform-dependent range. This range is typically that of 32-bit signed integers. Unsigned integers are converted to signed values in certain situations; this behavior is different from other programming languages. Integer variables can be assigned using decimal (positive and negative), octal, and hexadecimal notations. Real numbers are also stored in a platform-specific range. They can be specified using floating point notation, or two forms of scientific notation. PHP has a native Boolean type that is similar to the native Boolean types in Java and C++. Using the Boolean type conversion rules, non-zero values are interpreted as true and zero as false as in Perl and C++. The null data type represents a variable that has no value. The only value in the null data type is NULL. Variables of the "resource" type represent references to resources from external sources. These are typically created by functions from a particular extension, and can only be processed by functions from the same extension; examples include file, image, and database resources. Arrays can contain elements of any type that PHP can handle, including resources, objects, and even other arrays. Order is preserved in lists of values and in hashes with both keys and values, and the two can be intermingled. PHP also supports strings, which can be used with single quotes, double quotes, or heredoc syntax.

Functions

PHP has hundreds of base functions and thousands more from extensions. Functions are not first-class functions and can only be referenced by their name. User-defined functions can be created at any time without being prototyped. Functions can be defined inside code blocks, permitting a run-time decision as to whether or not a function should be defined. Function calls must use parentheses, with the exception of zero argument class constructor functions called with the PHP new operator, where parentheses are optional. PHP supports quasi-anonymous functions through the create function() function, although they are not true anonymous functions because anonymous functions are nameless, but functions can only be referenced by name, or indirectly through a variable \$function_name();, in PHP.]

Objects

Basic object-oriented programming functionality was added in PHP 3. Object handling was completely rewritten for PHP 5, expanding the feature set and enhancing performance. In previous versions of PHP, objects were handled like primitive types. The drawback of this method was that the whole object was copied when a variable was assigned or passed as a parameter to a method. In the new approach, objects are referenced by handle, and not by value. PHP 5 introduced private and protected member variables and methods, along with abstract classes and final classes as well as abstract methods and final methods. It also introduced a standard way of declaring constructors and destructors, similar to that of other object-oriented languages such as C++, and a standard exception handling model. Furthermore, PHP 5 added interfaces and allowed for multiple interfaces to be implemented. There are special interfaces that allow objects to interact with the runtime system. Objects implementing ArrayAccess can be used with array syntax and objects implementing Iterator or IteratorAggregate can be used with the foreach language construct. There is no virtual table feature in the engine, so static variables are bound with a name instead of a reference at compile time.

If the developer creates a copy of an object using the reserved word clone, the Zend engine will check if a __clone () method has been defined or not. If not, it will call a default __clone () which will copy the object's properties. If a __clone () method is defined, then it will be responsible for setting the necessary properties in the created object. For convenience, the engine will supply a function that imports the properties of the source object, so that the programmer can start with a by-value replica of the source object and only override properties that need to be changed.

Resources

PHP includes free and open source libraries with the core build. PHP is a fundamentally Internet-aware system with modules built in for accessing FTP servers, many database servers, embedded SQL libraries such as embedded MySQL and SQLite, LDAP servers, and others. Many functions familiar to C programmers such as those in the stdio family are available in the standard PHP build. PHP has traditionally used features such as "magic_quotes_gpc" and "magic_quotes_runtime" which attempt to escape apostrophes (') and quotes (") in strings in the assumption that they will be used in databases, to prevent SQL injection attacks. This leads to confusion over which data is escaped and which is not and to problems when data is not in fact used as input to a database and when the escaping used is not completely correct. To make code portable between servers which do and do not use magic quotes, developers can preface their code with a script to reverse the effect of magic quotes when it is applied.

PHP allows developers to write extensions in C to add functionality to the PHP language. These can then be compiled into PHP or loaded dynamically at runtime. Extensions have been written to add support for the Windows API, process management on Unix-like operating systems, multibyte strings (Unicode), cURL, and several popular compression formats. Some more unusual features include integration with Internet relay chat, dynamic generation of images and Adobe Flash content, and even speech synthesis. The PHP Extension Community Library (PECL) project is a repository for extensions to the PHP language.

