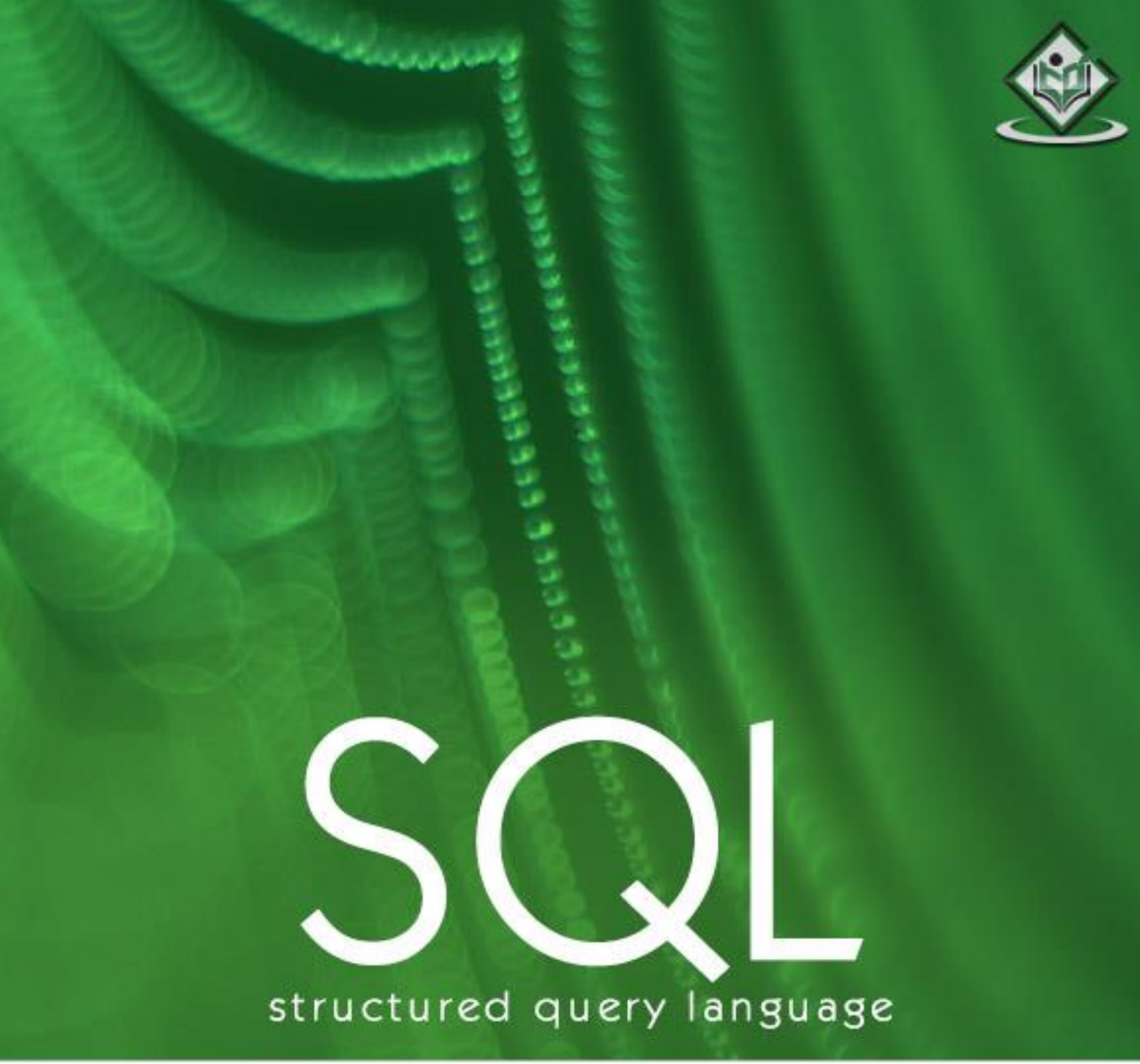
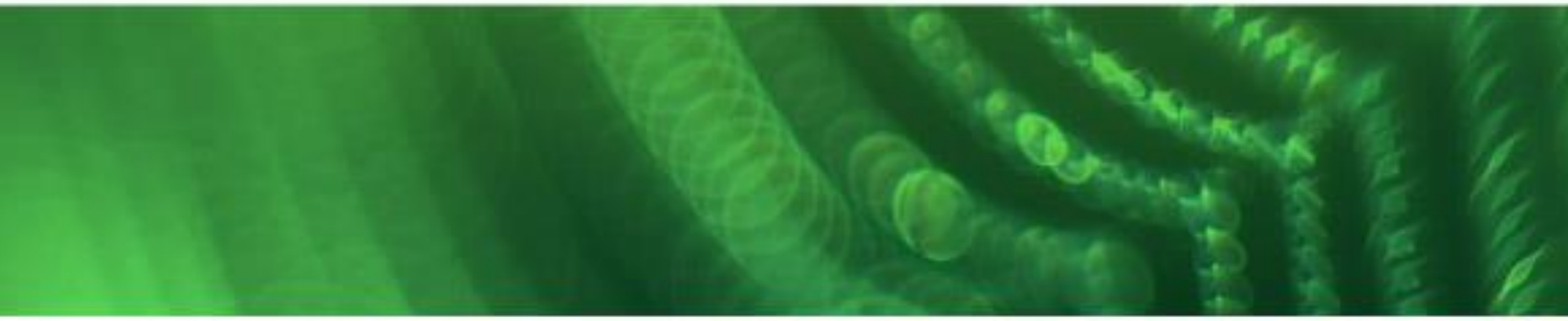


