NAMA : ARDILLAH

NIM : 200250502015

MAKUL : SISTEM BASIS DATA

TUGAS

**SQL TUTORIAL**

SQL adalah bahasa standar untuk menyimpan, memanipulasi dan mengambil data dalam database. Tutorial SQL kami akan mengajarkan Anda cara menggunakan SQL di: MySQL, SQL Server, MS Access, Oracle, Sybase, Informix, Postgres, dan sistem database lainnya.

SELECT \* FROM Customers;

**SQL HOME**

SQL (structure Query Languange) adalah bahasa kueri yang dirancang untuk membuat dalam pengmabilan dan mengelolah informasi pada sebuah database. Untuk yang masih pemula dalam dunia IT, biasanya diartikan sebagai bahasa untuk mengakses sebuah data dalam basis relasional.

**SQL INTRO**

SQL adalah singkatan dari Structured Query Language

SQL memungkinkan Anda mengakses dan memanipulasi database

SQL menjadi standar American National Standards Institute (ANSI) pada 1986, dan International Organization for Standardization (ISO) pada 1987

SQL dapat mengeksekusi query terhadap database

SQL dapat mengambil data dari database

SQL dapat menyisipkan catatan dalam database

SQL dapat memperbarui catatan dalam database

SQL dapat menghapus catatan dari database

SQL dapat membuat database baru

SQL dapat membuat tabel baru dalam database

SQL dapat membuat prosedur tersimpan dalam database

SQL dapat membuat tampilan dalam database

SQL dapat mengatur izin pada tabel, prosedur, dan tampilan

**SQL SYNTAX**

Database paling sering berisi satu atau lebih tabel. Setiap tabel diidentifikasi dengan nama (misalnya "Pelanggan" atau "Pesanan"). Tabel berisi catatan (baris) dengan data

SELECT \* FROM Customers;

**SQL SELECT**

Perintah SELECT merupakan perintah dasar SQL yang di gunakan untuk memilih data dari database. Data yang di kembalikan di simpan dalam tabel yang di sebut result-set.

*SELECT kolom1, kolom2, … FROM nama\_tabel;*  
*SELECT \* FROM nama\_tabel;*

**SQL SELECT DISTINCT**

Perintah SELECT DISTINCT merupakan perintah dasar SQL yang di gunakan untuk mengembalikan hanya nilai yang berbeda dari dalam sebuah tabel, dengan kata lain semua record duplikat (record dengan nilai yang sama) yang terdapat pada tabel akan di anggap sebagai satu record/nilai.

*SELECT DISTINCT kolom1, kolom2, … FROM nama\_tabel;*

**SQL WHERE SQL ,AND, OR, NOT**

Perintah WHERE merupakan perintah dasar SQL yang di gunakan untuk mem-filter hasil SELECT dengan mengekstrak record yang memenuhi persyaratan tertentu.

*SELECT kolom1, kolom2, … FROM nama\_tabel WHERE kondisi;*

Operator AND, OR dan NOT merupakan perintah dasar SQL yang biasanya di kombinasikan dengan perintah WHERE. Ketiganya di gunakan untuk mem-filter record berdasarkan suatu kondisi, operator AND akan menampilkan record apabila semua kondisi bernilai TRUE, operator OR akan menampilkan record apabila salah satu kondisi bernilai TRUE, sedangkan operator NOT akan menampilkan record apabila semua kondisi bernilai FALSE.

*SELECT kolom1, kolom2, … FROM nama\_tabel WHERE kondisi1 AND kondisi2 AND kondisi3;*  
*SELECT kolom1, kolom2, … FROM nama\_tabel WHERE kondisi1 OR kondisi2 OR kondisi3 …;*

**SQL ORDER BY**

Perintah ORDER BY merupakan perintah dasar SQL yang di gunakan untuk mengurutkan result-set dalam pengurutan ‘ascending’ atau ‘descending’. Secara default perintah ORDER BY menampilkan record dalam pengurutan ‘ascending’ (‘ASC’). Untuk mengurutkan ‘descending’, gunakan kata kunci ‘DESC’.