As with many scripting languages, PHP scripts are normally kept as human-readable source code, even on production web servers. While this allows flexibility, releasing scripts in source form is undesirable for commercial software developers, and can raise issues with security of web servers; as an example, if a hacker acquires control of a server,

PHP

database passwords may be quickly discovered, and undesirable changes to scripts may be made that remain undiscovered indefinitely.

Code optimizers improve the quality of the compiled code by reducing its size and making changes that can reduce the execution time and improve performance. The nature of the PHP compiler is such that there are often opportunities for code optimization, and an example of a code optimizer is the Zend Optimizer PHP extension.

PHP accelerators can offer significant performance gains by caching the compiled form of a PHP script in shared memory to avoid the overhead of parsing and compiling the code every time the script runs.

Certification

Zend provides a certification program for programmers to become certified PHP developers.

MySQL

MySQL is a relational database management system (RDBMS) which has more than 11 million installations. The program runs as a server providing multi-user access to a number of databases.

MySQL is owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now a subsidiary of Sun Microsystems, which holds the copyright to most of the codebase. The project's source code is available under terms of the GNU General Public License, as well as under a variety of proprietary agreements.

"MySQL" is officially pronounced /maɪˈsɪkjʊəl/, not "My sequel" /maɪˈsiːkwəl/. This adheres to the official ANSI pronunciation; SEQUEL was an earlier IBM database language, a predecessor to the SQL language. The company does not take issue with the pronunciation "My sequel" or other local variations.

Uses

MySQL is popular for web applications and acts as the database component of the LAMP, BAMP, MAMP, and WAMP platforms (Linux/BSD/Mac/Windows-Apache-MySQL-PHP/Perl/Python), and for open-source bug tracking tools like Bugzilla. Its popularity for use with web applications is closely tied to the popularity of PHP and Ruby on Rails, which are often combined with MySQL. PHP and MySQL are essential components for running popular content management systems such as Drupal, e107, Joomla!, WordPress and some BitTorrent trackers. Wikipedia runs on MediaWiki software, which is written in PHP and uses a MySQL database.

Platforms and interfaces

MySQL is written in C and C++. The SQL parser uses yacc and a home-brewed lexer.

MySQL works on many different system platforms, including AIX, BSDi, FreeBSD, HP-UX, i5/OS, Linux, Mac OS X, NetBSD, Novell NetWare, OpenBSD, eComStation, OS/2 Warp, QNX, IRIX, Solaris, SunOS, SCO OpenServer, SCO UnixWare, Tru64, Windows 95, Windows 98, Windows ME, Windows NT, Windows 2000, Windows XP, and the 32-bit version of Windows Vista (but not the 64-bit version). A port of MySQL to OpenVMS is also available.]

Libraries for accessing MySQL databases are available in all major programming languages with language-specific APIs. In addition, an ODBC interface called MyODBC allows additional programming languages that support the ODBC interface to communicate with a MySQL database, such as ASP or ColdFusion. The MySQL server and official libraries are mostly implemented in ANSI C/ANSI C++.

To administer MySQL databases one can use the included command-line tool (commands: mysql and mysqladmin). Also downloadable from the MySQL site are GUI administration tools: MySQL Administrator and MySQL Query Browser. Both of the GUI tools are now included in one package called tools/5.0.html MySQL GUI Tools.

In addition to the above mentioned tools developed by MySQL AB, there are several other commercial and non-commercial tools available. One can try phpMyAdmin a free software webbased administration interface implemented in PHP or SQLyog Community Edition a Free Desktop based GUI tool.