SQL

structured query language

tutorialspoint

SIMPLYEASYLEARNING



www.tutorialspoint.com

About the Tutorial

SQL is a database computer language designed for the retrieval and management of data in a relational database. SQL stands for Structured Query Language.

This tutorial will give you a quick start to SQL. It covers most of the topics required for a basic understanding of SQL and to get a feel of how it works.

Audience

This tutorial is prepared for beginners to help them understand the basic as well as the advanced concepts related to SQL languages. This tutorial will give you enough understanding on the various components of SQL along with suitable examples.

Prerequisites

Before you start practicing with various types of examples given in this tutorial, I am assuming that you are already aware about what a database is, especially the RDBMS and what is a computer programming language.

Compile/Execute SQL Programs

If you are willing to compile and execute SQL programs with Oracle 11g RDBMS but you don't have a setup for the same, do not worry. Coding Ground is available on a high-end dedicated server giving you real programming experience. It is free and is available online for everyone.

Copyright & Disclaimer

Table of Contents

1. SQL – Overview

SQL is a language to operate databases; it includes database creation, deletion, fetching rows, modifying rows, etc. SQL is an **ANSI** (American National Standards Institute) standard language, but there are many different versions of the SQL language.

What is SQL?

SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in a relational database.

SQL is the standard language for Relational Database System. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

Also, they are using different dialects, such as:

- MS SQL Server using T-SQL,
- Oracle using PL/SQL,
- MS Access version of SQL is called JET SQL (native format) etc.

Why SQL?

SQL is widely popular because it offers the following advantages:

- Allows users to access data in the relational database management systems.
- Allows users to describe the data.
- Allows users to define the data in a database and manipulate that data.
- Allows to embed within other languages using SQL modules, libraries & pre-compilers.
- Allows users to create and drop databases and tables.
- Allows users to create view, stored procedure, functions in a database.
- Allows users to set permissions on tables, procedures and views.

A Brief History of SQL

- **1970 –** Dr. Edgar F. "Ted" Codd of IBM is known as the father of relational databases. He described a relational model for databases.
- **1974 –** Structured Query Language appeared.
- **1978 –** IBM worked to develop Codd's ideas and released a product named System/R.

- **1986 –** IBM developed the first prototype of relational database and standardized by ANSI. The first relational database was released by Relational Software which later came to be known as Oracle.