*SELECT kolom1, kolom2, … FROM nama\_tabel ORDER BY column DESC;*

*SELECT nis, nama FROM siswa ORDER BY tahun\_lahir DESC;*

**SQL INSERT INTO**

Dalam SQL, perintah INSERT INTO merupakan perintah dasar SQL bagian dari perintah untuk DML (Data Manipulation Language) Saya asumsikan Anda telah faham perbedaan DDL, DCL, dan DML. Perintah INSERT INTO dapat di gunakan untuk menambahkan record baru ke dalam tabel.

*INSERT INTO nama\_tabel VALUES (nilai1, nilai2, nilai3, …);*  
*INSERT INTO nama\_tabel (kolom1, kolom2) VALUES (nilai1, nilai2)*

**SQL NULL VALUES**

**SQL UPDATES SQL DELETE**

Perintah UPDATE merupakan perintah dasar SQL yang di gunakan untuk memperbarui atau mengubah nilai suatu record berdasarkan kriteria tertentu.

*UPDATE nama\_tabel SET kolom1 = nilai1, kolom2 = nilai2, … WHERE kondisi;*

Hampir sama dengan perintah UPDATE, perintah DELETE juga merupakan perintah dasar SQL yang di gunakan untuk menghapus nilai suatu record berdasarkan kriteria tertentu.

*DELETE FROM table\_name WHERE condition;*

**SQL SELECT TOP**

SELECT TOP bisa untuk menspesifikasi angka maksimal atau persentase data entri dalam set hasil pengolahan data.

**SELECT** \* **FROM** Produksi

**LIMIT** 5;

**SELECT** **TOP** 70 **PERCENT** \* **FROM** Produks;

**SQL MIN AND MAX**

Fungsi MIN() merupakan perintah dasar SQL yang di gunakan untuk mendapatkan nilai terkecil dari suatu kolom, Anda dapat menerapkannya pada kolom ‘harga’, ‘nilai’, ‘qty’ atau kolom yang semisal dengan itu, berbeda dengan perintah ORDEY BY, fungsi MIN() hanya menampilkan satu record saja yang memenuhi kriteria yang Anda tentukan.

*SELECT MIN(nama\_kolom) FROM nama\_tabel WHERE kondisi;*

Fungsi MAX() merupakan perintah dasar SQL yang di gunakan untuk mendapatkan nilai terbesar dari suatu kolom, seperti halnya fungsi MIN() Anda dapat menerapkannya pada kolom ‘harga’, ‘nilai’, ‘qty’ atau kolom yang semisal dengan itu.

*SELECT MAX(nama\_kolom) FROM nama\_tabel WHERE kondisi;*

**SQL COUNT, AVG, SUM**

Fungsi COUNT() merupakan perintah dasar SQL yang di gunakan untuk mendapatkan jumlah hitungan record yang memenuhi suatu kriteria.

*SELECT COUNT(nama\_kolom) FROM nama\_tabel WHERE kondisi;*

Fungsi AVG() merupakan perintah dasar SQL yang di gunakan untuk mendapatkan rata-rata record yang memenuhi suatu kriteria, tentunya nilai pada kolom harus numerik.

*SELECT AVG(nama\_kolom) FROM nama\_tabel WHERE kondisi;*

Fungsi SUM() merupakan perintah dasar SQL yang di gunakan untuk mendapatkan jumlah record yang memenuhi suatu kriteria, tentunya nilai pada kolom harus numerik.

*SELECT SUM(nama\_kolom) FROM nama\_tabel WHERE kondisi;*

**SQL LIKE**

Gabungkan fungsi LIKE dan WHERE untuk menemukan pola data yang spesifik di kolom database.

**SELECT** \* **FROM** Produks **WHERE** Kategori LIKE '%Kursi%';

**SQL WILD CARDS**

**SQL IN**

Kode operasi IN bekerja dengan melibatkan nilai multiple dari WHERE.

**SELECT** \* **FROM** Produks

**WHERE** Kategori IN ('Kursi', 'Keyboard', 'Sepatu');

**SQL BETWEEN**

BETWEEN digunakan untuk memfilter hasil pengelolaan database dan mengembalikan (return) hasil yang paling sesuai dengan range (rentang) spesifik data. Anda bisa menggunakan kode BETWEEN untuk mengelola data berupa angka, tanggal atau teks.