Features

As of August 2007, MySQL offers MySQL 5.0 in two different variants: the MySQL Community Server and Enterprise Server. They have a common code base and include the following features:

- A broad subset of ANSI SQL 99, as well as extensions
- Cross-platform support
- Stored procedures
- Triggers
- Cursors
- Updatable Views
- True VARCHAR support
- INFORMATION_SCHEMA
- Strict mode
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using Oracle's InnoDB engine
- Independent storage engines (MyISAM for read speed, InnoDB for transactions and referential integrity, MySQL Archive for storing historical data in little space)
- Transactions with the InnoDB, BDB and Cluster storage engines; savepoints with InnoDB
- SSL support
- Query caching
- Sub-SELECTs (i.e. nested SELECTs)
- Replication with one master per slave, many slaves per master, no automatic support for multiple masters per slave.
- Full-text indexing and searching using MyISAM engine
- Embedded database library
- Partial Unicode support (UTF-8 sequences longer than 3 bytes are not supported; UCS-2 encoded strings are also limited to the BMP)
- ACID compliance using the InnoDB, BDB and Cluster engines
- Shared-nothing clustering through MySQL Cluster

The MySQL Enterprise Server is released once per month and the sources can be obtained either from MySQL's customer-only Enterprise site or from MySQL's Bazaar repository, both under the GPL license. The MySQL Community Server is published on an unspecified schedule under the GPL and contains all bug fixes that were shipped with the last MySQL Enterprise Server release. Binaries are no longer provided by MySQL for every release of the Community Server.

Distinguishing features

The following features are implemented by MySQL but not by some other RDBMS software:

- Multiple storage engines, allowing you to choose the one which is most effective for each table in the application (in MySQL 5.0, storage engines must be compiled in; in MySQL 5.1, storage engines can be dynamically loaded at run time):
- Native storage engines (MyISAM, Falcon, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, Cluster, BDB, EXAMPLE, and Maria)
- Partner-developed storage engines (InnoDB, solidDB, NitroEDB, BrightHouse)
- Community-developed storage engines (memcached, httpd, PBXT)
- Custom storage engines

- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

Server compilation type

There are 3 types of MySQL Server Compilations for Enterprise and Community users:

- Standard: The MySQL-Standard binaries are recommended for most users, and include the InnoDB storage engine.
- Max: (not MaxDB, which is cooperation with SAP AG) is mysqld-max Extended MySQL Server. The MySQL-Max binaries include additional features that may not have been as extensively tested or are not required for general usage.
- The MySQL-Debug binaries have been compiled with extra debug information, and are not intended for production use, because the included debugging code may cause reduced performance. Beginning with MySQL 5.1, MySQL AB has stopped providing these different package variants. There will only be one MySQL server package, which includes a mysqld binary with all functionality and storage engines enabled. Instead of providing a separate debug package, a server binary with extended debugging information is also included in the standard package.

History

Milestones in MySQL development include:

- MySQL was first released internally on 23 May 1995
- Windows version was released on January 8, 1998 for Windows 95 and NT
- Version 3.23: beta from June 2000, production release January 2001
- Version 4.0: beta from August 2002, production release March 2003 (unions)
- Version 4.1: beta from June 2004, production release October 2004 (R-trees and B-trees, subqueries, prepared statements)
- Version 5.0: beta from March 2005, production release October 2005 (cursors, stored procedures, triggers, views, XA transactions)
- Version 5.1: currently pre-production (since November 2005) (event scheduler, partitioning, plugin API, row-based replication, server log tables)
- Sun Microsystems acquired MySQL AB on 26 February 2008.

Future releases

The MySQL 5.1 roadmap outlines support for:

- Pluggable storage engine API
- Partitioning
- Event Scheduling
- XML functions
- Row-based replication

Support for parallelization is also part of the roadmap for future versions.

Foreign key support for all storage engines is targeted for release in MySQL 6.1 (although it has been present since version 3.23.44 for InnoDB).

The current MySQL 5.1 development release is 5.1.25-rc.

A new storage engine is also in the works, called Falcon. A preview of Falcon is already available on MySQL's website.