SQL Process

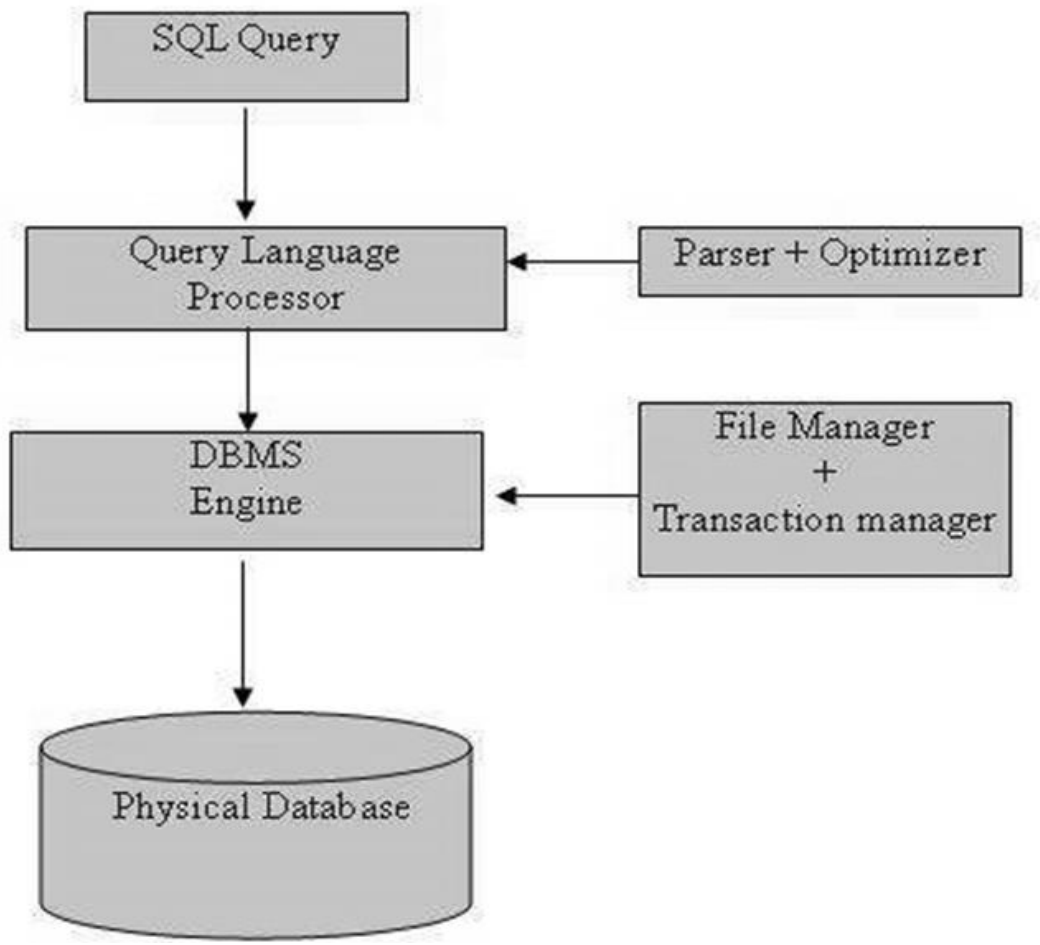
When you are executing an SQL command for any RDBMS, the system determines the best way to carry out your request and SQL engine figures out how to interpret the task. There are various components included in this process.

These components are –

- Query Dispatcher
- Optimization Engines
- Classic Query Engine
- SQL Query Engine, etc.

A classic query engine handles all the non-SQL queries, but a SQL query engine won't handle logical files.

Following is a simple diagram showing the SQL Architecture:



SQL Commands

The standard SQL commands to interact with relational databases are CREATE, SELECT, INSERT, UPDATE, DELETE and DROP. These commands can be classified into the following groups based on their nature:

DDL - Data Definition Language

Command	Description
CREATE	Creates a new table, a view of a table, or other object in the database.
ALTER	Modifies an existing database object, such as a table.
DROP	Deletes an entire table, a view of a table or other objects in the database.

DML - Data Manipulation Language

Command	Description
SELECT	Retrieves certain records from one or more tables.
INSERT	Creates a record.
UPDATE	Modifies records.
DELETE	Deletes records.

DCL - Data Control Language

Command	Description
GRANT	Gives a privilege to user.
REVOKE	Takes back privileges granted from user.

2. SQL – RDBMS Concepts

What is RDBMS?

RDBMS stands for **R**elational **D**atabase **M**anagement **S**ystem. RDBMS is the basis for SQL, and for all modern database systems like MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access.

A Relational database management system (RDBMS) is a database management system (DBMS) that is based on the relational model as introduced by E. F. Codd.

What is a table?

The data in an RDBMS is stored in database objects which are called as **tables**. This table is basically a collection of related data entries and it consists of numerous columns and rows.

Remember, a table is the most common and simplest form of data storage in a relational database. The following program is an example of a CUSTOMERS table:

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

What is a field?

Every table is broken up into smaller entities called fields. The fields in the CUSTOMERS table consist of ID, NAME, AGE, ADDRESS and SALARY.

A field is a column in a table that is designed to maintain specific information about every record in the table.

What is a Record or a Row?