**SELECT** \* **FROM** Produks

**WHERE** ProdukID BETWEEN 5 AND 25;

**SQL ALIESES**

Alias ​​​​SQL digunakan untuk memberi tabel, atau kolom dalam tabel, nama sementara. Alias ​​​​sering digunakan untuk membuat nama kolom lebih mudah dibaca. Alias ​​​​hanya ada selama kueri itu. Alias ​​​​dibuat dengan ASkata kunci.

SELECT column\_name AS alias\_name  
FROM table\_name;

**SQL JOINS**

Sebuah JOINklausul digunakan untuk menggabungkan baris dari dua atau lebih tabel, berdasarkan kolom terkait antara mereka.

SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate  
FROM Orders  
INNER JOIN Customers ON Orders.CustomerID=Customers.CustomerID;

**SQL INNER JOIN**

INNER JOIN merupakan perintah dasar SQL yang di gunakan untuk menggabungkan beberapa tabel dan mengambil nilai yang cocok (identik) di antara kedua tabel tersebut.

*SELECT nama\_kolom1, nama\_kolom2, … FROM tabel1 INNER JOIN tabel2 ON tabel1.nama\_kolom = tabel2.nama\_kolom;*

**SQL LEFT JOIN**

LEFT JOIN merupakan perintah dasar SQL yang di gunakan untuk menggabungkan beberapa tabel dan mengambil nilai yang cocok (identik) di antara kedua tabel tersebut dan nilai lain dari tabel pada ruas kiri meskipun tak ada nilai yang cocok dengan tabel pada ruas kanan.

*SELECT nama\_kolom1, nama\_kolom2, … FROM tabel1 LEFT JOIN tabel2 ON tabel1.nama\_kolom = tabel2.nama\_kolom;*

**SQL RIGHT JOIN**

**Inner Join** mengembalikan baris-baris dari dua tabel atau lebih yang memenuhi syarat.

*SELECT columns  
FROM TableA  
INNER JOIN TableB  
ON A.columnName = B.columnName;*

**SQL FULL JOIN**

**Full Outer Join** atau sering disingkat dengan **Full Join** akan mengembalikan seluruh baris dari kedua tabel yang dikenai ON termasuk data-data yang bernilai NULL.

*SELECT columns  
FROM TableA  
FULL JOIN TableB  
ON A.columnName = B.columnName;*

**SQL SELF JOIN**

Self JOIN adalah gabungan reguler, tetapi tabel bergabung dengan dirinya sendiri.

Contoh:

SELECT A.NamaCustomer AS NamaCustomer1, B.NamaCustomer AS NamaCustomer2, A.Kota

FROM Customers A, Customers B

WHERE A.IdCustomer <> B.IdCustomer

AND A.Kota = B.Kota

ORDER BY A.Kota;

**SQL UNION**

Anda bisa mengombinasikan set hasil secara multiple (ganda) menggunakan UNION dengan dua atau lebih statement SELECT.

**SELECT** NamaProduk **FROM** Produks

**UNION**

**SELECT** NamaProduk **FROM** Produks

**ORDER BY** NamaProduk;

**SQL GROUP BY**

Kombinasikan GROUP BY dengan perintah SELECT untuk mengatur data-data identik (semisal baris data dengan nilai sama) ke dalam grup yang sama.

**SELECT** **COUNT**(ProdukID), Kategori

**FROM** Produks

**GROUP** **BY** Kategori;

**SQL HAVING**

HAVING digunakan untuk melakukan filter hasil pengelolaan data dalam baris yang telah ditentukan sebelumnya. Fungsi dari HAVING tidak jauh berbeda dengan WHERE. Bedanya, HAVING digunakan untuk mengeksekusi fungsi agregat dari data.

**SELECT** **COUNT**(ProdukID), NamaProduk

**FROM** Produks

**GROUP** **BY** NamaProduk

**HAVING** **COUNT**(ProdukID) > 5;

**SQL EXISTS**

EXISTS memungkinkan Anda untuk mengecek kembali ketersediaan sebuah data dengan menulis sebuah subquery saja. Jika data ditemukan, maka hasilnya akan ditampilkan dalam statement yang sedang dijalankan. Anda bisa mengombinasikan fitur ini dengan fitur SQL lain seperti INSERT, SELECT, DELETE dan UPDATE.