Support and licensing

Via MySQL Enterprise MySQL AB offers support itself, including a 24/7 service with 30-minute response time, the support team has direct access to the developers as necessary to handle problems. In addition it hosts forums and mailing lists, employees and other users are often available in several IRC channels providing assistance.

Buyers of MySQL Enterprise enjoy access to binaries and software that is certified for their particular operating system, and access to monthly binary updates with the latest bug fixes. Several levels of Enterprise membership are available, with varying response times and features ranging from how to and emergency support through server performance tuning and system architecture advice. The MySQL Network Monitoring and Advisory Service monitoring tool for database servers is available only to MySQL Enterprise customers.

MySQL Server is available as free software under the GNU General Public License (GPL), but MySQL Enterprise subscriptions are offered for business users and dual-licensing under traditional proprietary licensing arrangements is available for cases where the intended use is incompatible with the GPL.

Both the MySQL server software itself and the client libraries are distributed under a dual-licensing format. Users may choose the GPL, which MySQL has extended with a FLOSS License Exception. It allows Software licensed under other OSI-compliant Open Source licenses, which are not compatible to the GPL, to link against the MySQL client libraries.

Customers that do not wish to be bound to the terms of the GPL may choose to purchase a proprietary license.

Like many open-source programs, the name "MySQL" is trademarked and may only be used with the trademark holder's permission.

Some users have independently continued to develop earlier versions of the client libraries, which was distributed under the less-restrictive GNU Lesser General Public License (LGPL).

Issues

There has been some controversy regarding the distribution of GPL licensed MySQL library files with other open source applications. The biggest controversy arose with PHP, which has a license incompatible with the GPL. This was later resolved when MySQL created a license exception that explicitly allows the inclusion of the MySQL client library in open source projects that are licensed under a number of OSI-compliant Open Source licenses, including the PHP License.

In September 2005, MySQL AB and SCO forged a partnership for "joint certification, marketing, sales, training and business development work for a commercial version of the database for SCO's new OpenServer 6 version of Unix". SCO raised controversy beginning in 2003 with a number of high-profile lawsuits related to the Linux Operating System. Various MySQL employees expressed that the company was committed to serving its end users, regardless of their operating system choice, that the company would leave it to the courts to resolve the SCO licensing controversy, and that other common open source databases have also been ported to, and support, SCO OpenServer.

In October 2005, Oracle Corporation acquired Innobase OY, the Finnish company that developed the InnoDB storage engine that allows MySQL to provide such functionality as transactions and foreign keys. A press release by Oracle that was issued after the acquisition, mentioned that the contracts that make the company's software available to MySQL AB come up for renewal (and presumably renegotiation) some time in 2006. During the MySQL Users Conference in April 2006, MySQL issued a press release which confirmed that MySQL and Innobase OY agreed to a multi-year extension of their licensing agreement.

In February 2006, Oracle Corporation acquired Sleepycat Software, makers of the Berkeley DB, a database engine onto which another MySQL storage engine was built.

Criticism

MySQL's divergence from the SQL standard on the subject of treatment of NULL values and default values has been criticized. Its handling of dates in versions prior to 5.0 allows storing a date with a day beyond the last day of a month with fewer than 31 days, and arithmetic operations are vulnerable to either integer overflow or floating point truncation. Since version 5 of the server, the treatment of illegal values varies according to use of the "SQL Mode" set in the server, which is by default set to the unusually tolerant state that critics dislike.

When the beta version of MySQL 5.0 was released in March 2005, David Axmark, a co-founder of MySQL, said that "People have been criticizing MySQL since we started for not having stored procedures triggers and views," and "We're fixing 10 years of criticism in one release." MySQL 5.0's 13 October build 5.0.15 was released for production use on 24 October 2005, after more than two million downloads in the 5.0 beta cycle.

SQL

SQL (Structured Query Language) (pronounced /ˈskjuːkwəl/ officially, although the unofficial pronunciation /ˈsiːkwəl/ is often used (see below)), is a database computer language designed for the retrieval and management of data in relational database management systems (RDBMS), database schema creation and modification, and database object access control management.