A record is also called as a row of data is each individual entry that exists in a table. For example, there are 7 records in the above CUSTOMERS table. Following is a single row of data or record in the CUSTOMERS table:

```
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
+----+----------+-----+-----------+----------+
```

A record is a horizontal entity in a table.

What is a column?

A column is a vertical entity in a table that contains all information associated with a specific field in a table.

For example, a column in the CUSTOMERS table is ADDRESS, which represents location description and would be as shown below:

```
+-----------+
| ADDRESS   |
+-----------+
| Ahmedabad |
| Delhi     |
| Kota      |
| Mumbai    |
| Bhopal    |
| MP        |
| Indore    |
+----+------+
```

What is a NULL value?

A NULL value in a table is a value in a field that appears to be blank, which means a field with a NULL value is a field with no value.

It is very important to understand that a NULL value is different than a zero value or a field that contains spaces. A field with a NULL value is the one that has been left blank during a record creation.

SQL Constraints

Constraints are the rules enforced on data columns on a table. These are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the database.

Constraints can either be column level or table level. Column level constraints are applied only to one column whereas, table level constraints are applied to the entire table.

Following are some of the most commonly used constraints available in SQL:

- NOT NULL Constraint: Ensures that a column cannot have a NULL value.
- DEFAULT Constraint: Provides a default value for a column when none is specified.
- UNIQUE Constraint: Ensures that all the values in a column are different.
- PRIMARY Key: Uniquely identifies each row/record in a database table.
- FOREIGN Key: Uniquely identifies a row/record in any another database table.
- CHECK Constraint: The CHECK constraint ensures that all values in a column satisfy certain conditions.
- INDEX: Used to create and retrieve data from the database very quickly.

Data Integrity

The following categories of data integrity exist with each RDBMS:

- **Entity Integrity:** There are no duplicate rows in a table.
- **Domain Integrity:** Enforces valid entries for a given column by restricting the type, the format, or the range of values.
- **Referential integrity:** Rows cannot be deleted, which are used by other records.
- **User-Defined Integrity:** Enforces some specific business rules that do not fall into entity, domain or referential integrity.

Database Normalization

Database normalization is the process of efficiently organizing data in a database. There are two reasons of this normalization process:

- Eliminating redundant data. For example, storing the same data in more than one table.
- Ensuring data dependencies make sense.

Both these reasons are worthy goals as they reduce the amount of space a database consumes and ensures that data is logically stored. Normalization consists of a series of guidelines that help guide you in creating a good database structure.

Normalization guidelines are divided into normal forms; think of a form as the format or the way a database structure is laid out. The aim of normal forms is to organize the database structure, so that it complies with the rules of first normal form, then second normal form and finally the third normal form.

It is your choice to take it further and go to the fourth normal form, fifth normal form and so on, but in general, the third normal form is more than enough.

- First Normal Form (1NF)

- Second Normal Form (2NF)
- Third Normal Form (3NF)

Database – First Normal Form (1NF)

The First normal form (1NF) sets basic rules for an organized database:

- Define the data items required, because they become the columns in a table.
- Place the related data items in a table.
- Ensure that there are no repeating groups of data.
- Ensure that there is a primary key.

First Rule of 1NF

You must define the data items. This means looking at the data to be stored, organizing the data into columns, defining what type of data each column contains and then finally putting the related columns into their own table.

For example, you put all the columns relating to locations of meetings in the Location table, those relating to members in the MemberDetails table and so on.

Second Rule of 1NF

The next step is ensuring that there are no repeating groups of data. Consider we have the following table:

```
CREATE TABLE CUSTOMERS(
       ID   INT              NOT NULL,
       NAME VARCHAR (20)     NOT NULL,
       AGE  INT              NOT NULL,
       ADDRESS  CHAR (25),
       ORDERS   VARCHAR(155)
);
```

So, if we populate this table for a single customer having multiple orders, then it would be something as shown below:

ID	NAME	AGE	ADDRESS	ORDERS
100	Sachin	36	Lower West Side	Cannon XL-200
100	Sachin	36	Lower West Side	Battery XL-200

100	Sachin	36	Lower West Side	Tripod Large

But as per the 1NF, we need to ensure that there are no repeating groups of data. So, let us break the above table into two parts and then join them using a key as shown in the following program:

CUSTOMERS Table

```
CREATE TABLE CUSTOMERS(
       ID   INT              NOT NULL,
       NAME VARCHAR (20)     NOT NULL,
       AGE  INT              NOT NULL,
       ADDRESS  CHAR (25),
       PRIMARY KEY (ID)
);
```