**SELECT** ProdukID, NamaProduk

**FROM** Produks

**WHERE** EXISTS (**SELECT** Produk\_id **FROM** Orders **WHERE** Orders.Produk\_id = ProdukID AND Produks.Kategori = "Kursi");

**SQL ANY, ALL**

Operator ANY dan ALL dapat digunakan dengan klausa WHERE atau [HAVING](https://dosenit.com/sql/klausa-having-sql-syntax-dan-contohnya). Operator ANY akan mengembalikan true jika salah satu nilai subquery memenuhi kondisi. Operator ALL akan mengembalikan true jika semua nilai subkueri memenuhi kondisi.

SELECT column\_name(s)

FROM table\_name

WHERE column\_name operator ANY

(SELECT column\_name FROM table\_name WHERE condition);

SELECT column\_name(s)

FROM table\_name

WHERE column\_name operator ALL

(SELECT column\_name FROM table\_name WHERE condition);

**SQL SELECT INTO**

Statement SELECT INTO digunakan untuk dapat menyalin data dari satu tabel ke tabel baru.

**SELECT \***

**INTO newtable [IN externaldb]**

**FROM oldtable**

**WHERE condition;**

**SQL INSERT INTO SELECT**

INSERT INTO SELECTperintah salinan data dari satu meja dan memasukkan ke meja lain.

INSERT INTO Customers (CustomerName, City, Country)  
SELECT SupplierName, City, Country FROM Suppliers;

**SQL CASE**

CASEpernyataan melewati kondisi dan mengembalikan nilai saat kondisi pertama bertemu (seperti pernyataan if-then-else). Jadi, begitu suatu kondisi benar, itu akan berhenti membaca dan mengembalikan hasilnya. Jika tidak ada kondisi yang benar, ia mengembalikan nilai dalam ELSEklausa. Jika tidak ada ELSEbagian dan tidak ada kondisi yang benar, ia mengembalikan NULL.

CASE  
    WHEN condition1 THEN result1  
    WHEN condition2 THEN result2  
    WHEN conditionN THEN resultN  
    ELSE result  
END;

**SQL NULL FUNCTIONS**

Microsoft ISNULL() adalah fungsi yang digunakan untuk menentukan bagaimana kita ingin memperlakukan nilai NULL.

The NVL() , IFNULL() , dan COALESCE() fungsi juga dapat digunakan untuk mencapai hasil yang sama.

Dalam hal ini kita ingin nilai NULL menjadi nol.

SELECT ProductName,UnitPrice\*(UnitsInStock+IFNULL(UnitsOnOrder,0))  
FROM Products

**SQL STORED PROSEDURES**

Stored Procedured (Prosedur tersimpan) adalah kode [SQL](https://dosenit.com/sql/sql) yang disiapkan untuk dapat di simpan, sehingga kode tersebut dapat digunakan kembali berulang kali. Jadi, jika memiliki kueri SQL yang ingin ditulis berulang kali, simpan sebagai store procedure (prosedur tersimpan), lalu panggil saja untuk menjalankannya. Kita juga bisa menggunakannya untuk meneruskan parameter ke store procedure (prosedur tersimpan), sehingga store procedure (prosedur tersimpan) dapat bertindak berdasarkan nilai parameter yang diteruskan.

CREATE PROCEDURE SelectAllCustomers

AS

SELECT \* FROM Customers

GO

**SQL COMMENTS**

comment sql adalah perintah dasar kita ingin membuat sebuah komentar pada script yang kita buat, komentar biasanya untuk membuat sebagai catatan mengenai script tersebut. atau juga bisa untuk menonaktifkan sebuah script.

a. menonaktifkan script dalam satu baris:

-- SELECT \* From nama\_tabel ;

 b. membuat sebuah catetan

-- Query untuk melihat tabel

**SQL OPERATORS**

Operator dalam SQL merupakan karakter atau kata yang dicadangkan serta digunakan terutama dalam klausa WHERE pernyataan SQL untuk melakukan operasi. Adapun beberapa operasi yang dilakukannya  seperti perbandingan dan operasi aritmatika. Operator dalam SQL digunakan untuk memastikan kondisi dalam statment SQL sehingga ia dapat berfungsi sebagai konjungsi untuk beberapa kondisi dalam statement atau pernyataan.