SQL is a standard interactive and programming language for querying and modifying data and managing databases. Although SQL is both an ANSI and an ISO standard, many database products support SQL with proprietary extensions to the standard language. The core of SQL is formed by a command language that allows the retrieval, insertion, updating, and deletion of data, and performing management and administrative functions. SQL also includes a Call Level Interface (SQL/CLI) for accessing and managing data and databases remotely.

The first version of SQL was developed at IBM by Donald D. Chamberlin and Raymond F. Boyce in the early 1970s. This version, initially called SEQUEL, was designed to manipulate and retrieve data stored in IBM's original relational database product, System R. The SQL language was later formally standardized by the American National Standards Institute (ANSI) in 1986. Subsequent versions of the SQL standard have been released as International Organization for Standardization (ISO) standards.

Originally designed as a declarative query and data manipulation language, variations of SQL have been created by SQL database management system (DBMS) vendors that add procedural constructs, control-of-flow statements, user-defined data types, and various other language extensions. With the release of the SQL: 1999 standard, many such extensions were formally adopted as part of the SQL language via the SQL Persistent Stored Modules (SQL/PSM) portion of the standard.

Common criticisms of SQL include a perceived lack of cross-platform portability between vendors, inappropriate handling of missing data (see Null (SQL)), and unnecessarily complex and occasionally ambiguous language grammar and semantics.

History

During the 1970s, a group at IBM's San Jose research center developed the System R relational database management system, based on the model introduced by Edgar F. Codd in his influential paper, A Relational Model of Data for Large Shared Data Banks. Donald D. Chamberlin and Raymond F. Boyce of IBM subsequently created the Structured English Query Language (SEQUEL) to manipulate and manage data stored in System R. The acronym SEQUEL was later changed to SQL because "SEQUEL" was a trademark of the UK-based Hawker Siddeley aircraft company.

The first non-commercial non-SQL RDBMS, Ingres, was developed in 1974 at the U.C. Berkeley. Ingres implemented a

query language known as QUEL, which was later supplanted in the marketplace by SQL.

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, CIA, and other government agencies. In the summer of 1979, Relational Software, Inc. introduced the first commercially available implementation of SQL, Oracle V2 (Version2) for VAX computers. Oracle V2 beat IBM's release of the System/38 RDBMS to market by a few weeks.

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.

Standardization

SQL was adopted as a standard by ANSI in 1986 and ISO in 1987. In the original SQL standard, ANSI declared that the official pronunciation for SQL is "es queue el". However, many English-speaking database professionals still use the nonstandard pronunciation /ˈsiːkwəl/ (like the word "sequel").

Until 1996, the National Institute of Standards and Technology (NIST) data management standards program was tasked with certifying SQL DBMS compliance with the SQL standard. In 1996, however, the NIST data management standards program was dissolved, and vendors are now relied upon to self-certify their products for compliance.

The SQL standard has gone through a number of revisions, as shown below:

Year	Name	Alias	Comments
1986	SQL-86		
1987	SQL-87		First published by ANSI. Ratified by ISO in 1987.
1989	SQL-89	FIPS 127-1	Minor revision, 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, triggers, support for procedural and control-of-flow statements, non-scalar types, and some object-oriented features.
2003	SQL:2003		
			Introduced XML-related features, window functions, standardized sequences, and columns with auto-generated values (including identity-columns).
2006	SQL:2006		

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 provides facilities that permit 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.

The SQL standard is not freely available. SQL:2003 and SQL:2006 may be purchased from ISO or ANSI. A late draft of SQL:2003 is freely available as a zip archive, however, from Whitemarsh Information Systems Corporation. The zip archive contains a number of PDF files that define the parts of the SQL:2003 specification.