This table would have the following record:

ID	NAME	AGE	ADDRESS
100	Sachin	36	Lower West Side

ORDERS Table

```
CREATE TABLE ORDERS(
       ID   INT              NOT NULL,
       CUSTOMER_ID INT       NOT NULL,
       ORDERS   VARCHAR(155),
       PRIMARY KEY (ID)
);
```

This table would have the following records:

ID	CUSTOMER_ID	ORDERS
10	100	Cannon XL-200
11	100	Battery XL-200

12	100	Tripod Large

Third Rule of 1NF

The final rule of the first normal form, create a primary key for each table which we have already created.

Database – Second Normal Form (2NF)

The Second Normal Form states that it should meet all the rules for 1NF and there must be no partial dependences of any of the columns on the primary key:

Consider a customer-order relation and you want to store customer ID, customer name, order ID and order detail and the date of purchase:

```
CREATE TABLE CUSTOMERS(
       CUST_ID    INT              NOT NULL,
       CUST_NAME VARCHAR (20)      NOT NULL,
       ORDER_ID   INT              NOT NULL,
       ORDER_DETAIL VARCHAR (20)  NOT NULL,
       SALE_DATE  DATETIME,
       PRIMARY KEY (CUST_ID, ORDER_ID)
);
```

This table is in the first normal form; in that it obeys all the rules of the first normal form. In this table, the primary key consists of the CUST_ID and the ORDER_ID. Combined, they are unique assuming the same customer would hardly order the same thing.

However, the table is not in the second normal form because there are partial dependencies of primary keys and columns. CUST_NAME is dependent on CUST_ID and there's no real link between a customer's name and what he purchased. The order detail and purchase date are also dependent on the ORDER_ID, but they are not dependent on the CUST_ID, because there is no link between a CUST_ID and an ORDER_DETAIL or their SALE_DATE.

To make this table comply with the second normal form, you need to separate the columns into three tables.

First, create a table to store the customer details as shown in the code block below:

```
CREATE TABLE CUSTOMERS(
       CUST_ID    INT              NOT NULL,
       CUST_NAME VARCHAR (20)      NOT NULL,
       PRIMARY KEY (CUST_ID)
```

```
);
```

The next step is to create a table to store the details of each order:

```
CREATE TABLE ORDERS(
       ORDER_ID   INT              NOT NULL,
       ORDER_DETAIL VARCHAR (20)  NOT NULL,
       PRIMARY KEY (ORDER_ID)
);
```

Finally, create a third table storing just the CUST_ID and the ORDER_ID to keep a track of all the orders for a customer:

```
CREATE TABLE CUSTMERORDERS(
       CUST_ID    INT              NOT NULL,
       ORDER_ID   INT              NOT NULL,
       SALE_DATE  DATETIME,
       PRIMARY KEY (CUST_ID, ORDER_ID)
);
```

Database – Third Normal Form (3NF)

A table is in a third normal form when the following conditions are met:

- It is in the second normal form.
- All non-primary fields are dependent on the primary key.

The dependency of these non-primary fields is between the data. For example, in the following table – the street name, city and the state are unbreakably bound to their zip code.

```
CREATE TABLE CUSTOMERS(
       CUST_ID       INT              NOT NULL,
       CUST_NAME     VARCHAR (20)      NOT NULL,
       DOB           DATE,
       STREET        VARCHAR(200),
       CITY          VARCHAR(100),
       STATE         VARCHAR(100),
       ZIP           VARCHAR(12),
       EMAIL_ID      VARCHAR(256),
       PRIMARY KEY (CUST_ID)
```

```
);
```

The dependency between the zip code and the address is called as a transitive dependency. To comply with the third normal form, all you need to do is to move the Street, City and the State fields into their own table, which you can call as the Zip Code table.

```
CREATE TABLE ADDRESS(
       ZIP            VARCHAR(12),
       STREET         VARCHAR(200),
       CITY           VARCHAR(100),
       STATE          VARCHAR(100),
       PRIMARY KEY (ZIP)
);
```

The next step is to alter the CUSTOMERS table as shown below.

```
CREATE TABLE CUSTOMERS(
       CUST_ID        INT              NOT NULL,
       CUST_NAME      VARCHAR (20)      NOT NULL,
       DOB            DATE,
       ZIP            VARCHAR(12),
       EMAIL_ID       VARCHAR(256),
       PRIMARY KEY (CUST_ID)
);
```

The advantages of removing transitive dependencies are mainly two-fold. First, the amount of data duplication is reduced and therefore your database becomes smaller.