Operator dalam SQL ini utamanya digunakan dalam klausa WHERE dari pernyataan SQL. Ini adalah bagian dari pernyataan yang digunakan untuk menyaring data berdasarkan kondisi atau kondisi tertentu. Ada enam jenis operator SQL yang akan kita bahas: Arithmetic, Bitwise, Comparison, Compound, dan Logical. Berikut ini adalah penjelasan masing-masing operator.

**SQL DATABASE**

Query dapat diartikan sebagai capability (kemampuan) database pada komputer untuk menampilkan/menyimpan informasi tertentu. Namun perlu diketahui bahwa kemampuan tersebut hanya berupa pre-defined code artinya jika permintaan tidak sesuai dengan informasi yang tersimpan di database, maka permintaan tidak dapat diproses.

Membuat Table (CREATE TABLE namaTable)

Database (CREATE DATABASE namaDatabase)

Index (CREATE [UNIQUE]  INDEX namaIndex ON namaTable (namaKolom))

View (CREATE VIEW namaView AS SELECT STATEMENT FROM namaTable)

**SQL CREATE DB**

CREATE DATABASE statement digunakan untuk membuat database baru. Akan tetapi untuk dapat menggunakannya, Anda harus sudah mengantongi izin akses sebagai admin.

**CREATE** **DATABASE** nama\_database

**SQL DROP DB**

DROP DATABASEpernyataan digunakan untuk menjatuhkan database SQL yang ada.

DROP DATABASE testDB;

**SQL BACKUP DB**

BACKUP DATABASEpernyataan digunakan dalam SQL Server untuk membuat kembali penuh atas sebuah database SQL yang ada.

BACKUP DATABASE testDB  
TO DISK = 'D:\backups\testDB.bak';

**SQL CREATE TABLE**

CREATE TABLE dipakai untuk membuat tabel baru dalam database.

**CREATE** **TABLE** nama\_tabel (

nama\_kolom1 tipe\_data(panjang data),

nama

**SQL DROP TABLE**

DROP TABLE digunakan untuk menghapus seluruh tabel data, lengkap dengan tipe data dan parameter yang telah ditentukan sebelumnya. Jika Anda ingin menghapus isi tabel saja, gunakan TRUNCATE TABLE.

**DROP** **TABLE** tbl\_name;

**SQL ALTER TABLE**

ALTER TABLEpernyataan digunakan untuk menambah, menghapus, atau memodifikasi kolom di tabel yang ada.

ALTER TABLE Customers  
ADD Email varchar(255);

**SQL CONSTRAINTS**

Batasan dapat ditentukan ketika tabel dibuat dengan CREATE TABLEpernyataan, atau setelah tabel dibuat dengan ALTER TABLEpernyataan.

CREATE TABLE table\_name (  
    column1 datatype *constraint*,  
    column2 datatype *constraint*,  
    column3 datatype *constraint*,  
    ....  
);

**SQL NUT NULL**

NOT NULLkendala memaksa kolom untuk TIDAK menerima nilai-nilai NULL. Ini memaksa bidang untuk selalu berisi nilai, yang berarti bahwa Anda tidak dapat menyisipkan catatan baru, atau memperbarui catatan tanpa menambahkan nilai ke bidang ini.

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255) NOT NULL,  
    Age int  
);

**SQL UNIQUE**

UNIQUEkendala memastikan bahwa semua nilai-nilai dalam kolom yang berbeda. Baik batasan UNIQUEmaupun PRIMARY KEYbatasan memberikan jaminan keunikan untuk kolom atau kumpulan kolom. Sebuah PRIMARY KEYkendala otomatis memiliki UNIQUEkendala. Namun, Anda dapat memiliki banyak UNIQUEbatasan per tabel, tetapi hanya satu PRIMARY KEYbatasan per tabel.

CREATE TABLE Persons (  
    ID int NOT NULL UNIQUE,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int  
);

**SQL PRIMARY KEY**

PRIMARY KEYkendala unik mengidentifikasi setiap record dalam sebuah tabel. Kunci utama harus berisi nilai UNIK, dan tidak boleh berisi nilai NULL. Sebuah tabel hanya dapatmemilikiSATU kunci utama; dan dalam tabel, kunci utama ini dapat terdiri dari satu atau beberapa kolom (bidang).