Scope and extensions

Procedural extensions

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 such as C or BASIC. However, there are extensions to Standard SQL which add procedural programming language functionality, such as control-of-flow constructs. These are:

- Source
- Common
- Name
- Full Name
- ANSI/ISO Standard
- SQL/PSM
- SQL/Persistent Stored Modules
- IBM
- SQL PL
- SQL Procedural Language (implements SQL/PSM)
- Microsoft/
- Sybase
- T-SQL
- Transact-SQL
- MySQL
- SQL/PSM
- SQL/Persistent Stored Module (as in ISO SQL:2003)
- Oracle
- PL/SQL
- Procedural Language/SQL (based on Ada)
- PostgreSQL
- PL/pgSQL
- Procedural Language/PostgreSQL Structured Query Language (based on Oracle PL/SQL)
- PostgreSQL
- PL/PSM
- Procedural Language/Persistent Stored Modules (implements SQL/PSM)

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 which were primarily written in C. Other database platforms, like MySQL and Postgres, allow functions to be

written in a wide variety of languages including Perl, Python, Tcl, and C.

Additional extensions

SQL: 2003 also defines several additional extensions to the standard to increase SQL functionality overall. These extensions include:

The SQL/CLI, or Call-Level Interface, extension is defined in ISO/IEC 9075-3:2003. This extension defines common interfacing components (structures and procedures) that can be used to execute SQL statements from applications written in other programming languages. The SQL/CLI extension 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.

The SQL/MED, or Management of External Data, extension is defined by ISO/IEC 9075-9:2003. SQL/MED 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.

The SQL/OLB, or Object Language Bindings, extension is defined by ISO/IEC 9075-10:2003. SQL/OLB defines the syntax and semantics of SQLJ, which is SQL embedded in Java. The standard also describes mechanisms to ensure binary portability of SQLJ applications, and specifies various Java packages and their contained classes.

The SQL/Schemata, or Information and Definition Schemas, extension is defined by ISO/IEC 9075-11:2003. SQL/Schemata 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.

The SQL/JRT, or SQL Routines and Types for the Java Programming Language, extension is defined by ISO/IEC 9075-13:2003. SQL/JRT specifies the ability to invoke static Java methods as routines from within SQL applications. It also calls for the ability to use Java classes as SQL structured user-defined types.

The SQL/XML, or XML-Related Specifications, extension is defined by ISO/IEC 9075-14:2003. SQL/XML 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.

The SQL/PSM, or Persistent Stored Modules, extension is defined by ISO/IEC 9075-4:2003. SQL/PSM 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").

Language elements

This chart shows several of the SQL language elements that compose a single statement.

The SQL language is sub-divided into several language elements, including:

- Statements which may have a persistent effect on schemas and data, or which may control transactions, program flow, connections, sessions, or diagnostics.
- Queries which retrieve data based on specific criteria.
- 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) Boolean truth values and which are used to limit the effects of statements and queries, or to change program flow?
- Clauses which are (in some cases optional) constituent components of statements and queries.

- Whitespace is generally ignored in SQL statements and queries, making it easier to format SQL code for readability.
- 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.

Queries

The most common operation in SQL databases is the query, which is performed with the declarative SELECT keyword. SELECT retrieves data from a specified table, or multiple related tables, in a database. While often grouped with Data Manipulation Language (DML) statements, the standard SELECT query is considered separate from SQL DML, as it has no persistent effects on the data stored in a database. Note that there are some platform-specific variations of SELECT that can persist their effects in a database, such as the SELECT INTO syntax that exists in some databases.

SQL queries allow the user to specify a description of the desired result set, but it is left to the devices of the database management system (DBMS) to plan, optimize, and perform the physical operations necessary to produce that result set in as efficient a manner as possible. An SQL query includes a list of columns to be included in the final result immediately following the SELECT keyword. An asterisk ("*") can also be used as a "wildcard" indicator to specify that all available columns of a table (or multiple tables) are to be returned. SELECT is the most complex statement in SQL, with several optional keywords and clauses, including:

- The FROM clause which indicates the source table or tables from which the data is to be retrieved. The FROM clause can include optional JOIN clauses to join related tables to one another based on user-specified criteria.
- The WHERE clause includes a comparison predicate, which is used to restrict the number of rows returned by the query. The WHERE clause is applied before the GROUP BY clause. The WHERE clause eliminates all rows from the result set where the comparison predicate does not evaluate to True.
- The GROUP BY clause is used to combine, or group, rows with related values into elements of a smaller set of rows. GROUP BY is often used in conjunction with SQL aggregate functions or to eliminate duplicate rows from a result set.
- The HAVING clause includes a comparison predicate used to eliminate rows after the GROUP BY clause is applied to the result set. Because it acts on the results of the GROUP BY clause, aggregate functions can be used in the HAVING clause predicate.
- The ORDER BY clause is used to identify which columns are used to sort the resulting data, and in which order they should be sorted (options are ascending or descending). The order of rows returned by an SQL query is never guaranteed unless an ORDER BY clause is specified.

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.

```
SELECT *
FROM Book
WHERE price > 100.00
ORDER BY title;
```

The example below demonstrates the use of multiple tables in a join, grouping, and aggregation in an SQL query, by returning a list of books and the number of authors associated with each book.

```
SELECT Book.title, count (*) AS Authors
FROM Book
JOIN Book_author ON Book.isbn = Book_author.isbn
GROUP BY Book.title;
```

Example output might resemble the following:

```
Title           Authors
-----
```

SQL Examples and Guide

The Joy of SQL
How to use Wikipedia
Pitfalls of SQL

Under the precondition that isbn is the only common column name of the two tables and that a column named title only exists in the Books table, the above query could be rewritten in the following form:

```
SELECT title, count (*) AS Authors
FROM Book
NATURAL JOIN Book_author
GROUP BY title;
```

However, many vendors either do not support this approach, or it requires certain column naming conventions. Thus, it is less common in practice.

Data retrieval is very often combined with data projection when the user is looking for calculated values and not just the verbatim data stored in primitive data types, or when the data needs to be expressed in a form that is different from how it's stored. 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.

```
SELECT isbn, title, price, price * 0.06 AS sales tax
FROM Book
WHERE price > 100.00
ORDER BY title;
```

Some modern day SQL queries may include extra WHERE statements that are conditional to each other. They may look like this example:

```
SELECT isbn, title, price, date FROM Book
WHERE price > 100.00
AND (date = '16042004' OR date = '16042005')
ORDER BY title;
```

Universal quantification is not explicitly supported by sql, and must be worked out as a negated existential quantification.

Data manipulation

First, there are the standard Data Manipulation Language (DML) elements. DML is the subset of the language used to add, update and delete data:

- INSERT is used to add rows (formally tuples) to an existing table, for example:
INSERT INTO My_table (field1, field2, field3) VALUES ('test', 'N', NULL);
- UPDATE is used to modify the values of a set of existing table rows, eg:
UPDATE My_table SET field1 = 'updated value' WHERE field2 = 'N';
- DELETE removes zero or more existing rows from a table, eg:
DELETE FROM My_table WHERE field2 = 'N';
- MERGE is used to combine the data of multiple tables. It is something of a combination of the INSERT and UPDATE elements. It is defined in the SQL:2003 standard; prior to that, some databases provided similar functionality via different syntax, sometimes called an "upsert".

Transaction controls

Transactions, if available, can be used to wrap around the DML operations:

- START TRANSACTION (or BEGIN WORK, or BEGIN TRANSACTION, depending on SQL dialect) can be used to mark the start of a database transaction, which either completes entirely or not at all.
- COMMIT causes all data changes in a transaction to be made permanent.
- ROLLBACK causes all data changes since the last COMMIT or ROLLBACK to be discarded, so that the state of the data is "rolled back" to the way it was prior to those changes being requested.

Once the COMMIT statement has been executed, the changes cannot be rolled back. In other words, its meaningless to have ROLLBACK executed after COMMIT statement and vice versa.