The second advantage is data integrity. When duplicated data changes, there is a big risk of updating only some of the data, especially if it is spread out in many different places in the database.

For example, if the address and the zip code data were stored in three or four different tables, then any changes in the zip codes would need to ripple out to every record in those three or four tables.

3. SQL – RDBMS Databases

There are many popular RDBMS available to work with. This tutorial gives a brief overview of some of the most popular RDBMS's. This would help you to compare their basic features.

MySQL

MySQL is an open source SQL database, which is developed by a Swedish company – MySQL AB. MySQL is pronounced as "my ess-que-ell," in contrast with SQL, pronounced "sequel."

MySQL is supporting many different platforms including Microsoft Windows, the major Linux distributions, UNIX, and Mac OS X.

MySQL has free and paid versions, depending on its usage (non-commercial/commercial) and features. MySQL comes with a very fast, multi-threaded, multi-user and robust SQL database server.

History

- Development of MySQL by Michael Widenius & David Axmark beginning in 1994.
- First internal release on 23rd May 1995.
- Windows Version was released on the 8th January 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 4.01: beta from August 2003, Jyoti adopts MySQL for database tracking.
- Version 4.1: beta from June 2004, production release October 2004.
- Version 5.0: beta from March 2005, production release October 2005.
- Sun Microsystems acquired MySQL AB on the 26th February 2008.
- Version 5.1: production release 27th November 2008.

Features

- High Performance.
- High Availability.
- Scalability and Flexibility Run anything.
- Robust Transactional Support.
- Web and Data Warehouse Strengths.
- Strong Data Protection.

- Comprehensive Application Development.
- Management Ease.
- Open Source Freedom and 24 x 7 Support.
- Lowest Total Cost of Ownership.

MS SQL Server

MS SQL Server is a Relational Database Management System developed by Microsoft Inc. Its primary query languages are:

- T-SQL
- ANSI SQL

History

- 1987 - Sybase releases SQL Server for UNIX.
- 1988 - Microsoft, Sybase, and Aston-Tate port SQL Server to OS/2.
- 1989 - Microsoft, Sybase, and Aston-Tate release SQL Server 1.0 for OS/2.
- 1990 - SQL Server 1.1 is released with support for Windows 3.0 clients.
- Aston - Tate drops out of SQL Server development.
- 2000 - Microsoft releases SQL Server 2000.
- 2001 - Microsoft releases XML for SQL Server Web Release 1 (download).
- 2002 - Microsoft releases SQLXML 2.0 (renamed from XML for SQL Server).
- 2002 - Microsoft releases SQLXML 3.0.
- 2005 - Microsoft releases SQL Server 2005 on November 7th, 2005.

Features

- High Performance
- High Availability
- Database mirroring
- Database snapshots
- CLR integration
- Service Broker
- DDL triggers
- Ranking functions
- Row version-based isolation levels
- XML integration

- TRY...CATCH
- Database Mail

ORACLE

It is a very large multi-user based database management system. Oracle is a relational database management system developed by 'Oracle Corporation'.

Oracle works to efficiently manage its resources, a database of information among the multiple clients requesting and sending data in the network.

It is an excellent database server choice for client/server computing. Oracle supports all major operating systems for both clients and servers, including MSDOS, NetWare, UnixWare, OS/2 and most UNIX flavors.

History

Oracle began in 1977 and celebrating its 32 wonderful years in the industry (from 1977 to 2009).

- 1977 - Larry Ellison, Bob Miner and Ed Oates founded Software Development Laboratories to undertake development work.
- 1979 - Version 2.0 of Oracle was released and it became first commercial relational database and first SQL database. The company changed its name to Relational Software Inc. (RSI).
- 1981 - RSI started developing tools for Oracle.
- 1982 - RSI was renamed to Oracle Corporation.
- 1983 - Oracle released version 3.0, rewritten in C language and ran on multiple platforms.
- 1984 - Oracle version 4.0 was released. It contained features like concurrency control - multi-version read consistency, etc.
- 1985 - Oracle version 4.0 was released. It contained features like concurrency control - multi-version read consistency, etc.
- 2007 - Oracle released Oracle11g. The new version focused on better partitioning, easy migration, etc.