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    PRIMARY KEY (ID)  
);

**SQL FROIGN KEY**

FOREIGN KEYadalah bidang (atau kumpulan bidang) dalam satu tabel, yang merujuk ke [PRIMARY KEY](https://www-w3schools-com.translate.goog/sql/sql_primarykey.asp?_x_tr_sl=en&_x_tr_tl=id&_x_tr_hl=id&_x_tr_pto=nui,sc)dalam tabel lain. Tabel dengan kunci asing disebut tabel anak, dan tabel dengan kunci utama disebut tabel referensi atau tabel induk.

CREATE TABLE Orders (  
    OrderID int NOT NULL,  
    OrderNumber int NOT NULL,  
    PersonID int,  
    PRIMARY KEY (OrderID),  
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)  
);

**SQL CHECK**

HECKkendala digunakan untuk membatasi rentang nilai yang dapat ditempatkan dalam kolom.Jika Anda menentukan CHECKbatasan pada kolom, itu hanya akan mengizinkan nilai tertentu untuk kolom ini.

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    CHECK (Age>=18)  
);

**SQL DEFAULT**

DEFAULTkendala digunakan untuk mengatur nilai default untuk kolom. Nilai default akan ditambahkan ke semua catatan baru, jika tidak ada nilai lain yang ditentukan.

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    City varchar(255) DEFAULT 'Sandnes'  
);

**SQL INDEX**

CREATE INDEXpernyataan digunakan untuk membuat indeks dalam tabel.Indeks digunakan untuk mengambil data dari database lebih cepat daripada sebaliknya. Pengguna tidak dapat melihat indeks, mereka hanya digunakan untuk mempercepat pencarian/kueri.

CREATE INDEX index\_name  
ON table\_name ( column1, column2, ...);

CREATE INDEX index\_name  
ON table\_name ( column1, column2, ...);

**SQL DATES**

MySQL hadir dengan tipe data berikut untuk menyimpan tanggal atau nilai tanggal/waktu dalamdatabase:

DATE - format YYYY-MM-DD

DATETIME - format: YYYY-MM-DD HH:MI:SS

TIMESTAMP - format: YYYY-MM-DD HH:MI:SS

YEAR - format YYYY atau YY

1.SQL Server

dilengkapi dengan tipe data berikut untuk menyimpan tanggal atau nilai tanggal/waktu dalam database:

DATE - format YYYY-MM-DD

DATETIME - format: YYYY-MM-DD HH:MI:SS

SMALLDATETIME - format: YYYY-MM-DD HH:MI:SS

TIMESTAMP - format: nomor unik

**SQL VIEWS**

Dalam SQL, tampilan adalah tabel virtual berdasarkan kumpulan hasil dari pernyataan SQL. Tampilan berisi baris dan kolom, seperti tabel nyata. Bidang dalam tampilan adalah bidang dari satu atau beberapa tabel nyata dalam database. Anda dapat menambahkan pernyataan dan fungsi SQL ke tampilan dan menyajikan data seolah-olah data berasal dari satu tabel.Tampilan dibuat dengan CREATE VIEWpernyataan.

CREATE VIEW [Brazil Customers] AS   
SELECT CustomerName, ContactName  
FROM Customers  
WHERE Country = 'Brazil';

**SQL INJECTION**

**SQL HOSTING**

**SQL DATA TYPES**

Setiap kolom dalam tabel database harus memiliki nama dan tipe data. Pengembang SQL harus memutuskan jenis data apa yang akan disimpan di dalam setiap kolom saat membuat tabel. Tipe data adalah pedoman bagi SQL untuk memahami tipe data apa yang diharapkan di dalam setiap kolom, dan juga mengidentifikasi bagaimana SQL akan berinteraksi dengan data yang disimpan.

**SQL REFESERENC**

**SQL KEYWORDS**

**MYSQL FUNCTIONS**

**SQL SERVER FUNCTIONS**

QL Server memiliki banyak fungsi bawaan. Referensi ini berisi string, numerik, tanggal, konversi, dan beberapa fungsi lanjutan di SQL Server.

**MS ACCES FUNCTIONS**

MS Access memiliki banyak fungsi bawaan. Referensi ini berisi fungsi string, numerik, dan tanggal di MS Access.

**SQL QUICK REF**