COMMIT and ROLLBACK interact with areas such as transaction control and locking. Strictly, both terminate any open transaction and release any locks held on data. In the absence of a START TRANSACTION or similar statement, the semantics of SQL are implementation-dependent. Example: A classic bank transfer of funds transaction.

```
START TRANSACTION;
UPDATE Account SET amount=amount-200 WHERE account_number=1234;
UPDATE Account SET amount=amount+200 WHERE account_number=2345;
IF ERRORS=0 COMMIT;
IF ERRORS<>0 ROLLBACK;
```

Data definition

The second group of keywords is the Data Definition Language (DDL). DDL allows the user to define new tables and associated elements. Most commercial SQL databases have proprietary extensions in their DDL, which allow control over nonstandard features of the database system. The most basic items of DDL are the CREATE, ALTER, RENAME, TRUNCATE and DROP statements:

- CREATE causes an object (a table, for example) to be created within the database.
- DROP causes an existing object within the database to be deleted, usually irretrievably.
- TRUNCATE deletes all data from a table (non-standard, but common SQL statement).
- ALTER statement permits the user to modify an existing object in various ways -- for example, adding a column to an existing table.

Example:

```
CREATE TABLE My_table (
My_field1 INT,
My_field2 VARCHAR (50),
My_field3 DATE NOT NULL,
PRIMARY KEY (my_field1, my_field2)
```

Data control

The third group of SQL keywords is the Data Control Language (DCL). DCL handles the authorization aspects of data and permits the user to control who has access to see or manipulate data within the database. Its two main keywords are:

- GRANT authorizes one or more users to perform an operation or a set of operations on an object.
- REVOKE removes or restricts the capability of a user to perform an operation or a set of operations.

Example:

```
GRANT SELECT, UPDATE ON My_table TO some_user, another_user;
```

Other

- ANSI-standard SQL supports double dash, --, as a single line comment identifier (some extensions also support

curly brackets or C style `/* comments */` for multi-line comments).

Example:

```
SELECT * FROM Inventory -- Retrieve everything from inventory table
```

- Some SQL servers allow user-defined functions.

Criticisms of SQL

Technically, SQL is a declarative computer language for use with "SQL databases". Theorists and some practitioners note that many of the original SQL features were inspired by, but in violation of, the relational model for database management and its tuple calculus realization. Recent extensions to SQL achieved relational completeness, but have worsened the violations, as documented in The Third Manifesto.

In addition, there are also some criticisms about the practical use of SQL:

- Implementations are inconsistent and, usually, incompatible between vendors. In particular date and time syntax, string concatenation, nulls, and comparison case sensitivity often vary from vendor to vendor.
- The language makes it too easy to do a Cartesian join (joining all possible combinations), which results in "run-away" result sets when WHERE clauses are mistyped. Cartesian joins are so rarely used in practice that requiring an explicit CARTESIAN keyword may be warranted.

SQL 1992 introduced the CROSS JOIN keyword that allows the user to make clear that a Cartesian join is intended, but the shorthand "comma-join" with no predicate is still acceptable syntax.

- It is also possible to misconstrue a WHERE on an update or delete, thereby affecting more rows in a table than desired.
- The grammar of SQL is perhaps unnecessarily complex, borrowing a COBOL-like keyword approach, when a function-influenced syntax could result in more re-use of fewer grammar and syntax rules. This is perhaps due to IBM's early goal of making the language more English-like so that it is more approachable to those without a mathematical or programming background. (Predecessors to SQL were more mathematical.)

Reasons for lack of portability

Popular implementations of SQL commonly omit support for basic features of Standard SQL, such as the DATE or TIME data types, preferring variations of their

own. As a result, SQL code can rarely be ported between database systems without modifications.

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

- The complexity and size of the SQL standard means that most databases do not implement the entire standard.
- The standard does not specify database behavior in several important areas (e.g. indexes, file storage...), leaving it up to implementations of the database 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 areas of ambiguity.
- Many database vendors have large existing customer bases; where the SQL standard conflicts with the prior behavior of the vendor's database, the vendor may be unwilling to break backward compatibility.