Features

- Concurrency
- Read Consistency
- Locking Mechanisms
- Quiesce Database
- Portability
- Self-managing database
- SQL*Plus

- ASM
- Scheduler
- Resource Manager
- Data Warehousing
- Materialized views
- Bitmap indexes
- Table compression
- Parallel Execution
- Analytic SQL
- Data mining
- Partitioning

MS ACCESS

This is one of the most popular Microsoft products. Microsoft Access is an entry-level database management software. MS Access database is not only inexpensive but also a powerful database for small-scale projects.

MS Access uses the Jet database engine, which utilizes a specific SQL language dialect (sometimes referred to as Jet SQL).

MS Access comes with the professional edition of MS Office package. MS Access has easy-to-use intuitive graphical interface.

- 1992 - Access version 1.0 was released.
- 1993 - Access 1.1 released to improve compatibility with inclusion the Access Basic programming language.
- The most significant transition was from Access 97 to Access 2000
- 2007 - Access 2007, a new database format was introduced ACCDB which supports complex data types such as multi valued and attachment fields.

Features

- Users can create tables, queries, forms and reports and connect them together with macros.
- Option of importing and exporting the data to many formats including Excel, Outlook, ASCII, dBase, Paradox, FoxPro, SQL Server, Oracle, ODBC, etc.
- There is also the Jet Database format (MDB or ACCDB in Access 2007), which can contain the application and data in one file. This makes it very convenient to distribute the entire application to another user, who can run it in disconnected environments.
- Microsoft Access offers parameterized queries. These queries and Access tables can be referenced from other programs like VB6 and .NET through DAO or ADO.

- The desktop editions of Microsoft SQL Server can be used with Access as an alternative to the Jet Database Engine.
- Microsoft Access is a file server-based database. Unlike the client-server relational database management systems (RDBMS), Microsoft Access does not implement database triggers, stored procedures or transaction logging.

4. SQL – Syntax

SQL is followed by a unique set of rules and guidelines called Syntax. This tutorial gives you a quick start with SQL by listing all the basic SQL Syntax.

All the SQL statements start with any of the keywords like SELECT, INSERT, UPDATE, DELETE, ALTER, DROP, CREATE, USE, SHOW and all the statements end with a semicolon (;).

The most important point to be noted here is that SQL is **case insensitive**, which means SELECT and select have same meaning in SQL statements. Whereas, MySQL makes difference in table names. So, if you are working with MySQL, then you need to give table names as they exist in the database.

Various Syntax in SQL

All the examples given in this tutorial have been tested with a MySQL server.

SQL SELECT Statement

```
SELECT column1, column2....columnN
FROM   table_name;
```

SQL DISTINCT Clause

```
SELECT DISTINCT column1, column2....columnN
FROM   table_name;
```

SQL WHERE Clause

```
SELECT column1, column2....columnN
FROM   table_name
WHERE  CONDITION;
```

SQL AND/OR Clause

```
SELECT column1, column2....columnN
FROM   table_name
WHERE  CONDITION-1 {AND|OR} CONDITION-2;
```

SQL IN Clause

```
SELECT column1, column2....columnN
FROM   table_name
WHERE  column_name IN (val-1, val-2,...val-N);
```

SQL BETWEEN Clause

```
SELECT column1, column2....columnN
FROM   table_name
WHERE  column_name BETWEEN val-1 AND val-2;
```

SQL LIKE Clause

```
SELECT column1, column2....columnN
FROM   table_name
WHERE  column_name LIKE { PATTERN };
```

SQL ORDER BY Clause

```
SELECT column1, column2....columnN
FROM   table_name
WHERE  CONDITION
ORDER BY column_name {ASC|DESC};
```

SQL GROUP BY Clause

```
SELECT SUM(column_name)
FROM   table_name
WHERE  CONDITION
GROUP BY column_name;
```

SQL COUNT Clause

```
SELECT COUNT(column_name)
FROM   table_name
WHERE  CONDITION;
```

SQL HAVING Clause

```
SELECT SUM(column_name)
FROM   table_name
WHERE  CONDITION
GROUP BY column_name
HAVING (arithematic function condition);
```

SQL CREATE TABLE Statement

```
CREATE TABLE table_name(
column1 datatype,
column2 datatype,
column3 datatype,
.....
columnN datatype,
PRIMARY KEY( one or more columns )
);
```

SQL DROP TABLE Statement

```
DROP TABLE table_name;
```

SQL CREATE INDEX Statement

```
CREATE UNIQUE INDEX index_name
ON table_name ( column1, column2,...columnN);
```

SQL DROP INDEX Statement

```
ALTER TABLE table_name
DROP INDEX index_name;
```

SQL DESC Statement

```
DESC table_name;
```

SQL TRUNCATE TABLE Statement

```
TRUNCATE TABLE table_name;
```

SQL ALTER TABLE Statement

```
ALTER TABLE table_name {ADD|DROP|MODIFY} column_name {data_ype};
```

SQL ALTER TABLE Statement (Rename)

```
ALTER TABLE table_name RENAME TO new_table_name;
```

SQL INSERT INTO Statement

```
INSERT INTO table_name( column1, column2....columnN)
VALUES ( value1, value2....valueN);
```

SQL UPDATE Statement

```
UPDATE table_name
SET column1 = value1, column2 = value2....columnN=valueN
[ WHERE  CONDITION ];
```

SQL DELETE Statement

```
DELETE FROM table_name
WHERE  {CONDITION};
```

SQL CREATE DATABASE Statement

```
CREATE DATABASE database_name;
```

SQL DROP DATABASE Statement

```
DROP DATABASE database_name;
```

SQL USE Statement

```
USE database_name;
```

SQL COMMIT Statement

```
COMMIT;
```

SQL ROLLBACK Statement

```
ROLLBACK;
```

5. SQL – Data Types

SQL Data Type is an attribute that specifies the type of data of any object. Each column, variable and expression has a related data type in SQL. You can use these data types while creating your tables. You can choose a data type for a table column based on your requirement.

SQL Server offers six categories of data types for your use which are listed below –

Exact Numeric Data Types

DATA TYPE	FROM	TO
bigint	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
int	-2,147,483,648	2,147,483,647
smallint	-32,768	32,767
tinyint	0	255
bit	0	1
decimal	-10^38 +1	10^38 -1
numeric	-10^38 +1	10^38 -1
money	-922,337,203,685,477.5808	+922,337,203,685,477.5807
smallmoney	-214,748.3648	+214,748.3647

Approximate Numeric Data Types

DATA TYPE	FROM	TO
float	-1.79E + 308	1.79E + 308
real	-3.40E + 38	3.40E + 38

Date and Time Data Types

DATA TYPE	FROM	TO
datetime	Jan 1, 1753	Dec 31, 9999
smalldatetime	Jan 1, 1900	Jun 6, 2079
date	Stores a date like June 30, 1991	
time	Stores a time of day like 12:30 P.M.	

Note – Here, datetime has 3.33 milliseconds accuracy where as smalldatetime has 1 minute accuracy.

Character Strings Data Types

DATA TYPE	Description
char	Maximum length of 8,000 characters.(Fixed length non-Unicode characters)
varchar	Maximum of 8,000 characters.(Variable-length non-Unicode data).
varchar(max)	Maximum length of 231characters, Variable-length non-Unicode data (SQL Server 2005 only).
text	Variable-length non-Unicode data with a maximum length of 2,147,483,647 characters.

Unicode Character Strings Data Types

DATA TYPE	Description
nchar	Maximum length of 4,000 characters.(Fixed length Unicode)
nvarchar	Maximum length of 4,000 characters.(Variable length Unicode)

nvarchar(max)	Maximum length of 231characters (SQL Server 2005 only).(Variable length Unicode)
ntext	Maximum length of 1,073,741,823 characters. (Variable length Unicode)

Binary Data Types

DATA TYPE	Description
binary	Maximum length of 8,000 bytes(Fixed-length binary data)
varbinary	Maximum length of 8,000 bytes.(Variable length binary data)
varbinary(max)	Maximum length of 231 bytes (SQL Server 2005 only). (Variable length Binary data)
image	Maximum length of 2,147,483,647 bytes. (Variable length Binary Data)

Misc Data Types

DATA TYPE	Description
sql_variant	Stores values of various SQL Server-supported data types, except text, ntext, and timestamp.
timestamp	Stores a database-wide unique number that gets updated every time a row gets updated
uniqueidentifier	Stores a globally unique identifier (GUID)
xml	Stores XML data. You can store xml instances in a column or a variable (SQL Server 2005 only).
cursor	Reference to a cursor object
table	Stores a result set for later processing

End of ebook preview

If you liked what you saw…

Buy it from our store @ https://store.tutorialspoint.com